High-Performance Memory Interfaces Made Easy
Transcript
March15, 2005

Slide 1
Hi, this is the fourth in our applications-oriented series of web-seminars that describe the 90-nm Virtex-4 family and its technology.
Previous web-seminars covered Performance, Power consumption, and PC-board Signal Integrity. Today's subject is interfacing to external memory and how Xilinx makes it Easy to do so.
S2
Many, if not most, FPGAs are somehow connected to external RAM.
Designing the FPGA interface to the memory may not be the most glamorous job, but it is getting increasingly more difficult and demanding. Access to external memory often is the main performance limiter. This means that everything possible must be done to optimize and streamline that interface. And usually this involves controlling many, up to a hundred or more, simultaneously switching address and data lines and also meeting more demanding design timing margins. We'll show you today how Virtex-4 and Xilinx has made your job easier with complete memory interface solutions.
For these and many other details, I now hand the microphone to my colleague, Adrian Cosoroaba.
S3

Thank you, Peter.
I would like to quickly overview today's agenda.
We'll start with a brief introduction of memory trends and follow with a detailed look at memory interface design challenges and our solutions.
We'll look in even greater detail at an actual implementation, the DDR2 SDRAM interface, and finally summarize our seminar and provide you with additional information on how to get started with your own design.
We'll wrap up our seminar with a 10 minute Q&A session following this presentation.

S4
Let's look first at the trend in memory interfaces, and will look at the mainstream memory market, the DRAM market. One of the performance related metrics that most of you designing memory interfaces look at is the data rate. Memory interfaces have evolved from single data rate SDRAMs, in the second half of the 90's, to double data rate; currently the fastest is DDR2 running at 533 Mbps per pin. Micron, one of our memory partners, predicts that these rates will increase in the future; we have seen on the average the rates double every four years and it is predicted to continue and may reach 1.6 Gbps/pin by 2010.

S5
What does it all mean to you as memory interface designer ?
Increasing data rates mean shrinking timing margins because data windows, data valid windows in particular, are getting smaller.
If you look at the two examples shown here, the 400 Mbps DDR and the current memory technology,533 Mbos DDR2, the data valid window is determined by the actual data period minus the various device and system uncertainties. What is challenging these days is that data valid windows are shrinking faster than the actual data period. Uncertainties associated with system and device parameters are not scaling down at the same rate. A 2.5 ns period has a data valid window of 2/3 ns while a 1.9 ns one is only 1/3 ns.
This is very small …. Interface timings are indeed becoming more demanding

S6

Let's look now at Design Challenges and how Xilinx and Virtex-4 FPGAs address them

S7
We have surveyed more than 300 customers and received a great deal of feedback regarding memory interface design challenges.

As many of you know, the Number one design challenge is the timing critical physical layer, and in particular, the read data capture This is due to higher data rates and especially the difficulty of capturing data with a very narrow data-valid window. Meeting the tight timings with enough margin to ensure a reliable design is also part of this great challenge.

A second challenge is achieving the high bandwidth that most systems require. We need to not only implement high data rates but also relatively wide data buses. The wider these buses become the more challenging the chip to chip interface is. Resolving signal integrity issues, I/O placement and board routing have also become part of this design challenge.

Finally, the complex memory controller design and its integration in the overall FPGA design sums up or TOP 3 challenges.

S8
We, at XILINX have made your design easy by doing most of this work for you.

We started by incorporating key features in the silicon, like Chipsync, a clever way of ensuring reliable data capture, that is part of every I/O
We'll look at Chipsync more closely in the upcoming sections of the seminar.
We have also implemented and verified in hardware the most commonly used high perf. memory interfaces like DDR2, DDR, QDR II, RLDRAM II and make them available to you with our ML461 development system.
In addition, we provide you a free tool, the Memory Interface Generator that generates your custom memory controller and physical layer interface in a matter of minutes using hardware verified designs.
S9
Let's spend some time now digging into some of these challenges, to better understand how we resolve them for you.
We'll start with the toughest one, the Timing-critical Physical Layer.
There are two parts to this challenge. The first one is providing a data valid window to the FPGA device inputs and many of you have seen eye charts that depict smaller data valid windows due to various device and system uncertainties. The second part of this challenge is the centering of the clock or strobe used to capture or latch the data inside the FPGA. Uncertainties are making data valid windows as small as 1/3 of a ns for today's 533 Mbps DDR2 SDRAMs. The clocks or strobes used to capture this data are also subject to phase shift uncertainties that can make it increasingly difficult to meet the set up and hold times.
S10
One method used for capturing read data is the fixed phase shift delay. This method has been used by our competitor's devices. It is a method that was adequate for lower data rates.
However, the major drawback is the fixed phase shift. It is determined at design time and usually a single value for multiple clocks or DQS signals that most systems employ. There are delays between DQS signals and data due to many factors like : process variations that change the device timings, voltage and temperature changes in the system environment, and board design variations.

The net result is an erosion in the design margins because the centering of the clock or strobe to the data-valid window is no longer valid for all DQS signals.
The result is that set up and hold times may not be met for all DQ signals and ultimately cause bit errors.
S11
On the other hand , Virtex-4 FPGAs use precise Data-to Clock centering.
Chipsync technology is built into every I/O and part of it is a block called IDELAY. It is a 64 tap delay chain with 78 ps resolution that is controlled by a state machine, which is part of the reference design.
This mechanism of centering the FPGA clock to the middle of the data valid window is done for all data inputs. It maximizes the design margins because it is also accomplished at "run time" ensuring that process, voltage and temperature variations are accounted for.
This method is a unique Xilinx method that is ***Not available in any other FPGA, ASIC or ASSP***
S12
Let's look now at the High Bandwidth System Requirements

Maximizing bandwidth, means not only Higher Data Rates but also Wider Buses to meet your system requirements.

It also means Resolving signal integrity issues

And Meeting I/O placement and board routing requirements
S13
In order to achieve a high bandwidth , both data rates and bus widths need to be maximized.
Virtex-4 FPGAs, unlike competing devices, do not have any restrictions and any I/O can be used for Data, Strobe/Clock or Address and Controls. Chipsync is built into every I/O and any data to strobe ratios are possible, providing ultimate flexibility.
Data rates for the single ended standards typical of memory interfaces are possible to 600 Mbps
Data Bandwidths of up to 259 Gbps can be accomplished with our larger packages as seen in this table.
These high bandwidths are also possible with our superior SSO performance.
Therefore, Virtex-4 FPGAs offer 3X higher bandwidth than competing solutions.
S14
Achieving superior SSO performance is a key ingredient in the quest for higher bandwidth.
The key feature is the innovative SparseChevron package wich is enabled by our Column based ASMLB architecture.
This makes uniformly spread power and ground pins possible.
The pinout diagram shows how Virtex-4 compares with a competing Stratix-II device that has many regions devoid of returns
This is a major reason Virtex-4 has better SSO performance.
S15
In a previous seminar, on March 1-st,
Dr. Howard Johnson showed a number of tests that measure SSO performance.
I am showing here one of them.
The Accumulating Test. The bottom line is that Virtex-4 shows ***7 x less crosstalk than competing solutions***

For more details and other tests please check the archived March 1-st Xilinx Tech on Line seminar on signal integrity
S16
Meeting I/O Placement Requirements is also important .
Routings on the PC boards have become a real challenge in today's designs
and this comparison shows that Virtex-4 FPGA devices provide you with **Unrestricted I/O placements**
Unlike competing Statix-II devices that have restricted I/O placements to the top and bottom of the package.
The benefit to you is more flexibility in the board routing and 3 x more data I/Os than competing solutions.

S17
We have talked about the silicon and package features of our Virtex-4 devices and the benefits they offer to you in designing memory interfaces, but as they say,
the proof is in the pudding !
Here we show our ML461 development system that Xilinx has used to verify the reference designs
for high bandwidth memory interfaces like DDR2, DDR SDRAM, QDR II, RLDRAM II
and you can see here the data rates that these designs run at as well as the data bus widths.
This development system is also NOW from xilinx.com.

S18
Let's Look at third challenge on our list
Controller state machines vary with the memory architecture and system parameters
State machine code to maximize performance canalso be complicated, and a function of many variables
For example:
Architecture (DDR, DDR2,QDR II, RLDRAM, etc.)
Or Number of Banks (be that external or internal to the memory device)
Data Bus Width

Or Device width
And sometimes special Bank and Page access algorithms
Finally, parameters like Data to Strobe ratios
The controller implementation CAN be Costly and time consuming
!!!

S19
Xilinx has the answer with our FREE tool, the Memory Interface Generator
It has a user friendly GUI and can generate complete memory interface and controller designs
It generates for you:

- HDL code
- Constraints file
- And also A synthesizable test bench to verify your interface and controller functionality

- It is available now at no COST from xilinx.com/memory

S20
The MIG generates your design in minutes and here is the Design Flow.
After you download the MIG you use the GUI to set your system and memory parameters.
The tool generates the rtl and ucf files, which are the HDL code and constraints files.
These are generated from a library of hardware-verified designs.

You also have an additional option. Unlike competing solutions, the code is not encripted and you have THE complete flexibility to change and customize the design.
After the optional code change you can perform additional simulations.

The next step is to import the files in the ISE project followed by synthesis using XILINX Synthesis TOOL ,XST, place and route, do timing simulation, and finally verify it in hardware.

S21
Let's look at the Memory Interface Generator options:

- You can select

The FPGA device, the package, speed grade that you need and
The Banks and signals per bank : address, data , or controls
You can also select the Memory Interface clock freq.
The other parameters of the memory system that you can select are :
Memory architecture and type of device or DIMM modules.
You can also pick the Data (bus) width and the depth or number of data loads
S22
The Memory Interface Generator will output for you from a library of hardware-verified designs
The Constraints file (ucf)
And Modular HDL code, the rtl files shown here like the
The Physical layer (including the IDELAY control)
The Controller state machine
and User Interface containing
The Read FIFO
And the Write FIFO
As well as THE Synthesizable test bench
You have Complete visibility to the HDL code
And the Option to further customize it

S23
Let's look now at some of the reference designs, an actual example, the DDR2 SDRAM interface.
AND NOW, I would like to introduce my colleague from the Applications team, who is one of the actual designers of the DDR2 SDRAM interface,

Maria George.

Thank you Adrian.
Hi, I am Maria George and I will be presenting the DDR2 SDRAM interface design that is provided by the Memory Interface Generator tool.
We selected DDR2 SDRAM since it is the most prevalent memory technology in the industry.

S24
The DDR2 SDRAM memory interface design comprises of the following blocks:
Physical layer interface that consists the write data path and the read data path
Controller State Machine
A User Interface to your back end application
And the Digital Clock Manager that generates the clock for all these logic blocks

The focus of this presentation will be on the physical layer interface.
During a write operation, the FPGA is required to send the strobe (DQS) to the memory device centered aligned with data (DQ) and this is shown in the waveform on the right

However, during a READ operation, the DDR2 SDRAM sends DQS (strobe) edge aligned with DQ (data) as shown in the waveform on the left. In addition, the strobe is non free-running thereby making read data capture and re-capture challenging to implement.

S25
This read data capture and re-capture challenges are made easy with Virtex-4 devices. The technique we use for read data capture is called Direct Clocking.
With this technique, read data is captured directly in the FPGA clock domain thereby eliminating the read data re-capture challenge. This also provides the ability to support any data to strobe ratio.

The first step in this technique is to determine the phase relationship between the FPGA clock and the read data received at the FPGA.
This is done using the memory read strobe.
Based on this phase relationship, the next step is to delay read data to center it with respect to the FPGA clock.
This delayed read data is then captured in Input DDR flip-flops directly in the FPGA clock domain.

S26
This slide explains how the phase relationship between FPGA clock and read data is determined using the read strobe. The phase detection is performed at run time by issuing dummy read commands after memory initialization. This is done to receive an un-interrupted strobe from the memory.
The goal is to detect 2 edges or transitions of the memory read strobe in the FPGA clock domain.
In order to do this, we input the strobe to the 64 tap IDELAY block that has a resolution of 78 ps.
We start at the 0 tap setting and increment it one tap at a time until we detect the first transition in the FPGA clock domain.
We record the number of taps it took to detect the first edge.
Then we continue incrementing the taps one tap at a time until we detect the second transition in the FPGA clock domain.
We then record the number of taps it took to detect the second edge.
Finally, the required data delay is computed.

The next slide illustrates how the data delay is computed.

S27
The red waveform on this slide is the FPGA clock
The green waveforms are the read strobe and the edge aligned read data received at the FPGA
This memory strobe is input to the IDELAY block and observed in the FPGA clock domain for transitions

The IDELAY taps are incremented one at a time until the first edge or transition is detected
This value is then recorded as ‘first edge taps’
The IDELAY taps are further incremented until the second edge or transition is detected
This value is then recorded as ‘ second edge taps’
The pulse center is computed with these recorded values as (second edge taps – first edge taps)/2
The required data delay is the sum of the first edge taps and the pulse center
This results in the delayed data being centered with respect to the FPGA clock and is shown highlighted in this waveform

S28
The block diagram of the memory strobe edge detection logic is shown here.
The delayed and registered memory strobe is compared with its previous value to detect transitions in the block called ‘Edge detection and control logic’
Each strobe goes through similar circuitry and determines the delay for its associated data bits.

S29
The block diagram of the read and write data path are shown here.
Let’s focus on the read data path since the write data path will be shown in a later slide.
Each Read data bit requires an IDELAY block, and an Input DDR primitive in the IO
Read data is directly captured in the input DDR flip-flops clocked by FPGA clock
Read data is re-captured into FIFOs that can be either implemented in fabric or implemented using the dedicated Block RAM FIFO feature

S30
Before we get into the timing analysis, let’s review the terminology used.
For DDR interfaces the data period is half the clock period as shown
The shaded regions are the uncertainties.
The uncertainties on the left of the rising clock edge affect the set up time and are called the leading edge uncertainties.
The uncertainties on the right of the rising clock edge affect the hold time and are called the trailing edge uncertainties.
The area between these uncertainties is the data-valid window

S31
This is the read timing analysis at 267 MHz for DDR2 SDRAM. The spread sheet lists the uncertainties due to the system, the memory device, and the FPGA.
The sum of the leading edge uncertainties is 685 ps and the sum of trailing edge uncertainties is also 685 ps.
The data valid window between these uncertainties is 317 ps. This is good margin with a 78 ps tap resolution.
S32
Here is the waveform showing the computed data-valid window of 317 ps. With a 78 ps tap resolution it is possible to accurately delay the data to center it with respect to the clock edge.

S33
Write data path is easy to implement because of the quadrature phase outputs of DCM and the Output DDR feature provided in every Virtex-4 IO
Write strobe is generated using ODDR clocked by CLK0 DCM output
Write data transmitted using ODDR clocked by CLK270 DCM output
Using CLK270 for write data and CLK0 for write strobe/clock ensures center alignment of strobe with respect to data as per memory specification
S34
This is the write timing analysis at 267 MHz for DDR2 SDRAM. The uncertainties due to the system, the memory device, and the FPGA are listed here.
The sum of the leading edge uncertainties is 325 ps and the sum of trailing edge uncertainties is 450 ps. The data valid window between these uncertainties is 912 ps. The write data path has more margin than a read and is easy to implement.

The DDR2 SDRAM memory interface and all the other interfaces provided by the Memory Interface Generator tool have been verified and characterized using the ML461 hardware platform. The ML461 development platform is available and Adrian will provide you with this information in the next couple of slides. Thank you.

S35
AND FOR THE SUMMARY I WOULD LIKE TO PASS THE MICROPHONE NOW BACK TO ADRIAN.

THANK YOU MARIA

S36
In summary, we have looked TODAY AT MEMORY INTERFACE CHALLENGES , THE THREE KEY CHALLENGES
Timing critical physical layer, the High bandwidth system requirements
And THE Complex memory controller design
THE Timing critical physical layer, toughest of all !!!
we have showed you how Virtex-4 with **Chipsync built in every I/O provides you with the unique capability of**
Clock-to-data centering at "run time"
Second challenge, the High bandwidth system requirements are also met with unique Virtex-4 capabilities
The Column based architecture and the superior packaging enables better signal integrity and routing making it possible to implement
600Mbps single-ended I/O rates with up to 432 bit wide buses
Finally the 3-rd challenge of Complex memory controller design is accomplished through
Our Hardware verified solutions for all popular memory types (DDR2, DDR SDRAM, QDR II SRAM, AND RLDRAM II)
They are ALL made available by a software tool
The Memory Interface Generator, That can Generate your design in a matter of minutes

S37
For your own design I recommend that you UTILIZE the Xilinx complete hardware proven solutions to assure first time design success

You may start by simply accessing the latest resources at the memory corner at xilinx.com/memory
We offer extensive application notes and reference designs and also the
Memory Interface Generator
Which you may download at NO COST
It is a full version !!
You may also purchase the ML461 development system that includes the board and complete documentation including gerber files

You can also contact your local FAE for an actual on site demo.