

DesignCon 2010

Accurately Timing Analyzing Memory Interfaces in the Presence of Calibrated Paths

Navid Azizi, Altera Corporation
nazizi@altera.com

Joshua Fender, Altera Corporation
jfender@altera.com

Ryan Fung, Altera Corporation
rfung@altera.com

Abstract

In an effort to increase IC bandwidth, particularly that of memories, the data rates of interface standards have been increasing. To meet timing at these high data rates, interfacing ICs must calibrate at power-up. Because the timing analysis of paths calibrated during runtime does not strictly meet the STA paradigm used predominately in EDA tools, new methods are necessary to model them accurately. This paper describes methods to obtain accurate timing of paths that are calibrated at runtime, and combines conventional STA with new methods to obtain timing that considers both calibration and post-calibration effects.

Author Biographies

Navid Azizi received the B.A.Sc., the M.A.Sc., and Ph.D. degrees in computer engineering from the University of Toronto in 2001, 2003, and 2007, respectively. His areas of interest include memory interfaces, specifically timing analysis, SSN, and submicron circuit design. In June 2007, he joined Altera's Toronto Technology Center, where he works in the I/O modeling and timing group.

Joshua Fender received the B.A.Sc. and M.A.Sc. degrees in computer engineering from the University of Toronto. His areas of expertise include I/O modeling, circuit simulation, and SSN. He currently manages Altera's I/O modeling group and is responsible for I/O timing, power, and signal integrity modeling within the Quartus® II software product.

Ryan Fung received the B.A.Sc. degree in Engineering Science (Computer option) from the University of Toronto in 2002. Currently, he is a senior member of technical staff at Altera Corporation's Toronto Technology Center. He holds eight issued U.S. patents from his work at Altera involving placement and routing algorithms, FPGA architecture, and timing analysis.

A. Introduction

In an effort to increase integrated circuit (IC) bandwidth, particularly that of memories, the data rates of interface standards have been increasing. To meet timing at these high data rates, interfacing ICs must calibrate at power-up to reduce skew between signals, to center-align clock and strobe signals, and to perform other operations.

Because timing analyzing paths which are calibrated does not strictly meet (STA) paradigm used predominately in EDA tools, new methods are necessary to accurately model them. Timing analyzing a path without considering calibration provides results that are too pessimistic. On the other hand, assuming calibration is perfect provides overly optimistic results because voltage and temperature variations over time effect the device's calibration.

A further subtype of a calibrated path is a path that is both calibrated at runtime and tracked continuously to update for operating condition variations. This subtype requires even further changes to the timing analysis paradigm.

Calibrated paths can be used in any part of an IC. However, when these paths are used to interface one IC to another (such as a memory), the timing analysis of those paths can be further upgraded to consider the effects where the process variation in the other device is calibrated out by the calibration process. Typically, external components provide specifications for their parts that comprise all types of variations. Without considering what types of variations can be removed through calibration, the timing analysis provides pessimistic results, which in turn leads to limitations in the operating speed.

This paper describes a method to obtain accurate timing analysis of paths that are calibrated at runtime, and optionally tracked throughout operation. The method combines conventional STA with new methods and procedures to obtain a timing analysis that considers the effects of calibration at runtime, the effects that degrade the effects of calibration over the operation of the device, and improvements to timing margins by calibrating out the process portion of variations on external devices.

The rest of this paper is organized as follows: Section B discusses some background on timing analysis. Section C explains how to timing analyze calibrated portions of one's design. Sections D through H cover the details of the process, which include calibration emulation, pessimism removal, calibration errors, the effect of changing operating conditions, and external interfaces respectively. Section I describes a case study in which Altera® Quartus® II software has integrated these algorithms and features in its timing analysis for a DDR3 interface. Finally, Section J concludes the paper.

B. Background

Static Timing Analysis

STA is a method used to compute the expected timing of a digital circuit without performing simulation [1]. Computing the expected timing of a digital circuit is important because it determines if a circuit can operate at a specific clock frequency. STA is also conventionally used as part of other algorithms in the digital circuit design process to help guide decisions and produce a design that works at a specified clock frequency.

Typically, STA is used instead of simulation because its runtime is orders of magnitude smaller than simulation techniques [1]. The runtime is important because obtaining accurate delay information is required by multiple other processes where STA is embedded. STA produces two values for each analyzed path: the setup margin and the hold margin, which refer respectively to the margin available on a signal path for which the signal has to be stable before and after a clock arrives [1].

To account for the operation of the design across different operating conditions that affect delay, STA is run across many extreme conditions or corners [2]. If the design works under each extreme condition, then, assuming monotonic behavior, the design is also qualified for all intermediate points [2].

Memory, Memory Controllers, and Calibration

A memory controller is a digital circuit that manages the flow of data to and from memory devices such as DDR3. Memory controllers can be separate devices, or they can be integrated onto other devices, such as FPGAs. The main role of the memory controller is to coordinate the signals necessary to read and write data from the memory.

In an effort to increase their bandwidth, many memory interface standards, such as DDR3 and RLDRAM II, have increased the data rates that they operate at [3][4]. To meet the timing requirements and operate at these high data rates, memory controllers may require calibration at power-up to reduce skew between signals and to center-align clock signals [5].

As an example, consider a DDR3 memory which can run at high data rates consisting of many parallel signals. For a memory controller to correctly interface with the DDR3 memory device it has to be able to capture data returning from the memory, which is known as a *read capture* operation. The read capture occurs when the memory's strobe signal (DQS) is used to clock the memory's data signals (multiple DQ signals). The region of time where the strobe signal can switch can correctly clock in all the DQ signals is known as the data valid window (DVW). The timing of the read capture path is critical as significant skew between various DQ pins causes the valid window to shrink or be non-existent, as shown in Figure 1 (left). Furthermore, even with the availability of a valid window, the DQS strobe may not be placed in an appropriate location within the window.

Per bit deskew is a process where delays on different DQ signal paths on the interfacing device are calibrated to make each of the DQ signal paths approximately equal. At the same time, DQS is placed in the middle of the valid window, resulting in the scenario shown in Figure 1 (right). The hardware calibrating process necessary to achieve that

goal involves increasing and decreasing delays on each signal path until the read capture fails, then picking a midpoint delay value.

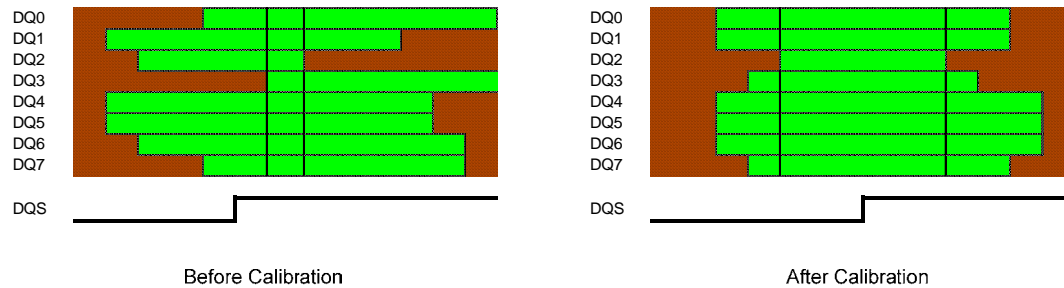


Figure 1: Effect of Calibration on Timing Margins

C. Timing Analyzing Calibrated Paths

STA works very well when defining the expected timing behavior of a circuit where the topology of the circuit is known. When the topology of the circuit changes during operation, as in the DDR3 calibration processes, the STA paradigm no longer holds and cannot be used to obtain accurate delay information. However, as with all other electronic designs, timing analysis must occur on interfaces that contain calibrated portions in order to compute the expected timing of the interface and determine if the design can operate at the desired clock frequency. Simulation of the design can provide accurate delay information, but simulation runtimes are orders of magnitude larger than STA [1], which make simulation unsuitable for real-time feedback or within other optimizations where STA is typically used.

One alternative to using simulation to determine the timing analysis of a calibrated path is to first assume that the calibration occurs perfectly, and then to remove certain uncertainties from the analysis. The problem with this methodology is that not every delay uncertainty is specified, so none of the STA analysis and data is used. In addition, because no STA calls are performed to obtain accurate delays, this type of analysis does not consider the actual operations that occur within the calibration process and how they affect the delays in the electronic device. Furthermore, this type of analysis cannot consider the effects on timing when the device is operating through multiple operating conditions.

A better technique is for the timing analysis of calibrated paths to use as much of the STA methodology as possible because the STA engines contain accurate delay information for the IC and know about the relationship between signals. The STA methodology must then be augmented with information about the operations performed during calibration and tracking, as well as the effects of any changes due to operating conditions variations. These types of operations allow for timing analysis to remain both quick (as opposed to simulation) and accurate (as opposed to assuming perfect calibration and uncertainty removal), so that it can be used in the optimization process.

The procedures described in this paper, and implemented in Quartus II software, timing analyze paths that are calibrated at runtime and include additional types of processes that work in conjunction with and around a conventional STA tool. The algorithm starts by obtaining an estimate of the timing margins using a conventional STA tool. The

continued use of the standard STA engine allows the timing analysis to obtain actual delays for different paths at different temperature and voltage corners as well as for different speed-grades. These types of delays allow a designer to obtain specific delay information for effects such as rise/fall imbalance (RF) and within-die variation, as well as other effects.

Once this estimate is obtained, the software emulates the calibration algorithm that occurs on the hardware (discussed in section D). Usually the calibration involves adding or subtracting delays to a path. Once this emulation step is complete, the timing analysis is closer to practice, but it contains both optimistic and pessimistic components. The analysis is optimistic because it does not take into account whether the calibration algorithm is accurate, nor does it consider the effects of noise on the calibration algorithm (discussed in Section O). It also doesn't include the effects of changes in the operating condition from when calibration occurs to when the design is operating (discussed in Section G). At the same time, the analysis is pessimistic because it contains within it some baseline assumptions from the STA tool that are no longer true when considering that the path's delay has been calibrated, such as the uncertainty due to within-die (WID) variations (discussed in Section E).

Both the optimistic and the pessimistic factors must be analyzed and removed to obtain an accurate timing analysis. Sections D through H explain each of these steps in detail to show how an accurate timing analysis for calibrated paths is obtained.

D. Calibration Emulation

Conventional STA cannot be used to timing analyze calibrated paths because the topology of the paths is dynamic and changes when the device is put into operation. Usually the calibration involves adding or subtracting delays to a path. To remove this limitation, the STA methodology must be changed to allow for the emulation of the calibration process.

The first method of emulating the calibration process is to integrate STA within a loop that includes calls to procedures containing algorithmic content that emulates the calibration process. Pseudo-code of this method is shown in Figure 2.

```
while (~calibration done) {  
    delays = run STA  
    calibration_done = calibration_algorithm(delays)  
}
```

Figure 2: Pseudo-Code of One Possible Method of Calibration Emulation

With this method, the delays obtained through STA are added to a procedure that emulates the calibration process occurring on the hardware and updates the topology of the circuitry. Since the topology of the circuitry has changed, the STA is re-evaluated to provide new delays, and this process iterates until the calibration process provides final timing values for the paths of interest. The benefit of this analysis is that the STA engine obtains all the delays and thus includes all the accuracy and runtime benefits. However, this method may be too slow to be used in an iterative process.

Instead of running multiple STA calls, a more efficient method of emulating the calibration process is to perform a single STA call and then post-process the delay information to approximate what occurs in calibration. Because there is only one call to STA, this method is faster than the previous method, but is somewhat less accurate since the precise delay effects of changing the topology are not calculated. However, this is not an issue because calibration algorithms are used in hardware devices when delay information is known to be incomplete (if one would know the delay information exactly, one could design specifically to it). In these cases, the amount of delay uncertainty does not negatively affect the timing analysis since the calibration algorithm calibrates to a near-optimum point and the calibration emulation assumes the same. The specific amount of uncertainty in the calibration process is covered by other portions of the algorithm (described in Sections F and G).

For example, when running timing analysis on the read capture of a memory controller, STA provides initial delays for each of the DQ signal paths. These delays and margins are used by the emulation algorithm to estimate the extra delay added during calibration to each signal path to make the paths relatively equal in delay and to center the DQS strobe. Using these estimates, the timing analysis emulates the hardware calibration, and obtains a better estimate of timing margins typically available.

E. Pessimism Removal

In conventional STA, each delay on the IC is modeled as a range of delays with a minimum value and a maximum value, as shown in Figure 3. The minimum and maximum values represent the range of delays that can occur for a specific path due to various factors such as within-die (WID) variations or the effects of circuit aging [1]. When STA is run, the value that provides for the worst-case margin is used in the analysis. For example, when timing analyzing the setup margin, the maximum value of the data path is used and the minimum value of the clock path is used. On the other hand, when timing analyzing the hold margin, the minimum value of the data path is used and the maximum value of the clock path is used.

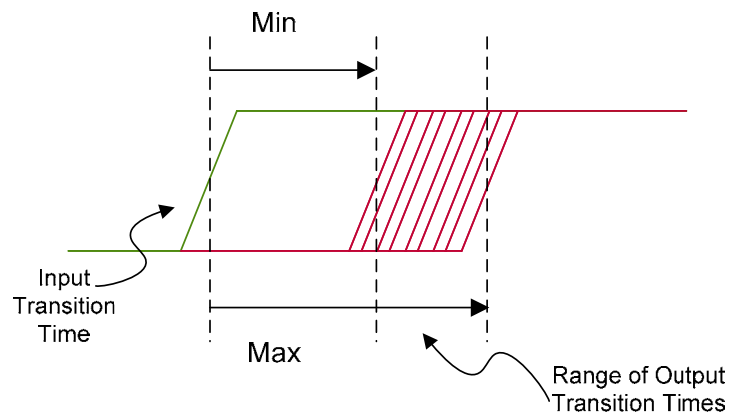


Figure 3: Range of Transition Times

Notice that for a single path, the setup- and hold-margins use opposite ends of the range of delays. While, in practice, this scenario cannot happen on a single IC (the data path delay is either towards the maximum or the minimum, but not both), the same path on

different ICs can have different delays and thus it is appropriate to timing analyze the path as explained previously. The use of a range of values to represent each delay results in the difference between the minimum and maximum value being removed from the total margin (the combination of setup and hold margin).

During the timing analysis of a calibrated path, the delay and margin information obtained from the STA engine implicitly contain the minimum and maximum values for each delay arc. However, using both the minimum and maximum values in computing different margins is pessimistic when considering a calibrated path. The purpose of calibration is to remove the uncertainty in the delay of a path and to pick the circuit topology that best meets the requirements. Thus, when considering the delay of a path after calibration, the path should no longer be considered to have a range of delays but a single delay.

To remove the pessimism that is inherent in using delay values obtained from the STA engine, the difference in delay between the maximum/minimum delay and nominal delay of the paths used to calculate the setup and hold margin should be added to the margin calculated. This step emulates the process of calibrating out the WID variation on the IC. Consider a path that has a nominal delay of 1 ns, with a minimum delay of 0.95 ns and a maximum delay of 1.05 ns. When considering the setup slack, the maximum delay of 1.05 ns is used instead of 1 ns, and thus the setup slack is reduced by 0.05 ns. Similarly, when considering the hold slack, the minimum delay of 0.95 ns is used instead of 1 ns, and thus the hold slack is reduced by 0.05 ns. When this same path is timing analyzed as part of a calibrated path analysis, the values provided by the STA tool (either 0.95 ns or 1.05 ns depending on whether the setup slack or hold slack is calculated) are pessimistic since the calibrated path has a single calibrated delay. To remove the pessimism, based on the assumptions of the STA engine, the range of delays due to WID variations or other effects should be added back to the total margin determined after calibration.

F. Calibration Errors and Noise Effects

After the calibration process has been emulated, another aspect that must be considered is calibration errors. Calibration algorithms are used in hardware devices when delay information is incomplete, because if one would know the delay information exactly, one could design specifically to it. Unknown delays in the memory controller include the delays between the memory and the controller, as well as delays in the memory. Since the delay information is unknown a priori, the timing analysis of the calibrated paths using STA is working with incomplete data. This missing information may cause the timing analysis calibration operations to pick different topologies than what would actually occur in hardware. These differences between what can occur in hardware and what occurs in the timing analysis must be quantified and included in the timing analysis of the calibrated paths.

In most cases, hardware calibration processes change the delays on various signal paths in specific increments or quanta. When increasing delays with specific increments, the calibration algorithm can result in delays that, in the worst-case, are offset from the optimal delay by the increment divided by two.

Figure 4 shows an example where a DQS strobe is centered within the valid window of all DQ pins using a quantized delay chain structure. In Case (a), the initial DQS position and the delay of the quantized steps allow the DQS strobe to be perfectly centered in the window, so there is no quantization error. On the other hand, in Case (b), the initial DQS position does not allow the strobe to be perfectly centered in the window. In this case, the calibration algorithm picks one of two locations for DQS, as well as a setting that reduces the hold margin by half of the quantization step. In practice, a well-designed calibration algorithm can always pick a setting that reduces either the setup or the hold margin by half of the quantization error, at most. However, since it is unknown what the initial DQS position is in regards to the DQ valid window, the quantization error must be removed from both the setup and the hold margins.

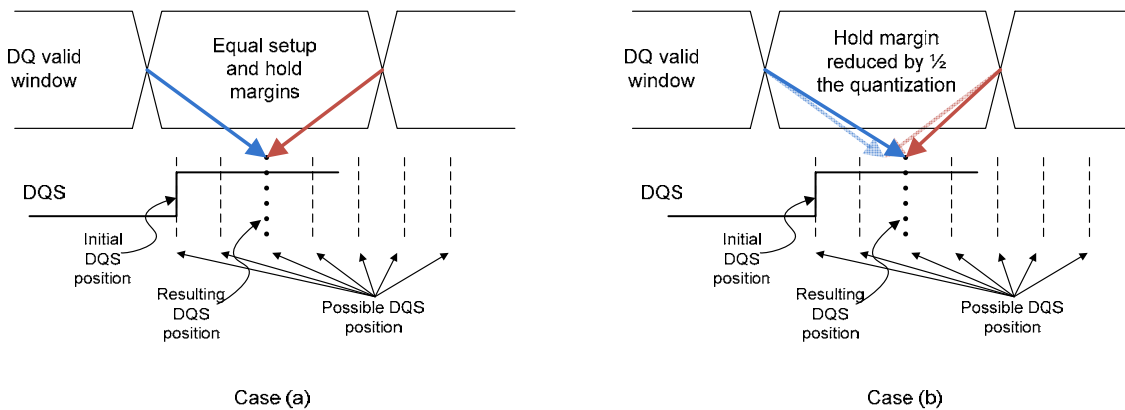


Figure 4: Quantization Error Effect

The delay increments come from various sources including delay chains (where the quantization error is the delay chain step) and PLL phases (where the quantization error is the minimum PLL adjustment). In both cases, during the timing analysis of the calibrated path, the worst-case quantization error must be assumed and removed from all timing margins produced.

A final aspect that must be considered is the effect of noise, specifically jitter. Jitter is the time variation of signals as a result of complex interactions of signals in an IC. The method of correctly accounting for jitter in the timing analysis depends on the calibration algorithm. Calibration algorithms that take many samples and use the mean to decide to make a change in the topology of a circuit remove the effects of jitter on their decisions. In this case, the worst-case jitter effect on each of the setup and hold margins is half of the peak-to-peak jitter. On the other hand, if the calibration algorithm takes a few samples to make a decision, it may use an errant timing signal to make its decision and thus calibrate non-optimally. In this case, the timing analysis must assume that it calibrated at a non-optimal location and remove the peak-to-peak jitter from both the setup and hold margins.

G. Considering Operating Conditions

Until now, the effect of changing operating conditions has not been included in the analysis. In a conventional STA, the design is analyzed at many extreme corners and the assumption of monotonic behavior is used to qualify all intermediate points. The timing

corners picked tend to be cases where circuits are faster or slower than nominal, including low temperatures and high voltage with fast devices, and high temperatures and low voltage with slow devices. In some cases, due to temperature inversion effects where certain devices slow down at low temperatures, other corners must also be included, such as low temperature and low voltage with slow devices [2]. As manufacturers produce ICs at increasingly smaller process nodes, further corners may be necessary to cover the space of possible timing [6]. This methodology accounting for operating condition variations cannot be used directly for the timing analysis of calibrated paths, because there are two different times at which operating conditions matter: *during* calibration time and *after* calibration time.

For the timing analysis of a calibrated path, the analysis must be run for a number of different corners, as in conventional STA. However, since the STA methodology has changed to consider the effects of calibration on the path it is analyzing, the operating condition used during analysis must be the same as the operating condition used *during* calibration. In this case, as with conventional STA, the design is analyzed at many extreme corners (i.e., calibration can occur in different corners) and the assumption of monotonic behavior qualifies all intermediate points during calibration. Operating condition changes *after* calibration are not considered in this analysis, but must be included for an accurate non-optimistic analysis.

The treatment of the effect of changes in the operating condition after calibration is quite different from that during calibration. The following subsection provides a background on the types of variations and how they differ from one another, while the subsequent subsection provides details on how these effects are found and applied to the results of the previous step in the timing analysis.

Types of Variation

Variations are typically grouped into three different types: process variations (P), voltage variations (V), and temperature variations (T). Together these compose PVT variations. The maximum P variation is determined by comparing different dies, while the maximum V and T variations are determined by operating a design at the endpoints of the range of voltage and temperature.

There are, however, some distinguishing characteristics between the three types of variations. In the first order, P variations do not change once the chip has been fabricated, while V and T variations change over time. In addition, T variations are typically low-frequency changes that gradually shift through the operation of the device, while V variations are either low-frequency changes that gradually change over time, or high-frequency changes that manifest themselves as jitter.

When considering operating condition variations, only the low-frequency VT variations must be considered, as P variations do not change with time, and high-frequency jitter effects were considered during calibration emulation. To simplify the analysis, we initially assume that the VT changes affect all the signal paths on the die (i.e. there is no VT gradient across the die) and that the delays in the path scale similarly with changes in VT. These assumptions may not be true in practice, but cause secondary and minor effects in the timing analysis.

Operating Condition Variation after Calibration

To determine the effects of changing operating conditions after calibration on the timing margin, two values must be defined and found: the percentage variation in delay that VT variations can cause, and the maximum absolute change in delay of a path due to the calibration process.

The variation in delay due to the VT variations is usually expressed as a percentage of the nominal delay (usually a function of the IC's silicon process), and the device's manufacturer-specified operating range of voltage and temperature. Determined for each product line, this value is stored and looked up during the timing analysis.

The second value, the maximum absolute change in delay due to the calibration process, is defined as the largest change in delay of a signal path as a result of calibration. This value is determined from the calibration algorithm and the hardware where the calibration is running. For example, if a delay chain is used to add delay to a signal path, the largest absolute change in delay of the calibrated path is equivalent to the longest delay of the delay chain. This value is also found a priori, and is stored and looked up during the timing analysis.

Once both values are determined, the amount to which the timing margins are reduced due to the effects of changes in the operating conditions after calibration is calculated as the percentage change in delay multiplied by the maximum change in delay due to calibration. The reason that only the maximum change in delay due to calibration is used in this analysis, and not the total absolute delay, is that, by definition, the conventional way of dealing with operating variations through corner analysis covers for variations in delays from topologies that are static and do not change during calibration. The changes in delay in calibrated portions of the design are not included anywhere in the analysis and must be included here.

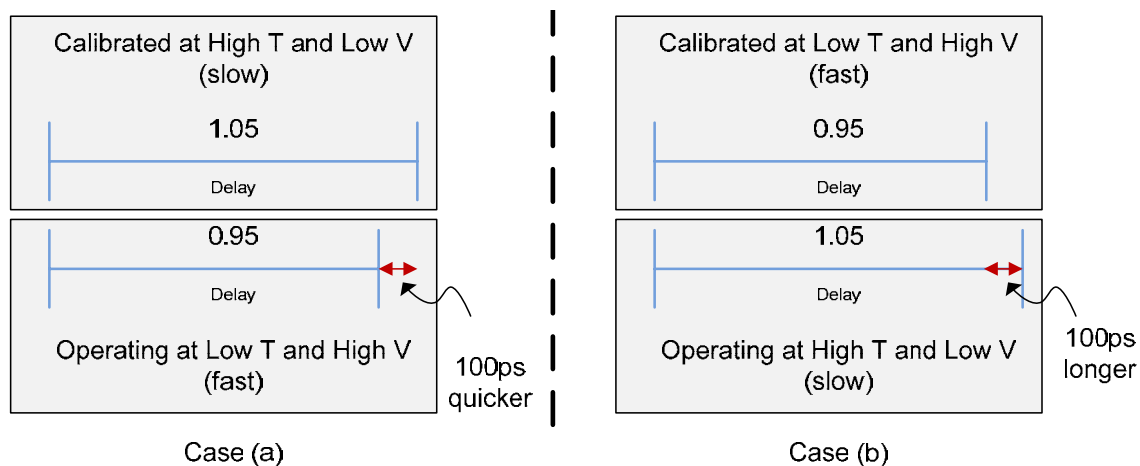


Figure 5: Effect of Variations in Operating Conditions After Calibration

As an example, consider the scenarios shown in Figure 5, where VT variations cause $\pm x = 5\%$ variation in delay from the nominal delays and the maximum change in delay due to calibration is 1 ns. In Case (a), the system was calibrated in slow conditions but operates in fast conditions, which causes the calibrated delay to change from 1.05 ns to

0.95 ns, a delay of 100 ps. If the delay is on a data path, the hold margin is reduced by 100 ps. The opposite occurs in Case (b), where the system was calibrated in fast conditions but operates in slow conditions. In this case, the calibrated delay has increased by 100 ps, resulting in a setup margin reduced by 100 ps. In all situations, the effect of changes in the operating margin must reduce both the setup and hold margins because if calibration occurs in one operating condition extreme, the device may operate in the opposite operating condition extreme.



Figure 6: Effect of VT Variations on Paths Matched to One Another

This analysis is pessimistic as it assumes delays added in calibration achieve some absolute delay, and in these cases, the variation in delay found must be removed from all margins. However, in cases where the calibration process adds delays to achieve a relative delay to some other signal path in the device, then VT variations on the additional delay track with the rest of the device, and thus do not need to be derated. Figure 6 shows an example with two DQ signals in the memory controller that terminate at the memory device. To improve the data valid window at the memory, it is best that the delay on each DQ signal be matched to one another. In the top part of the figure, there is a large mismatch in the static delay in the controller for the DQ0 and DQ1 pins, and thus calibration enables additional delay on DQ1 to match the delay in both paths. In this case, where the operating conditions change after calibration, as shown in the bottom part

of the figure, all delays in the controller scale similarly, and the paths are still matched. Any changes in operating condition should not affect the timing margin of the calibrated path.

Using the absolute maximum delay increase due to calibration in determining the margin loss is quite pessimistic. Instead of using the maximum delay increase due to calibration to remove the pessimism in the timing analysis, the maximum delay increase that does not track with the rest of the device must be used in the analysis. In the specific example of a memory controller, the delays that do not track with the rest of the device tend to be skewed in the package, board, or in some cases, PVT-compensated portions of the memory controller.

The above analysis of operating conditions assumes that VT changes affect all the signal paths on the die and that delays in the path scale similarly with changes in VT. These assumptions may not be true in practice, but are not considered because they cause second-order effects. To remove the dependence on this assumption, a further derating factor is included in the timing analysis. To determine this derating factor, a further correlation value must be defined for each pair of paths. This value represents the correlation in operating conditions between two paths, and is used to calculate the derating factor for calibrated paths that do not share the same VT conditions. The further derating value is calculated as follows: the correlation value is subtracted from 1 (perfect correlation) and multiplied by the percentage change in delay due to variation, and the maximum absolute change in delay due to calibration. For example, if the correlation value was 0.95 and the change in delay due to operating conditions was found to be 100 ps, a further derating factor of $(1 - 0.95) * 100 \text{ ps} = 5 \text{ ps}$ should be removed from every margin result.

Tracking

Changes in operating conditions after calibration can reduce much of the margin improvement obtained through calibration, as discussed previously. To reduce the margin loss due to changes in operating conditions, some ICs contain tracking algorithms in addition to calibration algorithms. These algorithms track changes in operating conditions and change the circuit topology during circuit operation, as opposed to calibration algorithms that solely change circuit topology before circuit operation.

To timing analyze paths that are subject to both calibration and tracking (instead of derating the timing analysis due to changes in operating conditions), a tracking uncertainty value must be removed from every margin. The tracking uncertainty is a function of the tracking algorithm; for tracking algorithms that continuously update the circuit topology, the tracking uncertainty value is close to 0, and for tracking algorithms that update the circuit topology periodically, the tracking uncertainty is larger. The tracking uncertainty is closely tied to the tracking algorithm and its implementation. It should be determined a priori so that timing analysis can remove it from the margins produced.

H. External Process Variation Calibration

While calibrated paths can be used in any part of an IC, when they are used to interface one IC to another IC such as a memory, the timing analysis of those paths is further upgraded to consider the effects where the other device's P variation is calibrated out.

Typically, when a memory IC is produced, the manufacturer also produces a report providing for specifications for the memory, including setup and hold times of registers on their input paths (t_{DH} and t_{DS} for DDR3) and worst-case skews on their output paths (t_{DQSQ} and t_{QH} for DDR3). These specifications come from uncertainties in the memory and include the P variations from one die to another along with the effects of VT variations that occur during operation. These specifications are then used in the timing analysis of the memory controller that is interfacing with the memory as input and output delays.

However, using the whole specification in the timing analysis of the controller's calibrated paths is pessimistic because the controller can calibrate to the specific P variation of the memory device with which it is interfacing. In other words, as the path changes, it calibrates out the process portion of the specification since it is only connected to the one die, rather than the many dies that the specification covers.

Consider the examples in Figure 7, where each case represents the uncertainty in arrival times of two signals. In Case (a), the percentage of the uncertainty attributed to PVT variation is unknown and thus the whole uncertainty must be considered. Case (b), on the other hand, shows a situation where P variation is known to compose 50% of the total variation. In this case, each pin's delay has a smaller uncertainty centered around a nominal point, but the total uncertainty over both pins is still the same as in Case (a). Finally, Case (c), shows a situation where the controller can calibrate to the memory and adds delay to Pin 2. In this case, the total uncertainty seen by the controller has been reduced by 50%.

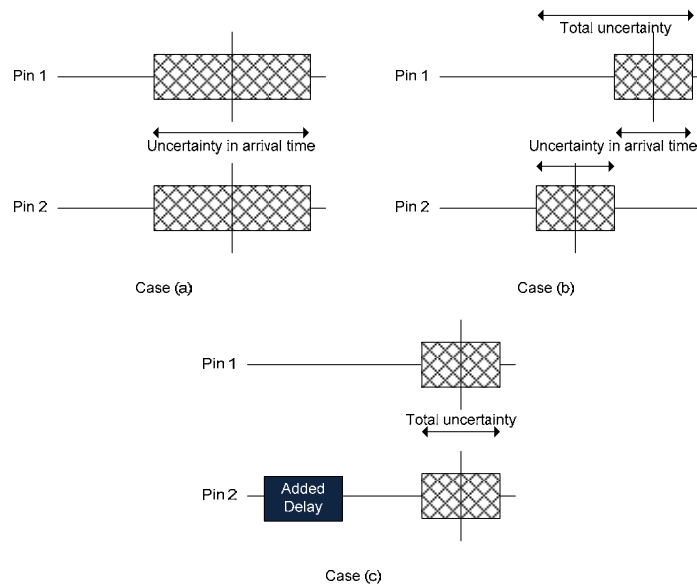


Figure 7: Effect of Calibration on Memory Specifications

Thus, if the percentage of the uncertainty that is due to P variation is known, then that portion of the specification is added to increase the timing margin. For example, consider the tDQSQ specification of a DDR3 interface representing the worst-case skew: 150 ps at 533 MHz. However, if it is known that 30% of the specification is due to P variations in the memory, then an additional 45 ps is added to the margins of the controller's read paths. Note that, as shown in Case (b) in Figure 7, solely knowing the percentage of the uncertainty due to P variation does not allow the timing margin to be increased. It can only be increased when that knowledge is combined with a calibrating controller.

I. Case Study

Altera has developed DDR3 IP that runs up to 533 MHz (1066 Mbps). However, to meet such high data rates, the IP performs calibration on various different paths including write leveling, resynchronization, and, most importantly, read capture and write. Timing analyzing the read capture and write paths accurately and with flexibility requires that these algorithms and features be implemented into Altera's Quartus II software and IP. These software tools include memory controllers, specifically the ALTMEMPHY DDR3 controller, and the TimeQuest timing analyzer tool.

The Read Capture and Write timing panels are of particular relevance to the timing analysis of calibrated paths, since the ALTMEMPHY DDR3 controller performs a per-bit deskew on the DQ and DQS pins in the design when the FPGA is turned on. The timing analysis of this type of path follows the algorithms and procedures described in this paper. First, a conventional STA is performed, and the results are displayed in the "Before Calibration" folder in the TimeQuest timing analyzer window, as seen in Figure 8.

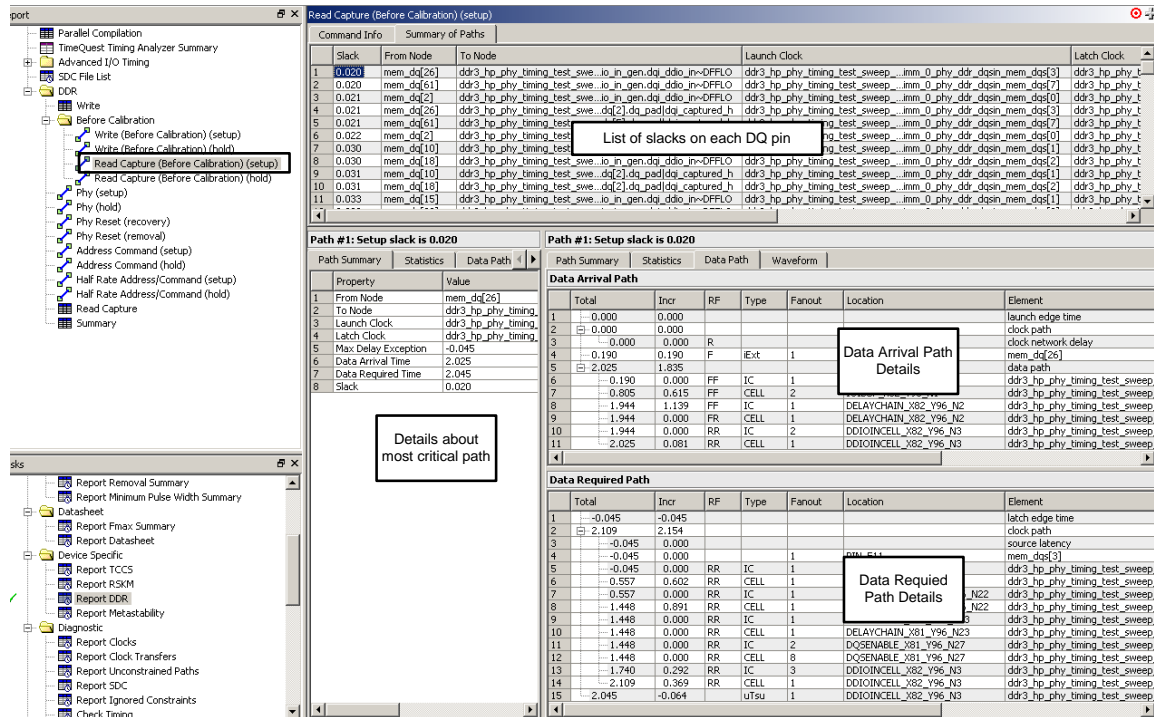


Figure 8: Before Calibration Read Capture Timing Analysis

With this base STA complete, TimeQuest continues to timing analyze the calibrated path. The Read Capture summary panel, shown in Figure 9, displays the results of this analysis. The “After Calibration Read Capture” setup and hold slacks are determined from the initial STA, noted in the “Before Calibration Read Capture” line, with various other effects included. The calibration emulation and the pessimism removal appear on the “Deskew Read” line, with the calibration error noted in the “Quantization error” line. The effect of changing operating conditions is included in the “Calibration uncertainty” line, and the effect of calibrating to the memory device is included in the “Memory Calibration” line. A similar algorithm and table exists for the write timing analysis.

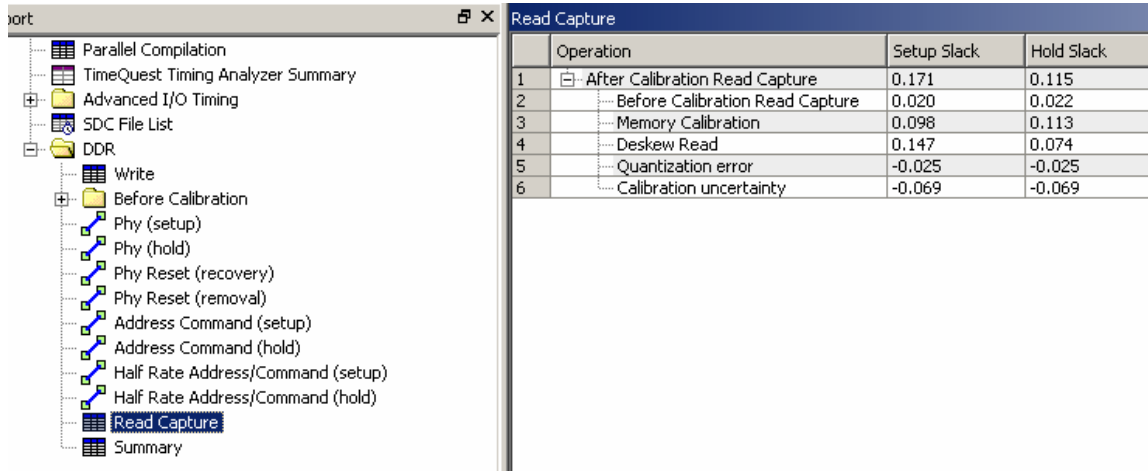


Figure 9: Calibrated Read Capture Path Timing Analysis

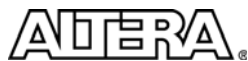
J. Conclusion

The push to higher interface bandwidths has decreased the available margin available to correctly read and write data between ICs. To meet timing at these high data rates, ICs, such as memory controllers, interfacing with other ICs are required to calibrate at power-up to reduce skew. However, the STA paradigm does not immediately apply to timing analyze these paths. This paper describes methods to obtain accurate timing analysis of paths that are calibrated at runtime, and optionally tracked throughout operation. The method combines conventional STA with new methods and procedures to obtain a timing analysis that considers the effects of calibration. The paper also describes how this new timing analysis method, included in Altera products, timing analyzes a DDR3 memory controller.

K. References

- [1] J. Bhasker, R. Chadha, *Static Timing Analysis for Nanometer Designs: A Practical Approach*, Springer Verlag, New York, 2009.
- [2] A. Dasdan, I. Hom, “Handling inverted temperature dependence in static timing analysis,” *ACM Transactions on Design Automation of Electronic Systems*, 11(2):306-324, 2006.
- [3] JEDEC DDR3 SDRAM Standard, JESD79-3D, September 2009.
- [4] CIO RLDRAM II MT49H64M9, Micron, June 2009.

- [5] Altera, *Utilizing Leveling Techniques in DDR3 SDRAM Memory Interfaces*, White Paper, November 2007, <http://www.altera.com/literature/wp/wp-01034-Utilizing-Leveling-Techniques-in-DDR3-SDRAM.pdf>.
- [6] K.R. Heloue, F.N. Najm, "Early Analysis and Budgeting of Margins and Corners Using Two-Sided Analytical Yield Models," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(10):1826–1839, 2008.



101 Innovation Drive
San Jose, CA 95134
www.altera.com

Copyright © 2010 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.