ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

DESIGN AND VERIFICATION OF DDR SDRAM MEMORY CONTROLLER USING SYSTEMVERILOG FOR HIGHER COVERAGE

[1]Shaik Suhanaajmi, [2]Dr.P.Sreelakshmi, [3]Mrs.K.Gayathri

[1]M.Tech Scholar, Dept of ECE, Audisankara College of Engineering and Technology, Gudur, Andhra Pradesh.

[2]Associate Professor,Dept of ECE, Audisankara College of Engineering and Technology, Gudur, Andhra Pradesh.

[3]Assistant professor, Dept of ECE, Audisankara College of Engineering and Technology, Gudur, Andhra Pradesh.

[1]suhanaajmishaik@gmail.com , [2] sree1612@gmail.com , [3] gayireddy22@gmail.com

ABSTRACT: Synchronous DRAM (SDRAM) has become a mainstream memory of choice in design due to its speed, burst access and pipeline features. For high-end applications using processors, the interface to the SDRAM is supported by the processor's built-in peripheral module. However, for other applications, the system designer must design a controller to provide proper commands for SDRAM initialization, read/write accesses and memory refresh. DDR SDRAM uses double data rate architecture to achieve high-speed data transfers. DDR SDRAM (referred to as DDR) transfers data on both the rising and falling edge of the clock. This DDR controller is typically implemented in a system between the DDR and the Processor. In this paper, the implementation has been done in Verilog by using Xilinx ISE 14.5.

I. INTRODUCTION

Image processing data pipelines require large amounts of buffered data, making memory selection critical in digital imaging systems. Memory ICs temporarily store data for processing pipelines, and different memory technologies offer different advantages and disadvantages. Static RAM (SRAM) is fast and consumes less power, but its low density and high transistor count increase cost and limit storage capacity. In contrast, Dynamic RAM (DRAM) provides higher density and lower cost, although it is comparatively slower and requires higher power consumption. Due to these characteristics, DRAM is widely preferred as the main memory in imaging and embedded systems.Earlier DRAM technologies such as Fast Page Mode DRAM (FPM DRAM) and Extended Data Out DRAM (EDO DRAM) were asynchronous and comparatively slow. Modern systems mainly use Synchronous DRAM (SDRAM), Double Data Rate (DDR) SDRAM, and RAMBus DRAM (RDRAM). SDRAM operates synchronously with the system clock, reducing latency and simplifying signal interfacing compared to previous DRAM technologies. However,

SDRAM is gradually being replaced by DDR SDRAM because DDR transfers data on both the rising and falling edges of the clock, effectively doubling the bandwidth while maintaining lower cost than RDRAM. Although RDRAM offers high bandwidth through a narrow 16-bit bus operating up to 800 MHz, its proprietary nature limits widespread adoption.DDR SDRAM has become highly suitable for embedded systems such as signal processing, networking, and image/video processing applications that require high-speed and cost-effective memory solutions. DDR SDRAM uses a 2n-bit prefetch architecture, transferring data on both clock edges to achieve data rates twice the clock frequency. Programmable burst lengths, auto-precharge functionality, and reduced power consumption further improve performance. Due to tight timing requirements, DDR systems often use phase-locked loops (PLLs) and calibration techniques for accurate operation.Modern Field Programmable Gate Arrays (FPGAs) from companies such as Xilinx and Altera now support high-speed DDR SDRAM interfacing. Advances in FPGA density, complexity, and performance have made them comparable to ASICs for demanding applications. Features such as SSTL-II compatible I/Os, DDR registers, PLLs, and delay chains enable efficient DDR memory controller implementation. FPGA-based IP cores provide memory controller solutions capable of initialization, refresh handling, and command translation for DDR SDRAM devices.A DDR SDRAM controller acts as an interface between the memory module and the embedded system. It performs initialization, refresh operations, and high-speed read/write command handling. Implemented commonly in Verilog HDL, the controller translates processor requests into DDR command sequences. Using Direct Memory Access (DMA), the DDR controller enables faster bulk data transfer than processor-controlled or interrupt-driven methods, significantly improving system throughput.RAM is generally classified into Static RAM (SRAM) and Dynamic RAM (DRAM). SRAM uses flip-flop structures with four to six transistors per memory cell and does not require refreshing, making it faster but more expensive and less dense. DRAM uses capacitors and transistors in each cell, requiring periodic refreshing to maintain stored data. Although slower, DRAM provides higher storage density and lower cost, making it suitable for system memory applications.Synchronous DRAM (SDRAM) organizes memory into rows, columns, and banks controlled through Row Address Select (RAS) and Column Address Select (CAS) signals. SDRAM supports pipelined operations synchronized with the system clock and requires periodic refresh cycles. SDR SDRAM transfers data only on the rising clock edge, whereas DDR SDRAM transfers data on both edges, doubling the transfer rate. Advanced versions such as DDR2 and DDR3 further improve performance and efficiency.Fault tolerance is another important consideration in modern memory systems, especially in aerospace, medical, defense, and safety-critical applications. Transient faults caused by cosmic rays, radiation, electromagnetic interference, and technology scaling can corrupt memory data and system

functionality. Examples include aircraft control failures and medical device malfunctions caused by single-bit memory errors. To improve reliability, designers use fault-tolerance techniques such as Triple Modular Redundancy (TMR) and Finite State Machine (FSM) encoding. Formal verification methods, including symbolic analysis and theorem proving, are used to ensure correctness and reliability under specified fault conditions.

This work focuses on gate-level circuit descriptions consisting of logic gates and flip-flops. Gate-level design simplifies formal verification, prevents synthesis optimization issues, and allows integration with commercial logic synthesis tools. The objective is to summarize FPGA design techniques for implementing high-speed DDR SDRAM controllers, describe memory controller architectures compatible with modern FPGAs, and discuss timing requirements necessary for reliable high-speed operation.

II. LITERATURE REVIEW:

The literature survey focused on storage architectures, flash memory systems, SDRAM controllers, routing algorithms, and fault-tolerant techniques for high-speed embedded applications.Sang-Woo Jun and Ming Liu et al. [1] proposed BlueDBM, a high-performance distributed flash storage architecture using FPGA-based flash controllers and PCIe interfaces. The design achieved high throughput and low latency through inter-FPGA communication and hardware acceleration.Duncan G. Elliott et al. [5] introduced Computational RAM (C-RAM), a processor-in-memory architecture that combines memory and SIMD processing to improve bandwidth and parallel computation while maintaining DRAM compatibility.Kermin Fleming et al. [6] developed a method for partitioning complex designs across multiple FPGAs using latency-insensitive communication links. Their approach improved scalability, synthesis time, and performance for DSP applications.

Flash Storage Architectures

Yeonseung Ryu et al. [9] proposed Filtering Flash Translation Layer (FFTL), which separates metadata and user data using block-level and page-level mapping to improve NAND flash efficiency and reduce unnecessary writes.Yang Ou et al. [10] designed a scalable multichannel NAND flash controller supporting burst operations and error correction. The architecture improved performance significantly through channel-level parallelism.

SDRAM Controllers

Jordi Sempere Agullo et al. [11] implemented a high-speed FPGA-based DDR SDRAM controller for image processing applications using dual clocks and optimized timing techniques.Deepali Sharma et al. [12] designed a DDR SDRAM controller with finite state machines for initialization, refresh, read/write operations, and signal generation.Li Wang, Ju Wang, and Qian Zhang [13] proposed an FPGA-based DDR SDRAM controller with improved read/write scheduling to increase processing efficiency.Santhoshima K.G et al. [14]

implemented an optimized DDR SDRAM controller operating above 150 MHz with reduced power consumption and efficient data transfer.

Routing Algorithms

Parag Parandkar et al. [17] explained deterministic and adaptive routing algorithms used in Network-on-Chip systems. XY routing is deterministic and deadlock-free, while Odd-Even routing is adaptive and improves flexibility.Shubhangi D. Chawade [18] compared XY, Odd-Even, and DyAD routing algorithms based on latency, throughput, and power consumption. DyAD combined advantages of both deterministic and adaptive routing.

Fault Tolerance

Technology scaling has increased susceptibility to soft errors caused by radiation and transient faults. Triple Modular Redundancy (TMR) and Error Correcting Codes (ECC) are commonly used protection techniques.Other methods such as Duplication with Comparison (DMR), self-checking circuits, RAZOR architecture, checkpointing, and rollback recovery improve system reliability in safety-critical applications. These techniques help detect and recover from transient faults while maintaining correct system operation.

III. EXISTING METHOD

A. CELL SCHEMATICS

1) Nwise CELL SCHEMATIC Fig. 2(a) shows the circuit structure of the Nwise cell [55]. A cross-coupled pair consisting of two NMOS transistors (N1 and N2) provides the main storage part of the cell while another cross-coupled pair of two NMOS transistors (N5 and N6) acts as the backup part. Therefore, the Nwise cell has four storage nodes Q, QB, P, and PB, where Q and QB keep the stored data correctly and P and PB provide the redundant stored data for Q and QB. The access transistors, N7 and N8, connect the bit-lines, BL and BLB, to the main storage nodes Q and QB, respectively. N7 and N8 are controlled by the word line (WL). Hence, when WL is in high mode, transistors N7 and N8 are turned on, and read/write operations can be done. Transistors P1-N4 and P2-N3 provide two feedback paths for the Nwise cell and help the storage nodes recover to their initial value after particle strikes and secure robustness under high-radiation conditions.

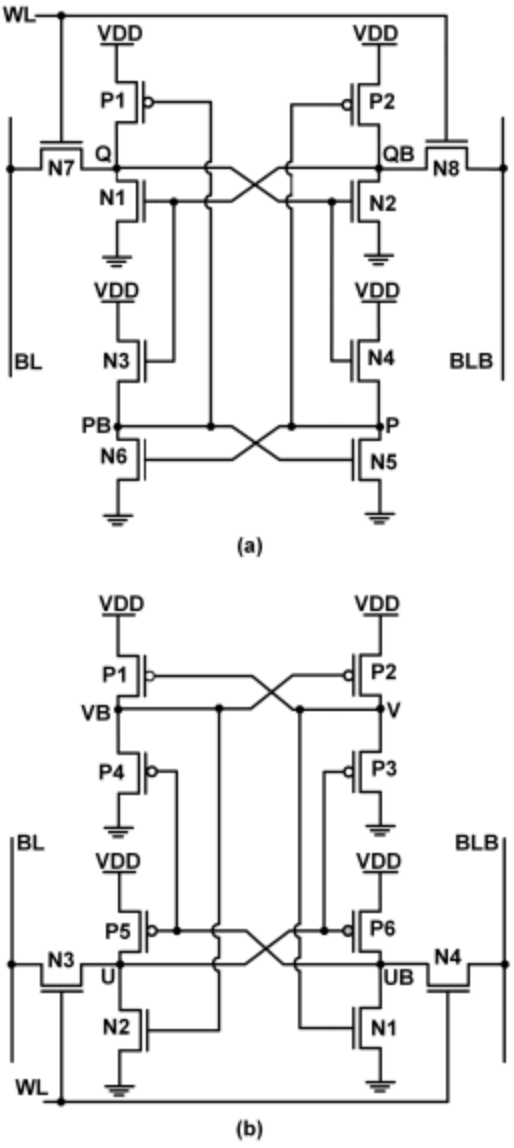


FIGURE 2. The circuit structure of (a) the Nwise cell [55] and (b) the new proposed Pwise cell.

2) Pwise CELL SCHEMATIC Fig. 2(b) shows the circuit structure of our proposed Pwise cell. Two cross-coupled PMOS transistor pairs, P1-P2 and P5-P6, shape two latches where one creates data redundancy for the other. Therefore, the Pwise cell has four storage nodes U, UB, V, and VB, where V and VB provide the redundant stored data for U and UB. Furthermore, the access transistors, NMOS pass gates N3 and N4, which are controlled by the word line (WL), connect the bit-lines (BL and BLB) to the main storage nodes U and UB, respectively. Like the Nwise cell, two feedback paths, N1-P4 and N2-P3 are provided for the Pwise cell to help storage nodes recover to their initial value after particle strikes, securing robustness under high-radiation conditions. The Nwise and Pwise cell feedback structures consist of both NMOS and PMOS transistors to increase the tolerance capability as confirmed through our simulations reported in Section IV. The Nwise and Pwise cell structures have similarities with the circuit structure of RHBD proposed in [10] with some important differences: 1- RHBD consists of two cross-coupled transistor pairs, but unlike Nwise and Pwise, one is made of two NMOS transistors and the other of two PMOS transistors where the PMOS-transistor pair creates data redundancy for the NMOS-transistor pair. 2- Similar to Nwise and Pwise cells, RHBD is equipped with two feedback paths to enhance the tolerance capability, but the feedbacks consist of two PMOS transistors instead of one NMOS and one PMOS.

IV. PROPOSED METHOD

With the high increase in development of the semiconductor processor's industry, major importance given to operation speed and capacity of the memory device. The DDR controller is an extension of old traditional memory controller synchronous DRAM (SDRAM). The DDR controller is capable of

transacting the data on both rising and falling edges of the clock cycles. Thereby, double the transfer of data rate in the memory device. The cost of DDR memory is low, because it is most commonly used in PC's, storage and buffers. The DDR SDRAM controller's capability of data widths of 16, 32 and 64 bits and its throughput increases by using pipelining command and bank management techniques. Changes in pipelining command and bank management techniques enhance DDR SDRAM memory module transfer data twice as speed as SDRAM memory. For an example, using a data rate of 133 MHz, DDR SDRAM memory transfer data rate at 266 MHz, that it means twice the frequency of clock cycles[1][2]. Compared to SDR SDRAM, DDR SDRAM have the burst length of 2, 4 and 8, thus one address is required to access sequential address [3]. There are two types of memories that can be purchased, namely Single In-line Memory Modules (SIMMs) and Double In-line Memory Modules (DIMMs). When in SIMM or DIMM both are small circuit that is capable to fit into common standard memory socket available in PC's. In DIMMs has two leads compared to the SIMMs has only one lead. They are soldered to the motherboard to supply desired memory. SDRAM memory usually comes in DIMMs. DIMMs are classified based on the data transfer rate, such has one is Single Data Rate (SDR) and other called Double Data Rate (DDR). SDR SDRAM transaction of data only on the rising edge of the clock cycle whereas DDR SDRAM transaction of data on both the rising as well as on the falling edge of the clock cycles, At present, other faster versions of available DDR SDRAMs, are DDR2, DDR3 and DDR4 [4].

DDR SDRAM MEMORY CONTROLLER ARCHITECTURE

A. DDR SDRAM Functional Blocks

The DDR SDRAM memory controller centre

consists of four primary components that are basically SDRAM controller, control interface, command and data path module. The one main DDR SDRAM controller module is the highlevel module. That mainly contains three lower level modules and gives out the entire outline to give exact outcomes. The Control Interface module of SDRAM makes an approach summons and that are related to memory addresses from the host and translating that charges and Transferring to the module called Command. The module command acknowledges from the SDRAM signals and address from the module called control interface module for creating the right way of building orders to the SDRAM controller. The module data path manages or makes sure of the information way activities WRITE and READ orders. The SDRAM control interface is the best level module that additionally instantiates two phase locked loops that are effectively use as a part of CLOCK_LOCKS [3].

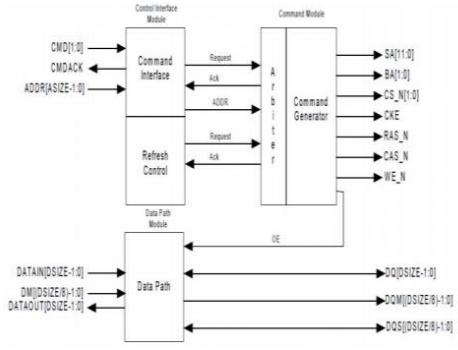


Figure 1: Block Diagram of proposed functional DDR SDRAM Memory Controller core.

B. DDR SDRAM Control Interface Module

The module control Interface is the primary module and it included the finite sate machine a programmable 16-bit counter, which is used to perform the auto refresh operation. The controller interface module accepts the commands from the processor and it will decode them and send the decoded WRITEA, NOP, READA, PRECHARGE, REFRESH, and LOAD_MODE signal commands are passed to the module command. There are two registers used, LOAD_REG1 and LOAD_REG2 signals are decoded and utilized to load the register REG1 and register REG2 from address ADDR signals The DDR SDRAM uses a different means separate clock signals: one is CLK and other is CLK` (the crossed of CLK going high and CLK` going low is just taken as the positive edge of the clock). The commands are taken down only on the positive edge of the clocks. The two signals READ and WRITE are the basic commands are used to access the SDRAM memory. The DDR SDRAM will support the burst length of 2, 4, or 8 locations for programmable READ/WRITE operations. When access to SDRAM started at any location and it will continue until it reaches the burst length. The READ and WRITE signal operation started by sending the ACTIVATE commands. The controller Interface consists of a finite state machine shown in figure 2. The initial state is considered as ideal state and the next state of the controller can be in either PRECHARGE, REFRESH, LOAD_MODE, or ACTIVATE state those all states depending on the request from the processor [5].

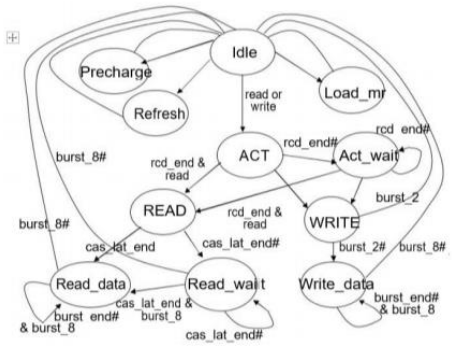


Figure 2: Finite state machine diagram

Those lines without indications in the figure 2 indicate an automatic sequence. The READ and WRITE command operation are followed by command ACT. The ACT command signal is used for particular bank to open a row, and where PRECHARGE command signal is used to close a bank which is open and if the open bank is other than that we want to access. When a WRITE command is passed and executed, the controller will first go to the ACT command state and then tit will enter to the WRITE state. The WRITE command operation will be executed till the burst length is completed or its Up to the burst terminate is inserted. When the READ operation is executed, the initial address is registered and during the READ operation the data will comes or arrival time is 1-3 clock cycles later on the data bus and this delay is because of the time that is actually needed to read the internal DRAM memory. This delay time is CAS latency. Once on completion of the burst READ and WRITE operations the controller will go settle back to the initial IDLE state.

C. DDR SDRAM Command Module

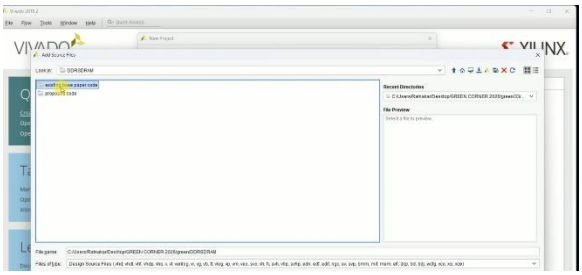
The command module mainly consists of an arbiter, multiplexer and three shift registers. An arbiter is used while in case if multiple

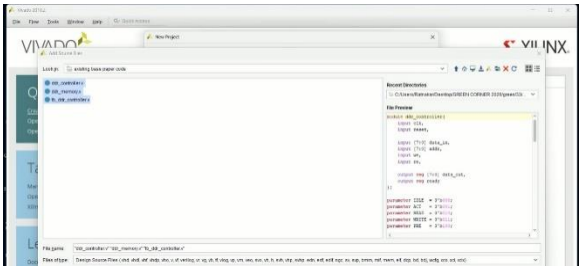
requests come from different-different devices. If both high- priority and low-priority device need to transfer data at the same time then high priority device gets higher priority. The low priority device not able to transfer data until the high priority device has transferred all its data. In my work high priority is REFRESH control operation. So, if the REFRESH operation is executing on and thereby command from the host comes, then the controller will first continue the refresh operation to be done and it hold command requests by not assert this command to the command acknowledge (CMDACK). Once refresh operation is finished the command module will start performing the requested commands from the host. Module inside command module is the shift registers and is used to maintain the timing between the command signals assigned to the SDRAM memory controller. Other module it contains is multiplexer which is used to multiplexing the address. Where the row address is multiplexed to SDRAM during when the signal ACTIVATE (RAS) command. Other column portion is also multiplex the address to SDRAM address outputs happen during a signal READ (WRITE) command [3][6].

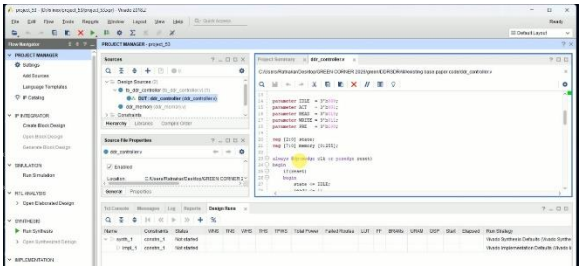
D. DDR SDRAM Datapath module

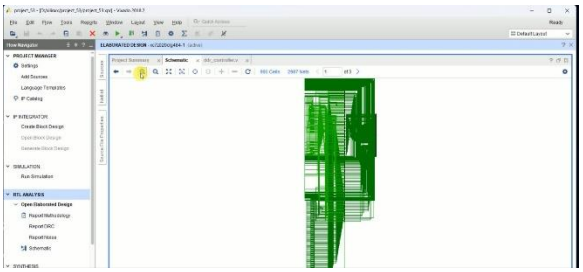
The basic function of module data path is to store the write data and calculation for the value of read data path. Its main operation is to data generation and sampling of data to DDR. When the READ operation is executed, the data is sampled from the module data path. Data will synchronize the clock signal and transacted as one-word per clock cycle. When the WRITE operation is executed, the user interface gets the data at one word per clock cycle. The data will be resynchronized by the module data path and it will transfer at a rate called DDR double data rate. The module data path performed as an interface between the processor and the memory. The data path module performs the two operations called latching of the data and dispatching of the data between the host and Memory. The module data path has tristate buffer which is controlled by OE signal, that comes from the module command. Whatever the data to be written is received on the DATAIN pin during the WRITE operation and during the READA operation data is provided on the DATAOUT pin [3]

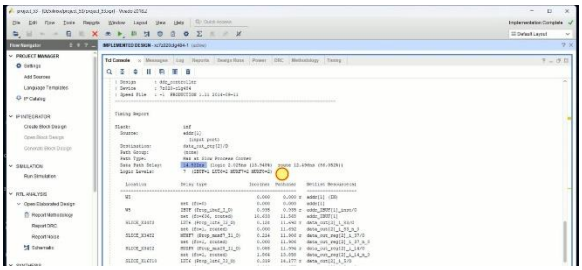
V. SIMULATION RESULTS

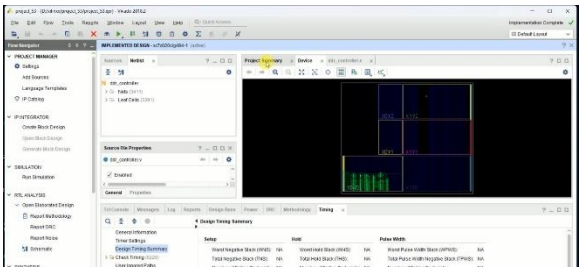


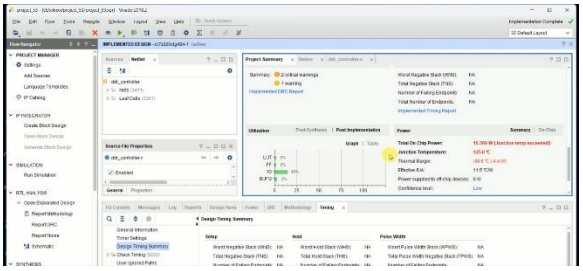


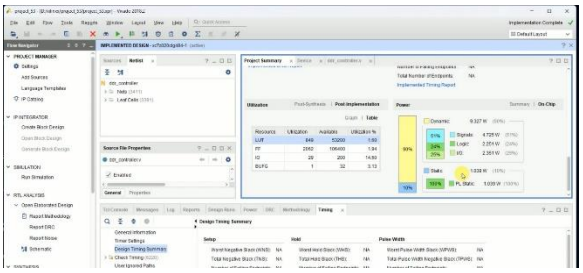


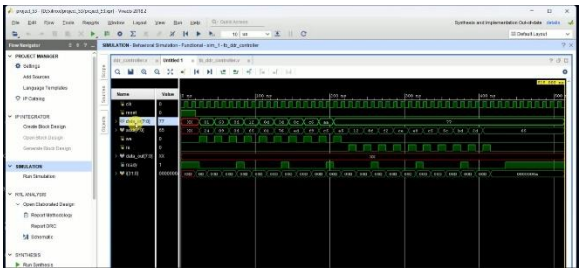












VI. CONCLUSION

The similarities between SDR and DDR SDRAM provide the DRAM manufacturer cost advantages and assure high production yields. These similarities also help the designer to better understand DDR and allow the most optimal design techniques from previous designs to be incorporated into the new DDR platforms. Although the addressing schemes, layout requirements, and device configurations are much the same for DDR, the performance gains are remarkable. For example, the power consumption for DDR is significantly less than for a comparable SDR device, yet peak transfer rates can exceed 2.1 GB/s for a standard x64 DDR. Traditional single data rate (SDR) SDRAM architectures are performance-limited in their interfaces; therefore, DDR SDRAMs were introduced as an enhancement. While most of the addressing and command control interface is identical, the fundamental difference is in the data interface. Thus, this enables high-speed operation as the internal column accesses are half the frequency of the external data transfer rate.

DDR SDRAM is a volatile and complex memory device. When the power is removed, all contents and operating configurations are lost. Each time the memory is powered up, the device requires a defined procedure to

initialize the internal state machines and to configure the various user-defined operating parameters. This project concentrates on the flow for the initialization sequence and the configurable device parameters.

VII. REFERENCES

[1] R. C. Baumann, "Soft errors in advanced computer systems," IEEE Design Test. Comput., vol. 22, no. 3, pp. 258–266, May/Jun. 2005.

[2] N. P. Rao and M. P. Desai, "A detailed characterization of errors in logic circuits due to single-event transients," in Proc. Euromicro Conf. Digit. Syst. Design, Funchal, Portugal, 2015, pp. 714–721.

[3] B. Narasimham et al., "Characterization of digital single event transient pulse-widths in 130-nm and 90-nm CMOS technologies," IEEE Trans. Nucl. Sci., vol. 54, no. 6, pp. 2506–2511, Dec. 2007.

[4] M. P. Baze and S. P. Buchner, "Attenuation of single event induced pulses in CMOS combinational logic," IEEE Trans. Nucl. Sci., vol. 44, no. 6, pp. 2217–2223, Dec. 1997.

[5] S. Buchner, M. Baze, D. Brown, D. McMorrow, and J. Melinger, "Comparison of error rates in combinational and sequential logic," IEEE Trans. Nucl. Sci., vol. 44, no. 6, pp. 2209–2216, Dec. 1997.

[6] J. Benedetto et al., "Heavy ion-induced digital single-event transients in deep submicron processes," IEEE Trans. Nucl. Sci., vol. 51, no. 6, pp. 3480–3485, Dec. 2004.

[7] M. A. Bajura et al., "Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs," IEEE Trans. Nucl. Sci., vol. 54, no. 4, pp. 935–945, Aug. 2007.

[8] T. Bonnoit, M. Nicolaidis, and N.-E. Zergainoh, "Using error correcting codes without speed penalty in embedded memories: Algorithm, implementation and case study," J. Electron. Test., vol. 29, no. 3, pp. 383–400, Jun. 2013.

[9] P. Papavramidou and M. Nicolaidis, "Test algorithms for ECC-based memory repair in ultimate CMOS and post-CMOS," IEEE Trans. Comput., vol. 65, no. 7, pp. 2284–2298, Jul. 2015.

[10] R. Vemu, A. Jas, J. A. Abraham, S. Patil, and R. Galivanche, "A lowcost concurrent error detection technique for processor control logic," in Proc. DATE, Munich, Germany, 2008, pp. 897–902.

[11] S. Ghosh, N. A. Touba, and S. Basu, "Synthesis of low power CED circuits based on parity codes," in Proc. 23rd IEEE VLSI Test Symp., May 2005, pp. 315–320.

[12] V. Sapozhnikov, V. Sapozhnikov, D. Efanov, and A. Blyudov, "On the synthesis of unidirectional combinational circuits detecting all single faults," in Proc. EWDTS, 2014, pp. 1–8.

[13] C. A. Lisboa, F. L. Kastensmidt, E. H. Neto, G. Wirht, and L. Carro, "Using built-in sensors to cope with long duration transient faults in future technologies," in Proc. ITC, 2007, pp. 1–10.

[14] R. P. Bastos, F. S. Torres, G. Di Natale, M. Flottes, and B. Rouzeyre, "Novel transient-fault detection circuit featuring enhanced bulk builtin current sensor with low-power sleep-mode," J. Microelectron. Rel., vol. 52, nos. 9–10, pp. 1781–1786, 2012.

[15] M. Nicolaidis, "Double-sampling design paradigm—A compendium of architectures," IEEE Trans. Device Mater. Rel., vol. 15, no. 1, pp. 10–23, Mar. 2015.

[16] D. Ernst et al., "Razor: A low-power pipeline based on circuit-level timing speculation," in Proc. MICRO, 2003, pp. 7–18.

[17] S. Das et al., "RazorII: In situ error detection and correction for PVT and SER tolerance," IEEE J. Solid-State Circuits, vol. 44, no. 1, pp. 32–48, Jan. 2009.

[18] M. R. Choudhury and K. Mohanram, "Low cost concurrent error masking using approximate logic circuits," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 8, pp. 1163–1176, Aug. 2013.

[19] M. Krsti´c, S. Weidling, V. Petrovic, and M. Goessel, "Improved circuitry for soft error correction in combinational logic in pipelined designs," in Proc. IOLTS, 2014, pp. 93–98.

[20] D. P. Siewiorek and R. S. Swarz, Reliable Computer Systems: Design and Evaluation. Newton, MA, USA: Digital, 1992.

[21] D. K. Pradhan, Fault-Tolerant Computer System Design. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.