

# ASIC Implementation of DDR SDRAM Memory Controller

Amit Bakshi  
School of Electronics  
Engineering.  
VIT University, Vellore-  
632014, India  
abakshi.ece@gmail.com

Sudhanshu Shekhar  
Pandey  
School of Electronics  
Engineering.  
VIT University, Vellore-  
632014, India  
sudhanshu\_mtech88@ya  
hoo.com

Tribikram Pradhan  
School of Information  
Tech. & Engineering  
VIT University, Vellore-  
632014, India  
tribikram14@gmail.com

Ratnadip Dey  
Electronics And Comm.  
Engineering  
Institute Of Engineering  
& Management  
ratnadip87@gmail.com

**Abstract-** A Dedicated Memory Controller is of prime importance in applications that do not contain microprocessors (high-end applications). The Memory Controller provides command signals for memory refresh, read and write operation and initialization of SDRAM. Our work will focus on ASIC Design methodology of Double Data Rate (DDR) SDRAM Controller that is located between the DDR SDRAM and Bus Master. The Controller simplifies the SDRAM command interface to standard system read/write interface and also optimizes the access time of read/write cycle. Double Data Rate (DDR) SDRAM Controller is implemented using Cadence RTL Compiler.

**Keywords-** DDR SDRAM Controller, Read/Write Data path, Cadence RTL Compiler.

## I. INTRODUCTION

Memory devices are almost found in all systems and nowadays high speed and high performance memories are in great demand. For better throughput and speed, the controllers are to be designed with clock frequency in the range of megahertz. As the clock speed of the controller is increasing, the design challenges are also becoming complex. Therefore the next generation memory devices require very high speed controllers like double data rate and quad data rate memory controllers. In this paper, the double data rate SDRAM Controller is implemented using ASIC methodology.

Synchronous DRAM (SDRAM) is preferred in embedded system memory design because of its speed and pipelining capability. In high-end applications, like microprocessors there will be specific built in peripherals to provide the interface to the SDRAM. But for other applications, the system designer must design a specific memory controller to provide command signals for memory refresh, read and write operation and initialization of SDRAM. In this paper, the SDRAM controller, located between the SDRAM and the bus master, minimizes the effort to deal with the SDRAM memory by providing a simple system to interact with the bus master. Figure 1 is the block diagram of the DDR SDRAM Memory Controller that is connected between the bus master and SDRAM [1].

SDRAM's are classified based on their data transfer rates. In Single data rate SDRAM, the data is transferred on every rising edge of the clock whereas in double data rate (DDR) SDRAM's the data is transferred on every rising edge and every falling edge of the clock and as a result the throughput is increased. DDR SDRAM Controllers are faster and efficient than its counterparts. They allow data transfer at a faster rate without much increase in clock frequency and bus width.

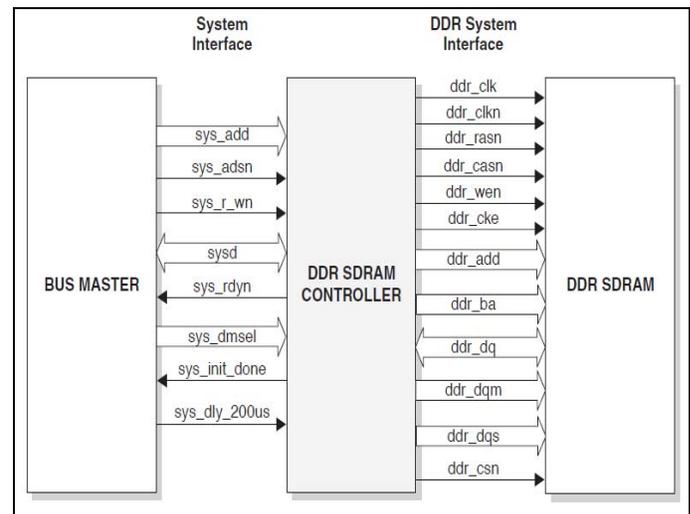


Fig. 1. DDR SDRAM controller system [1]

## II. IMPLEMENTATION METHODOLOGY

The DDR SDRAM Controller architecture is implemented using Verilog HDL. The methodology followed is ASIC design flow. The basic steps that an ASIC design must go through are Design entry and Analysis, Technology Optimization and Floor planning, Design Verification, Layout [5]. The RTL Synthesis and Simulations are performed using existing tools like Cadence RTL Compiler.

### III. DDR SDRAM CONTROLLER ARCHITECTURE

DDR SDRAM Controller module receives addresses and control signals from the BUS Master. The Controller generates command signals and based on these signals the data is either read or written to a particular memory location. The DDR SDRAM Controller architecture is shown in Figure 2. It consists of three modules: 1) Main control module 2) signal generation module 3) data path module. The main control module has two state machines and a refresh counter. The two state machines are for initialization of the SDRAM and for generating the commands to the SDRAM. They generate iState and cState outputs according to the system interface control signals. The signal generation module now generates the address and command signals depending upon the iState and cState. The data path module performs the read and write operations between the bus master and DDR. Following are some of the important features of DDR SDRAM Controller:

- i. The DDR SDRAM Read and Write operations are simplified by the controller.
- ii. For initializing the DDR SDRAM controller, separate state machines are designed internally.
- iii. The access time for read and the write cycle is optimised based on the CAS latency and burst length of the DDR SDRAM.
- iv. The auto refresh for the DDR SDRAM is done by the controller.

The main control module consists of three sub modules: 1) Initialization FSM module (INIT\_FSM). 2) Command FSM module (CMD\_FSM) 3) Counter module.

#### A. Main control module

The DDR SDRAM Controller has to undergo an initialization process by a sequence of command signals before the normal memory access. The initialization finite state machine in the main control module is responsible for the initialization of the DDR SDRAM controller. Figure 3 shows the state diagram of the initialization FSM (INIT\_FSM). Whenever reset signal is high, the initialization FSM will switch to *i\_IDLE* state. Once the reset signal goes low, the controller has to wait for 200us clock stabilization delay. This is constantly checked by *sys\_dly\_200us* signal and a high on the *sys\_dly\_200us* will indicate that the clock stabilization delay is complete. The DDR initialization sequence will begin immediately after the clock/power stabilization is complete and then the INIT\_FSM will change its state from *i\_IDLE* to *i\_NOP* state. From the *i\_NOP* state, the initialization FSM will switch to the *i\_PRE* state on the next clock cycle.

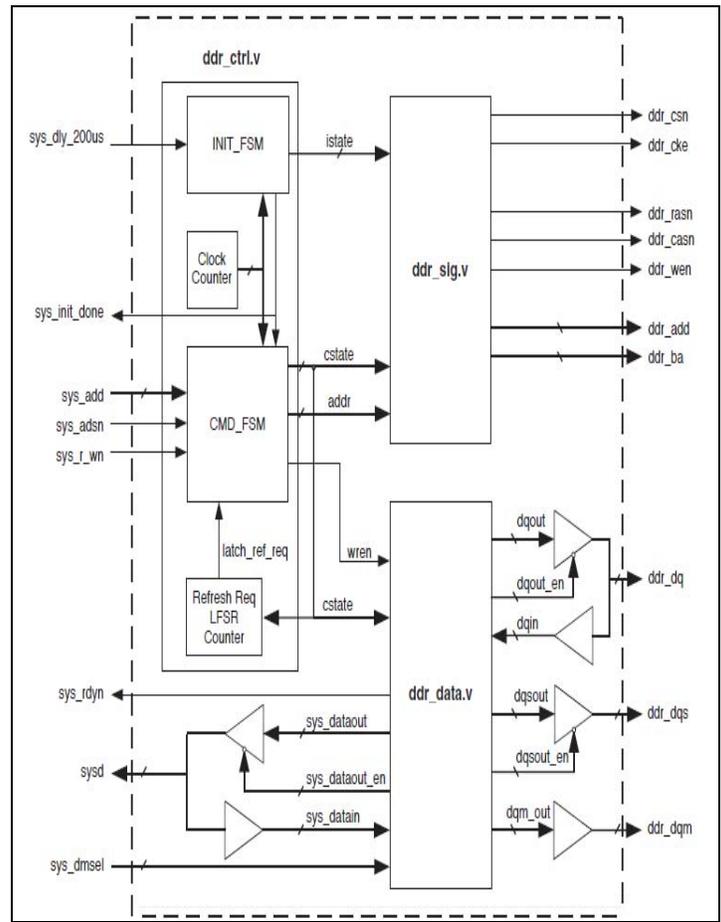


Fig. 2. Functional block diagram of DDR SDRAM controller [2].

In the *i\_PRE* state, the main control module will generate the PRECHARGE command. The PRECHARGE command generated during this state will be applied to all the banks in the device. Once the PRECHARGE command is generated by the initialization FSM, it will switch to the next state. The next state in the design of initialization FSM is two AUTO REFRESH commands. These refresh commands are generated to refresh the DRAM memory. After the two refresh state, the initialization FSM will switch to *i\_MRS* state. During this state LOAD MODE REGISTER command is generated to configure the DDR SDRAM to a specific mode of operation. After satisfying the *i\_tMRD* timing delay the initialization FSM will switch to *i\_ready* state. The initialization FSM will remain in the *i\_ready* state for normal memory access. And also, when the initialization FSM switches to *i\_ready* state signal *sys\_INIT\_DONE* is set to high to indicate that DDR SDRAM controller initialization is completed. The *i\_PRE*, *i\_AR1*, *i\_AR2*, *i\_EMRS* and *i\_MRS* states are used for issuing DDR commands.

CMD\_FSM handles the read, write and refresh of the SDRAM. The CMD\_FSM state machine is initialized to *c\_IDLE* during reset. After reset, CMD\_FSM stays in *c\_IDLE* as long as *sys\_INIT\_DONE* is low which indicates the



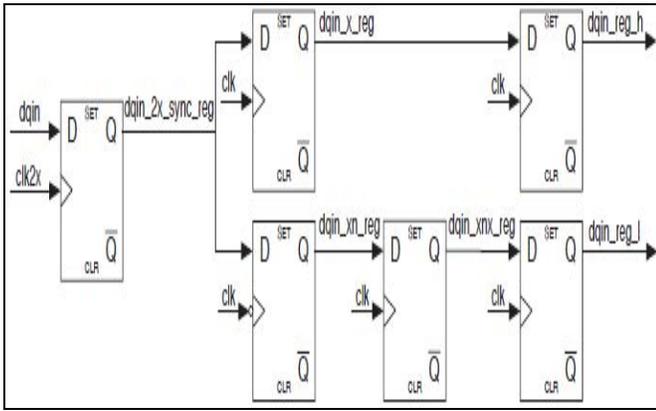


Fig. 5. Read data path for DDR SDRAM Controller [1].

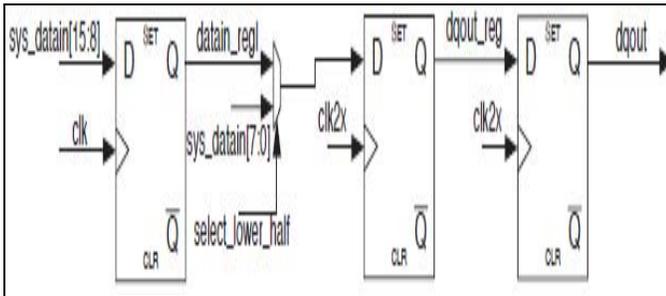


Fig. 6. Write data path for DDR SDRAM Controller [1].

### C. Signal Generation Module

The DDR SDRAM Signal generation module should generate the command signals to the DDR SDRAM. The command signals include `ddr_add`; for generating addresses, `ddr_casn` and `ddr_rasn`; for selecting particular column and row address. These signals are generated based on the `iState` and `cState` received from the `CMD_FSM` and `INIT_FSM` present in the Main Control module.

## IV. EXPERIMENTAL RESULTS

The Design is simulated after combining the BUS Master, DDR SDRAM Controller and the DDR SDRAM. Micron's MT46V16M8 [3] is chosen as the DDR SDRAM for this design. The Controller is designed using Verilog HDL. Cadence NCSIM is used for Simulation and Cadence RTL Compiler for the Synthesis of Controller [2].

Figures below show the timing diagram of Controller Initialization, Read Cycle and Write Cycle. In figure 7, once the initialisation sequence for the DDR SDRAM Controller is completed, `sys_INIT_DONE` becomes high. In figure 8 and 9, on both positive edge and negative edge of the clock, the read and write operations are performed.

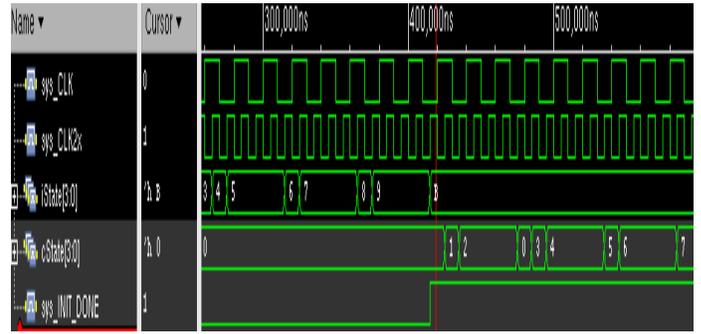


Fig. 7. Simulation result of the Initialization sequence of DDR SDRAM Controller.

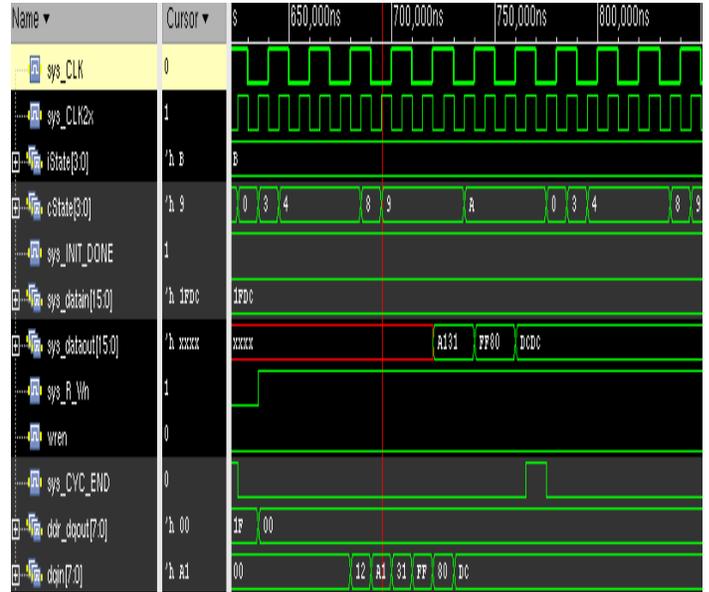


Fig. 8. Simulation result of the Read Cycle of DDR SDRAM Controller.

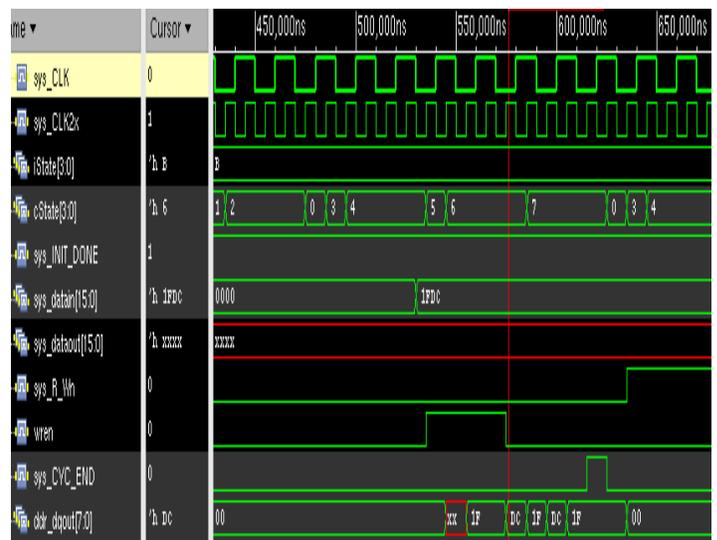


Fig. 9. Simulation result of the Write Cycle of DDR SDRAM Controller.

The Synthesis summary for the DDR SDRAM Controller is shown in Table I. The synthesis reports are obtained in Cadence RTL Compiler.

TABLE I  
SYNTHESIS RESULTS OF DDR SDRAM CONTROLLER OBTAINED IN CADENCE RTL COMPILER

Synthesis Summary	
Area (in Cells)	384
Leakage Power (in nW)	72.145
Dynamic Power (in mW)	15.86
Timing Slack (in ps)	7898

## V. CONCLUSION

ASIC Design methodology is opted for Double Data Rate (DDR) SDRAM Controller. The DDR SDRAM Controller architecture is implemented in Verilog HDL. The RTL Simulation and Synthesis results are obtained using Cadence NCSIM and RTL Compiler. Future work may include comparison of DDR/DDR1/DDR2 architectures.

- [1] DDR SDRAM Controller white paper, *Lattice Semiconductor Corporation*, Reference Design: RD1020, April 2004.
- [2] SDR SDRAM Controller white paper, *Lattice Semiconductor Corporation*, Reference Design: RD1010, April 2011.
- [3] Chen Shuang-yan, Wang Dong-hui, Shan Rui Hou Chao, "An Innovative design of DDR/DDR2 SDRAM Compatible Controller", *ASICON International Conference*, pp 62-66 24th Oct 2005.
- [4] Micron Technology, Micron's Synchronous DRAM (2002).
- [5] Cadence RTL Compiler help documents version 10.1, from [www.downloads.cadence.com](http://www.downloads.cadence.com).
- [6] Michael John Sebastian Smith, "*Application Specific Integrated Circuits*", 2004.