ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KỸ THUẬT MÁY TÍNH

BÁO CÁO BÀI THỰC HÀNH LAB04 - KIỂM TRA THIẾT KẾ SỬ DỤNG TESTBENCH

HỌ VÀ TÊN: NGUYỄN GIA BẢO NGỌC – 21520366
NGUYỄN QUỐC TRƯỜNG AN – 21521810
NGUYỄN TRƯƠNG TIẾN ĐẠT – 21521946

LỚP: CE213.O11.2

GIẢNG VIÊN HƯỚNG DẪN:

HỒ NGỌC DIỄM

TP. HỒ CHÍ MINH – Tháng 11 năm 2023

MỤC LỤC

MỤC LỤC ẢNH

MỤC LỤC BẢNG

I. Mục tiêu .

Sinh viên làm quen với việc kiểm tra thiết kế bằng phương pháp viết Testbench và sử dụng phần mềm ModelSim-Altera để kiểm tra thiết kế.

II. Chuẩn bị thực hành.

- Sinh viên đọc trước và thực hành sử dụng phần mềm ModelSim-Altera trong file "huong dan su dung phan mem ModelSim-Altera.pdf"

- Sinh viên phải chuẩn bị các phần được yêu cầu trong mỗi câu của bài Lab và nộp vào đầu buổi học.

- Điểm bài chuẩn bị được tính vào điểm bài báo cáo của Lab.

III. Nội dung thực hành.

Câu1:

Sử dụng ngôn ngữ Verilog HDL, thiết kế bộ ALU 32-bit có các chức năng như bên dưới.

- Viết testbench tạo giá trị cho các tín hiệu input và quan sát kết quả sử dụng các thủ tục $display và $monitor, chạy mô phỏng kiểm tra chức năng của thiết kế dùng phần mềm ModelSim-Altera.

M	S_1	S_0	ALU Operations
0	0	0	Complement A
0	0	1	AND
0	1	0	EX-OR
0	1	1	OR
1	0	0	Decrement A
1	0	1	Add
1	1	0	Subtract
1	1	1	Increment A

Hình 1 – Tín hiệu điều khiển ALU

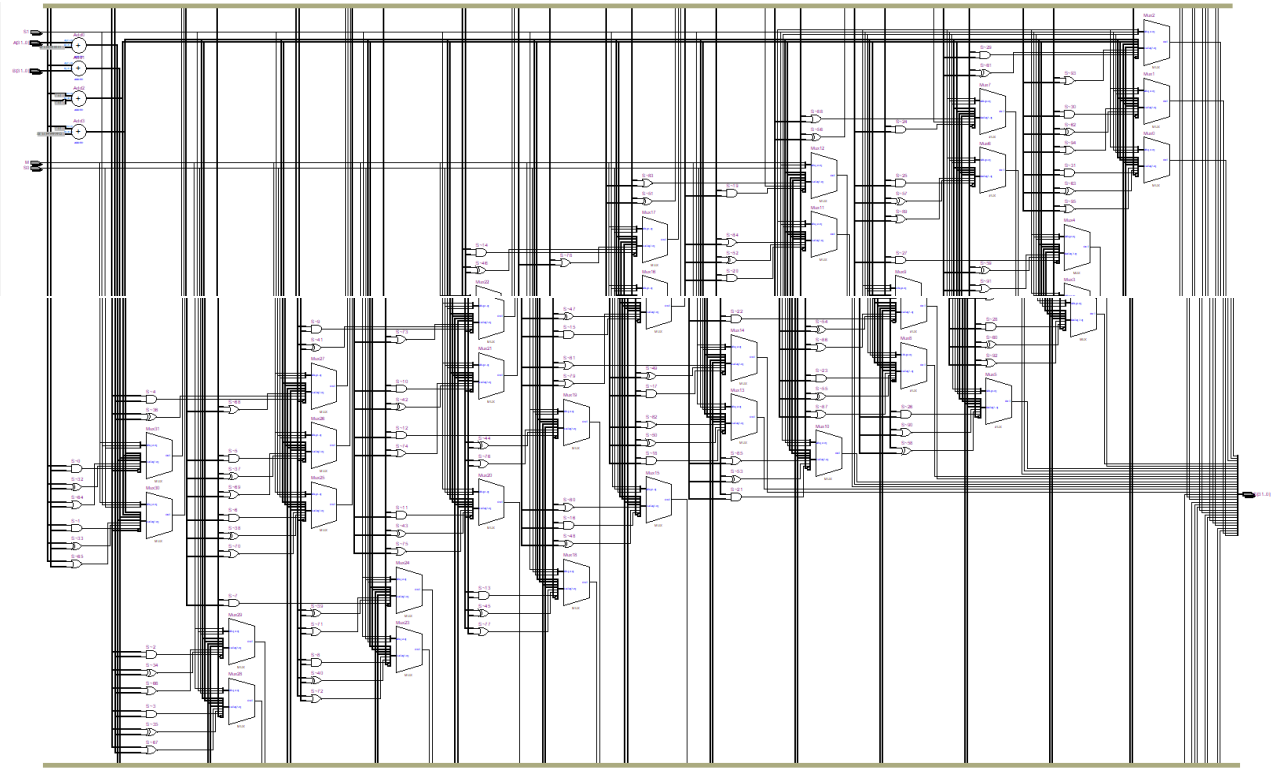
1.1 Thiết kế mạch:

- Code verilog:

```
module ALU_32bits ( S,
                                        A,
                                        B,
                                        M,
                                        S1,
                                        S0
);
//----------------------------------------
        output reg[31:0] S;
        input[31:0] A;
        input[31:0] B;
        input M;
        input S1;
        input S0;
//----------------------------------------
        always @(A, B, M, S1, S0) begin
                case ({M, S1, S0})
                        3'b000: S= ~A;
                        3'b001: S= A&B;
                        3'b010: S= A^B;
                        3'b011: S= A|B;
                        3'b100: S= A+1;
                        3'b101: S= A+B;
                        3'b110: S= A-B;
                        3'b111: S= A-1;
                        default: S=0;
                endcase
        end
endmodule
```

Bảng 1 – Code Verilog thiết kế ALU 32bits

- Mạch RTL:



Hình 2 – Mạch RTL ALU 32bits

1.2 Testbench:

- Code Verilog:

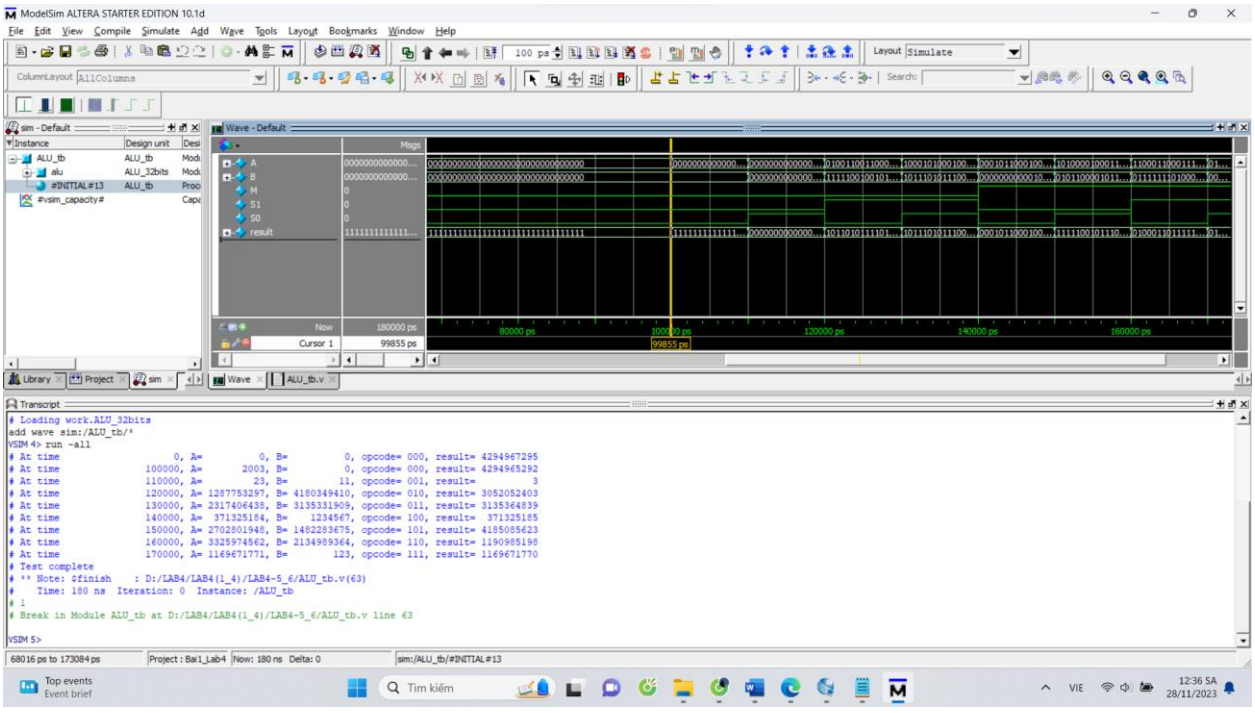
```
`timescale 1ns/1ps
module ALU_tb;
//--------------------Input&Output----------------------
                wire[31:0] result;
                reg[31:0]  A;
                reg[31:0]  B;
                reg M;
                reg S1;
                reg S0;
//----------------------Unit under test ------------------
        ALU_32bits alu ( result, A, B, M, S1, S0);
//----------------------Test session---------------------
        initial begin
  //-------initialize inputs--------
                A =  0;
                B =  0;
```

```
                M =  0;
                S1 = 0;
                S0 = 0;
                #100;
//-------start test session--------
                // Complement A
                A = 2003;
                {M,S1,S0} = 3'b000;
                #10;
                //  AND
                A = 23;
                B = 11;
                {M,S1,S0} = 3'b001;
                #10;
                // XOR
                A = 1287753297;
                B = 4180349410;
                {M,S1,S0} = 3'b010;
                #10;
                // OR
                A = 2317406438;
                B = 3135331909;
                {M,S1,S0} = 3'b011;
                #10;
                // DEC A
                A = 371325184;
                B = 1234567;
                {M,S1,S0} = 3'b100;
                #10;
                // ADD
                A = 2702801948;
                B = 1482283675;
                {M,S1,S0} = 3'b101;
                #10;
                // SUB
                A = 3325974562;
                B = 2134989364;
                {M,S1,S0} = 3'b110;
```

```
            #10;
            // INC
            A = 1169671771;
            B = 123;
            {M,S1,S0} = 3'b111;
            #10;

            $display ("Test complete");
            $finish;
        end
  // Monitor changes and display them
        initial begin
            $monitor("At time %t, A= %d, B= %d, opcode= %b%b%b,
result= %d", $time, A, B, M, S1, S0, result);
        end
endmodule
```

Bảng 2 - Code Verilog Testbench cho ALU 32bits

- Chạy mô phỏng trên ModelSim:



Hình 3 – Mô phỏng Testbench ALU 32bits

Câu2:

Sử dụng ngôn ngữ Verilog HDL, thiết kế một tập gồm 32 thanh ghi, mỗi thanh ghi 4 byte. Tập thanh ghi (Register File) có các tín hiệu sau: ReadAddress1[4:0], ReadAddress2[4:0], WriteAddress[4:0], WriteData[31:0], ReadData1[31:0], ReadData2[31:0], ReadWriteEn(1: cho phép ghi, 0 : cho phép đọc) .

- Viết testbench tạo giá trị cho các tín hiệu input và chạy mô phỏng kiểm tra chức năng của thiết kế.

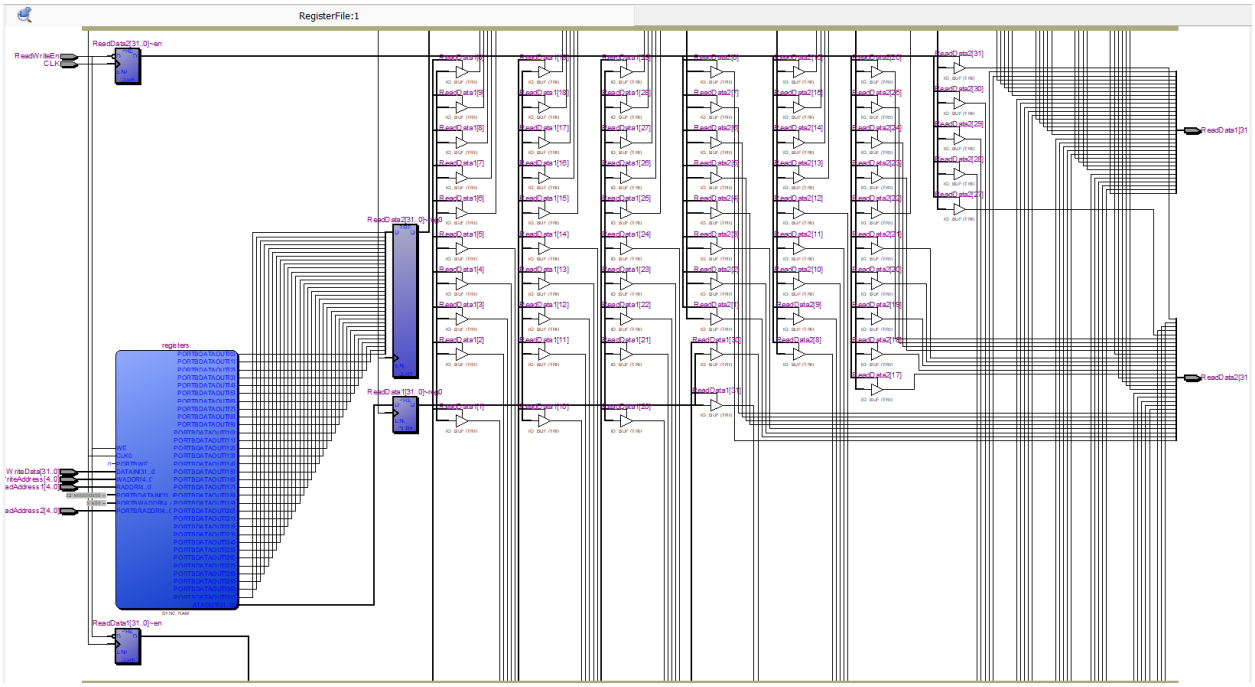
2.1 Thiết kế mạch:

- Code Verilog:

```
module RegisterFile(
        input CLK,
        input [4:0] ReadAddress1,
        input [4:0] ReadAddress2,
        input [4:0] WriteAddress,
        input [31:0] WriteData,
        output reg [31:0] ReadData1,
        output reg [31:0] ReadData2,
        input ReadWriteEn
);
        // Khai báo mảng thanh ghi
        reg [31:0] registers [0:31];

        always @(posedge CLK) begin
                if (ReadWriteEn) begin          // Write data into RegisterFile
                        registers[WriteAddress] <= WriteData;
                        ReadData1 <= 32'bz;     // When write data, output read is
unknown (32'bz)
                        ReadData2 <= 32'bz;
                end
                else begin   // Read data from RegisterFile
                        ReadData1 <= registers[ReadAddress1];
                        ReadData2 <= registers[ReadAddress2];
                end
        end
endmodule
```

Bảng 3 - Code Verilog thiết kế RegisterFile

- Mạch RTL:



Hình 4 – Mạch RTL RegisterFile 32 thanh ghi 32bits

2.2 Testbench:

- Code Verilog:

```
`timescale 1ns/100ps

module RegisterFileTestbench;
    reg CLK;
    reg [4:0] ReadAddress1, ReadAddress2, WriteAddress;
    reg [31:0] WriteData;
    reg ReadWriteEn;
    wire [31:0] ReadData1, ReadData2;

    RegisterFile dut (
        .CLK(CLK),
        .ReadAddress1(ReadAddress1),
        .ReadAddress2(ReadAddress2),
        .WriteAddress(WriteAddress),
        .WriteData(WriteData),
```

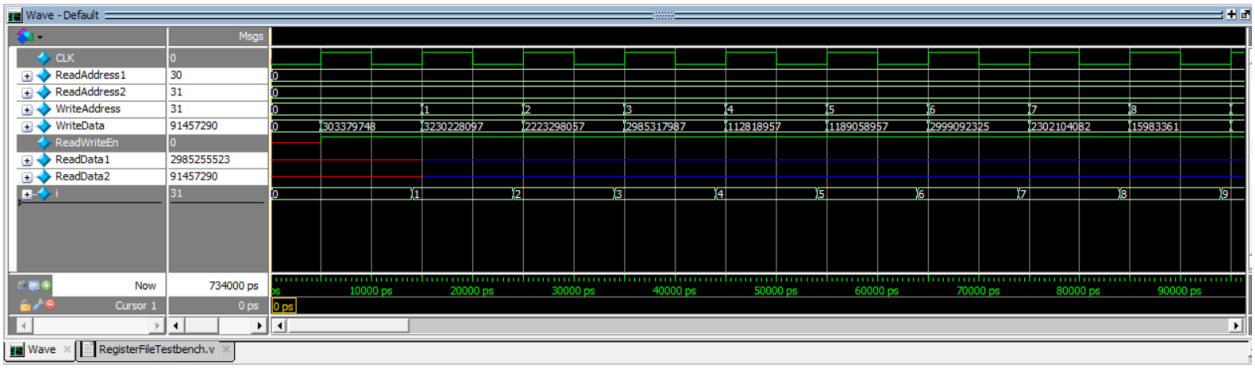
```
    .ReadData1(ReadData1),
    .ReadData2(ReadData2),
    .ReadWriteEn(ReadWriteEn)
);
      integer i;
// Set first value for input
initial begin
    {CLK, ReadAddress1, ReadAddress2, WriteAddress, WriteData} = 0;
end
      // Generate clock
      always #5 CLK = ~CLK;

initial begin

    // Write data into RegisterFile
    for (i = 0; i <= 31; i = i + 1) begin
                        repeat (1) @(posedge CLK) begin
                              WriteAddress = i;
                          ReadWriteEn = 1;  //Set to write mode
                              WriteData = $random;  //Get random value
for data
                        end
                        #10;  // wait for a few clock cycles
            end

            // Read data from RegisterFile
            for (i = 0; i <= 30; i = i + 1) begin
                        repeat (1) @(posedge CLK) begin
                              ReadWriteEn = 0;        // Set to read mode
                              ReadAddress1 = i;
                              ReadAddress2 = i + 1;
                        end
                        #10;  // wait for a few clock cycles
    end
    // Stop simulation after a delay
    #100 $stop;
end
```

```
endmodule
```

Bảng 4 - Code Verilog Testbench cho RegisterFile

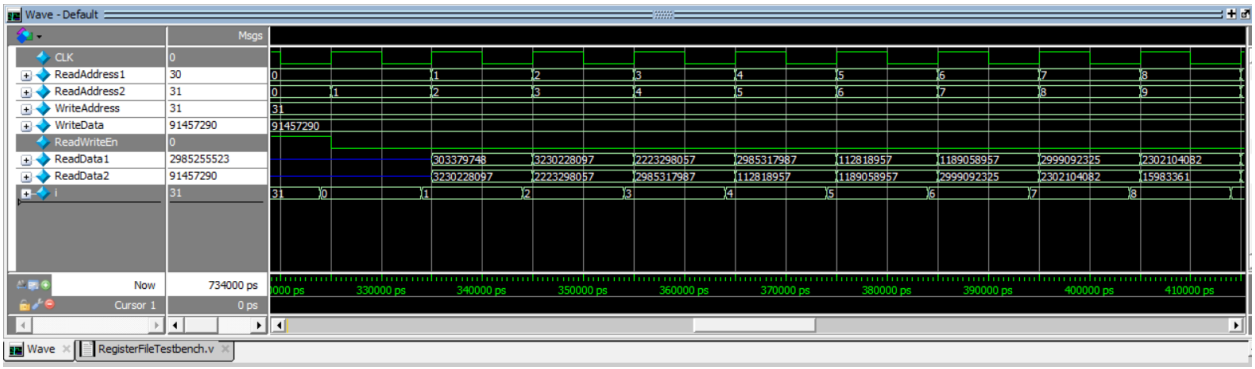
- Chạy mô phỏng trên ModelSim:

+ WriteData:



Hình 5 – Mô phỏng Testbench (Ghi dữ liệu vào RegisterFile)

+ ReadData:



Hình 6 – Mô phỏng Testbench (Đọc dữ liệu từ RegisterFile)

Câu3:

Sử dụng ngôn ngữ Verilog HDL, hiện thức thiết kế bộ nhớ dữ liệu (Data Memory) dung lượng 1024 bytes có các tín hiệu sau: Address[9:0], WriteData[7:0], ReadData[7:0], WriteEn, ReadEn và viết testbench kiểm tra chức năng trên trên phần mềm mô phỏng ModelSim.

- Viết testbench tạo giá trị cho các tín hiệu input và chạy mô phỏng kiểm tra chức năng của thiết kế.

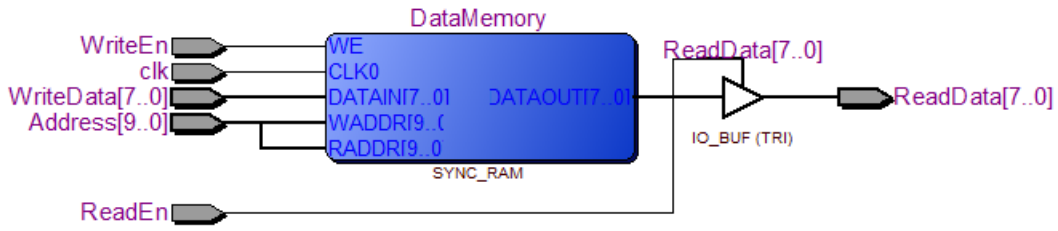
3.1 Thiết kế mạch:

```
module DataMemory(ReadData,
```

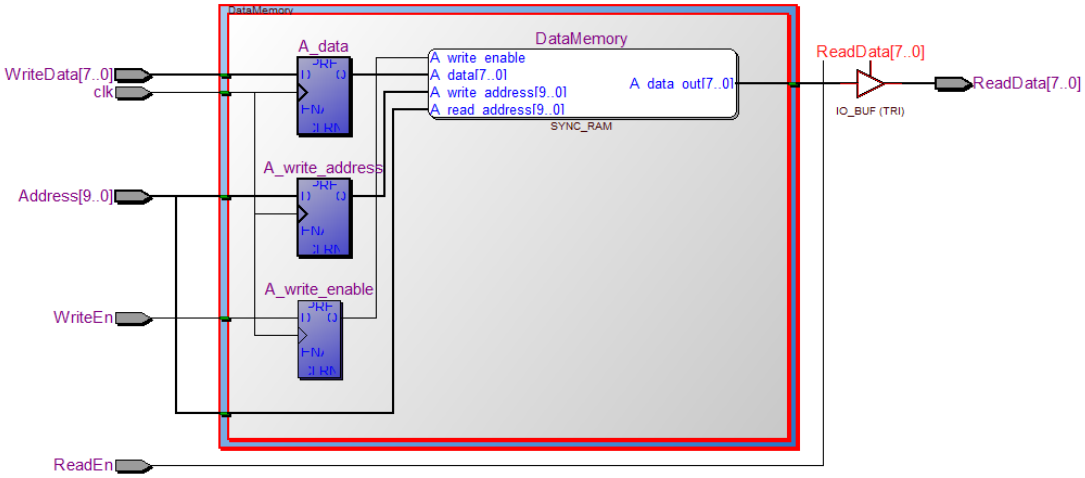
```
                                    clk,
                                    WriteEn,
                                    ReadEn,
                                    WriteData,
                                    Address
                                    );

    parameter DATA_WIDTH = 8;
    parameter ADDR_WIDTH = 10;

    output [DATA_WIDTH-1 : 0] ReadData;

    input clk;
    input WriteEn;
    input ReadEn;
    input [DATA_WIDTH-1 : 0] WriteData;
    input [ADDR_WIDTH-1 : 0] Address;

    reg [DATA_WIDTH-1:0] DataMemory[2**ADDR_WIDTH-1:0];

    always @(posedge clk) begin

        if(WriteEn)
            DataMemory[Address] <= WriteData;

    end

    assign ReadData = (ReadEn) ? DataMemory[Address] :
{(DATA_WIDTH){1'bz}};

endmodule
```

Bảng 5 - Code Verilog thiết kế DataMemory

- Mạch RTL:



Hình 7 – Mạch RTL DataMemory (Đóng gói)



Hình 8 – Mạch RTL DataMemory

3.2 Testbench:

```
`timescale 1ns/100ps
module DataMemory_Testbench();

        parameter DATA_WIDTH = 8;
        parameter ADDR_WIDTH = 10;

        wire [DATA_WIDTH-1 : 0] ReadData;

        reg clk;
        reg WriteEn;
        reg ReadEn;
```

```
reg [DATA_WIDTH-1 : 0] WriteData;
reg [ADDR_WIDTH-1 : 0] Address;

integer i;

// initial at start time = 0
initial begin

	{clk, WriteEn, ReadEn, WriteData, Address} = 0;

end

// generate clock width T = 10, Duty = 50%
always #10 clk = ~clk;

// instance module DataMemory
DataMemory #(
				.DATA_WIDTH(DATA_WIDTH),
				.ADDR_WIDTH(ADDR_WIDTH)
			)
			dmem_dut
			(
				.ReadData(ReadData),
				.clk(clk),
				.WriteEn(WriteEn),
				.ReadEn(ReadEn),
				.WriteData(WriteData),
				.Address(Address)
			);

// generate testcase
initial begin

	// write data into memory cells indexed from 1-50;
	for(i = 1; i <= 50; i = i + 1) begin

		repeat (1) @(posedge clk) begin
```

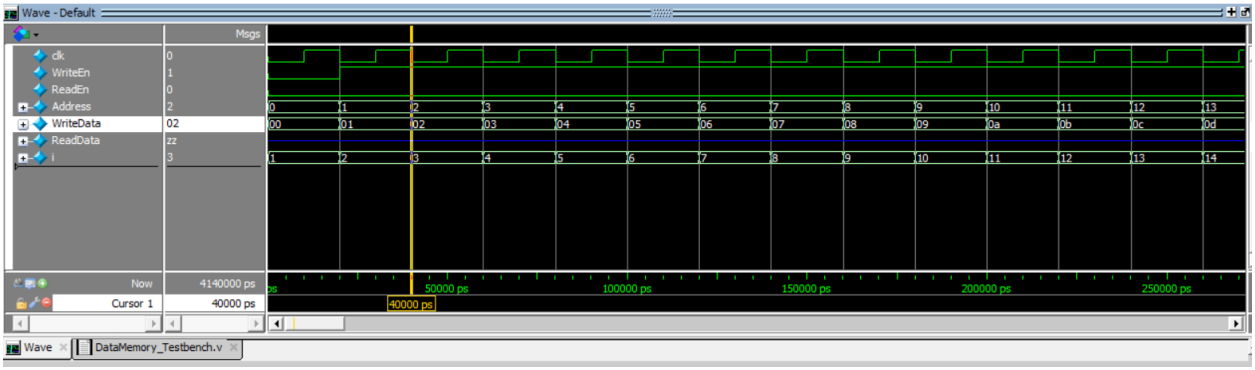
```
                #10
                Address = i;
                WriteData = i;
                WriteEn = 1;

        end

end

// read data from memory cells indexed from 1-50;
for(i = 1; i <= 50; i = i + 1) begin

        repeat (1) @(posedge clk) begin

                #10
                Address = i;
                WriteEn = 0;
                ReadEn = 1;

        end

end

// write data into memory cells indexed from 50-100;
for(i = 50; i <= 100; i = i + 1) begin

        repeat (1) @(posedge clk) begin

                #10
                Address = i;
                WriteData = i;
                WriteEn = 1;
                ReadEn = 0;

        end

end
```

```
            // read data from memory cells indexed from 50-100;
            for(i = 50; i <= 100; i = i + 1) begin

                repeat (1) @(posedge clk) begin

                    #10
                    Address = i;
                    WriteEn = 0;
                    ReadEn = 1;

                end

            end

            #100 $stop;

        end

endmodule
```

Bảng 6 - Code Verilog Testbench cho DataMemory

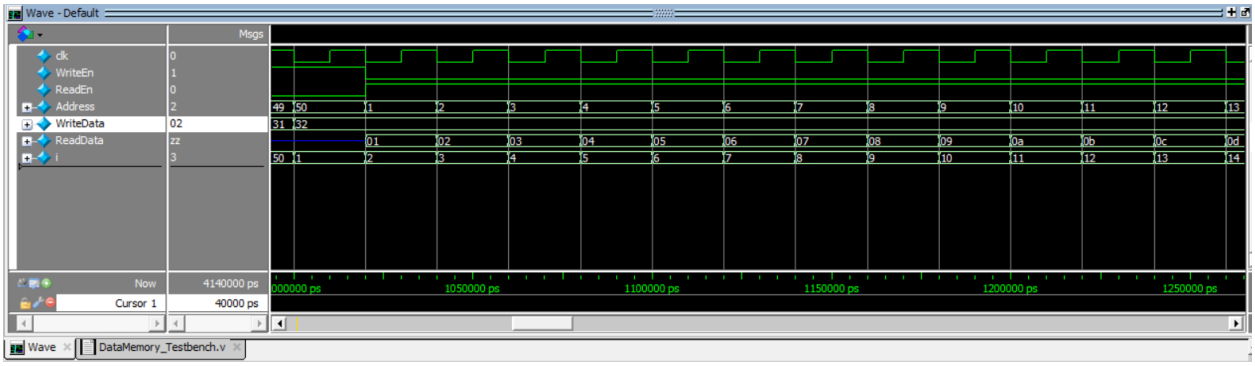
- Chạy mô phỏng trên ModelSim:

+ Write data from 1 to 50:



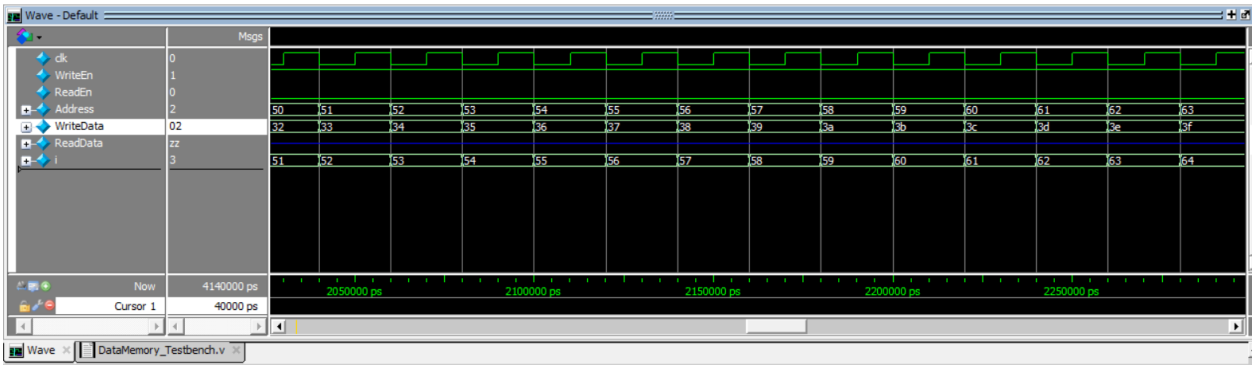
Hình 9 - Mô phỏng TestBench (Ghi dữ liệu vào DataMemory)

+ Read data from 1 to 50:



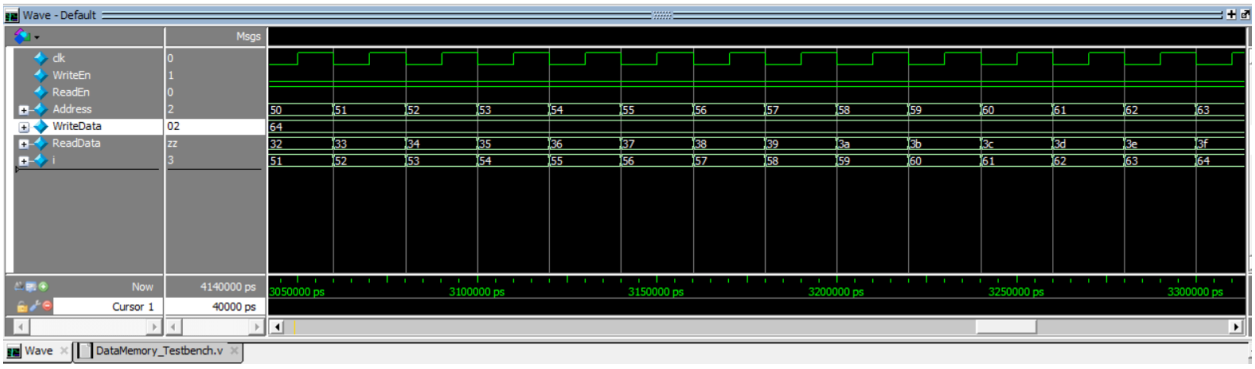
Hình 10 - Mô phỏng Testbench (Đọc dữ liệu ra từ DataMemory)

+ Write data from 50 to 100:



Hình 11 - Mô phỏng TestBench (Ghi dữ liệu vào DataMemory)

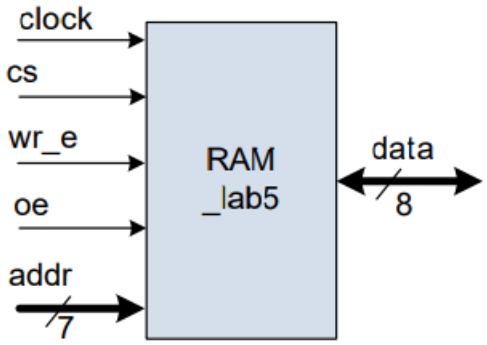
+ Read data from 50 to 100:



Hình 12 - Mô phỏng TestBench (Đọc dữ liệu ra từ DataMemory)

Câu4:

Thiết kế một single port RAM đồng bộ read/write có sơ đồ như bên dưới:



Hình 13 - Sơ đồ thiết kế single port RAM

Biết rằng:

- clock: kích cạnh lên

- cs: chip_select

- wr_e = 1: cho phép ghi

- wr_e = 0: cho phép đọc

- oe: Output enable

- addr: address (7-bit → RAM 128 byte)

- data: kiểu inout 8-bit

Yêu cầu:

• Viết testbench để kiểm tra thiết kế theo mô hình quan sát dạng sóng bằng phần mềm ModelSim-Altera

• Viết testbench để kiểm tra thiết kế theo mô hình tự kiểm tra (Self-checking) bằng phần mềm ModelSim-Altera

4.1 Thiết kế mạch:

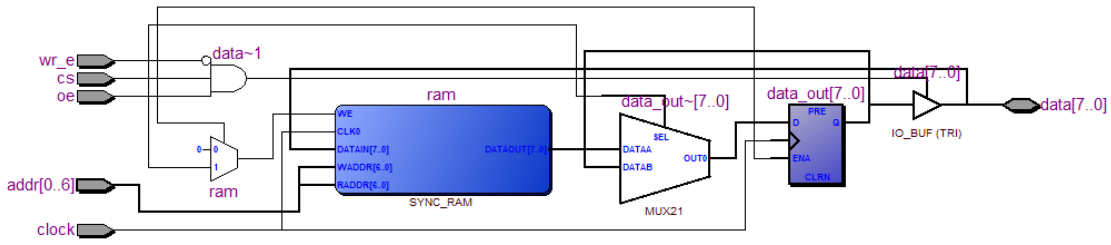
- Code Verilog:

```
module single_port_RAM ( data,
                         clock,
                         cs,
                         wr_e,
                         oe,
                         addr
);
```

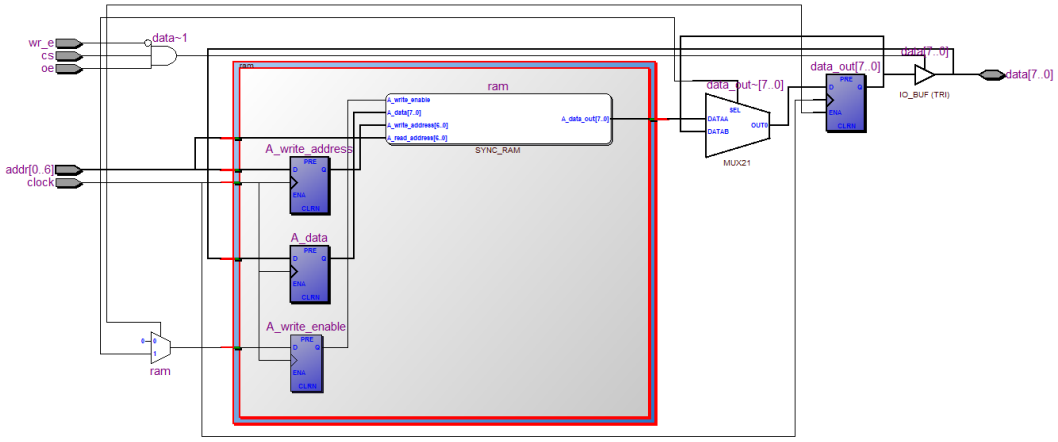
```
//--------------------Input&Output port----------------------
        parameter addr_width = 7;
        parameter data_width = 8;
        input clock;
        input cs;
        input wr_e;
        input oe;
        input [0:addr_width -1] addr;
        inout [0:7] data;
        reg[0:data_width-1] data_out;
        reg[0:data_width-1] ram[0:127];
//-------------------------------------------------------------
        assign data = (cs &&oe &&!wr_e) ? data_out : 8'bzzzzzzzz;
        always @(posedge clock) begin
                if (cs) begin
                        if(wr_e)
                                ram[addr]= data;
                        else
                                data_out = ram[addr];
                end
        end

endmodule
```

Bảng 7 - Code Verilog thiết kế single port RAM

- Mạch RTL:

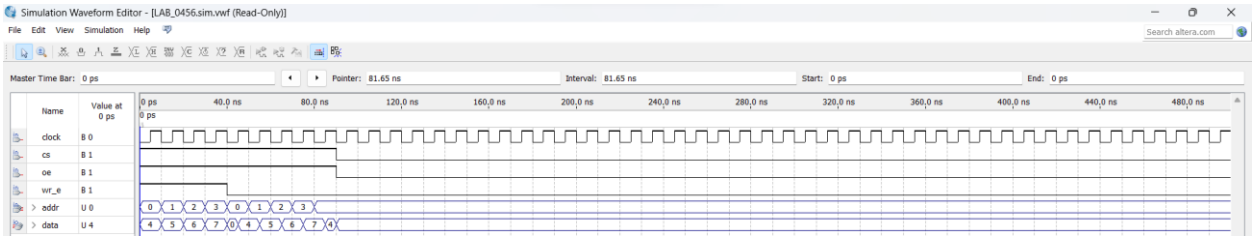


Hình 14 - Mạch RTL single port RAM (Đóng gói)



Hình 15 - Mạch RTL single port RAM

- Mô phỏng trên Quartus:



Hình 16 - Mô phỏng RAM trên Quartus

4.2 Testbench:

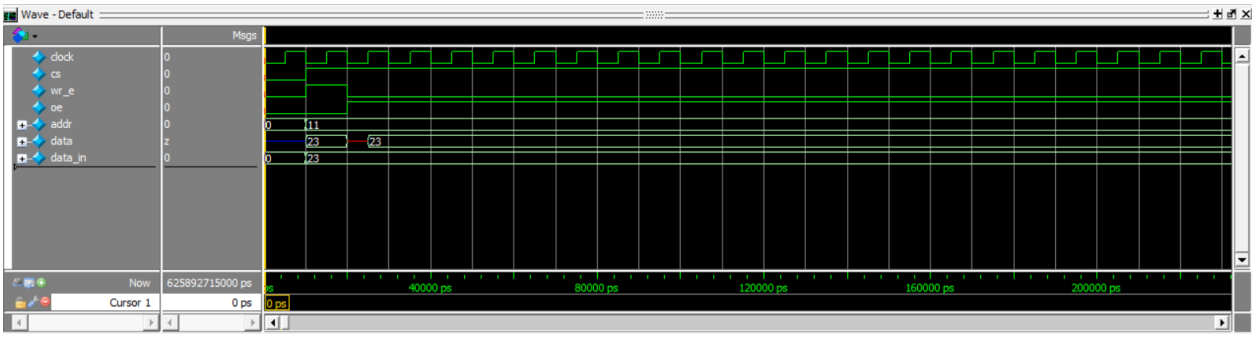
- Code Verilog:

```
`timescale 1ns/1ps
module RAM_tb ();
//--------------------Input&Output----------------------
  parameter addr_width = 7;
        parameter data_width = 8;
        reg clock;
        reg cs;
        reg wr_e;
        reg oe;
        reg [0:6] addr;
        wire [0:7] data;
        reg[0:7] data_in;
        integer i;
//---------------------Unit under test -----------------
```

```
        always #10 clock=~clock;
        assign data = (cs &!oe &wr_e) ? data_in : 8'bzzzzzzzz;
  single_port_RAM ram(data, clock, cs, wr_e, oe, addr);

//----------------------Test session--------------------
        initial begin
          {cs, clock, wr_e, oe} = 0;
          for (i =0; i<10; i=i+1)begin
            @(posedge clock) addr= i; wr_e= 1; cs=1; oe=0; data_in=
$random;
          end
          #10;
          clock=1;
          for (i =0; i<10; i=i+1)begin
            @(posedge clock) addr= i; wr_e= 0; cs=1; oe=1;
          end
        #10;
  $display ("Test complete");
  $finish;
end
  initial begin
    $monitor ("At time %t, Address= %d, Data_in= %d, Data_out= %d, cs=
%b, wr_e= %b, oe= %b", $time, addr, data_in, data, cs, wr_e, oe);
  end
endmodule
```

Bảng 8 - Code Verilog Testbench cho single port RAM

- Mô phỏng trên ModelSim:



Hình 17 - Mô phỏng Testbench single port RAM