**Code:**

* **Design Module**
* **parameter Module**

`define mem\_addr\_len 8

`define mem\_locations 8

* **interface Module**

`include "parameters.sv"

interface Ports(input clock);

logic reset;

logic op;

logic [$clog2(`mem\_addr\_len) -1 : 0] addr;

logic [(`mem\_locations -1) : 0] data\_in;

logic [(`mem\_locations-1) : 0] out\_val;

modport tb(input out\_val,clock, output reset,op,addr,data\_in);

modport dut(input reset,op,addr,data\_in,output out\_val,clock);

endinterface

* **Main Module**

// Code your design here

// Code your design here

// lec 20

`include "parameters.sv"

`include "interface.sv"

// parameter addr\_bits = $clog2(mem\_addr\_len);

module Memory(Ports ports\_memory);

reg [ (`mem\_locations-1) : 0] out;

reg [`mem\_locations-1:0] memory\_register [`mem\_locations]; // unpacked array - unorganized memory block/cells

// reg [7:0] memory\_register [7:0] // packed array - continous memory block/cells

always @(negedge ports\_memory.clock)

begin

if(ports\_memory.reset)

begin

out <= 0 ;

for (int i = 0 ; i<= (`mem\_locations -1); i++)

begin

memory\_register[i] <= 0;

end

end

else

begin

if (ports\_memory.op)

begin

memory\_register [ports\_memory.addr] <= ports\_memory.data\_in;// write

end

else

begin

out <= memory\_register [ports\_memory.addr]; // read

end

end

end

assign ports\_memory.out\_val = out;

endmodule

* **TestBench**

module tb();

bit clock;

always #5 clock = ~clock;

Ports ports\_memory(clock); //interface

Memory dut(ports\_memory.dut); //ports from modports

initial

begin

clock = 1;

setup(); // task from inside module

end

initial

begin

ports\_memory.dut.reset = 1;

#15

ports\_memory.dut.reset = 0;

end

initial

begin

ports\_memory.dut.data\_in <= 00110010;

#60

ports\_memory.dut.data\_in <= 01111110;

end

initial

begin

wave(); // task from outside module

end

task setup();

for (int i = 0; i<=(`mem\_addr\_len) -1 ; i++) begin @(negedge clock)

ports\_memory.dut.addr = $random;

end

for(int i = 0; i<= 500; i++) begin @(posedge clock)

ports\_memory.dut.op = $random;

end

endtask

endmodule

task wave();

$dumpfile("dump.vcd");

$dumpvars();

#500

$finish;

endtask

**Output:**

Timeline

Description automatically generated