DAYANANDA SAGAR UNIVERSITY

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING SCHOOL OF ENGINEERING**

**DAYANANDA SAGAR UNIVERSITY KUDLU GATE**

**BANGALORE - 560068**



## MINI PROJECT REPORT

***ON***

# “Implementation of RAM & ROM”

**SUBMITTED TO THE 3rd SEMESTER DIGITAL ELECTRONICS & LOGIC DESIGN (19CS203) COURSE**

**BACHELOR OF TECHNOLOGY**

***IN***

### COMPUTER SCIENCE & ENGINEERING

***Submitted by***

Anand Singh Tanwar (ENG19CT0005)

Piyush Sharma (ENG19CT0025)

Sakar Dubey (ENG19CT0031)

Syed Suhail Salaha (ENG19CT0043)

***Under the supervision of***

### Mahesh Bharamashette

**DAYANANDA SAGAR UNIVERSITY**

### School of Engineering, Kudlu Gate, Bangalore-560068



**CERTIFICATE**

#### This is to certify that Anand Singh Tanwar, Piyush Sharma, Sakar Dubey and Syed Suhail Salaha bearing USN ENG19CT0005, ENG19CT0025, ENG19CT0031 and ENG19CT0043 has satisfactorily completed his/her Mini Project as prescribed by the University for the 3RD semester B.Tech. programme in Computer Science & Engineering for Digital Electronics & Logic Design (19CS203) course during the year 2020 at the School of Engineering, Dayananda Sagar University., Bangalore.

Date:

Signature of the faculty in-charge

|  |  |
| --- | --- |
| Max Marks | Marks Obtained |
|  |  |

Signature of Chairman Department of Computer Science & Engineering

**ACKNOWLEDGEMENT**

From the very core of our heart, we would like to express our sincere gratitude to **Mahesh Bharamashette** for his invaluable guidance, support, motivation and patience during the course of this mini- project work. We are always indebted to him for his kind support and constant encouragement.

We extend our sincere thanks to our **Chairman Dr. Sanjay Chitnis** who continuously helped throughout the project and without his guidance, this project would have been an uphill task.

It requires lots of efforts in terms of cooperation and support to fulfil various tasks involved during the project. We are always grateful to our peers and friends who have always encouraged us and guided us whenever we needed assistance.

Anand Singh Tanwar ENG19CT0005

Piyush Sharma ENG19CT0025

Sakar Dubey ENG19CT0031

Syed Suhail Salaha ENG19CT0043

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **SL.NO** | **TITLE** | **PAGE. NO** |
| 1. | Cover Page | 1 |
| 2. | Certificate | 2 |
| 3. | Acknowledgement | 3 |
| 4. | Content | 4 |
| 5. | Abstract | 5 |
| 6. | Introduction | 6 |
| 7. | Problem Statement | 7 |
| 8. | S/W & H/W Requirements | 8 |
| 9. | Logic Circuit | 9 |
| 10. | Verilog/VHDL Code | 10-13 |
| 11. | Output | 14 |
| 12. | Conclusion | 15 |
| 13. | References | 16 |

**ABSTRACT**

Memory is needed to store data and retrieve it when needed. Computers have two kind of memory; one is primary and another is secondary.  
Secondary memory is commonly known as just storage. It's used to store files when they are not needed by the processors. The examples are HDD, SSD, USB Drives, SD Cards etc.

Primary memory is needed to execute any program. Your program when executed loads itself into primary memory. When program runs, it needs to work on variables, objects, data structure etc. to store data. For example, if it needs to store value 10, it'll need an integer datatype while for storing text, it'll need a string data type. These data types store these values into primary memory. Apart from that some programs needs to access files which are also gets loaded into primary memory for execution and then after releasing them, they go back with changes (if any) to secondary memory.

Primary memory examples could be **RAM**, **ROM**, L1 & L2 Cache, Registers etc. Processor works as a primary memory too since they have limited storage too.

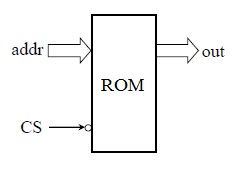
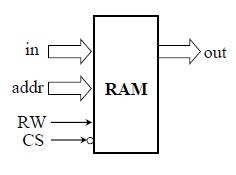
Now the question is why do we need Primary memory like RAM at all? Can't we just use Secondary memory for this?  
We can, in fact we do. When RAM ran out of space to open new files, the OS uses Hard disks as primary memory by creating page files on that. If you have seen page swap or page files in your system's hidden files then you have seen the proof.

Now why do we need it? We need processors to run the commands, however processors are really costly and they cannot hold much data in them. Processors are really fast, but we need to store the data processors are not currently using, even in the running program. Caches are faster kind of memory which can be used by Processors very easily. However, they are costly too (L1 is faster and costly than L2). Registers are next faster memory and RAM is the next fastest and the cheapest option.  
RAMs can't hold the data, so we need a persistent data storage such as HDDs. HDD works on magnetic resonance principles and are really slow. However, they are faster than Optical disks such as CDs and DVDs. SSDs are faster, but they still can't compete with RAMs. RAMs are electronic drives which can connect with Processors with ease.

Consider an example of slow-moving traffic. If the fastest car is behind slow moving cars, it will move as fast as the slowest car in the lane. That's why the slowness of HDDs affect the Processors' execution speed. Thus, we need RAM for this.

## INTRODUCTION

* **Read-only memory** (**ROM**) is a type of non-volatile memory used in computers and other electronic devices.
* **Random-access memory** (**RAM**) is a form of computer memory that can be read and changed in any order, typically used to store working data and machine code. RAM contains multiplexing and demultiplexing circuitry, to connect the data lines to the addressed storage for reading or writing the entry.

## PROBLEM STATEMENT

RAM and ROM are two types of memory which are used to store information.

Information stored on RAM are temporary and accounts as long as you keep your computer ON only. Once you shut down your computer, information stored on RAM is lost, Its Random-Access Memory.

Whereas, information stored on ROM are permanent and its Read Only Memory which means it is not lost even if you shut down your computer neither it could be modified easily.

RAM accounts for your memory needs to run the OS (Operating System) and other programmes on your processes while your system is awake.

ROM accounts for your memory needs to boot up your computer.

RAM keeps data only when your computer is awake which means it needs some little amount of power to work (Electricity)

ROM keeps data even in the powered off stage of your computer which means no power is required for data storage.

Both of these memories are required for the smooth functioning and running of a computer

## SOFTWARE AND HARDWARE REQUIREMENTS

### Software requirements:

* OS – Windows 7 and above/Ubuntu
* Programming Language – Verilog/VHDL
* IDE (Integrated Development Environment) – EDA Playground

### Hardware requirements:

## LOGIC CIRCUIT

## VERILOG/VHDL CODE

## RAM

## Design Code:

## module single\_port\_sync\_ram

## # (

## parameter ADDR\_WIDTH = 4,

## parameter DATA\_WIDTH = 32,

## parameter DEPTH = 16

## )

## 

## (

## input clk,

## input [ADDR\_WIDTH-1:0] addr,

## inout [DATA\_WIDTH-1:0] data,

## input cs,

## input we,

## input oe

## );

## 

## reg [DATA\_WIDTH-1:0] tmp\_data;

## reg [DATA\_WIDTH-1:0] mem [0:DEPTH];

## 

## always @ (posedge clk) begin

## if (cs & we)

## mem[addr] <= data;

## end

## 

## always @ (posedge clk) begin

## if (cs & !we)

## tmp\_data <= mem[addr];

## end

## 

## assign data = cs & oe & !we ? tmp\_data : 'hz;

## endmodule

## Test Bench Code:

## module tb;

## parameter ADDR\_WIDTH = 4;

## parameter DATA\_WIDTH = 16;

## parameter DEPTH = 16;

## 

## reg clk;

## reg cs;

## reg we;

## reg oe;

## reg [ADDR\_WIDTH-1:0] addr;

## wire [DATA\_WIDTH-1:0] data;

## reg [DATA\_WIDTH-1:0] tb\_data;

## 

## single\_port\_sync\_ram

## #(.DATA\_WIDTH(DATA\_WIDTH)) u0

## (

## .clk(clk),

## .addr(addr),

## .data(data),

## .cs(cs),

## .we(we),

## .oe(oe)

## );

## 

## 

## always #10 clk = ~clk;

## assign data = !oe ? tb\_data : 'hz;

## 

## integer i;

## initial begin

## $dumpfile("dump.vcd");

## $dumpvars(1);

## {clk, cs, we, addr, tb\_data, oe} <= 0;

## 

## repeat (2) @ (posedge clk);

## 

## for (i = 0; i < 2\*\*ADDR\_WIDTH; i= i+1)

## begin

## repeat (1) @(posedge clk) addr <= i; we <= 1; cs <=1; oe <= 0; tb\_data <= $random;

## end

## 

## for (i = 0; i < 2\*\*ADDR\_WIDTH; i= i+1)

## begin

## repeat (1) @(posedge clk) addr <= i; we <= 0; cs <= 1; oe <= 1;

## end

## 

## #20 $finish;

## end

## endmodule

## ROM

## Design Code:

## module simple\_rom(

## input [1:0] addr,

## output reg [7:0] data

## );

## 

## always @(\*)

## begin

## case (addr)

## 0: data = 8'd05;

## 1: data = 8'd10;

## 2: data = 8'd15;

## 3: data = 8'd20;

## endcase

## end

## endmodule

## Test Bench Code:

## module simple\_rom\_tb();

## 

## reg [1:0] addr;

## wire [7:0] data;

## 

## simple\_rom simple\_rom(

## .addr(addr),

## .data(data)

## );

## 

## initial begin

## $dumpfile("dump.vcd");

## $dumpvars;

## addr = 2'd3;

## #10;

## 

## addr = 2'd2;

## #10;

## 

## addr = 2'd1;

## #10;

## 

## addr = 2'd0;

## #10;

## 

## end

## endmodule

## OUTPUT :

## RAM

## 

## ROM:

## 

## CONCLUSION :

## Computer random access memory (RAM) is one of the most important components in determining your system's performance. RAM gives applications a place to store and access data on a short-term basis. It stores the information your computer is actively using so that it can be accessed quickly.

## ROM provides the necessary instructions for communication between various hardware components. As mentioned before, it is essential for the storage and operation of the BIOS, but it can also be used for basic data management, to hold software for basic processes of utilities and to read and write to peripheral devices.

## In this project we have seen the successful implement of RAM and ROM using Verilog code with output EPwaveform.

## REFERENCES

* <http://www.asic-world.com/verilog/veritut.html>
* <https://www.youtube.com/watch?v=FWE0-FOoE4s&list=PLUtfVcb-iqn-EkuBs3arreilxa2UKIChl>