**Experiment 4 – Memory and Interface**

**Aim:**

Write the SystemVerilog testbench code for 256 word memory. Use the concept of System Verilog “interface”.

Memory has following ports:

Input ports: clk, rst, ce, we

Input ports: data\_in, addr\_in of 8bits

Output port: data\_out of 8-bit size.

**Part 1:** **Write the SV code for interface.**

**Code:**

interface mem\_int(

  input wire clk,rst,ce,we,

  input wire [7:0] data\_in,addr\_in,

  output logic [7:0] data\_out);

endinterface

**Part 2:** **Write the SV code for Memory with interface i.e. DUT with interface.**

**Code:**

module ram(mem\_int inf);

  logic [7:0] memory [256];

  always@(posedge inf.clk) begin

    if(inf.rst) inf.data\_out <= 8'd0;

    else if (inf.ce) begin

      if(inf.we) memory[inf.addr\_in] <= inf.data\_in;

      else if(!inf.we) inf.data\_out <= memory[inf.addr\_in];

      else inf.data\_out <= 8'd0;

    end

  end

endmodule

**Part 3:** **Write the SV testbench to perform the write operation (store 10 random values in the locations with address 0, 1,2….9. Also perform the read operations to display the content written in the memory during write operation.**

**Code:**

module tb();

  logic clk=0,rst,ce,we;

  logic [7:0] data\_in,addr\_in;

  wire [7:0] data\_out;

  mem\_int int\_tb(clk,rst,ce,we,data\_in,addr\_in,data\_out);

  ram uut(int\_tb);

  initial forever #5 clk = ~clk;

  task reset();

    begin

      rst <= 1'b1;

      ce <= 1'b0;

      repeat(2) @(posedge clk);

      rst <= 1'b0;

    end

  endtask

  task read(input [7:0] addr);

    logic [7:0] test;

    begin

      @(negedge clk);

      ce <= 1'b1;

      we <= 1'b0;

      addr\_in <= addr;

      @(posedge clk) test <= uut.inf.data\_out;

      @(negedge clk);

      ce <= 1'b0;

      $display("the data out is %d at %d",uut.inf.data\_out,uut.inf.addr\_in);

    end

  endtask

  task write(input [7:0] addr);

    begin

      @(negedge clk);

      ce <= 1'b1;

      we <= 1'b1;

      addr\_in <= addr;

      data\_in <= $random;

      @(negedge clk);

      ce <= 1'b0;

      $display("value written is %d at %d", uut.inf.data\_in,uut.inf.addr\_in);

    end

  endtask

initial begin

    reset;

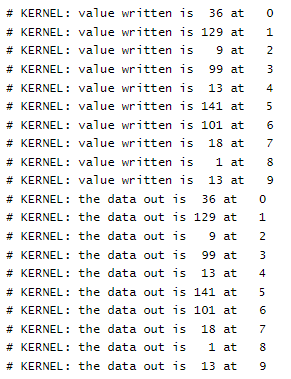
    for(int i=0; i<10; i++) write(i[7:0]);

    for(int i=0; i<10; i++) read(i[7:0]);

  end

endmodule

**Output:**

****

**Result:**

The given problem statement is executed and verified to be correct.