

**lab report 5**

Andrew Nady

900184042

Question 1

Technical summary

Steps:

* In the experiment 1, we added in the source code some testcases, then we wrote the testbench to test the code.
* In experiment 2, we wrote the testbench that can write and then read from the data memory

|  |
| --- |
| mem[**0**]=**32'd17**;  mem[**1**]=**32'd9**;  mem[**2**]=**32'd25**;  mem[**3**]=**32'd55**;  mem[**4**]=**32'd40**; |

Functionalities:

* In experiment 1, it is the instruction memory which will contain the instructions that will be implemented and executed.
* In experiment 1, it is the Data memory which will contain some data that were stored and we can load any data from it

Components:

* Verilog
* VNC

Question 2:

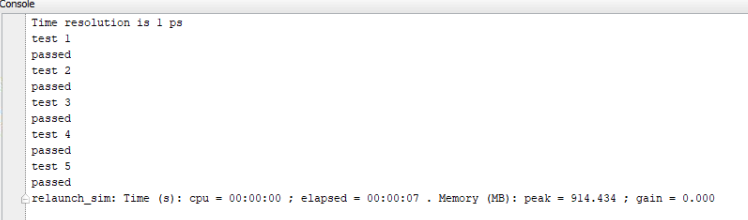
Code:

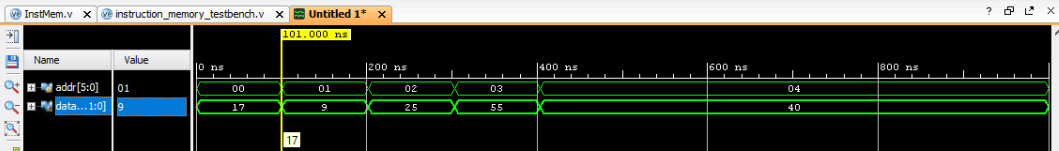
* Note: I attached all the code in a separate folder

Question 3:

Results

Exp1:

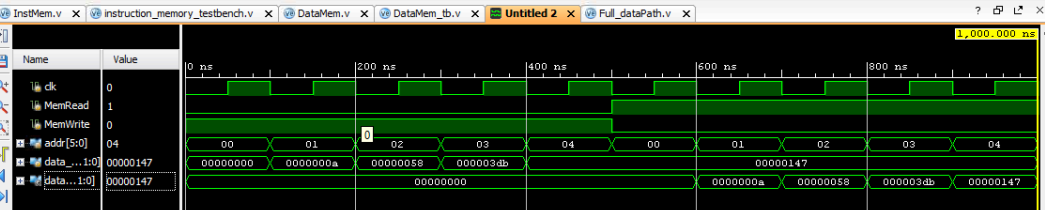




* As I told later that we added 5 test cases in the source file and in the testbench we tried to read them and we successfully read them.

|  |
| --- |
| mem[**0**]=**32'd17**;  mem[**1**]=**32'd9**;  mem[**2**]=**32'd25**;  mem[**3**]=**32'd55**;  mem[**4**]=**32'd40**; |

Exp2:



* We wrote in the data memory using writing flags 5 times with 5 different values
* Then we tried to read them all again, and we successful read and wrote them correctly.

Question 4:

Exercise 3

|  |
| --- |
| `timescale 1ns / 1ps  module Full\_dataPath( input clk, input rst );  wire [31:0] PCin, PCout0,PCout1,PCmux;  wire cout1\_ripple1,cout1\_ripple2;  wire [31:0]ints\_out;  wire [31:0]readdata1, readdata2 ;  reg [31:0] gen\_out; //should be reg || wire  reg branch;  reg memread;  reg memtoreg;  reg [1:0]aluop;  reg memwrite;  reg alusrc;  reg regwrite;  reg [3:0]aluS;  wire [31:0]outmux\_input\_alu;  reg [31:0]ALU\_result;  wire zero;  wire [31:0] dataMem\_out;  wire [31:0] writingData;  wire [31:0] shiftout;  //instantiate 32bitreg PCin PCout  ripple\_carry ripplecar0(PCin,  32'd4,  0,  PCout0,  cout1\_ripple1  );  //adder tany  ripple\_carry ripplecar1(PCin,  shiftout,  0,  PCout1,  cout1\_ripple2  );  //mux  ThirtytwoMUX mux3(PCout0,PCout1,branch&zero ,PCmux);  //register (PCmux, PCin) //reload the PC  //instmem  InstMem instmem(PCin[7:2],ints\_out);  //control unit  Control\_unit controlunit(ints\_out[6:0], branch,  memread, memtoreg,  aluop, memwrite, alusrc, regwrite );  //alu\_control  ALU\_Control aluControl(aluop, ints\_out[14:12],ints\_out[30] ,aluS);  //registerfile  RegisterFile regfile( clk, rst,  ints\_out[19:15], ints\_out[24:20], ints\_out[11:7],  writingData,  regwrite,  readdata1, readdata2 );  //immGenerator  immediate\_generator immgen( gen\_out, ints\_out );  //shifting left  shiftLeft(gen\_out ,shiftout);  //mux  ThirtytwoMUX mux (readdata2,gen\_out,alusrc ,outmux\_input\_alu);  //ALU  ALU alu(  readdata1, outmux\_input\_alu,  aluS,  ALU\_result, zero );  //dataMem  DataMem datamem( clk, memread, memwrite,  ALU\_result[7:2] , readdata2, dataMem\_out);  //mux after the datamem  ThirtytwoMUX(dataMem\_out,dataMem\_out,memtoreg ,writingData);  endmodule |

* We still need to add a register for the PC
* And modify some parts that we used the PC as not a register