



**S. B. JAIN INSTITUTE OF TECHNOLOGY, MANAGEMENT &
RESEARCH, NAGPUR.**

Practical No. 09

Aim: Design a RAM using VHDL.

Name of Student : Aadesh R. Motghare
Roll No. : 41(ET20065)
Semester/Year : 6th Sem/3rd Year
Academic Session : 2022-23
Date of Performance :
Date of Submission :

AIM: Design a RAM using VHDL.

OBJECTIVE:

- To verify the functionality RAM.
- RAM used to design complex digital System like Memory designing applications. Vending machines, washing machines.

SOFTWARE: - Xilinx ISE9.1.

THEORY:-

RAM is called main memory, primary memory, or system memory, RAM (random-access memory). It is a hardware device that allows information to be **stored** and **retrieved** on a computer. RAM is a volatile memory and requires power to keep the data accessible. If the computer is turned off, all data contained in RAM is lost.

Memory is divided into cells, and they are stored in the storage space present in the computer. Every cell has its unique location/address.

Types of RAM

RAM is further divided into two types, SRAM – Static Random Access Memory and DRAM- Dynamic Random Access Memory.

- **SRAM (Static Random Access memory)**

SRAM is used for Cache memory, it can hold the data as long as the power availability is there. It is refreshed simultaneously to store the present information. It is made with CMOS technology. It contains 4 to 6 transistors and it also uses clocks. It does not require a periodic refresh cycle due to the presence of transistors. Although SRAM is faster, it requires more power and is more expensive in nature.

- **DRAM (Dynamic Random Access memory)**

DRAM is used for the Main memory, it has a different construction than SRAM, it uses one transistor and one capacitor (also known as a conductor), which is needed to get recharged in milliseconds due to the presence of the capacitor. DRAM has one transistor and one capacitor in 1 memory bit. Although DRAM is slower, it can store more bits per chip, for instance, for the same amount of memory stored in SRAM, DRAM requires one less chip. DRAM requires less power and hence, less heat is produced.

RAM can be viewed logically in different configurations as **single** or **dual** port as follows

In a single port RAM, writing and reading can be done through one port only. It has one *en* input and *we* input. When *en* and *we* are both high, data are written into RAM and if *en* is high but *we* is low, reading through RAM can be done. The block diagram and Verilog code of a RAM block is shown below. (*en* input is optional)

LOGICAL DIAGRAM:-

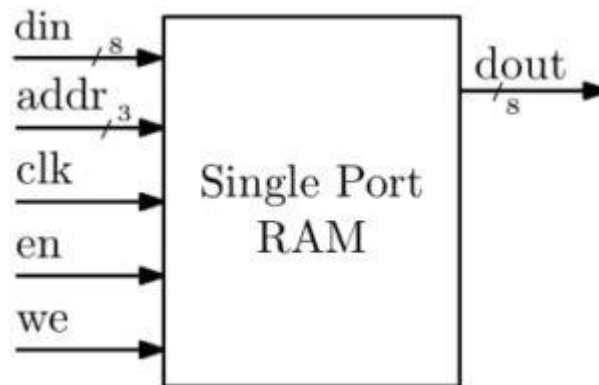


Fig 1. Single Port RAM

Dual Port Memory

In a dual port memory reading and writing can be through 2 different ports. It is possible to access the same address locations through two ports. The dual port memories have separate control line for both the ports.

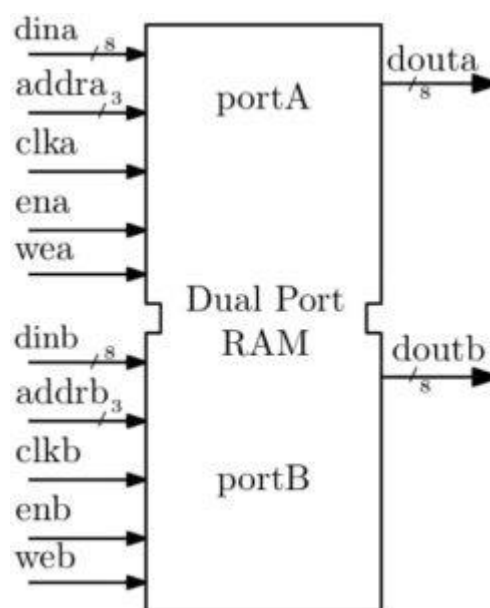


Fig 2. Dual Port RAM

Many different RAM configurations are possible depending on number ports of available, whether inputs and outputs are clocked or not, and other control signals.

RAM has a width and a depth. Depth indicates the number of words it can store and width indicates the number of bits in each word.

VHDL CODE:-

STEPS FOR PROGRAM:-

Step 1.Library /Package Declaration :Involves declaration of all libraries and respective packages used in the design.

```
LIBRARY library_name;  
USE library_name.package_name.all;
```

Step 2.Entity:

```
ENTITY entity_name is  
    PORT(signal_name(s): mode signal_type;  
        signal_name(s): mode signal_type;  
        ...);  
end ENTITY entity_name;
```

Signals of the same mode and signal_type can be grouped on 1line.

MODE describes the direction data is transferred through port

- in – data flows into the port
- out – data flows out of port *only*
- buffer – data flows out of port *as well as read* internally.
- inout – bi-directional data flow into and out of port

SIGNAL_TYPE defines the data type for the signal(s)

- bit – single signals that can have logic values 0 and 1.
- bit_vector – bus signals(vector form of bit) that can have logic values 0 and 1.
- std_logic – part of std_logic_1164 package of IEEE library. Used to represent 2 value logical values i.e 0 and 1 as well as other values such as high impedance ,don't care and others as described below.
- std_logic_vector – bus signals (vector form of std_logic) but IEEE standard for simulation and synthesis note that all vectors must have a range specified example for a 4 bit bus: bit_vector (3 downto 0) or std_logic_vector (3 downto 0).

In order to use std_logic and std_logic_vector we must include the library and package usage declarations in the VHDL model before the entity statement as follows:

```
LIBRARY ieee;  
USE ieee.std_logic_1164.all;
```

Values for std-logic:

U un-initialized (undefined logic value)
X forced unknown logic value
0 Logic low
1 Logic High
Z high impedance (tri-state)
W weak unknown
L weak 0
H weak 1
- don't care value (for synthesis minimization)

Step 3: Architecture Declaration

```
architecture architecture_name of entity_name  
  
    architecture_declarative_part;  
  
begin  
  
    Statements;  
  
end architecture_name;
```

Here we should specify the entity name for which we are writing the architecture body. The architecture statements should be inside the begin and end keyword.

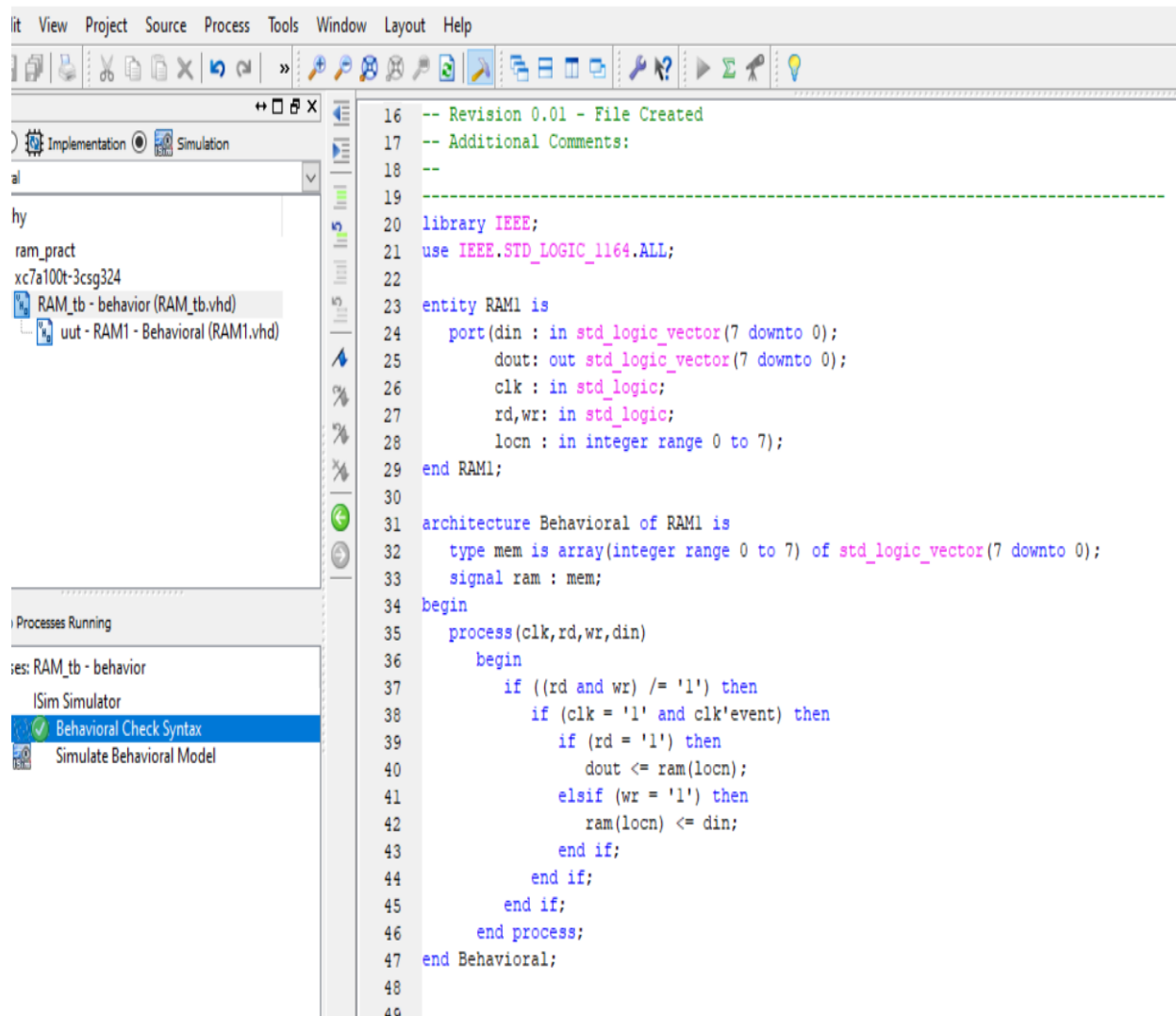
Architecture declarative part may contain variables, constants, or component declaration.

Step 4: Simulate the VHDL code and remove the syntax errors if any.

Step 5: Write the testbench and verify the design .

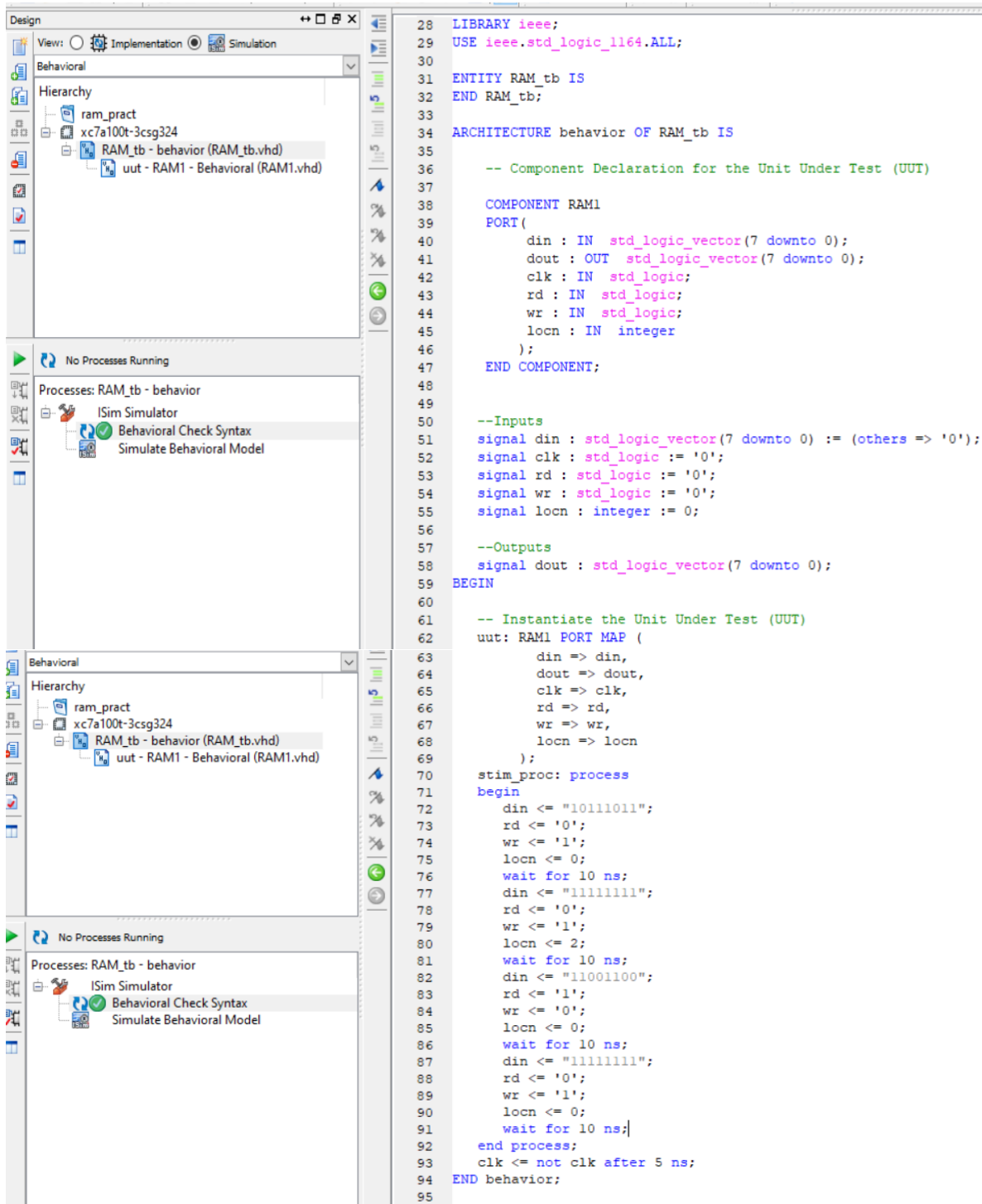
CODE:

ct Navigator (P.20131013) - C:\Users\ACER\OneDrive\Desktop\DSD practs\ram_pract\ram_pract.xise - [RAM1.vhd]



```
16 -- Revision 0.01 - File Created
17 -- Additional Comments:
18 --
19 -----
20 library IEEE;
21 use IEEE.STD_LOGIC_1164.ALL;
22
23 entity RAM1 is
24     port(din : in std_logic_vector(7 downto 0);
25          dout: out std_logic_vector(7 downto 0);
26          clk : in std_logic;
27          rd,wr: in std_logic;
28          locn : in integer range 0 to 7);
29 end RAM1;
30
31 architecture Behavioral of RAM1 is
32     type mem is array(integer range 0 to 7) of std_logic_vector(7 downto 0);
33     signal ram : mem;
34 begin
35     process(clk,rd,wr,din)
36     begin
37         if ((rd and wr) /= '1') then
38             if (clk = '1' and clk'event) then
39                 if (rd = '1') then
40                     dout <= ram(locn);
41                 elsif (wr = '1') then
42                     ram(locn) <= din;
43                 end if;
44             end if;
45         end if;
46     end process;
47 end Behavioral;
```

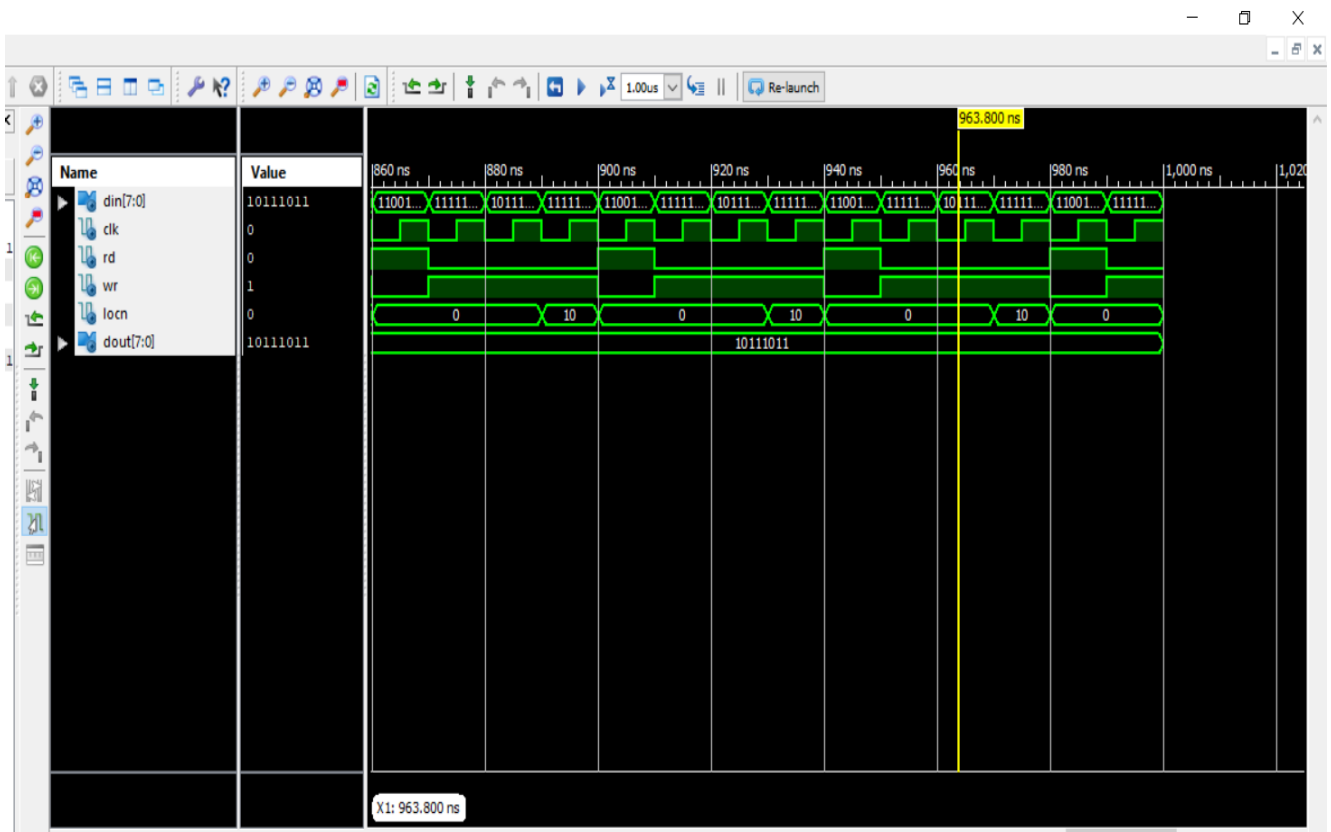
TESTBENCH:



```

28  LIBRARY ieee;
29  USE ieee.std_logic_1164.ALL;
30
31  ENTITY RAM_tb IS
32  END RAM_tb;
33
34  ARCHITECTURE behavior OF RAM_tb IS
35
36      -- Component Declaration for the Unit Under Test (UUT)
37
38      COMPONENT RAM1
39      PORT(
40          din : IN  std_logic_vector(7 downto 0);
41          dout : OUT std_logic_vector(7 downto 0);
42          clk : IN  std_logic;
43          rd : IN  std_logic;
44          wr : IN  std_logic;
45          locn : IN integer
46      );
47      END COMPONENT;
48
49      --Inputs
50      signal din : std_logic_vector(7 downto 0) := (others => '0');
51      signal clk : std_logic := '0';
52      signal rd : std_logic := '0';
53      signal wr : std_logic := '0';
54      signal locn : integer := 0;
55
56      --Outputs
57      signal dout : std_logic_vector(7 downto 0);
58
59  BEGIN
60
61      -- Instantiate the Unit Under Test (UUT)
62      uut: RAM1 PORT MAP (
63          din => din,
64          dout => dout,
65          clk => clk,
66          rd => rd,
67          wr => wr,
68          locn => locn
69      );
70      stim_proc: process
71      begin
72          din <= "10111011";
73          rd <= '0';
74          wr <= '1';
75          locn <= 0;
76          wait for 10 ns;
77          din <= "11111111";
78          rd <= '0';
79          wr <= '1';
80          locn <= 2;
81          wait for 10 ns;
82          din <= "11001100";
83          rd <= '1';
84          wr <= '0';
85          locn <= 0;
86          wait for 10 ns;
87          din <= "11111111";
88          rd <= '0';
89          wr <= '1';
90          locn <= 0;
91          wait for 10 ns;
92      end process;
93      clk <= not clk after 5 ns;
94  END behavior;
95
  
```

WAVEFORM:



RESULT:-

The VHDL code for RAM is executed and desired output is obtained.

CONCLUSION:

Here, I successfully design and verify the RAM using VHDL code.

DISCUSSION & VIVA VOCE

- 1) Explain the working of RAM.
- 2) How to design RAM in VHDL
- 3) What are the applications of RAM?

REFERENCE:

- VHDL Primer–J Bhasker –Pearson Education