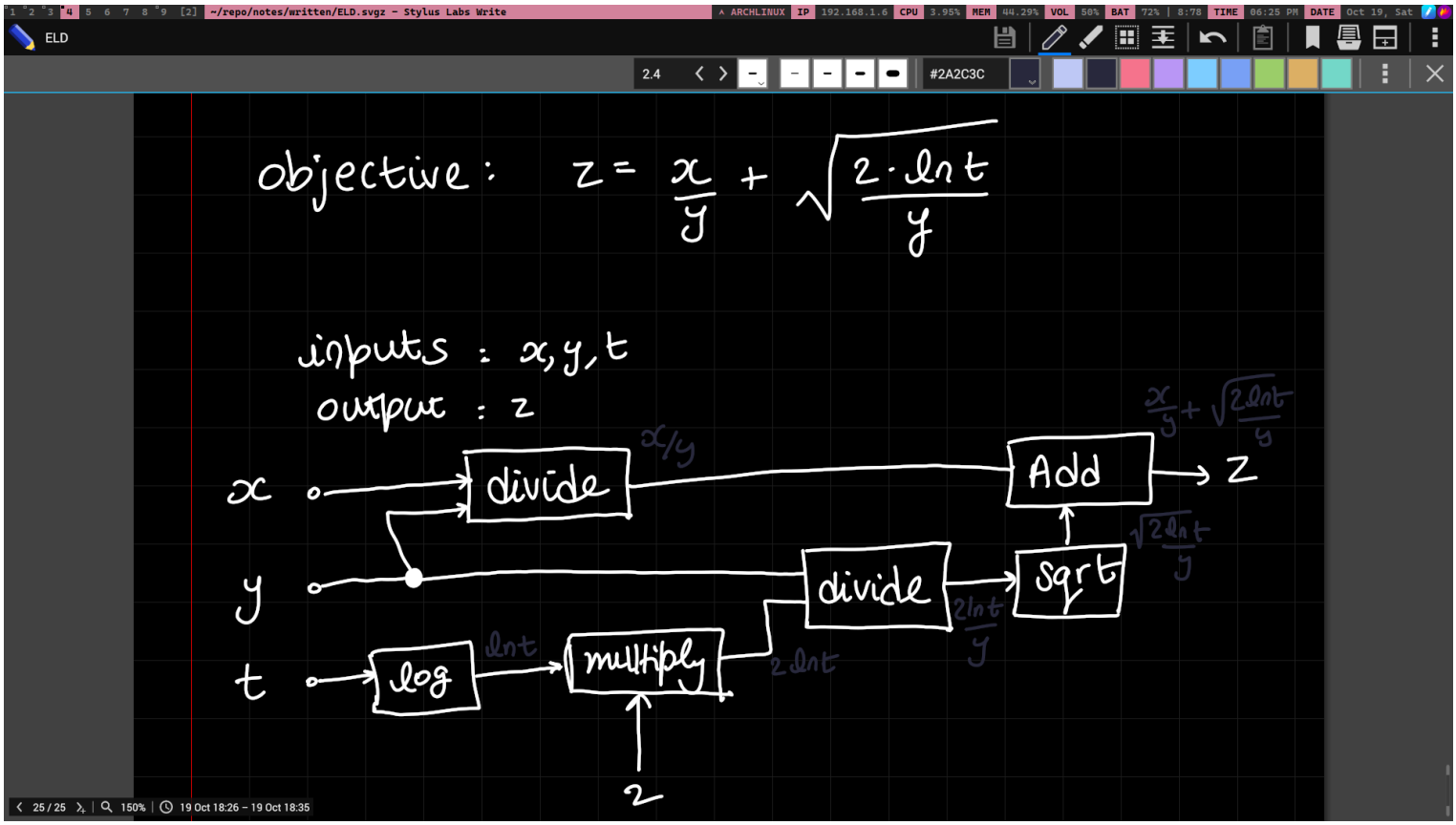
ELD LAB 7 HOMEWORK 2

Aditya Gautam
2023043

Project Idea



Project Structure

```
top_module.v
|__ FL_divide
|__ FL_log
|__ FL_multiply
|__ FL_divide (for the second division operation)
|__ FL_sqrt
|__ FL_add
```

Code

- top_module.v

```
`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 Slave_x_valid,
  output Slave_x_ready,

  input [31:0] Slave_y_data,
  input Slave_y_valid,
  output Slave_y_ready,

  input [31:0] Slave_t_data,
  input Slave_t_valid,
  output Slave_t_ready,

  output [31:0] Master_data,
  output Master_valid,
  input Master_ready

  );

  wire intermediate_x_by_y_valid;
  wire intermediate_x_by_y_ready;
  wire [31:0] intermediate_x_by_y_data;
```

```
wire intermediate_ln_t_valid;
wire intermediate_ln_t_ready;
wire [31:0] intermediate_ln_t_data;

wire intermediate_two_ln_t_valid;
wire intermediate_two_ln_t_ready;
wire [31:0] intermediate_two_ln_t_data;

wire two_data_valid;
wire two_data_ready;
wire [31:0] two_data = 32'b01000000000000000000000000000000;

wire intermediate_two_ln_t_by_y_valid;
wire intermediate_two_ln_t_by_y_ready;
wire [31:0] intermediate_two_ln_t_by_y_data;

wire intermediate_sqrt_two_ln_t_by_y_valid;
wire intermediate_sqrt_two_ln_t_by_y_ready;
wire [31:0] intermediate_sqrt_two_ln_t_by_y_data;

// x / y
FL_divide divide_x_y (
     .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

     .s_axis_b_tvalid(Slave_y_valid),               // input wire s_axis_b_tvalid
     .s_axis_b_tready(Slave_y_ready),               // output wire s_axis_b_tready
     .s_axis_b_tdata(Slave_y_data),                 // input wire [31 : 0] s_axis_b_tdata

     .m_axis_result_tvalid(intermediate_x_by_y_valid),  // output wire m_axis_result_tvalid
     .m_axis_result_tready(intermediate_x_by_y_ready),  // input wire m_axis_result_tready
     .m_axis_result_tdata(intermediate_x_by_y_data) // output wire [31 : 0] m_axis_result_tdata
);

// ln(t)
FL_log ln_t (
     .aclk(Clock),                                  // input wire aclk

     .s_axis_a_tvalid(Slave_t_valid),               // input wire s_axis_a_tvalid
     .s_axis_a_tready(Slave_t_ready),               // output wire s_axis_a_tready
     .s_axis_a_tdata(Slave_t_data),                 // input wire [31 : 0] s_axis_a_tdata

     .m_axis_result_tvalid(intermediate_ln_t_valid),  // output wire m_axis_result_tvalid
     .m_axis_result_tready(intermediate_ln_t_ready),  // input wire m_axis_result_tready
     .m_axis_result_tdata(intermediate_ln_t_data)   // output wire [31 : 0] m_axis_result_tdata
);

// 2 * ln(t)
FL_multiply two_ln_t (
     .aclk(Clock),                                  // input wire aclk
```

```
      .s_axis_a_tvalid(two_data_valid),             // input wire s_axis_a_tvalid
      .s_axis_a_tready(two_data_ready),             // output wire s_axis_a_tready
      .s_axis_a_tdata(two_data),             // input wire [31 : 0] s_axis_a_tdata

      .s_axis_b_tvalid(intermediate_ln_t_valid),           // input wire s_axis_b_tvalid
      .s_axis_b_tready(intermediate_ln_t_ready),           // output wire s_axis_b_tready
      .s_axis_b_tdata(intermediate_ln_t_data),             // input wire [31 : 0] s_axis_b_tdata

      .m_axis_result_tvalid(intermediate_two_ln_t_valid),  // output wire m_axis_result_tvalid
      .m_axis_result_tready(intermediate_two_ln_t_ready),  // input wire m_axis_result_tready
      .m_axis_result_tdata(intermediate_two_ln_t_data)     // output wire [31 : 0] m_axis_result_tdata
  );

  // 2 * lnt / y
  FL_divide two_lnt_by_y (
      .aclk(Clock),                                 // input wire aclk

      .s_axis_a_tvalid(intermediate_two_ln_t_valid),             // input wire s_axis_a_tvalid
      .s_axis_a_tready(intermediate_two_ln_t_ready),             // output wire s_axis_a_tready
      .s_axis_a_tdata(intermediate_two_ln_t_data),               // input wire [31 : 0] s_axis_a_tdata

      .s_axis_b_tvalid(Slave_y_valid),              // input wire s_axis_b_tvalid
      .s_axis_b_tready(Slave_y_ready),              // output wire s_axis_b_tready
      .s_axis_b_tdata(Slave_y_data),                // input wire [31 : 0] s_axis_b_tdata

      .m_axis_result_tvalid(intermediate_two_ln_t_by_y_valid),  // output wire m_axis_result_tvalid
      .m_axis_result_tready(intermediate_two_ln_t_by_y_ready),  // input wire m_axis_result_tready
      .m_axis_result_tdata(intermediate_two_ln_t_by_y_data) // output wire [31 : 0] m_axis_result_tdata
  );

  // sqrt(2*ln(t)/y)
  FL_sqrt sqrt_2_lnt_by_y (
      .aclk(Clock),                                 // input wire aclk

      .s_axis_a_tvalid(intermediate_two_ln_t_by_y_valid),         // input wire s_axis_a_tvalid
      .s_axis_a_tready(intermediate_two_ln_t_by_y_ready),         // output wire s_axis_a_tready
      .s_axis_a_tdata(intermediate_two_ln_t_by_y_data),           // input wire [31 : 0] s_axis_a_tdata

      .m_axis_result_tvalid(intermediate_sqrt_two_ln_t_by_y_valid),  // output wire m_axis_result_tvalid
      .m_axis_result_tready(intermediate_sqrt_two_ln_t_by_y_ready),  // input wire m_axis_result_tready
      .m_axis_result_tdata(intermediate_sqrt_two_ln_t_by_y_data)  // output wire [31 : 0]
m_axis_result_tdata
  );

  // sqrt(2*ln(t)/y) + x/y
  FL_add final_step (
      .aclk(Clock),                                 // input wire aclk

      .s_axis_a_tvalid(intermediate_sqrt_two_ln_t_by_y_valid),          // input wire s_axis_a_tvalid
      .s_axis_a_tready(intermediate_sqrt_two_ln_t_by_y_ready),          // output wire s_axis_a_tready
      .s_axis_a_tdata(intermediate_sqrt_two_ln_t_by_y_data),            // input wire [31 : 0]
s_axis_a_tdata
```

```
      .s_axis_b_tvalid(intermediate_x_by_y_valid),          // input wire s_axis_b_tvalid
      .s_axis_b_tready(intermediate_x_by_y_ready),          // output wire s_axis_b_tready
      .s_axis_b_tdata(intermediate_x_by_y_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

/*
* For the test bench, i will have these values so that the final value comes
* out neat →
*
* t = 54.5981500331
* so that ln(t) = 4
* 2*ln(t) = 8
* y = 2
* so that 2*ln(t) / y = 4
* and sqrt of that will be 2
* and x = 16
* so, z = 10
*/
```

```
1 2 3 4 5 6 7 8 9 [2] nvim  ARCHLINUX IP 192.168.1.6 CPU 3.49% MEM 33.25% VOL 0% BAT 44% | 8:61 TIME 07:28 PM DATE Oct 19, Sat
  1 `timescale 1ns / 1ps
  2 //////////////////////////////////////////////////////////////////////////////////
  3 // Company:
  4 // Engineer:
  5 //
  6 // Create Date: 09/24/2024 09:57:28 AM
  7 // Design Name:
  8 // Module Name: top_module
  9 // Project Name:
 10 // Target Devices:
 11 // Tool Versions:
 12 // Description:
 13 //
 14 // Dependencies:
 15 //
 16 // Revision:
 17 // Revision 0.01 - File Created
 18 // Additional Comments:
 19 //
 20 //////////////////////////////////////////////////////////////////////////////////
 21
 22
 23 module top_module(
 24
 25   input Clock,
 26
 27   input [31:0] Slave_x_data,
 28   input Slave_x_valid,
 29   output Slave_x_ready,
 30
 31   input [31:0] Slave_y_data,
 32   input Slave_y_valid,
 33   output Slave_y_ready,
 34
 35   input [31:0] Slave_t_data,
 36   input Slave_t_valid,
 37   output Slave_t_ready,
 38
 39   output [31:0] Master_data,
 40   output Master_valid,
 41   input Master_ready
 42
 43   );
 44
 45   wire intermediate_x_by_y_valid;
 46   wire intermediate_x_by_y_ready;
repo/notes/sem3/ELD2024_LAB/lab7_hw_2/lab7_hw_2.srcs/sources_1/new/top_module.v
177 lines yanked
```

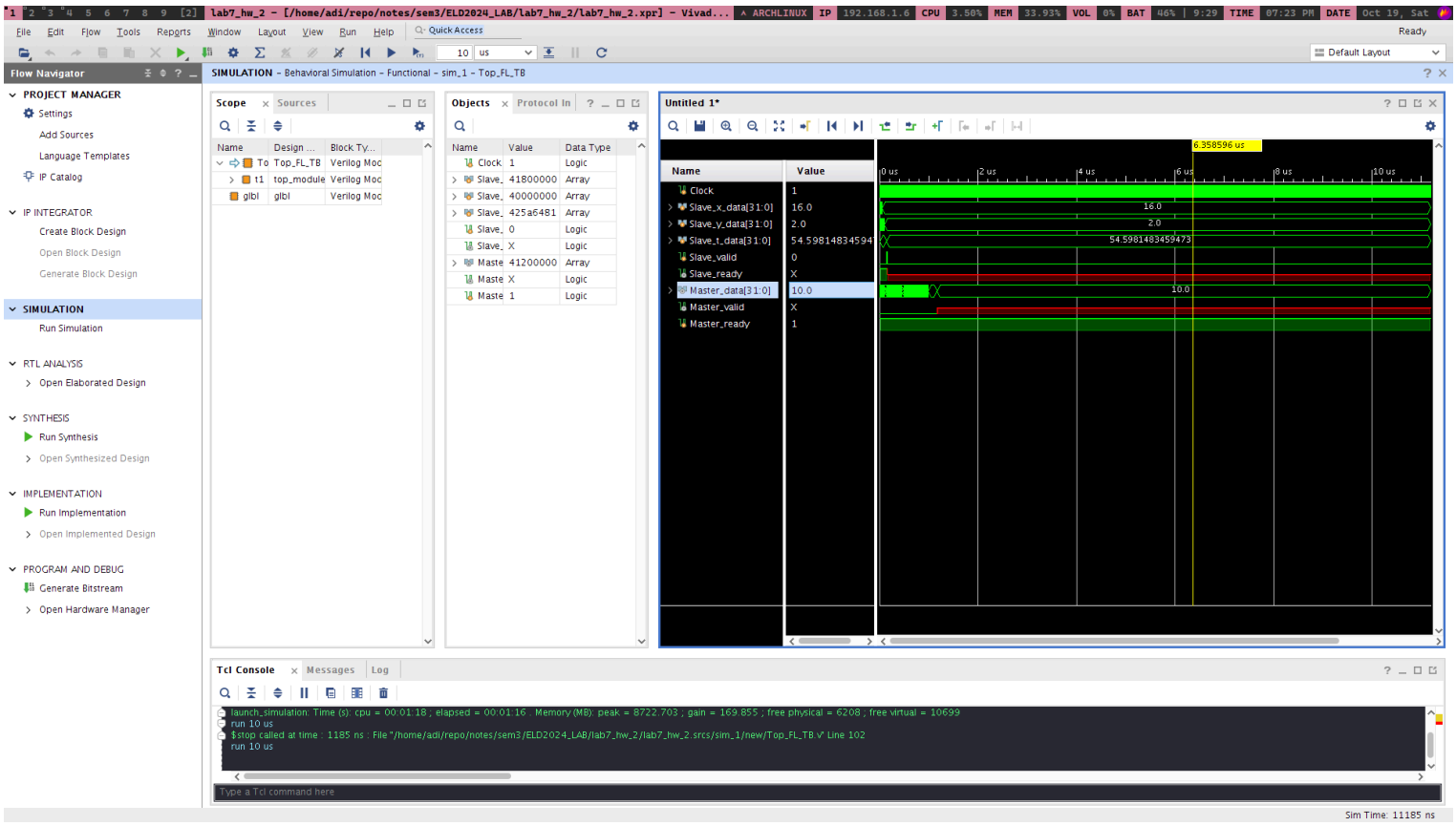
- Top_FL_TB.v

```
`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 [31:0] Slave_t_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),

     .Slave_t_data(Slave_t_data),
     .Slave_t_valid(Slave_valid),
     .Slave_t_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;
     Slave_t_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

  initial
  begin
     #50 Slave_x_data = 32'b01000001100000000000000000000000;
     #50 Slave_y_data = 32'b01000000000000000000000000000000;
     #50 Slave_t_data = 32'b01000010010110100110010010000001;
     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
```

```
`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 [31:0] Slave_t_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),
repo/notes/sem3/ELD2024_LAB/lab7_hw_2/lab7_hw_2.srcs/sim_1/new/Top_FL_TB.v
105 lines yanked
```

OUTPUT



We get the intended output 10

(reason for getting 10 as the output -> mentioned in the top_module.v file as comments in the bottom)