Performance Verification of DDR2 Memory Controller using System Verilog

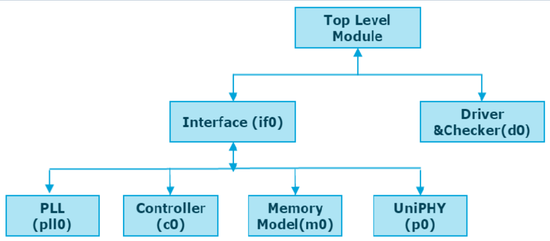
Chinmay Tambat, Professor Morris Jones

Department of Electrical Engineering, San Jose State university, San Jose, California 95192.

Introduction

The project performed System Verilog performance verification of a DDR2 SDRAM Memory Controller by Altera. Performance parameters considered are Read Latency, Write Latency, CAS to RAS latency etc. Performance verification is essential in determining system memory controller effectiveness and data handling capability of the memory. The simulations were observed using ModelSim. Verification takes about 70% of the design cycle and is definitely a major step before the product released actually into the market. The amount of time spent in verifying a particular design clearly indicates the importance of verification. Functional verification also plays an important role in the entire process. Once it is functioning the way it was expected to, the performance defines of the controller defines its actual worth in the market. System verilog as a verification language is extremely powerful because of the wide range of features it provides. It has great in built functions or properties, which makes the verification process very systematic and convenient to perform. System verilog not only helps in generating the wide range of random data to check all possible conditions of data or address but, also allows to constraint these values as per the need. System verilog being an object oriented programming language makes it easier for verification with features adopted from languages like C or C++.

Block Diagram



Top Level module – Instantiates the interface module and driver. Signals in the top level module were observed on ModelSim.

Driver & Checker (d0) – Driver module generates the pseudo random numbers for write and read operation. The checker module checks if the values written is read back correctly.

Interface (if0) – Interface connects all the modules that are responsible for processing the data being sent by the driver through the top level module.

The module instantiations inside Interface are:

- PLL (pll0) – PLL produces the clock for the UniPHY and the Controller. It uses the reference clock supplied to it and generates clocks of different frequencies as per requirement.

Block Diagram

- Controller (c0) – Controller module contains the actual architecture model of the controller that is responsible for handling all the data and address values associated with the operations and to give out the correct output.
- Memory model (m0) – Memory model is included to store the values written and to enable the comparison to take place.
- UniPHY (p0) – UniPHY is responsible to handle the transfer of data/address out of the FPGA. Consists of sequencer, read/write data path handler etc. Sequencer within the UniPHY used to handle the calibration of the design.

System Verilog

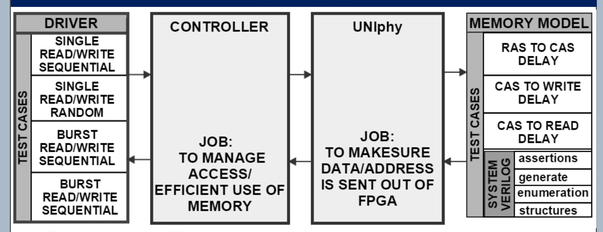
System Verilog was used as the verification language as it provide a variety of data types like C and a lot of built in functions, classes etc. As a result it is a powerful language and helps in the verification process. The following system verilog constructs were used:

Assertion – assertion are used to check conditions and if they are not met it can be used to give an error.

Enumeration – Enumeration is a way to group a set of variables that hold the same type of values and represent it using a single name.

Struct - Structures help to group variables of different data types together under a struct block.

Test Cases



The following Test cases were used to verify the performance of DDR2 Memory Controller.

Single Read/Write Sequential – In this Single Read and Write signals are sent sequentially. This king of access has the highest performance.

Single Read/Write Random – In this Single Read and Write signals are sent in random format. This transfer has moderate performance.

Burst Read/Write Sequential – In this Burst Read and Write signals are sent in a sequential format. This transfer has moderate performance.

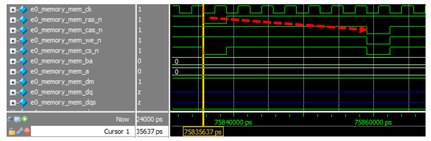
Burst Read/Write Random– In this Burst Read and Write signals are sent at random. Amongst all four this type of transfer has the lowest performance.

Results

For calculating the Latency, signals like RAS, CAS, WriteEnable, ChipSelect, MemDQ Bus and MemDQS strobe were monitored.

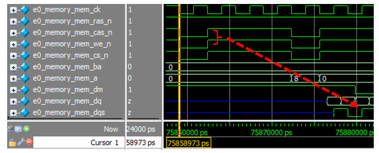
RAS to CAS Latency

This is the latency between when the RAS is asserted for selection of Bank and the CAS is asserted for selection of Memory.



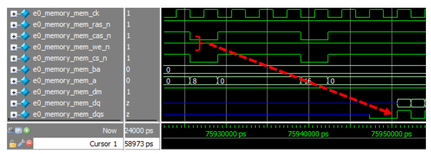
CAS to Write Latency

This is the latency between when the CAS is asserted for selection of Memory and the data is written into the Memory on MemDQ Bus.

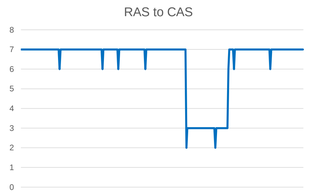


CAS to Read Latency

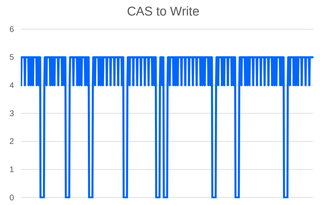
This is the latency between when the CAS is asserted for selection of Memory and the data is Read from the Memory on MemDQ Bus



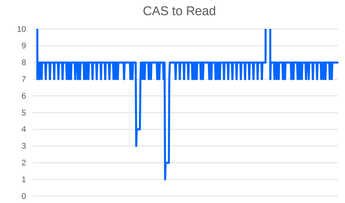
The average RAS to CAS Latency is calculated to be 7.



The Average CAS to Write Latency is calculated to be between 4 and 5.



The Average CAS to Write Latency is calculated to be between 4 and 5.



Summary

To summarize, the RAS to CAS latency is calculated to be 7, CAS to Write latency is between 4 to 5 and CAS to Read latency is between 7 to 8.

The CAS to RAS and CAS to Write latency of the DDR2 Controller is very Good.

The CAS to Read latency can be better if the latency value is reduce by 1.

$$Write\ Latency = Read\ Latency - 1$$

Key References

[1] Altera Corporation, "Introduction to the Quartus 2 Software. Version 10.0," [Online]. Available: https://www.altera.com/en_US/pdfs/literature/manual/intro_to_quartus2.pdf.

[2] V. Thyagarajan, K. Kariya and S. Menon, "A Performance Architecture Exploration and Analysis Platform for Memory Sub-systems," Design & Reuse, [Online]. Available: http://www.design-reuse.com/articles/30944/a-performance-architecture-exploration-and-analysis-platform-for-memory-sub-systems.html.

[3] G. Torres, "Everything You Need to Know About DDR, DDR2 and DDR3 Memories," Hardware Secrets, 27 August 2009. [Online]. Available: http://www.hardwaresecrets.com/printpage/Everything-You-Need-To-Know-About-DDR-DDR2-and-DDR3-Memories/167.

Acknowledgements

The authors wish to thank Prof. Morris Jones for his assistance in helping us understand the concepts of System Verilog and DDR2 Memory Controller. Thanks to Altera for providing us the DDR2 Core for verification.