Implementation of AXI Design Core with DDR3 Memory Controller for SoC

[1]Darshana Dongre, [2]Anil Kumar Sahu
[1,2]Dept. of ECE, Shri. Shankaracharya college of Engineering & Technology, Bhilai, India

Abstract
This paper discusses the overall architecture of AMBA AXI design core along with its advantage with DDR3 memory controller and operation of its individual sub blocks. It takes care of the DDR3 initialization and various timing requirements of the DDR3 memory. The memory controller works as an intelligent bridge between the AXI host and DDR3 memory. Our design has been implemented with respect to latency reduction and improvement in various performance parameters and the design is simulated on Modelsim and synthesized on Xilinx successfully.

Keywords
DDR3 memory, AXI Interconnect, AXI interface, AXI access Manager, DDR3 memory controller, AXI design core.

I. Introduction

Presently, Communication between processors and memories is often a major bottleneck, making the design of the memory controller a critical task in determining overall system-level performance [1-3]. The AXI compliant DDR3 Controller permits access of DDR3 memory through AXI Bus interface. It describes the controller and the data capture technique for high-performance DDR3 interfaces. The memory controller, in addition to collecting the requests from the masters and forwarding them to the memory, is tasked with re-ordering these requests in order to optimize specific system characteristics, such as latencies, power consumption and throughput [3]. Numerous optimization techniques have been developed over the last few years. In this project we have utilized the advanced feature of AMBA 3 protocol–AXI (Advanced eXtensible Interface) is targeted at high-performance, high-frequency system designs and includes a number of features such as a unidirectional channel architecture, which enables the efficient use of register slices to pipeline the connection for higher speeds, or to enable the use of multiple clock domains for low power. The support of multiple outstanding transactions and out-of-order transaction completion, together with the efficient use of the read/write and address/control channels enable systems to achieve levels of performance and Efficiency limited only by the capabilities of the peripherals themselves.

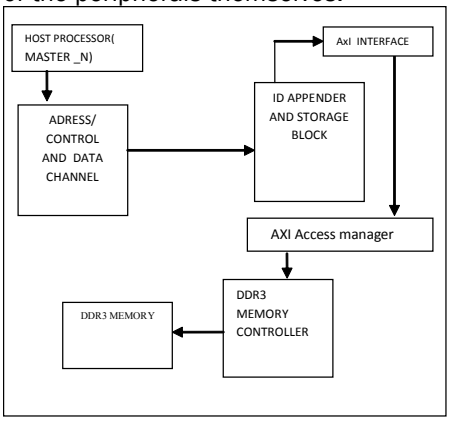


Fig. 1: Axi Design Core with ddr3 Memory Controller

II. AXI Compliant DDR3 Controller Architecture

The architecture of the design is shown in the fig. 1. The design consists of following blocks-
A. AXI Interconnect
B. AXI interface
C. AXI access Manager
D. DDR3 Memory Controller Bridge

A. AXI interconnect

It is responsible for communication between AXI protocol and device interface. Its major parts are ID appender and storage as in fig. 2. Its task is to store the ID from the AXI host and arbiter is for scheduling the multiple AXI interface and slaves are memory interface. The AXI interconnect is responsible for all five channel operation according to AXI.

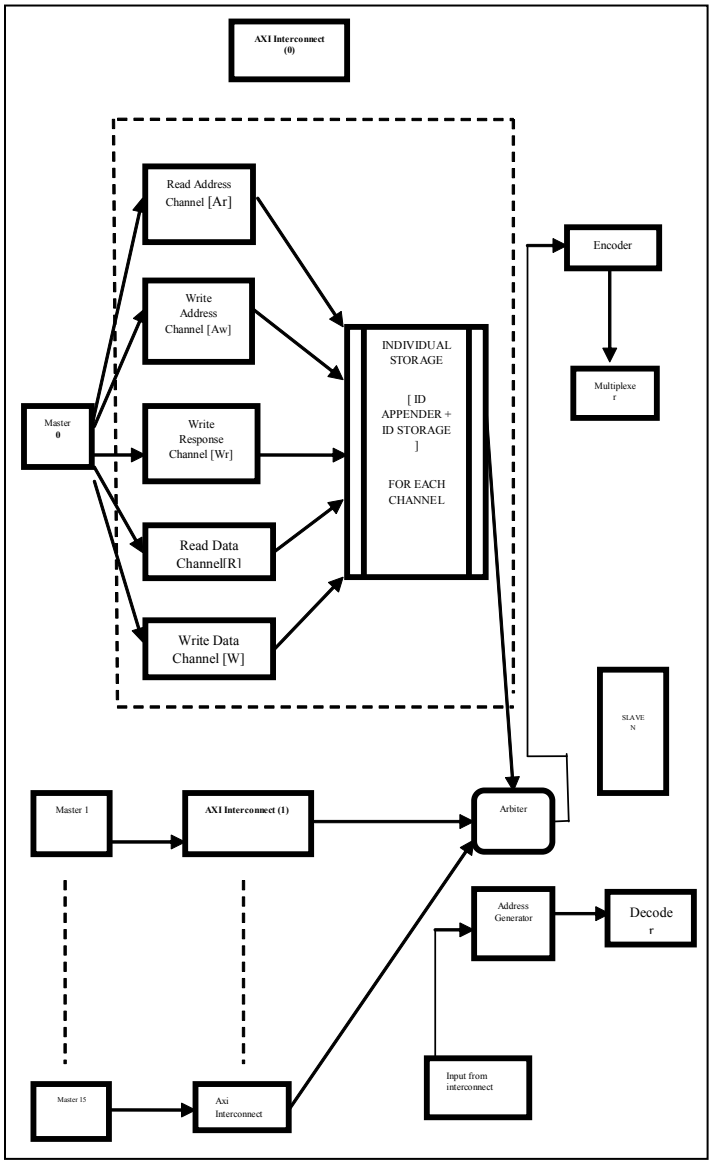


Fig. 2: AXI Interconnect

B. AXI Interface

AXI-Interface act as a bridge between AXI interconnects and AXI access manager. It accepts the AXI commands and interprets the AXI commands. The AXI-interface responds to Read / Write requests in AXI protocol as requested by processor. It consists of an arbiter which is responsible for arbitrating between Read/Write commands. Arbitration is required between the commands because of parallel/independent Read/Write command received at the AXI interconnect to AXI interface. It maintains asynchronous FIFO's to store the command and the data. Whenever AXI access manager is free, the stored commands are sent to the AXI access manager. Now since FIFO consists of both read and writes commands, stored in their respective blocks, thus it requires an arbiter which arbitrates between read and write commands and whenever Burst Manager is free one of the present command is supplied to the Burst Manager. The fig. 3, shows AXI Interface block:

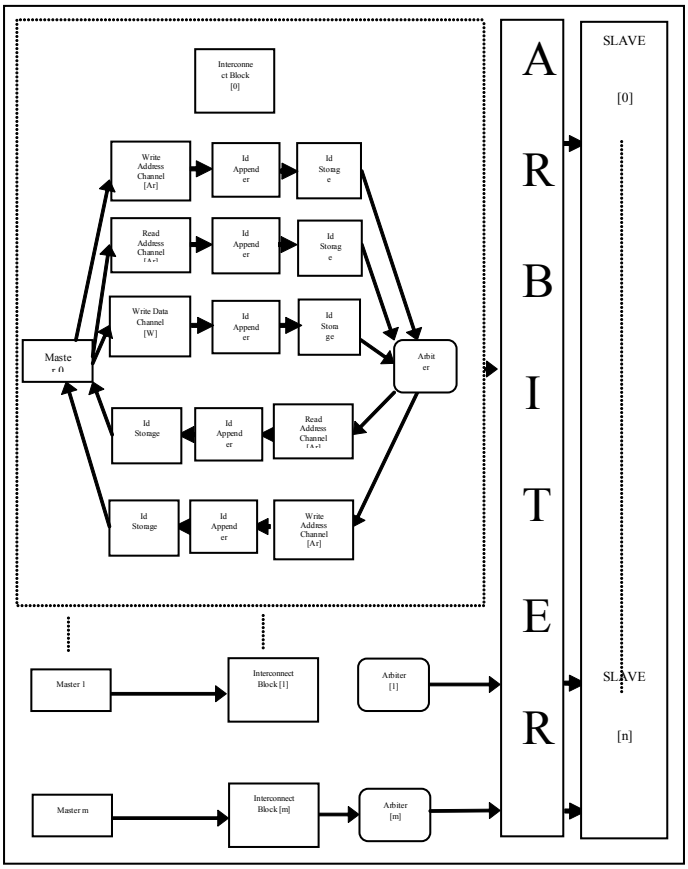


Fig. 3: AXI Interface Block

C. AXI Access Manager

The AXI-Access Manager performs the task to convert AXI commands into memory access commands for efficient utilization of the DDR3 Bandwidth. The memory manipulates command only in burst 4 or 8 mode whereas the AXI command could be of variable burst. To improve the overall performance the AXI access manager combines the command wherever possible and sends this command to DDR3 controller. It performs parallelism mechanism in order to attain maximum throughput. The commands are fetched from the AXI interface and convert into memory commands and controlled locally. Whenever controller is free, these stored commands are supplied to it in the very next clock. The prefetch command block interacts with the AXI interface block and receives commands which are stored in the Storage control block. The Burst Manager converts the AXI command into DDR3 burst. The address control block generates required address. The Control Logic controls operation of various blocks in the AXI Access Manager. The AXI access Manager Block is shown in fig. 4.

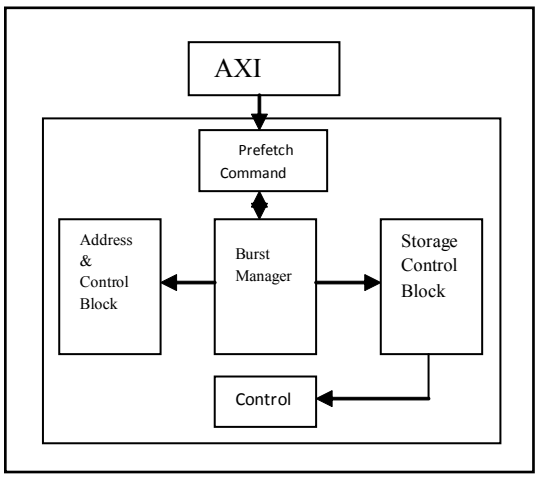


Fig. 4: AXI Access Manager Block

D. DDR3 memory Controller

The DDR3 controller acts as interface between AXI access manager and the DDR3 memory. It is responsible for understanding the DDR3 protocol and communicating with the DDR3 memory.DDR3 Controller also performs other functions such as Refresh, Power down, Self refresh command along with the read or writes command [1]. The internal blocks of DDR3 Controller Bridge are shown in fig. 5. The Power down Control Block generates the power down command to the DDR3 memory whenever the host commands it to go to Power saving mode. The refresh control block refreshes the DDR3 memory as per the user supplied configuration. The initialization control block initializes the DDR3 memory after reset. To control the timings of DDR3 banks it contains central Bank manager which track their timing requirements. The address control block is responsible for generating the address to the DDR3 memory.

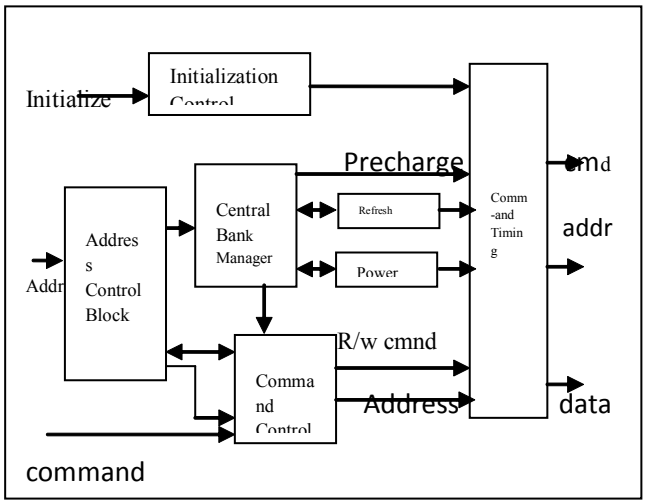


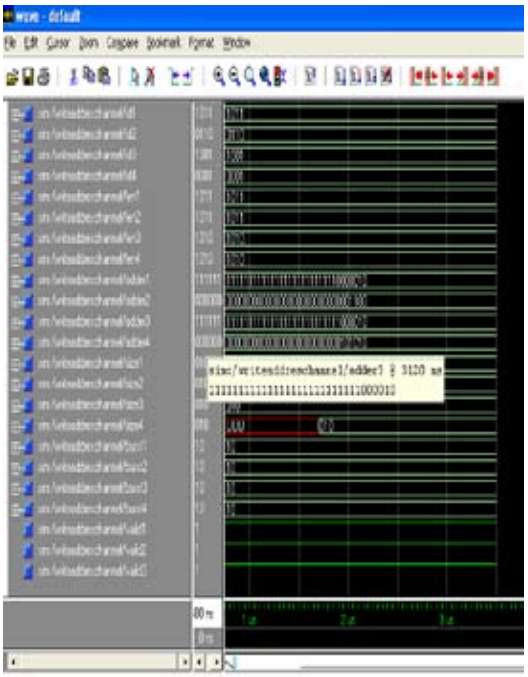
Fig. 5: DDR3 memory controller block

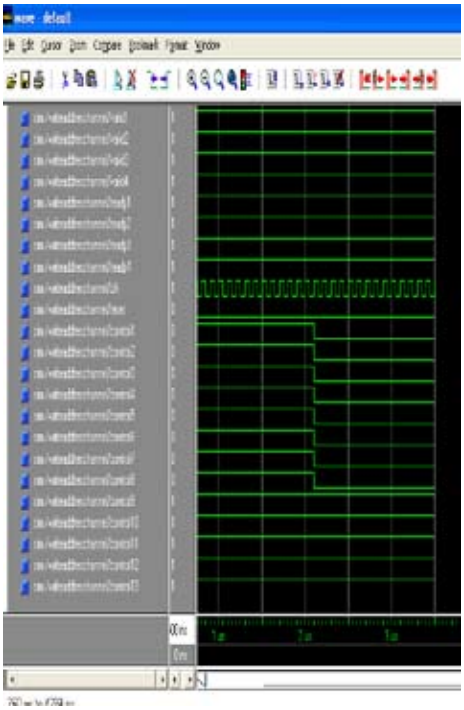
Since the activation of a command is time consuming it is vital to keep track of if a bank is open and in that case which row that are activated so that if a consecutive command is to the same bank and row it does not have to be activated . The

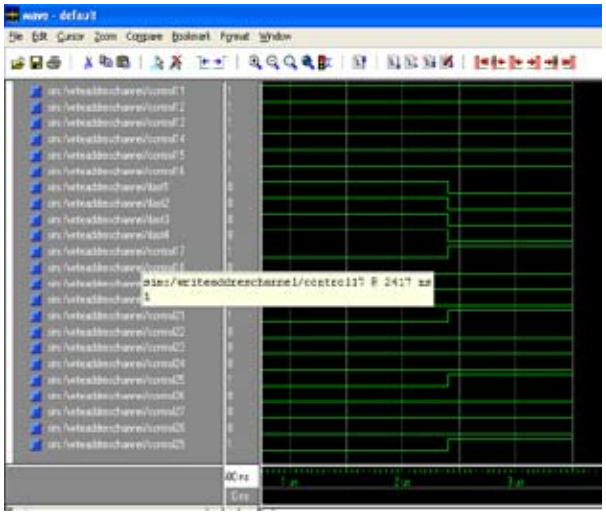
address control block is responsible for generating the address to the DDR3 memory. The Command control block interacts with the DDR3 memory and drives the DDR3 bus. It is also responsible for receiving the data during the Read operation. The Central Bank manager coordinates maintains the overall timing requirement of DDR3 memory.

III. Simulation Results

The below figures are showing snapshot of the DDR3 Controller operation in various modes. The design has been coded in "Verilog HDL" language. Our design has been implemented with respect to low area and high performance parameter and the design is simulated on Modelsim and synthesized on Xilinx Virtex-4 XC4VFX25SF363-12 series successfully, about 60% area utilization has been achieved.







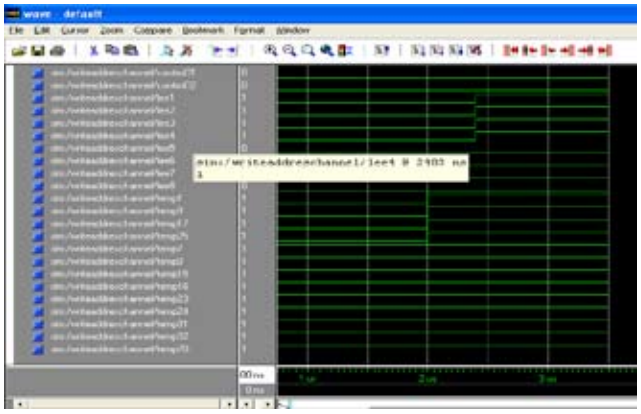


Fig. 6 Simulation Results

IV. Conclusion

The design has been verified by the exhaustive functional verification. We examined the performance of the design by generating different type of AXI commands and noting down the time taken by the DDR3 controller in finishing them. In most of the scenario the throughput of the design is close to the theoretical max. The latency of the design is between 10 to 35 clocks based on the command generated and the internal DDR state.

V. Future Improvments

Future improvements in AXI interface block is to add more features like fixed address mode, address wrapping mode and write response signal generation other than OKEY response. In fixed burst, the address remains all the same for every transfer in the burst. This burst type is for every repeated accesses to the same location such as when loading or emptying a peripheral FIFO and wrapping burst is similar to an incrementing burst, in that the address for each transfer in the burst is an increment of the previous transfer address and thus can improve the design.

References

[1] Churoo (Chul-Woo) Park, HoeJu Chung, Yun-Sang Lee, Jun-Ho Shin, Jin-Hyung Cho, Seunghoon Lee, Ki-Whan Song, Kyu-Hyoun Kim, Jung-Bae Lee, Changhyun Kim, Soo-In Cho, "A 512-Mb DDR3 SDRAM Prototype and Self-Calibration Techniques", Proc. IEEE Journal of Solid-State Circuits, Vol. 41, NO. 4, 2006

[2] K. Kim et al, "A 1.4 Gb/s DLL using 2nd order chargepump scheme with low phase/duty error for high-speed DRAM application", in IEEE Int. Solid-state Circuits Conf. (ISSCC) Dig. Tech. Papers, 2004, pp. 212-523.

[3] S.Lee et al, "A. 1.6 Gbs/pin double data rate SDRAM with wavepiplined CAS latency control", in IEEE Int.Solid-State Circuits conf. (ISSCC) Dig. Tech. Papers, 2004, pp. 210-213.

[4] H. Song et al, "A 1.2 Gbs/pin double data rate SDRAM with on die-termination", in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, 2003, pp. 31