

FASTER CLOCKS MEAN

EVER SMALLER DATA-VALID WINDOWS.

Overcoming DDR-2-interface challenges

DDR (DOUBLE-DATA-RATE) memory devices do an admirable job of unclogging the memory bottleneck between external memory and ASICs, but the nature of DDR is stretching the skills of designers creating controllers to transfer data between ASIC and SDRAM through nanosecondwide DVWs (data-valid windows). For example, the data-bit width for 400-Mbps DDR 2-400 is 2.5 nsec. The design challenge intensifies for higher-data-rate products based on the second-generation DDR-2 specification. Design glitches can reduce data rates and cause fatal system errors that crash computer or telecommunications systems.

Minuscule DVWs are challenging enough. To complicate matters, design contingencies, such as jitter and other noises, setup-and-hold time, clock skew, and package skew, even further reduce the time available for a valid transfer. In addition, process, temperature, and voltage variations affect signal-relative timings. The good news is that, once a DDR or DDR-2 controller and memory buffers are in place, you can turn more of your attention to circuits that implement product features and provide a competitive advantage during succeeding designs.

DDR BASICS

Understanding the underlying DDR technology is essential to developing innovative design techniques that can ensure that the ASIC-SDRAM interface is both robust and failsafe. In the late 1990s, SDRAMs began delivering twice the memory bandwidth previously available by transferring data on both the rising and the falling edges of the clock signal. This approach doubled the data rate; hence the term “DDR.” To achieve double the data rate and maintain a low-cost implementation, DDR- SDRAM designers use a 2-bit, prefetch-datapath architecture, which allows the internal column-access frequency to be only one-half the external data-transfer frequency. DDR-2 designers use a 4-bit, prefetch-datapath architecture to further speed transfers.

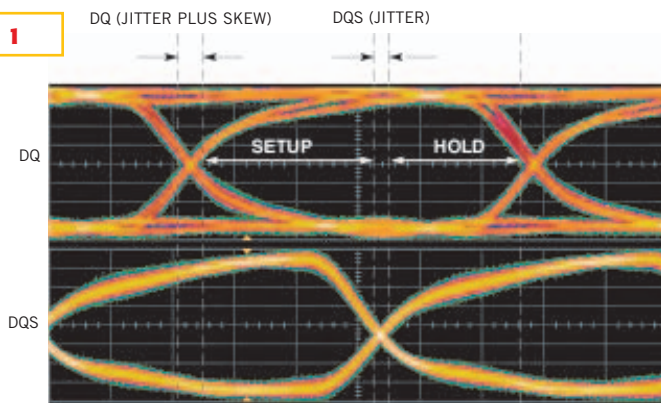
The first step to designing any high-speed interface system is to define a timing budget that meets all the device and the system-board-timing requirements. DDR devices transfer data on both rising and

falling clock edges, so the data-bit width for DDR2-533 is just 1.875 nsec and not the 3.75 nsec that you would ordinarily expect for its 266-MHz system clock. However, that 1.875-nsec time window is a theoretical maximum because several factors, such as noise and skew, combine to considerably narrow the window (**Table 1**). You must reduce noise, skew, and the other detrimental factors to maximize the timing margins, especially for DDR-2 as you ramp up the data-transfer rates from 400 Mbps to 533 Mbps, 667 Mbps, and beyond.

You can superimpose these timing contingencies onto a clocking diagram to represent and visualize their cumulative effects (**Figure 1**). The setup-and-hold times comprise the largest amount of time—about 350 psec at either edge. If you take the other two factors together, they subtract 900 psec more from the data window. These numbers represent a typical example; each design differs. Data transfer is successful only when the data strobe falls within the data eye with sufficient setup-and-hold margins.

Another challenging aspect of designing DDR controllers comes from memory vendors implementing the standard by sending aligned rather than phase-shifted DQ (data) and DQS (strobe) signals. Because the DQS and DQ arrive simultaneously, you must phase-shift the DQS signals by approximately 90° for latching the data. This step leaves you with the task of figuring out how to latch the data by for-

Figure 1



Data transfers must occur within the timing limits of the data eye.

warding the strobe by 90° without a continuous reference clock from the memory. You need a custom DLL (delay-lock loop) because the DQS is a strobe and not a continuous running clock. Therefore, you cannot use a traditional PLL (phase-lock loop) for the 90° phase shift.

TIMING IS EVERYTHING

You can fashion a DDR implementation in the memory controller and the I/O buffers that interface with the SDRAM device. The controller design can partially address the primary design goals: minimizing on-chip clock skew, maintaining a balanced duty cycle, minimizing noise, and implementing the phase shift. Likewise, optimizing the memory-buffer design for noise, duty cycle, and clock skew is essential. Within the DDR controller, a PHY hard macrocell implementation is the only real choice because you need a precise physical layout of the circuits.

Crosstalk and skews on the system board that carries the ASIC, DIMM, and other chips also narrow the data eye. You must address this aspect of the design by carefully matching the trace length of critical paths on the board. The main effect system-board design has on the chip design, however, is to force ASIC designers to minimize signal skews.

Minimizing on-chip signal skew requires a custom layout but is a relatively straightforward process of carefully laying out datapaths to achieve identical delays for each trace on the data bus. The impact of carefully eliminating on-chip skew can save as much as 200 psec in the data eye. Achieving a balanced duty cycle (the rise and fall intervals of the data signal) can net you 200 psec more in the timing margin. If the rise and fall cycles are not of identical intervals, you must assume the worst-case scenario, which reduces the size of the data eye (Figure 2).

Noise generated inside the core is considerably more difficult to handle. Power droop and crosstalk are major causes of noise. High-end place-and-route tools from major EDA vendors can anticipate and avoid these problems, but they are expensive. You should also add decoupling capacitors to limit core-switching noise. As an alternative, ASIC vendors offer prestructured ASIC platforms

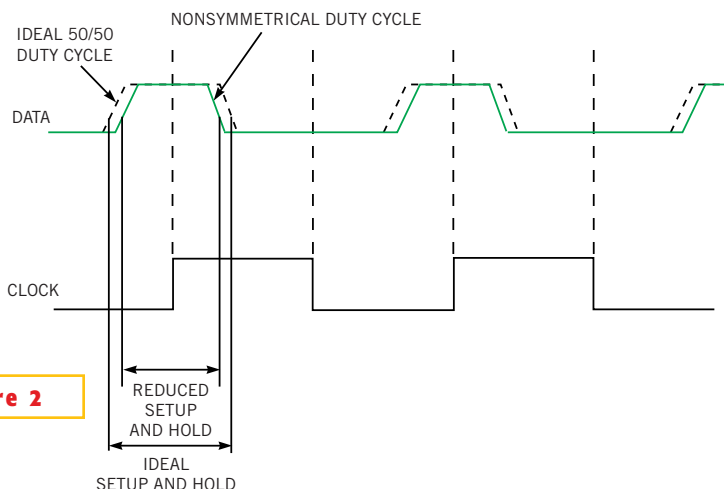


Figure 2

Duty cycles that are not evenly split between rise and fall reduce the timing window.

or DDR-interface hard macros as plugins for standard-cell ASICs.

PHASE-SHIFTING TECHNIQUES

Implementing a 90° phase shift of the DQ with DQS calls on your experience and creativity. Writing data to memory is easier than reading it from memory because you have more control over writing. The basic technique uses a PLL to generate a 2× clock for the DQS signal (Figure 3). You must design a low-jitter PLL to maximize the data eye. For example, in a DDR2-533 implementation, PLL jitter should be less than 50 psec.

Although a conventional PLL is an ideal approach to writing data to memory, you cannot use this method during the read cycle. The reason for this seeming anomaly is that, although you have virtually complete control over what happens on the ASIC, such as in a write, that level of control is missing when reading from the SDRAM device. This fact is particularly important for data reads, because DQS is a strobed signal and not a continuously running clock. Specifically,

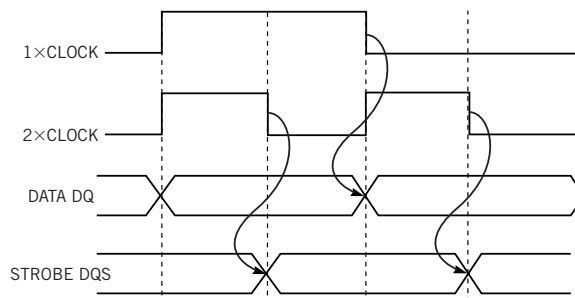


Figure 3

Generating a 2× clock creates a 90° phase shift between the DQ and DQS signals.

due to the long lock time of a PLL, you cannot use the DQS signal as the reference clock for the PLL in the read cycle. Because DQ and DQS arrive simultaneously, there is no chance of locking the analog PLL in time to read the data.

Depending on your DDR-performance goals, you have at least three options for read operations. For DDR200, which runs on a 100-MHz clock and has a 5-nsec data window, enough timing margin may be available to use simple delay cells in a memory controller. Whether there is margin available depends greatly on predictably good performance over the process variations of the ASICs, once they have entered production and their operating conditions in the application. PVT (process, voltage, and temperature) variations affect the actual delay of the delay cells.

For higher performance systems, such as DDR333, DDR400, and DDR-2, there is insufficient timing margin in the DDR controller for simple delay cells to work. You have at least two options. You can use an analog approach that implements PVT-compensated DLLs that will probably require special attention during the design to optimize IR drop, noise, and other parameters. Another approach uses a DDLL (digital-delay-lock loop) to monitor and adjust the delay cell within 20 psec over the PVT range (Figure 4).

You should use the more sophisticated technique of eye-training/painting for interfaces that operate faster than 400 Mbps. Employ an

eye-painting algorithm to detect the exact data edges during power-up and carefully place the strobe at the center of the data eye. You should place the strobe as a delta to the DLL delay; otherwise, overwriting the DLL delay loses the voltage-and-temperature-compensation effect.

MEMORY-BUFFER TECHNIQUES

In addition to meeting JEDEC (Joint Electronic Device Engineering Council) memory-buffer requirements, creating and maintaining a balanced duty cycle should be key objectives when you are designing the SSTL2 (stub-series-terminated-logic-for-2.5V) memory-interface buffers for DDR and SSTL18 (stub-series-terminated-logic-for-1.8V) buffers for DDR2. The voltage difference between the core-logic operating voltage, which may be 1.8 or 1.2V, depending on the process technology, and the I/O voltage of 2.5 or 1.8V is a major complicating factor for achieving a balanced duty cycle. The DPD (delta-propagation delay) performance metric measures the differences in the rise and fall propagation delays of the receiver or driver to measure acceptable performance. In DDR2-533, the DPD cannot be more than 80 psec.

In the write cycle, propagation delays through the flip-flop and driver circuits are inevitable. To maintain a relatively balanced duty cycle and minimize skews between bits, PVT variations cannot excessively affect the delay. Design the flip-flop inside the driver and eliminate the interconnects. This approach helps to minimize skew and delay between the data bits. In addition, you should design the buffer with a PVT-compensation technique to provide stable performance.

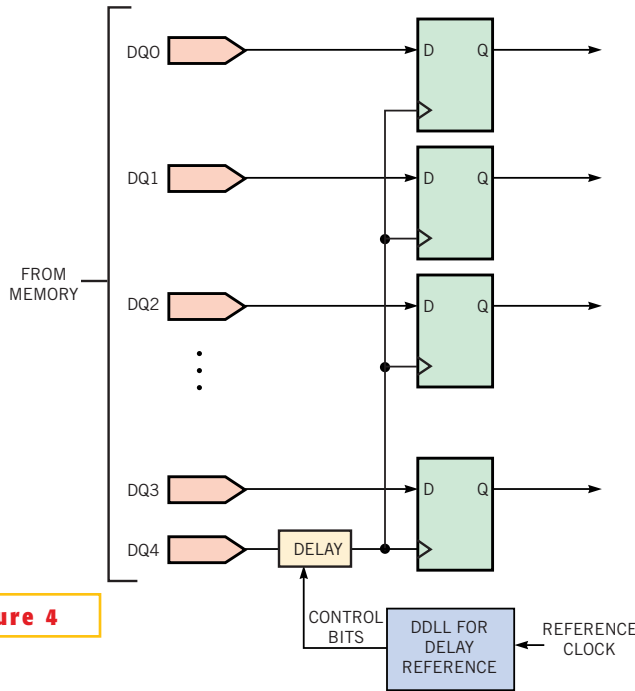


Figure 4

A digital delay lock loop creates phase shift in the read cycle.

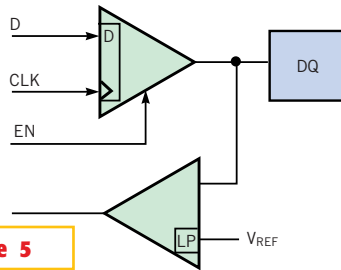


Figure 5

Placing a flip-flop inside the driver and using a lowpass filter in the receiver help minimize signal skew and reduce noise from the pc board.

In a read cycle, you need to achieve a balanced duty cycle, but you face the additional challenge of handling system noise coming from the system board. Putting a lowpass filter on each V_{REF} pin is a valuable technique. You can combine this technique with using as few V_{REF} pins as possible to limit noise sources. Using one V_{REF} pin for approximately every 80 to 100 DQ signals is a good goal in this respect. You must isolate the routing of the V_{REF} signal on the die to keep it from

coupling noise with other high-speed signals (Figure 5). DDR-2 calls for ODT (on-die termination) to more effectively reduce signal jitters. You should also design such an ODT circuit with PVT-compensation techniques to maintain a constant termination resistance. With so much of the timing budget riding on package-trace lengths, it is not surprising that finding the best place for the hard macrocell and I/O on the periphery of the ASIC die is an important and difficult design challenge. Minimizing simultaneous-switching-output noise and crosstalk can be time-consuming. When designing the memory controller, you should also consider gating DQS to avoid false triggers and faking a data

valid signal to time data arrival. Off-die conditions affect signal skew as much as on-die conditions. You must consider signal integrity to minimize system noise; the electrical properties of the signal path; package parasitics, including bond wires and package traces; and board traces. Important considerations during board design include cost-versus-performance trade-offs, such as whether to use single- or dual-stripline traces and other stackup issues, trace geometries, and via size and placement. These issues are beyond the scope of this article, but the overall signal-travel paths must be close to equal for every bus trace, and signal integrity must be as good as the design's budget allows. You must perform a full-system signal-integrity analysis with accurate buffer, package, system, and memory models to predict the results. □

AUTHOR'S BIOGRAPHY

William Lau is an engineering director at LSI Logic Corp (Milpitas, CA), where he is responsible for the development and technical support of memory and high-speed-interface physical-layer and I/O buffers. He received a bachelor's degree in electrical engineering from San Jose State University, and he received a master's in electrical engineering and a master's in business administration from Santa Clara University.

TABLE 1—DESIGN CONTINGENCIES FOR THE DATA-VALID WINDOW

Description	Value (psec)
Data-bit window (533 Mbps)	1875
Minus data jitter and noise and strobe jitter and noise	700
Minus SDRAM setup-and-hold times	700
Minus ASIC and system-board signal skews	200
Total margin left	275