NXP Women In Tech Batch - II

Design of Synchronous Dual Port RAM using Verilog HDL

Project Report

Group Members: Aishani
Amritha Anujan
Ananya Singhal
Sameeksha

Mentor: Mr. Sanjay Kumar Saini

I. Introduction

Dual-port RAM (DPRAM) has become integral to digital systems requiring high-speed and concurrent data access, such as multi-core processors and parallel processing applications. Unlike single-port memory, DPRAM facilitates simultaneous read and write operations on two separate ports, addressing the performance bottlenecks of traditional memory architectures.

The demand for high-performance memory in real-time processing and high-speed data applications necessitates flexible, synchronous memory design. Verilog HDL offers a robust framework for modeling DPRAM at the RTL level, providing a structured approach to handle concurrent data access. Through this project we aim to design a Verilog-based synchronous DPRAM module, ensuring optimized memory usage, priority-based conflict resolution, and efficient simulation.

II. Background Research

Recent studies on DPRAM have explored synchronous and asynchronous designs for FPGA-based implementations. Some studies, like Muehlegg and Schuetz et al. [1], introduced flexible DPRAM architectures, while others like Pandey et al. [2] focused on energy-efficient synchronous memory design with clock gating. There are works like Aditi et al. [3] which explore the System Verilog and UVM implementation of Dual Port RAM. There has been advancement in various scales and aspects in the dual-port memory architecture. In this work, we intend to get familiarized with the basic structure of a synchronous dual-port memory architecture using its Verilog implementation and further explore the various ways it can be improved and designed.

III. Design Methodology

The design of the Dual-Port RAM (DPRAM) module leverages a synchronous memory architecture that supports independent and concurrent access to two memory ports. The following key design elements ensure efficient and reliable operation:

1. Dual-Port Structure:
   The DPRAM module is structured to allow simultaneous operations on two ports (Port A and Port B). Each port has its own clock, write enable signal, data input, and address inputs. This dual-port configuration allows for independent read and write operations on each port without data conflicts, crucial for applications requiring parallel access.
2. Synchronous Memory Access:
   Each port operates synchronously with its dedicated clock signal. The synchronous design ensures data stability and predictability, as read and write operations are aligned with the rising edge of their respective clocks. This setup allows the memory module to be integrated into larger systems with timing constraints.

3. Conflict-Free Operation with Write Enable Control:
   To prevent conflicts when both ports access the same memory location, the design employs write enable signals for each port. These signals control when data is written to the memory, allowing selective access and ensuring data integrity when both ports attempt simultaneous operations.
4. Read-After-Write Functionality:
   The design ensures that each port can read data immediately after a write operation, thanks to dedicated registers that store the most recent address accessed by each port. This read-after-write capability enhances the module's usability in high-performance systems.

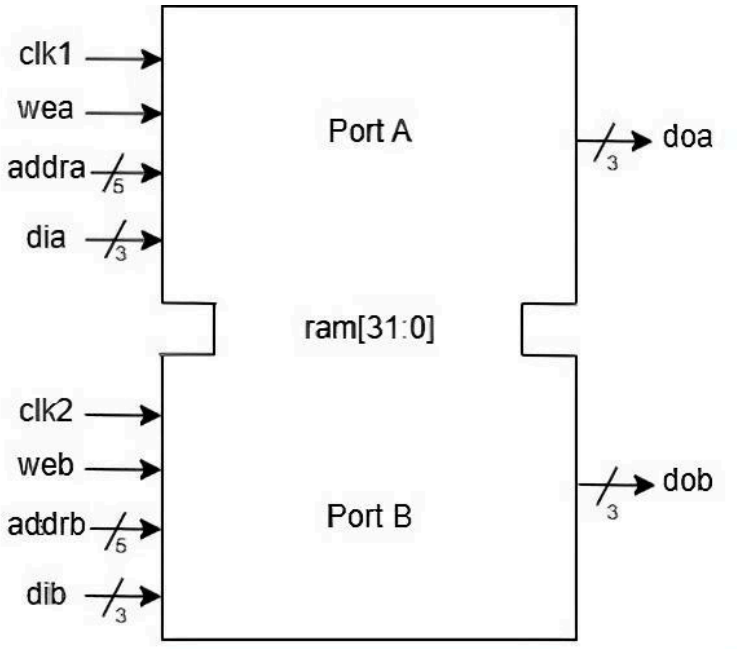

Figure 1. Block Diagram of the design

IV. Implementation and Testbench

The **`raminfr`** Verilog module implements the design principles outlined above, translating each aspect into HDL code:

1. Memory Array Definition:
   The module includes a 32-entry by 4-bit register array (**`ram`**), representing the DPRAM's core memory. This array is accessible through both ports, with each entry addressable via a 5-bit input (**`addra`** for Port A and **`addrb`** for Port B).
2. Synchronous Read and Write Operations:
   The blocks operate on the positive edge of their respective clocks (**`clk1`** for Port A and **`clk2`** for Port B). When the write enable signal (**`wea`** or **`web`**) is active, the data input (**`dia`** or **`dib`**) is stored at the specified address (**`addra`** or **`addrb`**). Additionally, the

current address is saved in **`read_addra`** or **`read_addrb`**, which drives the data output (**`doa`** or **`dob`**) for each port.

3. Data Output Logic:
   The outputs **`doa`** and **`dob`** are assigned using continuous assignments linked to **`ram[read_addra]`** and **`ram[read_addrb]`**, respectively. This ensures that each port's data output reflects the most recent data at its designated read address.

The test bench (**`top`**) verifies the DPRAM module's functionality by simulating various read and write scenarios:

1. Clock Signal Generation:
   A forever loop toggles the clock signals (**`clk1`** and **`clk2`**) every 5 time units, providing timing for each port to ensure synchronous operation during testing.
2. Initialization and Write Testing:
   The test bench initializes all signals to zero and sets the write enable signals (**`wea`** and **`web`**) high to test write functionality. It writes:
   - **`4'hA`** to address 6 via Port A.
   - **`4'hB`** to address 7 via Port B.
3. Data Verification and Simulation Control:
   The test bench includes **`$dumpfile`** and **`$dumpvars`** directives to generate a **`top.vcd`** file for waveform analysis, allowing verification of data integrity and concurrent read/write operations. The simulation runs for 100 time units and then terminates with **`$finish`**.

The implementation and test bench confirm the DPRAM's dual-port capabilities, allowing independent and synchronous access to memory locations, demonstrating the module's suitability for parallel processing applications.

V. Observation and Results

Code: https://github.com/Sameeksha2323/Dual-Port-RAM

The following observations were made from the project:

1. Clock Synchronization:
   - Each port operates on different clocks (**`clk1`** for Port A and **`clk2`** for Port B), ensuring that the dual-port RAM can handle operations independently and synchronously.
2. Signal Initialization and Behavior:
   - Initializing signals to zero and setting appropriate values for testing ensured that write and read operations worked as expected, with the written data appearing correctly on the output.

3. Data Output Consistency:
    - The outputs (**doa** and **dob**) reliably reflected the most recent data at their respective read addresses, confirming the correct implementation of continuous assignments for data output.
4. Simulation Time Constraints:
    - The simulation was conducted over 100 time units, which was sufficient to validate the primary operations of the module without running into timing issues.
5. Parallel Processing Capabilities:
    - The dual-port design demonstrated its suitability for parallel processing applications, allowing independent access to memory locations from both ports without interference.

The following results were derived from the simulation performed:

1. Initialization:
    - Both clock signals (**clk1** and **clk2**) are initialized to **0** and toggle every 5 time units.
    - Initial values for **wea**, **web**, **dia**, **dib**, **addra** and **addrb** are set to zero.
2. Write Operations:
    - In the initial block, write operations are performed by setting **wea** and **web** to **1**.
    - Data 4'ha is written to address **6** (**addra**), and data **4'hb** is written to address 7 (**addrb**).
3. Read Operations:
    - Read operations occur at each positive edge of **clk1** and **clk2**.
    - The values stored at addresses **6** and **7** are read and assigned to **doa** and **dob**, respectively.
4. Simulation Output:
    - The simulation ends after 100 time units, capturing the results in the **top.vcd** file.

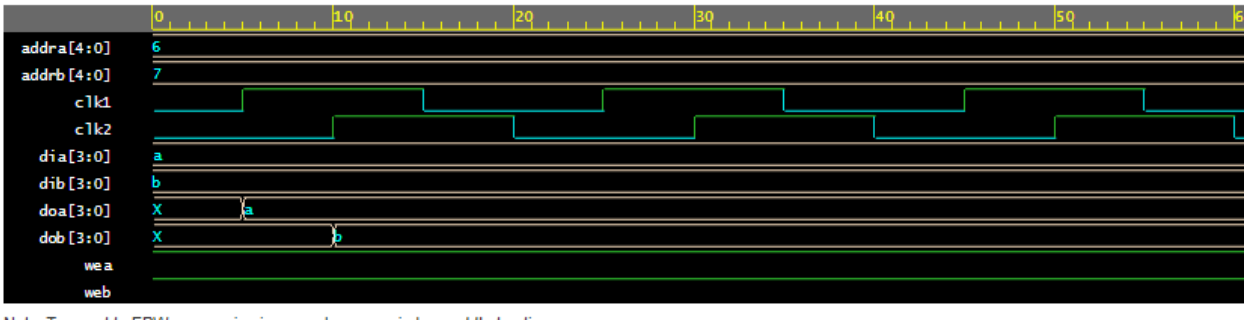

Figure 2. Simulation Waveforms of Design

VI. Conclusion

For our project on designing a Verilog-based synchronous Dual-Port RAM (DPRAM), we successfully implemented a memory module that allows concurrent data access through two independent ports, significantly improving data throughput in comparison to single-port memory architectures. The Verilog-based design framework provided an efficient and structured approach for modeling DPRAM at the RTL level, addressing performance bottlenecks and meeting high-speed data access needs.

Overall, the project demonstrated the practical advantages of synchronous DPRAM in applications requiring real-time, concurrent data processing, and served as a valuable exercise in memory optimization. Future work may involve refining these aspects to further increase efficiency, positioning DPRAM as a viable solution for advanced multi-core and parallel processing applications.

VII. References

[1] Muehlegg, Franz, and Alfred Schuetz. "A highly flexible dual-port-RAM compiler." [Proce edings] EURO ASIC90. IEEE, 1990.

[2] Pandey, Aayush, P. Bhargava, and Sowmya Madhavan. "Design and Implementation of Synchronous Dual-Port Memory." 2024 Third International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE). IEEE, 2024.

[3] Aditi, Pawan Kumar Dahiya. "Design and Verification of Dual Port RAM using System Verilog Methodology." International Journal for Research in Applied Science & Engineering Technology (IJRASET), IEEE (2019).