Memory Controllers: A comparative study of SDRAM and DDR SDRAM Controller

Pavankumar N C, Jeeru Dinesh Reddy

Abstract— **Random access memory is an essential resource required by the computational hardware. As the processor speed has attained GHz clock frequency, memory throughput can be a bottleneck to achieve high performance. DRAM can deliver a reasonable solution for such data storage. Typical computational system consists of multiple hardware modules that perform different operations on the data. These modules attempt to access the data concurrently. This leads to a requisite for a memory controller that arbitrates amid requests queried by different modules and exploits maximum throughput. The memory controller interfaces DRAM and other subsystems. Hence it manages the data into and out of memory. The access latency or access speed solely depends on the implementation of memory controller. The work concentrates on the relative study of two memory controllers viz., SDRAM and DDR SDRAM controller. The study comprises area, power and timing analysis of the both. Synopsys Design Compiler tool is used to obtain the necessary results.**

Index Terms—**SDRAM, DDR, ASIC, Latency.**

I. Introduction

Any computational hardware or commonly computer system requires a minimum storage. The storage requirement can be fulfilled by two different classes of memories viz., Static RAM (SRAM) and Dynamic RAM (DRAM). A flip-flop is used in SRAM to retain the information. A single bit SRAM cell is made of 6 transistors and stores the information as a logic level in a cross connection of transistors. Benefits of SRAM are no refresh mechanism, low power consumption and no address multiplexing. Hence making it suitable for higher levels of the memory pyramid where memory must be quick, such as in scratchpads. SRAM has drawback of low memory density and expensive. When there is a want for mass storage and is not time critical, the DRAMS can be employed in the storage.

DRAMs are the main memory in all computing systems. Robert Dennard of IBM invented DRAM concept in 1967. DRAM is an acronym that stands for dynamic random access memory. In a DRAM, a combination of a transistor usually an NMOSFET and a capacitor, called a memory cell stores a bit of information. This allowed the memory designer to accommodate large memory cells, therefore escalating the memory density. The bit of data is stored as charge on the capacitor. Reading the data on the capacitor can disrupt the data in the memory cell, consequently it requires a precharge mechanism to maintain the stored data. A practical capacitor is a charge leaky, so the information may get lost. Therefore a memory cell is refreshed regularly. Accordingly the tag is dynamic RAM.

DRAM is an array of memory cells. In comparison with the architecture of SRAM, the architecture of DRAM packs more memory cells into the memory. This is the reason for the bulky width of address lines. This causes more pin count to house increased address lines. More pin count poses signal integrity problem and pricey too. To avoid these complications, the address is allocated into row and column address bus. The architecture is as shown in Fig. 1 with three aspect row, column and bank.

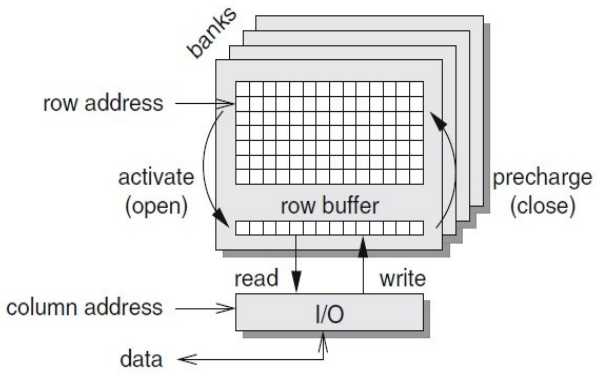


Fig. 1 DRAM Architecture

Because of address multiplexing and constraint for the precharge and refresh mechanisms, DRAM is characteristically slow. To match the swiftness of microprocessor, there is a necessity of an extra hardware to match the processor speed and response of the DRAM. The hardware may be named as a memory controller as it regulates the data into and out of DRAM. Besides memory controller confirms the protocol compliance, DRAM device specific electrical and timing characteristics.

To compensate for low speed of operation, several DRAMs are staggered in parallel. Though individual of each device is low, the parallel configuration of DRAMs enables more data access at the same time ensuing in higher bandwidth. This practice is an easy way to improve the performance of DRAM based memory systems. The high bandwidth permits DRAM to manage high throughput

***Pavankumar NC**, M Tech Scholar, Dept. of ECE, BMS College of Engineering, Bangalore, India.*

***Jeeru Dinesh Reddy**, Asst. Professor, Dept. of ECE, BMS College of Engineering, Bangalore, India.*

ISSN: 2278 – 7798

although latency is still high. The arrangement to increase the bandwidth by means of parallel pattern of DRAM devices is replicated and arranged as banks exterior to the integrated component. A bank is a set of memory arrays that operates independently of other sets.

In the novice stage of development DRAMs implemented asynchronous protocol for interfacing and communication. This protocol is inherently slow. To reduce latency and to sustain the same bandwidth synchronous protocol is adopted in the design of DRAMs devices and hence they became synchronous DRAMs (SDRAMs).

Based on the bandwidth, SDRAMs are classified as single data rate and double data rate (DDR) SDRAMs. Single data rate SDRAMs or just SDRAMs transfer the data usually on the positive edge of the system clock and all the protocols are synchronous to the same system clock. The protocol covers six commands: activate (ACT), read (RD), write (WR), precharge (PRE), refresh (REF), and no-operation (NOP). The ACT command instructs, with a row and a bank as argument, the chosen bank to copy the requested row to its buffer. The requested row is opened; column accesses i.e. read or write bursts can be issued to contact the columns in row buffer. Burst length is the volume of data read/written after a read/write command is offered by the memory controller.

The major difference between SDRAM and DDRx SDRAM is that DDR transfers the data on both edge of the data strobe signal which is synchronized with the system clock. As latency is inflexible to improve, this way data rate is doubled and high bandwidth is accomplished. SDRAM and DDR SDRAM are internally multi-bank architecture. Programmability is a distinguishing characteristic of these memories. This allows memory controller to transfer the data in bursts consequently higher speed of operation. Both technologies have registers to program the various attributes for data transfers.

II. IMPLEMENTATION METHODOLOGY

Typical ASIC practice is followed in carrying out SDRAM and DDR SDRAM Controller architectures implementation using Verilog HDL. The ASIC implementation flow includes following steps: Specification capture and design entry, logical simulation and analysis, placement and floor planning, design verification, layout. The RTL synthesis and simulations are executed Cadence RTL Compiler.

III. CONTROLLERS ARCHITECTURE

The controller modules accept addresses and control signals from the BUS Master. The Controller produces command signals and based on these signals the data is either read or written to a particular memory location. The DDR SDRAM and SDRAM controller have similar architectures internally. Few modifications are encompassed in DDR SDRAM Controller architecture to accomplish double bandwidth compared to SDRAM Controller. Therefore only DDR SDRAM controller is described in following sections. The controllers has similar architecture is shown in Fig. 2 and Fig. 3.

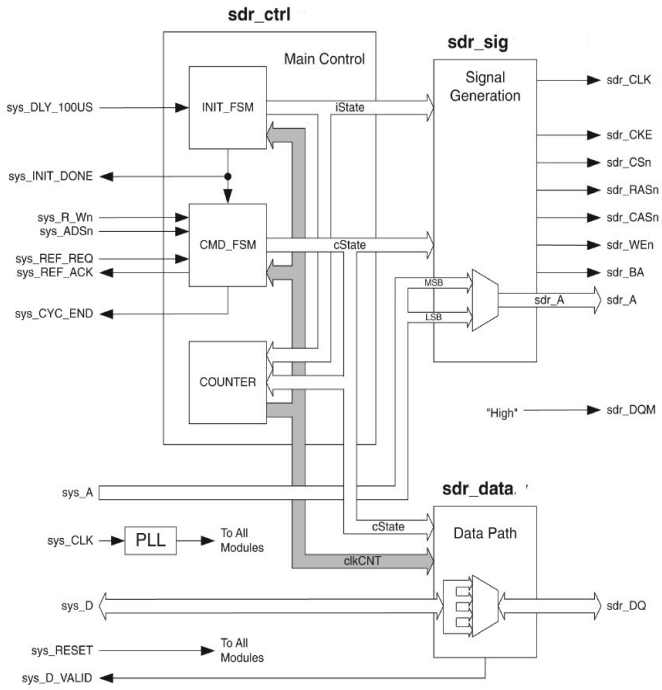


Fig. 2. Functional diagram of SDR SDRAM controller [1]

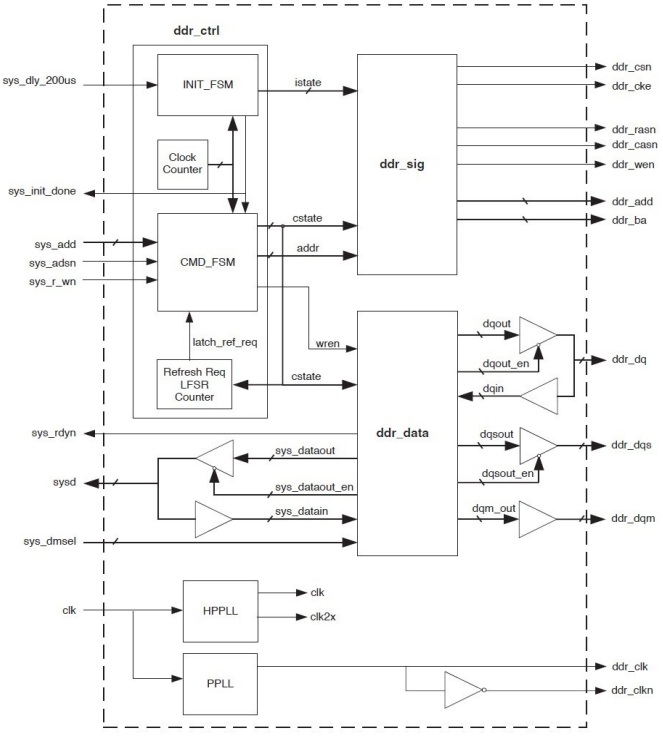


Fig. 3. Functional diagram of DDR SDRAM controller [2]

The memory controller has three modules:

1) Main control module
2) Signal generation module
3) Data path module.

The main control module is with two state machines and a refresh counter. The two state machines initialize the SDRAM and generate the commands to the SDRAM. According to the signals from system interface state machines generate iState and cState signals. Based on the

ISSN: 2278 – 7798

iState and cState the signal generation module now creates the address and command signals.

The data path module completes the read and writes operations between the bus master and SDRAM. Few essential features of memory controller are as below:

1) The controller simplifies read and write operations.
2) Internally separate state machines are designed to initialize the memory controller,.
3) Based on the CAS latency and burst length of the SDRAM the access time for read and the write cycle is optimized.
4) The controller does auto refresh for the SDRAM.

Three sub modules make main control module and they are:

1) Initialization FSM module (INIT_FSM).
2) Command FSM module (CMD_FSM)
3) Counter module.

B. Main control module

The Controller has to go through 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. Fig. 4 shows the state diagram of the initialization FSM (INIT_FSM). When reset signal is high, the initialization FSM will switch to i_IDLE state. Once the reset signal goes low, the controller has to take a pause for 200us clock stabilization delay. The sys_dly_200us signal constantly checks and a high on the sys_dly_200us indicates the end of clock stabilization delay. The initialization sequence initiates immediately after the conclusion of clock/power stabilization and then the INIT_FSM will change its state from i_IDLE to i_NOP state. The initialization FSM will switch from the i_NOP state to the i_PRE state on the next clock cycle.

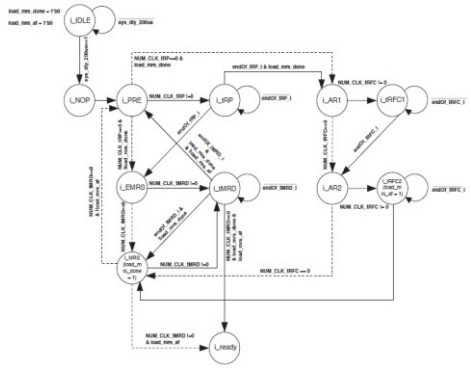


Fig. 4.Initialization FSM (INIT_FSM) state diagram [2]

In the i_PRE state, the main control module generates the PRECHARGE command and this command is applied to all the banks in the device. After the PRECHARGE command, INIT_FSM will switch to the next state. The next state in the design of initialization FSM is two AUTO REFRESH commands. These commands will refresh the DRAM memory. After the two refresh cycles, the initialization FSM will shift to i_MRS state. In this state LOAD MODE REGISTER command is generated to conFig. the 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. When the initialization FSM switches to i_ready state signal sys_INIT_DONE is set to high to indicate that 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 SDRAM initialization sequence is not yet completed. A high on sys_INIT_DONE indicates the system initialization is complete. The controller now waits for latch_ref_req, sys_INIT_DONE signals and enters auto refresh, read and write mode depending upon these signals. When the initialization is complete and when the latch_ref_req goes high the controller will refresh by entering into refresh state. After the refresh is complete, when the latch_ref_req and sys_ADSn signal goes low, the controller will go to active state. The ACTIVE command is issued for each read or writes access to open the row.

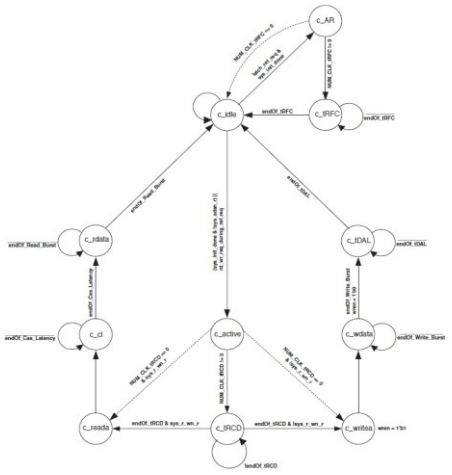


Fig. 5. Command FSM (CMD_FSM) [2]

Once the specified interval is satisfied, READ or WRITE commands are issued. The sys_R_Wn signal determines read or write operation. Logic high switches the state machine to c_READA. If a logic low is sampled, the state machine switches to c_WRITEA. For read cycles, the state machine changes from c_READA to c_cl for specified delay, then switches to c_rdata for transferring data from SDRAM to bus master. After the data transfer, it switches back to c_IDLE. For write cycles, the state machine switches from c_WRITEA to c_wdata to transfer the data from bus master to SDRAM, and then switches to c_tDAL. After writing the data, it switches back to c_IDLE state.

C. Signal generation module

The signal generation module generates the command signals to SDRAM. The signals comprise of ddr_add; to generate the addresses, ddr_casn and ddr_rasn; to select particular column and row address. These signals are initiated based on the iState and cState established from the CMD_FSM and INIT_FSM present in the main control module.

ISSN: 2278 – 7798

D. Datapath module

The data path module for read and write are shown in Fig.. 5. The data path module reads/writes according to cState. The cState is a signal from the CMD_FSM present in the main control module.

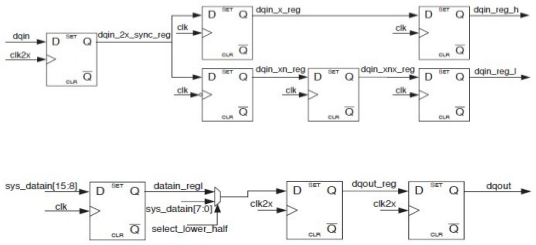


Fig. 6. Read and write datapath

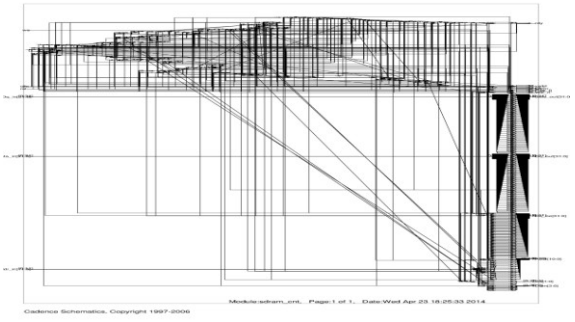
IV. Simulation and Synthesis

Connecting the compatible controller, bus interface, and the simulation models together, simulates the design. The controller is written in Verilog language, Cadence Native Compiler is used for the simulation process.

After simulation, the controller is synthesized. Using Cadence synthesis tool RTL-Compiler, both controllers are synthesized. The power and area reports are obtained from Cadence SoC Encounter and are shown.

	Cell Area	Dynamic Power (nW)	Leakage Power (nW)	Total Power (nW)
SDRAM Controller	3331	7998.085	7998.085	61818.985
DDR SDRAM Controller	7457	64801.848	18906.381	83708.229

Table 1. Area and Power Report of the controllers



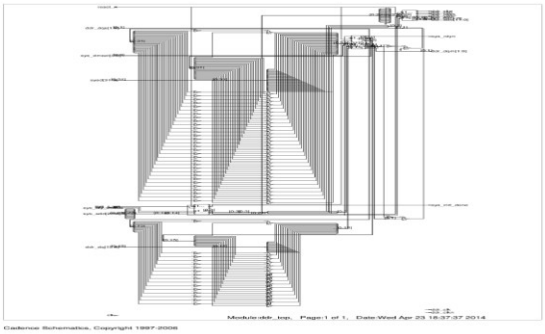


Fig. 7. Synthesized modules of SDRAM and DDR SDRAM Controller

V. Conclusion

To summarize, to meet the demand of the computer market, there is a need of the design with the feature of good performance, low power consumption and low cost. Memory will remain an inevitable component and a crucial device in the computational arena. With this motive, an implementation of the SDRAM and DDR SDRAM controller is presented in this paper. Power and area are two prime gauges in the evaluation of the performance of any digital system. To conclude with the results, both controllers consume comparable power while DDR controller takes extra cell area. As device integration doubles in every 18 months cell area will not be an overhead. And power efficiency can be improved applying low power practices. This work assesses the two prominent memory controllers in terms of power and area and results can be used to improve the implementation practices.

Future work would be verification of these memory controllers using SystemVerilog. SystemVerilog is an industry standard verification language as it ensures 100% functional and code coverage in any design. Other value addition would be application of low power design techniques in the memory controller design. Low power methods greatly reduce the overall power intake of the system.

References

[1] "SDR SDRAM Controller White Paper", *Lattice Semiconductor Corporation, Reference Design*: December 2012

[2] DDR SDRAM Controller White Paper", *Lattice Semiconductor Corporation, Reference Design*: RD1020, April 2004

[3] DDR SDRAM Controller White Paper", *Lattice Semiconductor Corporation, Reference Design*: RD1020, April 2004.

[4] "JESD79C" *JEDEC Solid State Technology Association* 2003

[5] B. Jacob et al., "Memory Systems: Cache, DRAM, Disk", Morgan Kaufmann Publishers, 2007

[6] "SoC Encounter RTL-to-GDSII System", *Full-chip implementation in a single system*, 2012.

[7] Michael John Sebastian Smith, "Application Specific Integrated Circuits", Pearson Education, 2008.

ISSN: 2278 – 7798

[8] J. M. Rabaey and M. Pedram, "Low Power Design Methodologies", Boston: Springer, 1996

[9] Pan Guangrong Feng Da ; Wang Qin ; Qi Yue ; Yu Meiqiang "Performance exploration and optimization of SDRAM-controller architecture on SDRAM access" WRI World Congress on Computer Science and Information Engineering, 2009

[10] Zhang Yu Ling Ming ; Pu Hanlai ; Zhou Fan "Performance exploration and optimization of SDRAM-controller architecture on SDRAM access" ASICON 2005. 6th International Conference on ASIC, 2005.

[11] Jiayi Zhu Peilin Liu ; Dajiang Zhou" An SDRAM controller optimized for high definition video coding application" IEEE International Symposium on Circuits and Systems, 2008. ISCAS 2008.

[12] Miura, S. Ayukawa, K. ; Watanabe, T. "A dynamic-SDRAM-mode-control scheme for low-power systems with a 32-bit RISC CPU " International Symposium on Low Power Electronics and Design, 2001

[13] Chen Shuang-yan Wang Dong-hui ; Shan Rui ; Hou Chao-huan "An innovative design of the DDR/DDR2 SDRAM compatible controller" 6th International Conference On (Volume:1)ASIC, 2005. ASICON 2005.

[14] Makam, D. Jayashree, H.V." An innovative design of the DDR/DDR2 SDRAM compatible controller" International Conference on Nanoscience, Engineering and Technology (ICONSET), 2011

[15] Qiu Daqiang Hu Bing ; Li Dandan "Design of SDRAM Controller in High-Speed Data Acquisition Based on PCI Bus" 8th International Conference on Electronic Measurement and Instruments, 2007. ICEMI '07.

[16] Pei-Jun Ma Ling-Fang Zhu ; Kang Li ; Jia-Liang Zhao ; Jiang-Yi Shi "The application and optimization of SDRAM controller in multicore multithreaded SoC" 10th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), 2010

[17] Jian Qituo Liu Liansheng ; Peng Yu ; Liu Datong "Optimized FPGA-based DDR2 SDRAM controller" IEEE 11th International Conference on Electronic Measurement & Instruments (ICEMI), 2013

[18] Bayliss, S. Constantinides, G.A." Methodology for designing statically scheduled application-specific SDRAM controllers using constrained local search" International Conference on Field-Programmable Technology, 2009. FPT 2009.

[19] Heithecker, S. Ernst, R." Traffic shaping for an FPGA based SDRAM controller with complex QoS requirements " . 42nd Proceedings on Design Automation Conference, 2005.

[20] Akesson, B. Goossens, K. ; Ringhofer, M. "Predator: A predictable SDRAM memory controller" 5th IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2007

[21] Goossens, S. Kuijsten, J. ; Akesson, B. ; Goossens, K. "A reconfigurable real-time SDRAM controller for mixed time-criticality systems" International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2013

[22] Miura, S. Akiyama, S. "A memory controller that reduces latency of cached SDRAM" IEEE International Symposium on Circuits and Systems, 2005. ISCAS 2005.

[23] Li Wang Ju Wang ; Qian Zhang" Design and implementation of DDR SDRAM controller based on FPGA in satellite navigation system " IEEE 11th International Conference on (Volume:1) Signal Processing (ICSP), 2012

[24] Goossens, S. Kouters, T. ; Akesson, B. ; Goossens, K. "Memory-map selection for firm real-time SDRAM controllers " Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012

[25] Szymanski, T. Kielbik, R. ; Napieralski, Andrzej "SDRAM controller for real time digital image processing systems " Proceedings of the 6th International Conference. The Experience of Designing and Application of CAD Systems in Microelectronics, 2001. CADSM 2001.

[26] Kahng, A.B. Srinivas, V." Mobile system considerations for SDRAM interface trends" 2011 13th International Workshop on System Level Interconnect Prediction (SLIP)

[27] Chen Chao Xu Weizhang ; Yang Zhan Xin "Design and Implementation of the Time-Partition SDRAM Controller" International Conference on Management and Service Science, 2009. MASS '09.

Pavankumar NC received the Bachelors in Engineering in Electrical and Electronics from BVB College of Engg. & Tech. Hubli in 2011. He is currently an M Tech Scholar in BMS College of Engineering, Bangalore. His areas of interests are Low Power VLSI Design, Mixed Signal Design and Design Automation.

Jeeru Dinesh Reddy received the B Tech in Electronics & Communication from Andhra University Vizag, Andhra Pradesh in 2003 and M Tech in VLSI Design from P S G College of Technology Coimbatore, Tamilnadu in 2005. Presently he is working as an Asst. Professor in the Dept. of ECE, BMS College of Engineering, Bangalore. He worked in VLSI Industry for a year. His areas of interests are Mixed Signal Design, Digital System Design and Verification.

ISSN: 2278 – 7798