ELD LAB HW 6

Aditya Gautam

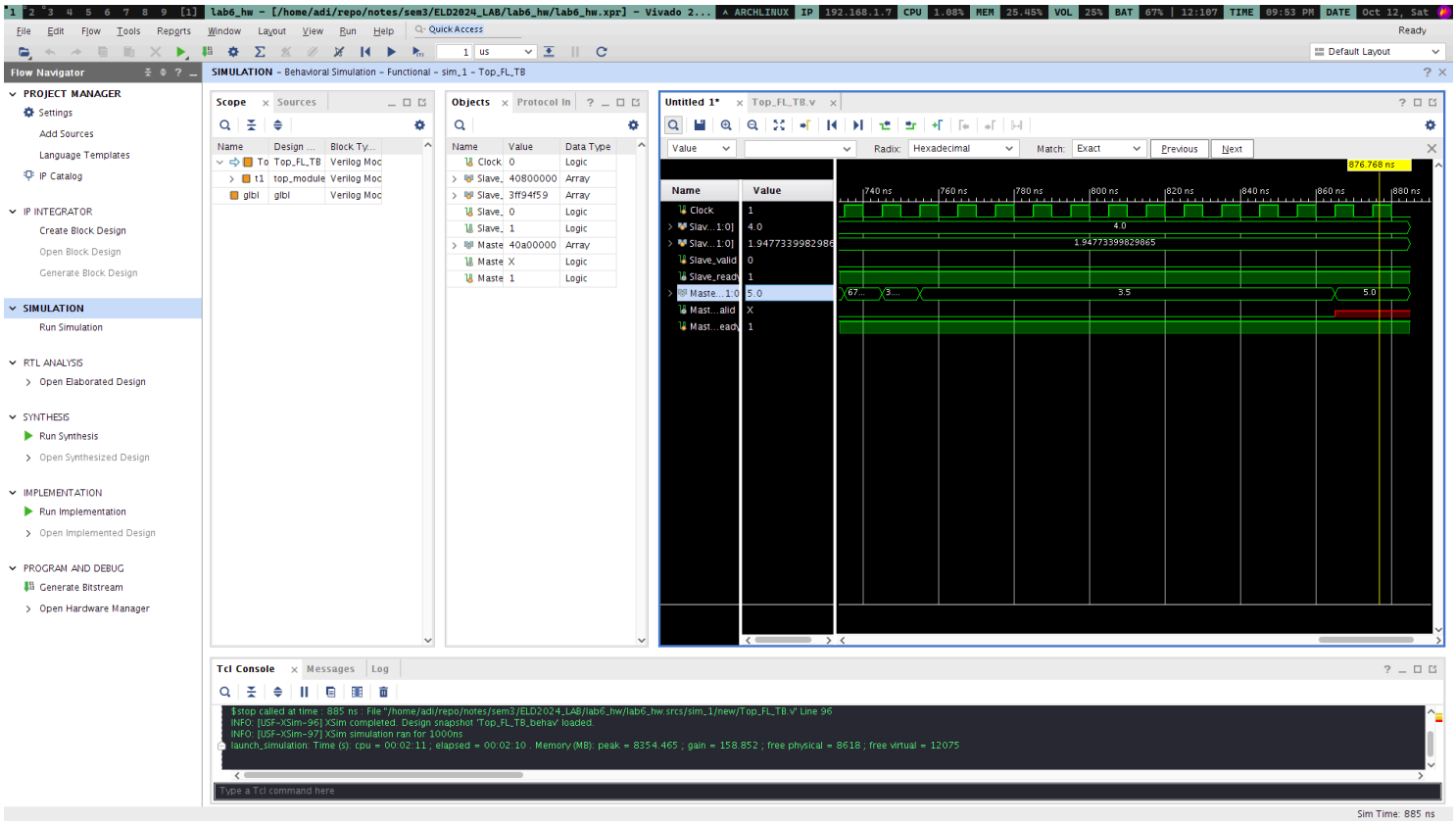
2023043

- TESTBENCH

Let x = 4 and y = 1.94773404105

So, sqrt(x) = 2 and 1 / ln(y) = 1.5

and z = 2 + 1.5 + 1.5 = 5

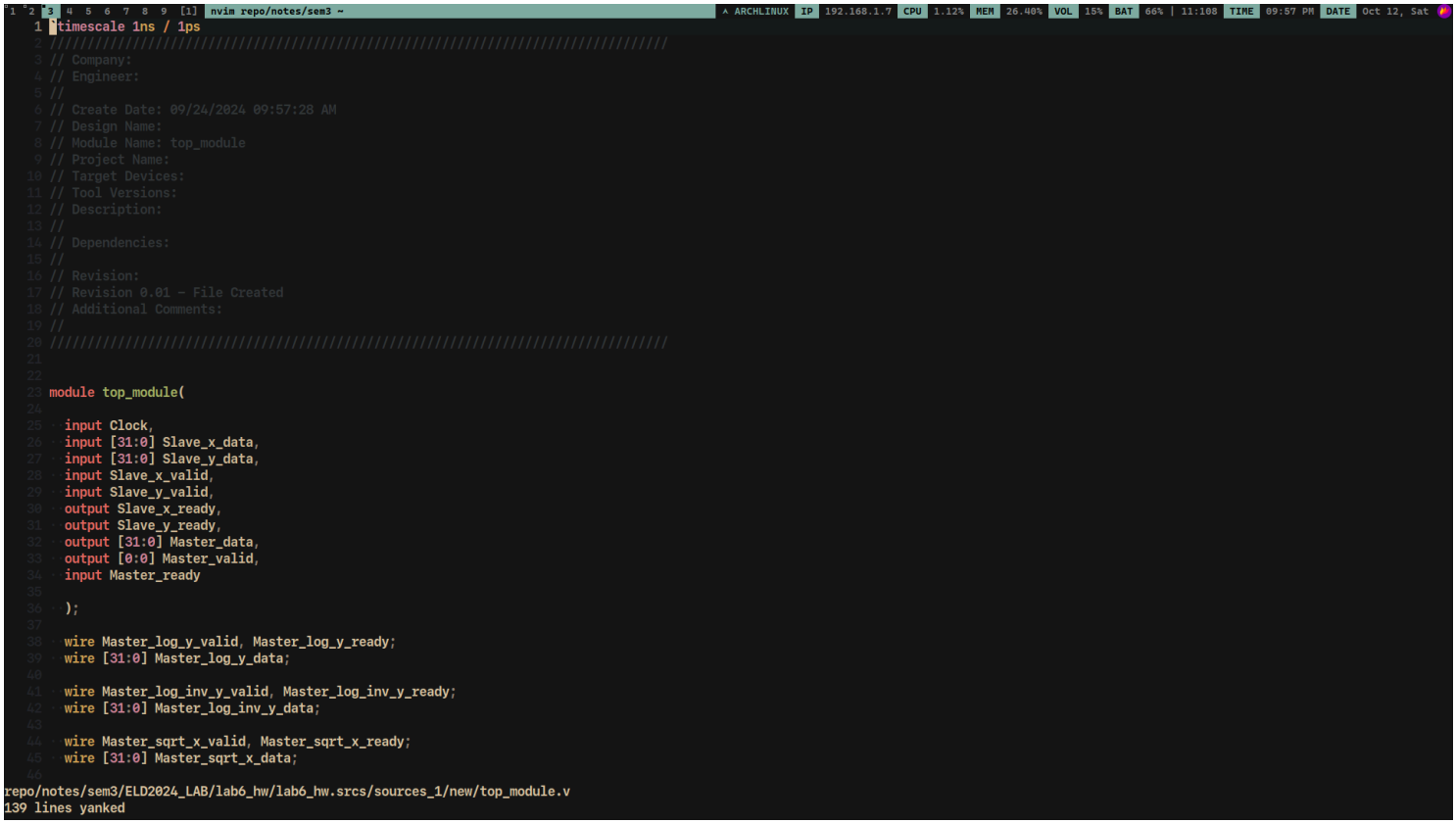


- Top_module.v (main source code)

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 09/24/2024 09:57:28 AM
// Design Name:
// Module Name: top_module
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
```

```
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module top_module(

  input Clock,
  input [31:0] Slave_x_data,
  input [31:0] Slave_y_data,
  input Slave_x_valid,
  input Slave_y_valid,
  output Slave_x_ready,
  output Slave_y_ready,
  output [31:0] Master_data,
  output [0:0] Master_valid,
  input Master_ready

  );

  wire Master_log_y_valid, Master_log_y_ready;
  wire [31:0] Master_log_y_data;

  wire Master_log_inv_y_valid, Master_log_inv_y_ready;
  wire [31:0] Master_log_inv_y_data;

  wire Master_sqrt_x_valid, Master_sqrt_x_ready;
  wire [31:0] Master_sqrt_x_data;

  wire Master_add_x_y_valid, Master_add_x_y_ready;
  wire [31:0] Master_add_x_y_data;

  wire Master_const_valid, Master_const_ready;
  wire [31:0] Master_const_data = 32'b00111111111000000000000000000000;

  /*
  * x → sqrt(x) [Master_sqrt_x_data]
  * y → ln(y) → 1/ln(y) [Master_log_y_data]
  * add 1.5 with the sum of the output of last two
  */

  // X
  FL_sqrt sqrt1 (

        .aclk(Clock),                                 // input wire aclk

        .s_axis_a_tvalid(Slave_x_valid),              // input wire s_axis_a_tvalid
        .s_axis_a_tready(Slave_x_ready),              // output wire s_axis_a_tready
        .s_axis_a_tdata(Slave_x_data),                // input wire [31 : 0] s_axis_a_tdata

        .m_axis_result_tvalid(Master_sqrt_x_valid),  // output wire m_axis_result_tvalid
        .m_axis_result_tready(Master_sqrt_x_ready),  // input wire m_axis_result_tready
        .m_axis_result_tdata(Master_sqrt_x_data)     // output wire [31 : 0] m_axis_result_tdata

  );

  // Y
  FL_log log1 (

        .aclk(Clock),                                 // input wire aclk

        .s_axis_a_tvalid(Slave_y_valid),              // input wire s_axis_a_tvalid
        .s_axis_a_tready(Slave_y_ready),              // output wire s_axis_a_tready
        .s_axis_a_tdata(Slave_y_data),                // input wire [31 : 0] s_axis_a_tdata

        .m_axis_result_tvalid(Master_log_y_valid),  // output wire m_axis_result_tvalid
        .m_axis_result_tready(Master_log_y_ready),  // input wire m_axis_result_tready
        .m_axis_result_tdata(Master_log_y_data)      // output wire [31 : 0] m_axis_result_tdata
```

```
  );

  FL_inverse inv1 (

        .aclk(Clock),                                 // input wire aclk

        .s_axis_a_tvalid(Master_log_y_valid),         // input wire s_axis_a_tvalid
        .s_axis_a_tready(Master_log_y_ready),         // output wire s_axis_a_tready
        .s_axis_a_tdata(Master_log_y_data),           // input wire [31 : 0] s_axis_a_tdata

        .m_axis_result_tvalid(Master_log_inv_y_valid),  // output wire m_axis_result_tvalid
        .m_axis_result_tready(Master_log_inv_y_ready),  // input wire m_axis_result_tready
        .m_axis_result_tdata(Master_log_inv_y_data)  // output wire [31 : 0] m_axis_result_tdata

  );

  FL_add add_x_y (

        .aclk(Clock),                                 // input wire aclk

        .s_axis_a_tvalid(Master_log_inv_y_valid),           // input wire s_axis_a_tvalid
        .s_axis_a_tready(Master_log_inv_y_ready),           // output wire s_axis_a_tready
        .s_axis_a_tdata(Master_log_inv_y_data),             // input wire [31 : 0] s_axis_a_tdata

        .s_axis_b_tvalid(Master_sqrt_x_valid),              // input wire s_axis_b_tvalid
        .s_axis_b_tready(Master_sqrt_x_ready),              // output wire s_axis_b_tready
        .s_axis_b_tdata(Master_sqrt_x_data),                // input wire [31 : 0] s_axis_b_tdata

        .m_axis_result_tvalid(Master_add_x_y_valid),  // output wire m_axis_result_tvalid
        .m_axis_result_tready(Master_add_x_y_ready),  // input wire m_axis_result_tready
        .m_axis_result_tdata(Master_add_x_y_data)    // output wire [31 : 0] m_axis_result_tdata

  );

  FL_add add_final (

        .aclk(Clock),                                 // input wire aclk

        .s_axis_a_tvalid(Master_add_x_y_valid),             // input wire s_axis_a_tvalid
        .s_axis_a_tready(Master_add_x_y_ready),             // output wire s_axis_a_tready
        .s_axis_a_tdata(Master_add_x_y_data),               // input wire [31 : 0] s_axis_a_tdata

        .s_axis_b_tvalid(Master_const_valid),         // input wire s_axis_b_tvalid
        .s_axis_b_tready(Master_const_ready),         // output wire s_axis_b_tready
        .s_axis_b_tdata(Master_const_data),           // input wire [31 : 0] s_axis_b_tdata

        .m_axis_result_tvalid(Master_valid),  // output wire m_axis_result_tvalid
        .m_axis_result_tready(Master_ready),  // input wire m_axis_result_tready
        .m_axis_result_tdata(Master_data)    // output wire [31 : 0] m_axis_result_tdata

  );

endmodule
```



- Top_FL_TB.v (test bench)

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 09/24/2024 10:07:23 AM
// Design Name:
// Module Name: Top_FL_TB
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module Top_FL_TB(

  );

  // inputs → reg
  // outputs → wire

  reg Clock;
  reg [31:0] Slave_x_data;
  reg [31:0] Slave_y_data;
  reg Slave_valid;
  wire Slave_ready;
```

```
  wire [31:0] Master_data;
  wire Master_valid;
  reg Master_ready;

  top_module t1(

        .Clock(Clock),

        .Slave_x_data(Slave_x_data),
        .Slave_x_valid(Slave_valid),
        .Slave_x_ready(Slave_ready),

        .Slave_y_data(Slave_y_data),
        .Slave_y_valid(Slave_valid),
        .Slave_y_ready(Slave_ready),

        .Master_data(Master_data),
        .Master_valid(Master_valid),
        .Master_ready(Master_ready)
  );

  initial
        begin
        Clock = 0;
        Slave_valid = 0;
        Slave_x_data = 0;
        Slave_y_data = 0;
        Master_ready = 1;
        // test bench is accepting data from the inverse block
        // hence we set Master_ready = 1 because it is always ready to accept
        // data
        end

  always
        begin
        #5 Clock = ~Clock;

        // Put data on data bus
        // Put valid signal to one
        // make the valid = 1 till the ready = 1
        // make valid = 0 at the next positive edge clock cycle after handshake (i.e. ready = 1)
        //      in order to prevent garbage values from going in

        end

  // setting x = 4 and y = 1.94773404105 so that i could get 1/ln(y) = 1.5 to get
  // z = 2 + 1.5 + 1.5 = 7

  initial
  begin
        #50 Slave_x_data = 32'b01000000100000000000000000000000;
        #50 Slave_y_data = 32'b00111111111111001010011110101100 1;
        Slave_valid = 1;
        while(Slave_ready == 0)
        Slave_valid = 1;
        // at the next clock cycle transmission happens and the valid should be
        // set to zero in order to prevent any garbage values from going in
        #5 Slave_valid = 0;
        // waiting for the inverse IP to set M_valid once the operation has been
        // carried out
        @(posedge Master_valid);
        #20 $stop;
  end

endmodule
```

```
1 2 3 4 5 6 7 8 9 [1] nvim repo/notes/sem3 ~  ARCHLINUX IP 192.168.1.7 CPU 1.09% MEM 26.13% VOL 20% BAT 67% | 12:76 TIME 09:55 PM DATE Oct 12, Sat
```

```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 09/24/2024 10:07:23 AM
// Design Name:
// Module Name: Top_FL_TB
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
//////////////////////////////////////////////////////////////////////////////////


module Top_FL_TB(

    );

    // inputs -> reg
    // outputs -> wire

    reg Clock;
    reg [31:0] Slave_x_data;
    reg [31:0] Slave_y_data;
    reg Slave_valid;
    wire Slave_ready;
    wire [31:0] Master_data;
    wire Master_valid;
    reg Master_ready;

    top_module t1(

        .Clock(Clock),

        .Slave_x_data(Slave_x_data),
        .Slave_x_valid(Slave_valid),
        .Slave_x_ready(Slave_ready),

```

```
repo/notes/sem3/ELD2024_LAB/lab6_hw/lab6_hw.srcs/sim_1/new/Top_FL_TB.v
99 lines yanked
```