ECEN 651 Lab Exercise 4 Report

Storage Elements in Verilog

Name: Pranav Anantharam

UIN: 734003886

Date: 10/03/2023

1. Design and simulate a 32-by-32 register file.

Block Diagram:

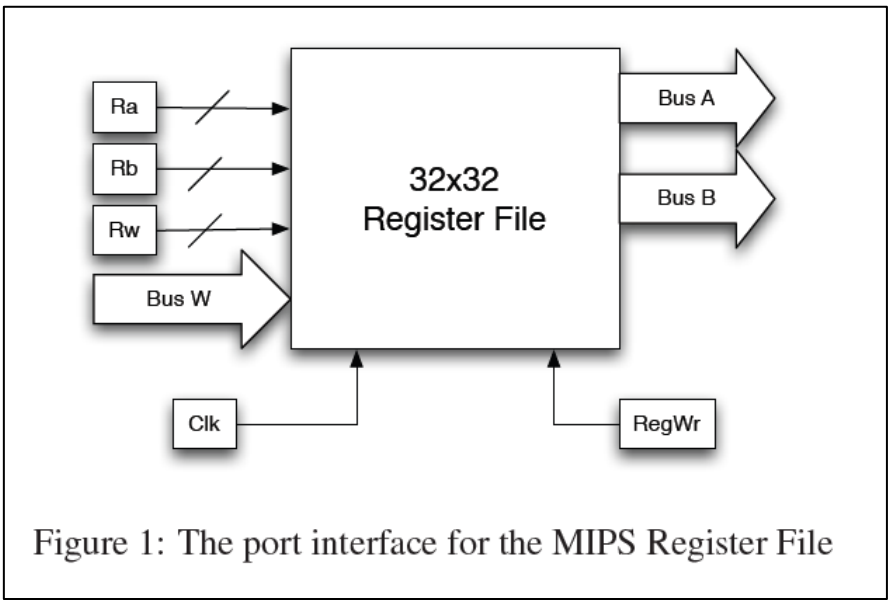


Figure 1: The port interface for the MIPS Register File

(a) Describe a register file in behavioral Verilog.

Design Source code filename: RegisterFile.v

```
`timescale 1ns / 1ps


module RegisterFile( input [4:0] RA, input [4:0] RB, input [4:0] RW, input [31:0] BusW,
input RegWr, input Clk, output reg [31:0] BusA, output reg [31:0] BusB );
// RA - Bus A Address
// RB - Bus B Address
// RW - Write Port Address
// BusW - Write Port Data Input
// RegWr - Write enable signal
// Clk - Clock signal
```

```
// BusA - Bus A output data register
// BusB - Bus B output data register

reg [31:0] register_file [31:0]; // 32 elements x 32 bit wide element

// 0th register is wired to value 0
initial begin
    register_file[0] = 32'b0;
end

always @(negedge Clk) begin
    // If Write Port address is not zero and Write enable signal is high
    if ( (RW != 5'b0) && (RegWr == 1'b1) ) begin
        // Write data into register
        register_file[RW] <= BusW;
    end
end

// Read values from registers
always @* begin
    BusA <= register_file[RA];
    BusB <= register_file[RB];
end
endmodule
```

(b) Test the behavioral model of the register file using the test bench provided.

Testbench Source code filename: RegisterFileTest.v

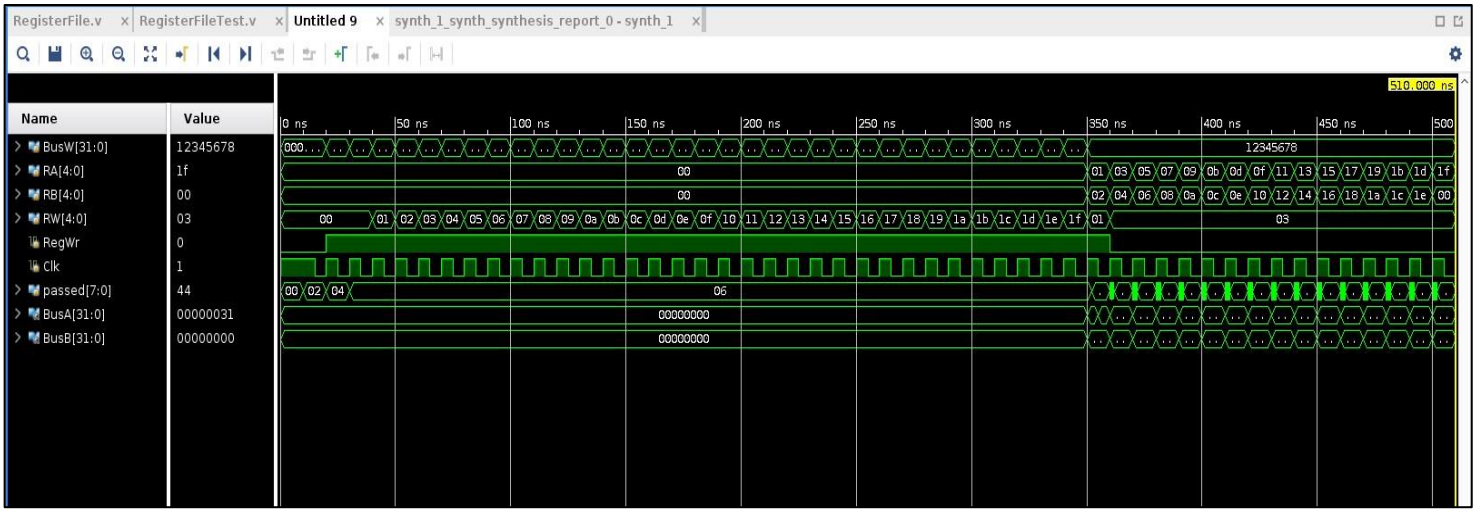
```
`timescale 1ns / 1ps

`define STRLEN 32
module RegisterFileTest_v;

      task passTest;
            input [31:0] actualOut, expectedOut;
            input [`STRLEN*8:0] testType;
            inout [7:0] passed;

            if(actualOut == expectedOut) begin $display ("%s passed", testType); passed =
passed + 1; end
            else $display ("%s failed: %d should be %d", testType, actualOut, expectedOut);
      endtask
```

```
task allPassed;
      input [7:0] passed;
      input [7:0] numTests;

      if(passed == numTests) $display ("All tests passed");
      else $display("Some tests failed");
endtask

// Inputs
reg [31:0] BusW;
reg [4:0] RA;
reg [4:0] RB;
reg [4:0] RW;
reg RegWr;
reg Clk;
reg [7:0] passed;

// Outputs
wire [31:0] BusA;
wire [31:0] BusB;

// Instantiate the Unit Under Test (UUT)
RegisterFile uut (.BusA(BusA), .BusB(BusB), .BusW(BusW), .RA(RA), .RB(RB), .RW(RW),
.RegWr(RegWr), .Clk(Clk));
initial begin
      // Initialize Inputs
      BusW = 0;
      RA = 0;
      RB = 0;
      RW = 0;
      RegWr = 0;
      Clk = 1;
      passed = 0;
      #10;
      // Add stimulus here
      {RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd0, 32'h0, 1'b0};
      passTest(BusA, 32'h0, "Initial $0 Check 1", passed);
      passTest(BusB, 32'h0, "Initial $0 Check 2", passed);
      #5; Clk = 0; #5; Clk = 1;

      {RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd0, 32'h12345678, 1'b1};
      passTest(BusA, 32'h0, "Initial $0 Check 3", passed);
      passTest(BusB, 32'h0, "Initial $0 Check 4", passed);
      #5; Clk = 0; #5; Clk = 1;
      passTest(BusA, 32'h0, "$0 Stays 0 Check 1", passed);
      passTest(BusB, 32'h0, "$0 Stays 0 Check 2", passed);

{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd0, 32'h0, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd1, 32'h1, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd2, 32'h2, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd3, 32'h3, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd4, 32'h4, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd5, 32'h5, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd6, 32'h6, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd7, 32'h7, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd8, 32'h8, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd9, 32'h9, 1'b1};#5; Clk = 0; #5; Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd10, 32'h10, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd11, 32'h11, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd12, 32'h12, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd13, 32'h13, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd14, 32'h14, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd15, 32'h15, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd16, 32'h16, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd17, 32'h17, 1'b1};#5; Clk = 0; #5;Clk = 1;
```

```
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd18, 32'h18, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd19, 32'h19, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd20, 32'h20, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd21, 32'h21, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd22, 32'h22, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd23, 32'h23, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd24, 32'h24, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd25, 32'h25, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd26, 32'h26, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd27, 32'h27, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd28, 32'h28, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd29, 32'h29, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd30, 32'h30, 1'b1};#5; Clk = 0; #5;Clk = 1;
{RA, RB, RW, BusW, RegWr} = {5'd0, 5'd0, 5'd31, 32'h31, 1'b1};#5; Clk = 0; #5;Clk = 1;

{RA, RB, RW, BusW, RegWr} = {5'd1, 5'd2, 5'd1, 32'h12345678, 1'b1};#2;
passTest(BusA, 32'h1, "Initial Value Check 1", passed);
passTest(BusB, 32'h2, "Initial Value Check 2", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h12345678, "Value Updated 1", passed);
passTest(BusB, 32'h2, "Value Stayed Same 1", passed);
{RA, RB, RW, BusW, RegWr} = {5'd3, 5'd4, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h3, "Initial Value Check 3", passed);
passTest(BusB, 32'h4, "Initial Value Check 4", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h3, "Value Not Updated 2", passed);
passTest(BusB, 32'h4, "Value Stayed Same 2", passed);
{RA, RB, RW, BusW, RegWr} = {5'd5, 5'd6, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h5, "Initial Value Check 5", passed);
passTest(BusB, 32'h6, "Initial Value Check 6", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h5, "Value Not Updated 3", passed);
passTest(BusB, 32'h6, "Value Stayed Same 3", passed);
{RA, RB, RW, BusW, RegWr} = {5'd7, 5'd8, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h7, "Initial Value Check 7", passed);
passTest(BusB, 32'h8, "Initial Value Check 8", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h7, "Value Not Updated 4", passed);
passTest(BusB, 32'h8, "Value Stayed Same 4", passed);
{RA, RB, RW, BusW, RegWr} = {5'd9, 5'd10, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h9, "Initial Value Check 9", passed);
passTest(BusB, 32'h10, "Initial Value Check 10", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h9, "Value Not Updated 5", passed);
passTest(BusB, 32'h10, "Value Stayed Same 5", passed);
{RA, RB, RW, BusW, RegWr} = {5'd11, 5'd12, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h11, "Initial Value Check 11", passed);
passTest(BusB, 32'h12, "Initial Value Check 12", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h11, "Value Not Updated 6", passed);
passTest(BusB, 32'h12, "Value Stayed Same 6", passed);
{RA, RB, RW, BusW, RegWr} = {5'd13, 5'd14, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h13, "Initial Value Check 13", passed);
passTest(BusB, 32'h14, "Initial Value Check 14", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h13, "Value Not Updated 7", passed);
passTest(BusB, 32'h14, "Value Stayed Same 7", passed);
{RA, RB, RW, BusW, RegWr} = {5'd15, 5'd16, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h15, "Initial Value Check 15", passed);
passTest(BusB, 32'h16, "Initial Value Check 16", passed);
#3; Clk = 0; #5; Clk = 1;
passTest(BusA, 32'h15, "Value Not Updated 8", passed);
passTest(BusB, 32'h16, "Value Stayed Same 8", passed);
{RA, RB, RW, BusW, RegWr} = {5'd17, 5'd18, 5'd3, 32'h12345678, 1'b0};#2;
passTest(BusA, 32'h17, "Initial Value Check 17", passed);
passTest(BusB, 32'h18, "Initial Value Check 18", passed);
```

```
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h17, "Value Not Updated 9", passed);
        passTest(BusB, 32'h18, "Value Stayed Same 9", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd19, 5'd20, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h19, "Initial Value Check 19", passed);
        passTest(BusB, 32'h20, "Initial Value Check 20", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h19, "Value Not Updated 10", passed);
        passTest(BusB, 32'h20, "Value Stayed Same 10", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd21, 5'd22, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h21, "Initial Value Check 21", passed);
        passTest(BusB, 32'h22, "Initial Value Check 22", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h21, "Value Not Updated 11", passed);
        passTest(BusB, 32'h22, "Value Stayed Same 11", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd23, 5'd24, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h23, "Initial Value Check 23", passed);
        passTest(BusB, 32'h24, "Initial Value Check 24", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h23, "Value Not Updated 12", passed);
        passTest(BusB, 32'h24, "Value Stayed Same 12", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd25, 5'd26, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h25, "Initial Value Check 25", passed);
        passTest(BusB, 32'h26, "Initial Value Check 26", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h25, "Value Not Updated 13", passed);
        passTest(BusB, 32'h26, "Value Stayed Same 13", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd27, 5'd28, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h27, "Initial Value Check 27", passed);
        passTest(BusB, 32'h28, "Initial Value Check 28", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h27, "Value Not Updated 14", passed);
        passTest(BusB, 32'h28, "Value Stayed Same 14", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd29, 5'd30, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h29, "Initial Value Check 29", passed);
        passTest(BusB, 32'h30, "Initial Value Check 30", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h29, "Value Not Updated 15", passed);
        passTest(BusB, 32'h30, "Value Stayed Same 15", passed);
        {RA, RB, RW, BusW, RegWr} = {5'd31, 5'd32, 5'd3, 32'h12345678, 1'b0};#2;
        passTest(BusA, 32'h31, "Initial Value Check 31", passed);
        #3; Clk = 0; #5; Clk = 1;
        passTest(BusA, 32'h31, "Value Not Updated 16", passed);
        allPassed(passed, 68);
end
endmodule
```

Simulation Output Waveform: (Register File)



Simulation Output Logs: (Register File)

```
restart
INFO: [Simtcl 6-17] Simulation restarted
run all
               Initial $0 Check 1 passed
               Initial $0 Check 2 passed
               Initial $0 Check 3 passed
               Initial $0 Check 4 passed
               $0 Stays 0 Check 1 passed
               $0 Stays 0 Check 2 passed
            Initial Value Check 1 passed
            Initial Value Check 2 passed
                  Value Updated 1 passed
              Value Stayed Same 1 passed
            Initial Value Check 3 passed
            Initial Value Check 4 passed
              Value Not Updated 2 passed
              Value Stayed Same 2 passed
            Initial Value Check 5 passed
            Initial Value Check 6 passed
              Value Not Updated 3 passed
              Value Stayed Same 3 passed
            Initial Value Check 7 passed
            Initial Value Check 8 passed
              Value Not Updated 4 passed
              Value Stayed Same 4 passed
            Initial Value Check 9 passed
           Initial Value Check 10 passed
              Value Not Updated 5 passed
              Value Stayed Same 5 passed
           Initial Value Check 11 passed
           Initial Value Check 12 passed
              Value Not Updated 6 passed
              Value Stayed Same 6 passed
           Initial Value Check 13 passed
           Initial Value Check 14 passed
              Value Not Updated 7 passed
              Value Stayed Same 7 passed
           Initial Value Check 15 passed
           Initial Value Check 16 passed
              Value Not Updated 8 passed
              Value Stayed Same 8 passed
           Initial Value Check 17 passed
           Initial Value Check 18 passed
              Value Not Updated 9 passed
```

```
              Value Stayed Same 9 passed
            Initial Value Check 19 passed
            Initial Value Check 20 passed
              Value Not Updated 10 passed
              Value Stayed Same 10 passed
            Initial Value Check 21 passed
            Initial Value Check 22 passed
              Value Not Updated 11 passed
              Value Stayed Same 11 passed
            Initial Value Check 23 passed
            Initial Value Check 24 passed
              Value Not Updated 12 passed
              Value Stayed Same 12 passed
            Initial Value Check 25 passed
            Initial Value Check 26 passed
              Value Not Updated 13 passed
              Value Stayed Same 13 passed
            Initial Value Check 27 passed
            Initial Value Check 28 passed
              Value Not Updated 14 passed
              Value Stayed Same 14 passed
            Initial Value Check 29 passed
            Initial Value Check 30 passed
              Value Not Updated 15 passed
              Value Stayed Same 15 passed
            Initial Value Check 31 passed
              Value Not Updated 16 passed
All tests passed
```

(c) In Xilinx, synthesize your register file with the following device properties:

Set the device properties to the following:
Device Family: Virtex5
Device: XC5VLX110T
Package: ff1136
Speed Grade: -1

Synthesis Report: (Register File)

```
--------------------------------------------------------------------------------

Start Writing Synthesis Report

--------------------------------------------------------------------------------

Report BlackBoxes:

+-+--------------+----------+

| |BlackBox name |Instances |

+-+--------------+----------+

+-+--------------+----------+


Report Cell Usage:

+------+-------+------+

|      |Cell   |Count |

+------+-------+------+
```

```
|1     |BUFG   |     1|
|2     |LUT1   |     1|
|3     |LUT6   |     1|
|4     |RAM32M |    12|
|5     |IBUF   |    49|
|6     |OBUF   |    64|
+------+-------+------+


Report Instance Areas:
+------+---------+-------+------+
|      |Instance |Module |Cells |
+------+---------+-------+------+
|1     |top      |       |   128|
+------+---------+-------+------+
```

Finished Writing Synthesis Report : Time (s): cpu = 00:00:36 ; elapsed = 00:03:24 . Memory (MB): peak = 1702.172 ; gain = 387.812 ; free physical = 185 ; free virtual = 16107

Synthesis finished with 0 errors, 0 critical warnings and 0 warnings.

Synthesis Optimization Runtime : Time (s): cpu = 00:00:36 ; elapsed = 00:03:24 . Memory (MB): peak = 1702.172 ; gain = 387.812 ; free physical = 183 ; free virtual = 16109

Synthesis Optimization Complete : Time (s): cpu = 00:00:36 ; elapsed = 00:03:24 . Memory (MB): peak = 1702.176 ; gain = 387.812 ; free physical = 192 ; free virtual = 16118

INFO: [Project 1-571] Translating synthesized netlist

INFO: [Netlist 29-17] Analyzing 61 Unisim elements for replacement

INFO: [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds

INFO: [Project 1-570] Preparing netlist for logic optimization

INFO: [Opt 31-138] Pushed 1 inverter(s) to 12 load pin(s).

INFO: [Project 1-111] Unisim Transformation Summary:

  A total of 12 instances were transformed.

  RAM32M => RAM32M (inverted pins: WCLK) (RAMD32, RAMD32, RAMD32, RAMD32, RAMD32, RAMD32, RAMS32, RAMS32): 12 instances

INFO: [Common 17-83] Releasing license: Synthesis

12 Infos, 0 Warnings, 0 Critical Warnings and 0 Errors encountered.

synth_design completed successfully

synth_design: Time (s): cpu = 00:00:39 ; elapsed = 00:03:27 . Memory (MB): peak = 1794.816 ; gain = 517.383 ; free physical = 229 ; free virtual = 16059

INFO: [Common 17-1381] The checkpoint '/home/grads/p/pranav_anantharam/lab3_register_file/project_1/project_1.runs/synth_1/RegisterFile.dcp' has been generated.

```
INFO: [runtcl-4] Executing : report_utilization -file RegisterFile_utilization_synth.rpt -pb
RegisterFile_utilization_synth.pb

report_utilization: Time (s): cpu = 00:00:00.27 ; elapsed = 00:00:00.32 . Memory (MB): peak =
1818.840 ; gain = 0.000 ; free physical = 228 ; free virtual = 16059

INFO: [Common 17-206] Exiting Vivado at Fri Sep 22 15:59:24 2023...
```

---

2. Design and simulate a simple data memory module.

(a) Describe a data memory module file in behavioral Verilog.

Design Source code filename: DataMemory.v

```
`timescale 1ns / 1ps


module DataMemory( output reg [31:0] ReadData, input [5:0] Address, input [31:0] WriteData,
input MemoryRead, input MemoryWrite, input Clock );
 // ReadData - Data output
 // Addresss - Address bus for read/write
 // WriteData - Data input
 // MemoryRead - Read signal
 // MemoryWrite - Write signal
 // Clock - Clock signal

 // Each word is 32 bits
 // Hence, we need 64 words for 256 bytes storage
 // 64 x 32 bits = 2048 bits = 256 bytes

reg [31:0] data_memory[63:0]; // 64 elements x 32 bit wide element

// Writes are synchronous on negative edge of clock
always @(negedge Clock) begin
    // Check if write signal is high
    if ( MemoryWrite == 1'b1 ) begin
        data_memory[Address] <= WriteData;
    end
end
```

```
// Reads are synchronous on positive edge of clock
always @(posedge Clock) begin
    // Check if read signal is high
    if ( MemoryRead == 1'b1 ) begin
        ReadData <= data_memory[Address];
    end
end
endmodule
```

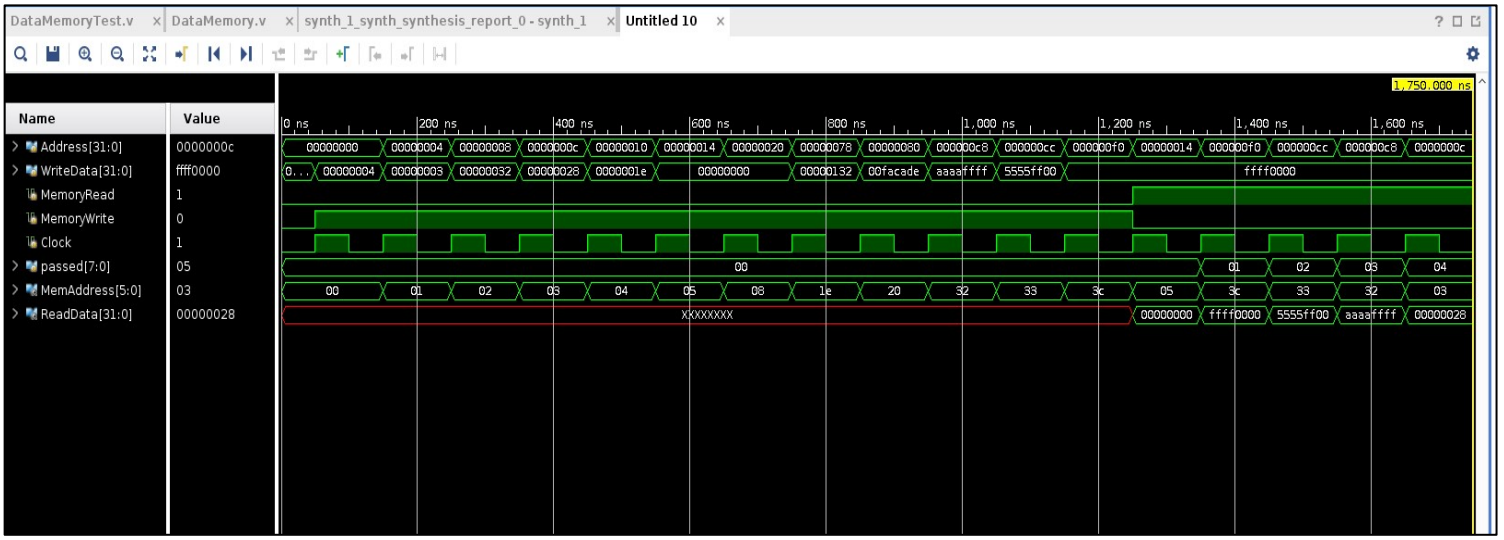
(b) Test the behavioral model of the memory module using the test bench provided.

Testbench code filename: DataMemoryTest.v

```
`timescale 1ns / 1ps

`define STRLEN 32
module DataMemoryTest_v;


      task passTest;
            input [31:0] actualOut, expectedOut;
            input [`STRLEN*8:0] testType;
            inout [7:0] passed;

            if(actualOut == expectedOut) begin $display ("%s passed", testType); passed =
passed + 1; end
            else $display ("%s failed: %d should be %d", testType, actualOut, expectedOut);
      endtask

      task allPassed;
            input [7:0] passed;
            input [7:0] numTests;

            if(passed == numTests) $display ("All tests passed");
            else $display("Some tests failed");
      endtask


      // Inputs
      reg [31:0] Address;
      reg [31:0] WriteData;
      reg MemoryRead;
      reg MemoryWrite;
      reg Clock;
      reg [7:0] passed;

      //intermediate nets
      wire [5:0] MemAddress;

      // Outputs
      wire [31:0] ReadData;

      assign MemAddress = Address[7:2];
```

```
	// Instantiate the Unit Under Test (UUT)
	DataMemory uut (
		.ReadData(ReadData),
		.Address(MemAddress),
		.WriteData(WriteData),
		.MemoryRead(MemoryRead),
		.MemoryWrite(MemoryWrite),
		.Clock(Clock)
	);

	initial begin
		// Initialize Inputs
		Address = 0;
		WriteData = 0;
		MemoryRead = 0;
		MemoryWrite = 0;
		Clock = 0;
		passed = 0;

		// Add stimulus here
		$display("Init Memory with some useful data");
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h0, 32'h4, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h4, 32'h3, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h8, 32'd50, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'hc, 32'd40, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h10, 32'd30, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h14, 32'h0, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h20, 32'h0, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h78, 32'h132, 2'h2};#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'h80, 32'd16435934, 2'h2};
#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'hc8,32'haaaaffff,2'h2};
#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'hcc, 32'd1431699200, 2'h2};
#50 Clock=0;
#50 Clock=1;{Address,WriteData,MemoryWrite,MemoryRead}={32'hf0, 32'hffff0000, 2'h2};
#50 Clock=0;
		#50 Clock = 1;
		{Address, WriteData, MemoryWrite, MemoryRead} = {32'h14, 32'hffff0000, 2'h1};
		#50 Clock = 0; #50 Clock = 1;
		passTest(ReadData, 32'h0, "Read address 0x14", passed);
		{Address, WriteData, MemoryWrite, MemoryRead} = {32'hf0, 32'hffff0000, 2'h1};
		#50 Clock = 0; #50 Clock = 1;
		passTest(ReadData, 32'hffff0000, "Read address 0xf0", passed);
		{Address, WriteData, MemoryWrite, MemoryRead} = {32'hcc, 32'hffff0000, 2'h1};
		#50 Clock = 0; #50 Clock = 1;
		passTest(ReadData, 32'd1431699200, "Read address 0xcc", passed);
		{Address, WriteData, MemoryWrite, MemoryRead} = {32'hc8, 32'hffff0000, 2'h1};
		#50 Clock = 0; #50 Clock = 1;
		passTest(ReadData, 32'haaaaffff, "Read address 0xc8", passed);
		{Address, WriteData, MemoryWrite, MemoryRead} = {32'hc, 32'hffff0000, 2'h1};
		#50 Clock = 0; #50 Clock = 1;
		passTest(ReadData, 32'd40, "Read address 0xc", passed);
		allPassed(passed, 5);
	end
endmodule
```

Simulation Output Waveform: (Data Memory Module)



Simulation Output Logs: (Data Memory Module)

```
restart
INFO: [Simtcl 6-17] Simulation restarted
run all
Init Memory with some useful data
                Read address 0x14 passed
                Read address 0xf0 passed
                Read address 0xcc passed
                Read address 0xc8 passed
                 Read address 0xc passed
All tests passed
```

(c) In Xilinx, synthesize your memory module with the aforementioned device properties.

Synthesis Report: (Data Memory Module)

```
---------------------------------------------------------------------------------

Start Writing Synthesis Report

---------------------------------------------------------------------------------


Report BlackBoxes:

+-+--------------+----------+

| |BlackBox name |Instances |

+-+--------------+----------+

+-+--------------+----------+
```

Report Cell Usage:

	Cell	Count
1	BUFG	1
2	RAMB18E1	1
3	IBUF	41
4	OBUF	32

Report Instance Areas:

	Instance	Module	Cells
1	top		75

Finished Writing Synthesis Report : Time (s): cpu = 00:00:36 ; elapsed = 00:03:25 . Memory (MB): peak = 1719.164 ; gain = 404.805 ; free physical = 202 ; free virtual = 16208

Synthesis finished with 0 errors, 0 critical warnings and 0 warnings.

Synthesis Optimization Runtime : Time (s): cpu = 00:00:36 ; elapsed = 00:03:25 . Memory (MB): peak = 1719.164 ; gain = 404.805 ; free physical = 204 ; free virtual = 16211

Synthesis Optimization Complete : Time (s): cpu = 00:00:36 ; elapsed = 00:03:25 . Memory (MB): peak = 1719.168 ; gain = 404.805 ; free physical = 211 ; free virtual = 16218

INFO: [Project 1-571] Translating synthesized netlist

INFO: [Netlist 29-17] Analyzing 42 Unisim elements for replacement

INFO: [Netlist 29-28] Unisim Transformation completed in 0 CPU seconds

INFO: [Project 1-570] Preparing netlist for logic optimization

INFO: [Opt 31-138] Pushed 0 inverter(s) to 0 load pin(s).

INFO: [Project 1-111] Unisim Transformation Summary:

No Unisim elements were transformed.

INFO: [Common 17-83] Releasing license: Synthesis

13 Infos, 0 Warnings, 0 Critical Warnings and 0 Errors encountered.

synth_design completed successfully

synth_design: Time (s): cpu = 00:00:39 ; elapsed = 00:03:28 . Memory (MB): peak = 1808.621 ; gain = 531.188 ; free physical = 224 ; free virtual = 16172

INFO: [Common 17-1381] The checkpoint '/home/grads/p/pranav_anantharam/lab3_data_memory/project_2/project_2.runs/synth_1/DataMemory.dcp' has been generated.

INFO: [runtcl-4] Executing : report_utilization -file DataMemory_utilization_synth.rpt -pb DataMemory_utilization_synth.pb

```
report_utilization: Time (s): cpu = 00:00:00.30 ; elapsed = 00:00:00.34 . Memory (MB): peak =
1832.645 ; gain = 0.000 ; free physical = 224 ; free virtual = 16171

INFO: [Common 17-206] Exiting Vivado at Fri Sep 22 15:54:32 2023...
```

---

4. Answer the following review questions.

(a) Suppose we could make the data memory module dual-ported (i.e. two separately addressed read ports). Could one design a microarchitecture which uses the dual-ported memory module in order to eliminate the register file? If so, would this be a good design? Explain your answer.

Answer:

Data memory module provides the microprocessor with a means of storing large sizes of data on a long-term basis. A dual-ported data memory module will support two read operations simultaneously on the same clock cycle. However, these read operations are synchronous on the positive edge of clock.

Register files provide high speed storage within the microprocessor. In order to perform high speed storage, register files support asynchronous read operations. Hence, register files cannot be eliminated by dual-ported memory module microarchitecture.

(b) What is the purpose of the data memory's MemRead, (i.e. the read enable signal)? Is it functionally necessary? What advantages might it provide? Why do we not implement a similar signal for the register file?

Answer:

The MemRead or read enable signal is used to enable/disable read operations from the data memory. It is also used to differentiate between read and write modes. The MemRead signal is not required in register file since the processor issues commands to perform read operations from the register file. Moreover, register file is of smaller size implying that there is a very low chance of reading unwanted data from the register file. Hence, MemRead signal is not required to be implemented for register files.

(c) Elaborate on the term synthesizable. What sort of constructs in Verilog are not synthesizable?

Answer:

The term 'synthesizable' refers to the code in hardware description languages that can be converted into a netlist of logic gates and other hardware elements by a synthesis tool. The goal of writing synthesizable code is to describe the behavior and functionality of digital logic circuits in a way that can be converted into a physical implementation on an FPGA or an ASIC.

The following constructs in Verilog are not synthesizable:

- Timing delays using '#'
- System tasks and functions – $display, $random, $finish
- Timing controls – wait, forever
- Fork and join statements
