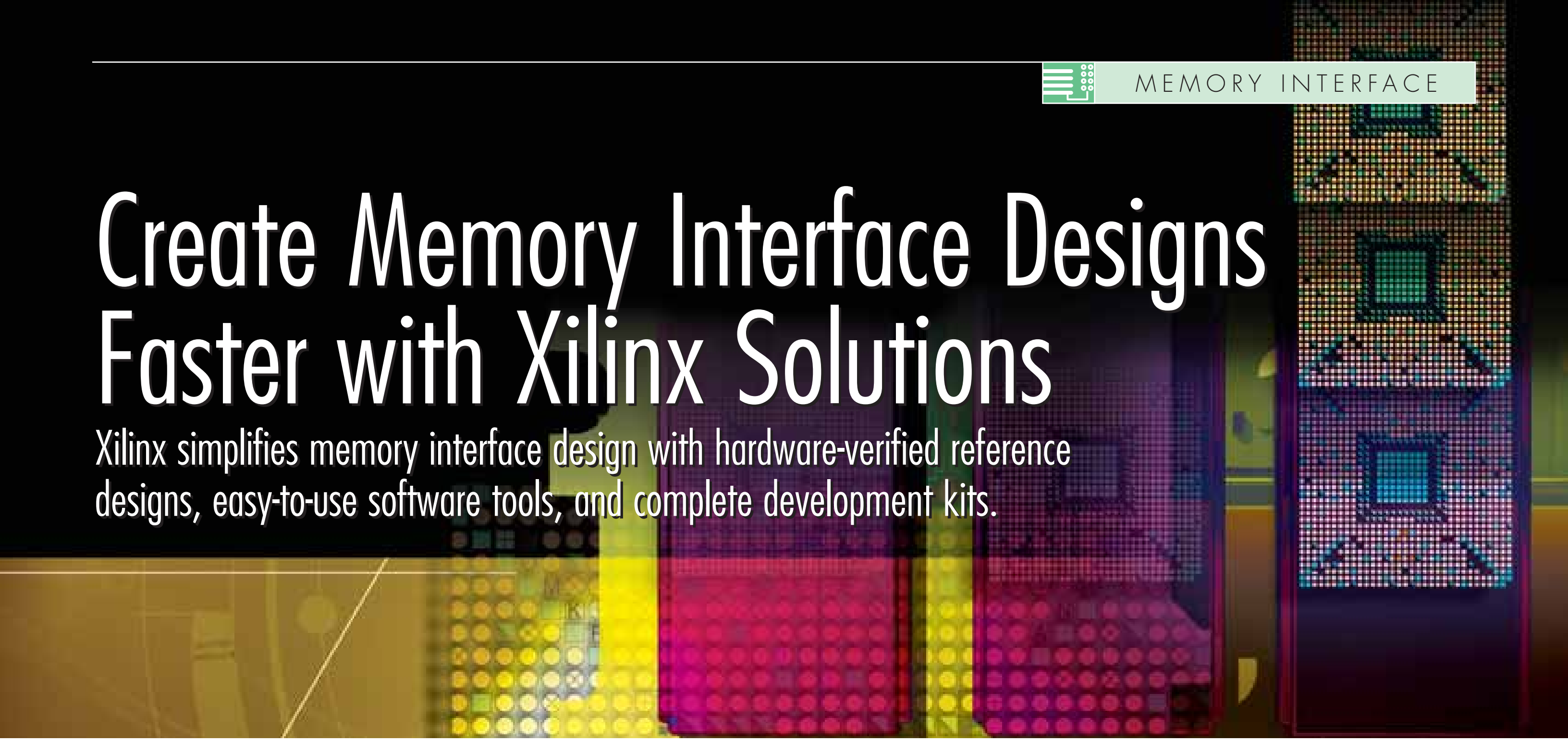

# Create Memory Interface Designs Faster with Xilinx Solutions

### Xilinx simplifies memory interface design with hardware-verified reference designs, easy-to-use software tools, and complete development kits.

by Adrian Cosoroaba
Marketing Manager
Xilinx, Inc.
*adrian.cosoroaba@xilinx.com*

Xilinx® FPGAs provide I/O blocks and logic resources that make interface design easier. Nonetheless, designers must still configure, verify, implement, and properly connect these I/O blocks – along with extra logic – to the rest of the system in the source RTL code, carefully simulated and verified in hardware.

In this article, I'll review the performance requirements, design challenges, and Xilinx solutions for memory interface design, from low-cost implementations with Spartan™-3 Generation FPGAs to the highest bandwidth interfaces using Virtex™-5 FPGAs.

## Performance Requirements and Xilinx Solutions

In the late 1990s, memory interfaces evolved from single-data-rate SDRAMs to double-data-rate (DDR) SDRAMs, with today's DDR2 SDRAMs running at 667 Mbps per pin or higher.

Applications can generally be classified in two categories:

- Low-cost applications, where the cost of the device is most important
- High-performance applications, where getting the highest bandwidth is paramount

DDR SDRAMs and low-end DDR2 SDRAMs running below 400 Mbps per pin are adequate to meet the memory bandwidth requirements of most low-cost systems. For these applications, Xilinx offers Spartan-3 Generation FPGAs: Spartan-3, 3E, 3A, and 3AN devices.

For applications that push the limits of memory interface bandwidths, like 667-Mbps-per-pin DDR2 SDRAMs, Xilinx offers Virtex-5 FPGAs.

Bandwidth is a factor related to both the data rate per pin and the width of the data bus. Both Spartan-3 Generation and Virtex-5 FPGAs offer distinct options, spanning a range from smaller low-cost systems with data bus widths of less than 72 bits to those as wide as 576 bits for the larger Virtex-5 packages (Figure 1).

Wider buses at these speeds make chip-to-chip interfaces all the more challenging, requiring larger packages and better power-to-signal and ground-to-signal ratios. Virtex-5 FPGAs have been built with advanced SparseChevron packaging that provides superior signal-to-power and ground-pin ratios. Every I/O pin is surrounded by sufficient power and ground pins and planes to ensure proper shielding for minimum crosstalk noise caused by simultaneously switching outputs (SSO).

## Memory Interfaces with Spartan-3 FPGAs

For low-cost applications where 400-Mbps bit rates per pin are sufficient, Spartan-3 Generation FPGAs coupled with Xilinx software tools provide an easy-to-implement, economical solution.

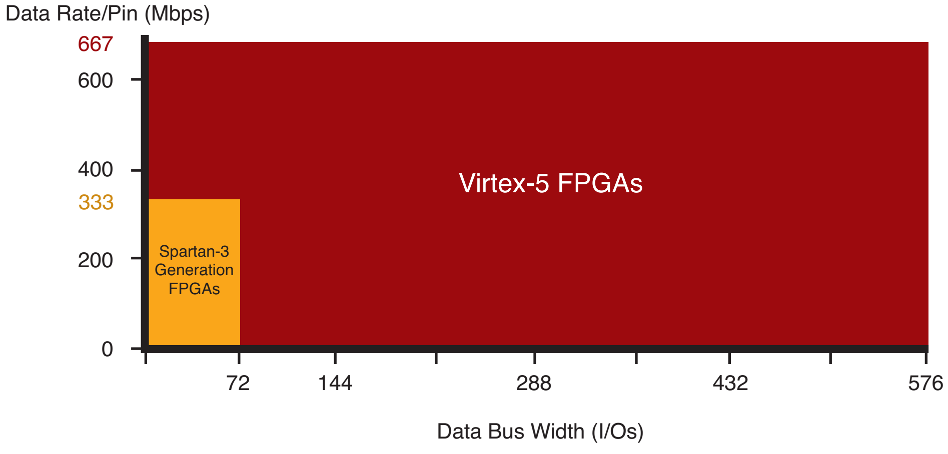

*Figure 1 – Xilinx FPGAs and memory interface bandwidth*

Three fundamental building blocks comprise a memory interface and controller for an FPGA-based design: the read and write data interface, the memory controller state machine, and the user interface that bridges the memory interface design to the rest of the FPGA design. Implemented in the fabric, these blocks are clocked by the output of the digital control manager (DCM) that, in the Spartan-3 Generation implementation, also drives the look-up table (LUT) delay calibration monitor (a block of logic that ensures proper timing for read data capture).

In the Spartan-3 Generation implementation, read data capture is implemented using the LUTs in the configurable logic blocks (CLBs). During a read transaction, the DDR2 SDRAM device sends the read data strobe (DQS) and associated data to the FPGA edge-aligned with the read data (DQ). Capturing the DQ is a challenging task in source-synchronous interfaces because data changes at every edge of the non free-running DQS strobe. The read data capture implementation uses a tap-delay mechanism based on LUTs.

Write data commands and timings are generated and controlled through the write data interface. The write data interface uses input/output block (IOB) flip-flops and the DCM's 90-, 180-, and 270-degree outputs to transmit the DQS strobe properly aligned to the command and data bits.

The implementation of a DDR2 SDRAM memory interface has been fully verified in hardware. The design was implemented in the Spartan-3A starter kit board using a 16-bit-wide DDR2 SDRAM memory device and the XC3S700A-FG484 device. The reference design uses only a small portion of the Spartan-3A FPGA's available resources: 13% of IOBs, 9% of logic slices, 16% of global buffer (BUFG) multiplexers (MUXs), and one of the eight DCMs.

You can easily customize Spartan-3 Generation memory interface designs to fit your application using the Memory Interface Generator (MIG) software tool.

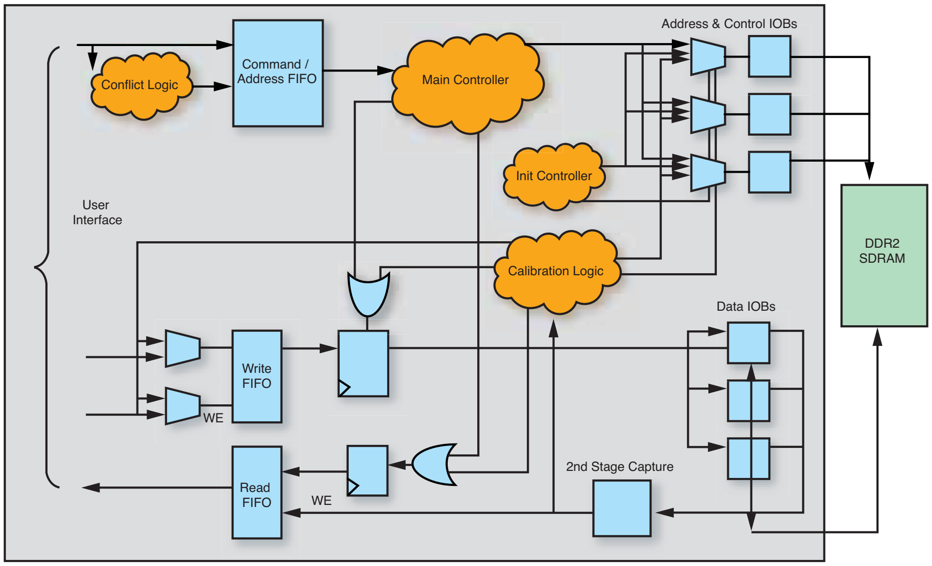

*Figure 2 – Virtex-5 FPGA memory interface architecture*

## Memory Interfaces with Virtex-5 FPGAs

With higher data rates, interface timing requirements become more challenging. The trend toward higher data rates presents a serious problem to designers in that the data valid window – that period within the data period during which DQ can be reliably obtained – is shrinking faster than the data period itself. This is because the various uncertainties associated with system and device performance parameters, which impinge on the size of the data valid window, do not scale down at the same rate as the data period.

This trend is readily apparent when comparing the data valid windows of DDR SDRAMs running at 400 Mbps and DDR2 memory technology, which runs at 667 Mbps. The DDR device with a 2.5-ns data period has a data valid window of 0.7 ns, while the DDR2 device with a 1.5-ns period has a mere 0.14 ns.

Virtex-5 FPGAs address this challenge with dedicated delay and clocking resources in the I/O blocks – called ChipSync™ technology. The ChipSync block built into every I/O contains a string of delay elements (also known as tap delays), called IODELAY, with a resolution of 75 ps.

The architecture of the implementation is based on several building blocks. The user interface that bridges the memory controller and physical layer interface to the rest of the FPGA design uses a FIFO architecture. The FIFOs hold the command, address, write data, and read data. The main controller block controls the read, write, and refresh operations. Two other logic blocks execute the clock-to-data centering for read operations: the initialization controller and the calibration logic (Figure 2).

The physical layer interface for address, control, and data is implemented in the IOBs. The DQ capture uses the memory strobe to capture corresponding DQ and register it with a delayed version of the strobe. This data is then synchronized to the system clock domain in a second stage of flip-flops. The input serializer/deserializer feature in the I/O block is used for read capture – the first pair of flip-flops transfer the data from the delayed strobe to the system clock domain. The technique involves the use of 75-ps tap delay (IODELAY) elements that are varied during a calibration routine implemented by the calibration logic. This routine is performed during system initialization to set the optimal phase between strobe, data, and the system clock to maximize timing margins.

There are other aspects of the design, such as the overall controller state machine logic generation and user interface. To make the complete design easier for the FPGA designer, Xilinx developed the Memory Interface Generator.

### Design and Integration with MIG

Integrating all of the building blocks, including the memory controller state machine, is essential for the completeness of your designs. Controller state machines vary with the memory architecture and system parameters. State machine code can also be complicated and a function of many variables, such as architecture, data bus width, depth, access algorithms, and data-to-strobe ratios.

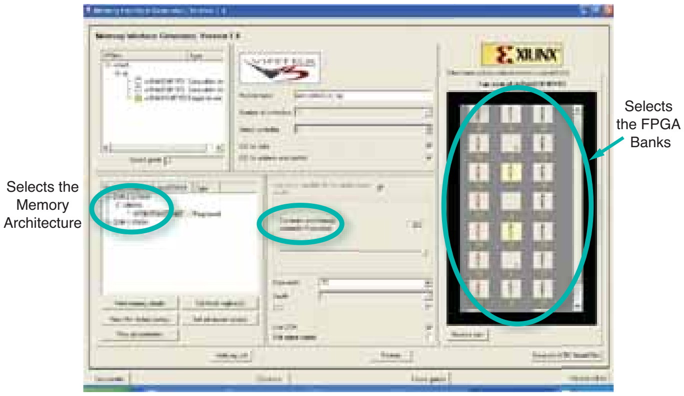

*Figure 3 – MIG GUI*

| Xilinx FPGA | Spartan-3E | Spartan-3A | Spartan-3AN | Virtex-5 |
|---|---|---|---|---|
| Development Board/Kit | Starter Kit | Development Kit | Starter Kit | ML-561 |
| Memory Interfaces Supported | DDR | DDR2 | DDR2 | DDR<br>DDR2<br>QDR-II<br>RLDRAM II |

*Table 1 – Development boards and kits for memory interfaces*

The complete design can be generated with the MIG, a software tool freely available from Xilinx as part of the ISE™ software CORE Generator™ suite of reference designs and IP. The MIG design flow is very similar to the traditional FPGA design flow. The benefit of the MIG for designers is that there is no need to generate RTL code from scratch for the physical layer interface or the memory controller.

You can use the MIG's GUI to set system and memory parameters (Figure 3). For example, after selecting the FPGA device, package, and speed grade, you can select the memory architecture and pick the actual memory device or dual in-line memory module (DIMM). The same GUI provides a selection of bus widths and clock frequencies. Other options provide control of the clocking method, CAS latency, burst length, and pin assignments.

The MIG tool can generate in a matter of minutes the RTL and UCF files, which are the HDL code and constraints files, respectively. These files are generated using a library of hardware-verified reference designs, with modifications based on user inputs.

You have complete flexibility to further modify the RTL code. Unlike other solutions that offer “black-box” implementations, the code is not encrypted, providing complete flexibility to change and further customize a design. The output files are categorized in modules that apply to different building blocks of the design: user interface, physical layer, or controller state machine. You may wish to customize the state machine that controls the bank access algorithm, for example. After the optional code change, you can perform additional simulations to verify the functionality of the overall design.

The MIG also generates a synthesizable test bench with memory checker capability. The test bench is a design example used in the functional simulation and the hardware verification of the Xilinx base design.

The final stage of the design is to import the MIG files in the ISE project, merge them with rest of your FPGA design files, conduct synthesis and place and route, run additional timing simulations if needed, and finally perform verification in hardware. MIG software also generates a batch file with the appropriate synthesis, map, and place and route options to help you optimally generate the final bit file.

### Development Boards and Kits

Hardware verification of reference designs is an important final step to ensure a robust and reliable solution. Xilinx has verified the memory interface designs for both Spartan-3 Generation and Virtex-5 FPGAs. Table 1 shows the memory interfaces supported for each of the development boards.

The development boards range from low-cost Spartan-3 Generation FPGA implementations to the high-performance solutions offered by the Virtex-5 FPGA family.

### Conclusion

You can complete your memory interface designs faster with a wide spectrum of Xilinx FPGA devices, dedicated software tools like the MIG, and development boards that fit your application needs, from low cost to high bandwidth.

For more information and details on these memory interface solutions, visit *www.xilinx.com/memory*.

**TAKE THE NEXT STEP**

- Download the Memory Interface Generator to generate your reference designs for Virtex-5, Virtex-4, and Spartan-3 devices, including HDL code and pin placements
- See a demo on Memory Interface Solutions with Xilinx FPGAs
- View an on-demand webcast on Low-Cost to High-Performance Memory Interface Design Made Easy with Xilinx FPGAs
- Contact Xilinx