521404A Digital Techniques 2

Laboratory Exercises Report 2025

Name and student number: Arafat Miah (2512200)

Other contributors (if done in cooperation):
In this whole lab, I have done all by myself, but as my english is not professional because I haven't came from a english speaking country, I sometime use AI to furnish my writing, except that everything is done by myself

Lab 1: Data Types

Q1. What is the meaning of the following values of the SystemVerilog data type logic.

```
0= logical 0

1= logical 1

X= unknown

Z= high-impedance
```

Q2. What is the value of variable **one_byte** or **word32** after execution of the following statements in **binary** and **hexadecimal** format:

a)

logic [7:0] one_byte;
one_byte = 8'b11111111;
one_byte = '0;

b)

logic [7:0] one_byte;
one_byte = 8'b11111110;
one_byte = 1;

c)

logic [3:0] [7:0] four_bytes;
logic [31:0]   word32;
four_bytes = c1;
four_bytes[3] = '0;
word32 = four_bytes;

```
a) one_byte = 8'b00000000; and in hexa decimal= 8'h00

b) one_byte =8'b00000001; and in hexa decimal= 8'h01

c) word32 = 32’b00000000_11111111_11111111_11111111 and in hex=32’h00ffffff
```

Q3. With how many bits would variable **colors** be represented in SystemVerilog, if the definition logic [1:0] were removed from the following variable declaration?

enum   logic [1:0] { RED = 2'b00, GREEN = 2'b01, BLUE = 2'b10, WHITE = 2'b11 } colors;

```
32 bits
```

Q4. Assume the following variable definition and initializations, and answer the questions presented in the text box. (tip: edit test Q4 in input/datatypes_tb.sv to check).

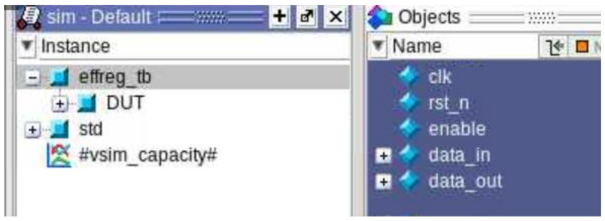
logic [3:0] a = 4'b0101;
logic [3:0] b = 4'hc;
logic [3:0] c;

```
a) After the operation c = a & b; the value of c is  ‘h4 and ‘b100


b) After the operation c = a && b; the value of c is     ‘h1 and ‘b0001


c) After the operation c = |a; the value of c is     ‘h1 and ‘b0001


d) After the operation c = a >> 2; the value of c is  ‘h1 and ‘b0001


e) After the operation c = { 2'b01 2'b11 }; the value of c is ‘h7 and ‘b0111

I did the hand calculation by myself.
```

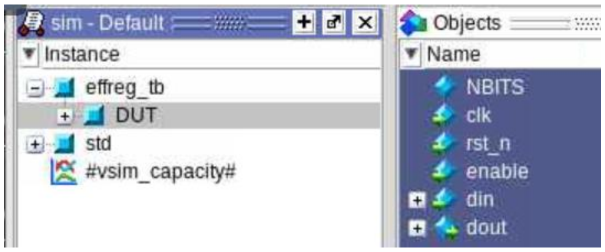
Lab 2: Structural Modeling

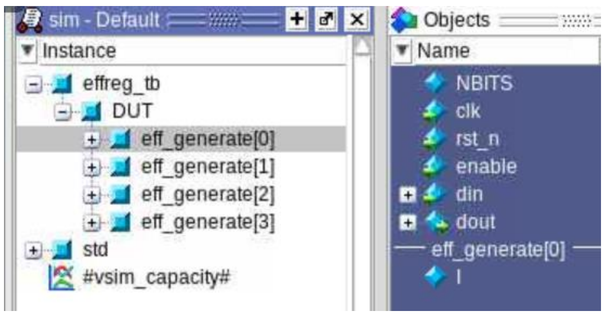
Q1. Design Hierarchy Exploration (Ex 3.2)

Design Hierarchy View	**Source Code**

 sim - Default
Instance
effreg_tb
DUT
std
#vsim_capacity#
Objects
Name
clk
rst_n
enable
data_in
data_out
	Name the file(s), and paste code that defines the selected top-level module (effreg_tb) and the items in the Object window.
	
	Example:
	
	File X.sv:
	/homedir01/amiah25/DT2_2025/labs/lab2_effreg/input/effreg_tb.sv (/effreg_tb)
	pasted code here: module effreg_tb;
	   logic clk, rst_n, enable;
	   logic [3:0] data_in;
	   logic [3:0] data_out;

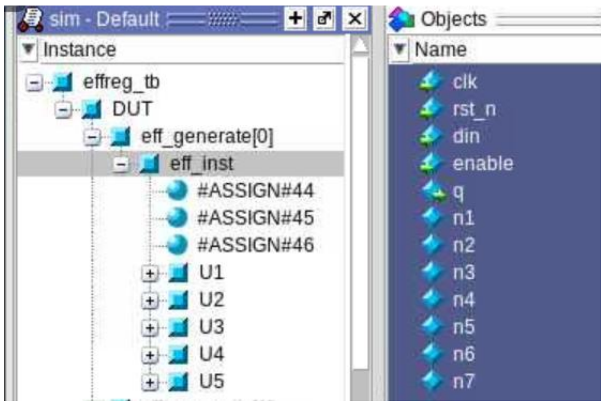
Design Hierarchy View	**Source Code**

 sim - Default
Instance
effreg_tb
DUT
std
#vsim_capacity#
Objects
Name
NBITS
clk
rst_n
enable
din
dout
	Name the file(s), and paste code that creates the selected module instance, the respective module and the items in the Object window.
	File X.sv:
	/homedir01/amiah25/DT2_2025/labs/lab2_effreg/input/effreg.sv (/effreg_tb/DUT)
	pasted code here: effreg #(.NBITS(4)) DUT
	     (.clk(clk),
	      .rst_n(rst_n),
	      .din(data_in),
	      .enable(enable),
	      .dout(data_out));

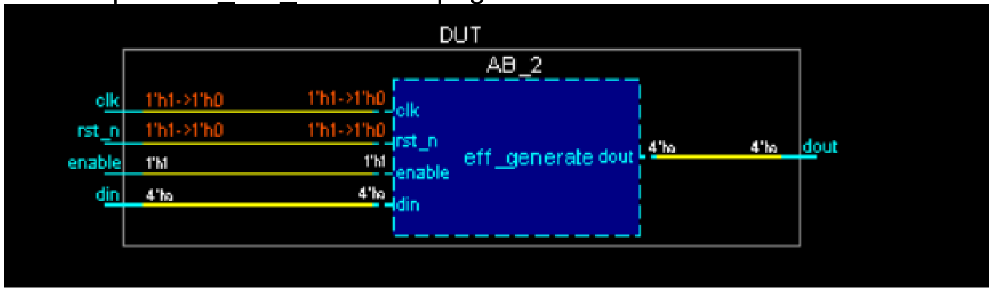
Design Hierarchy View	**Source Code**
 sim - Default Instance effreg_tb DUT eff_generate[0] eff_generate[1] eff_generate[2] eff_generate[3] std #vsim_capacity# Objects Name NBITS clk rst_n enable din dout eff_generate[0] I	Name the file(s), paste code that defines the selected object in the Sim window, and explain the purpose of the variable I shown in the Objects window. File X.sv: /homedir01/amiah25/DT2_2025/labs/lab2_effreg/input/effreg.sv (/effreg_tb/DUT/eff_generate[3]) pasted code here: generate for (genvar I = 0; I < NBITS; ++I) begin : eff_generate eff eff_inst (.clk(clk), .rst_n(rst_n), .din(din[I]), .enable(enable), .q(dout[I])); end

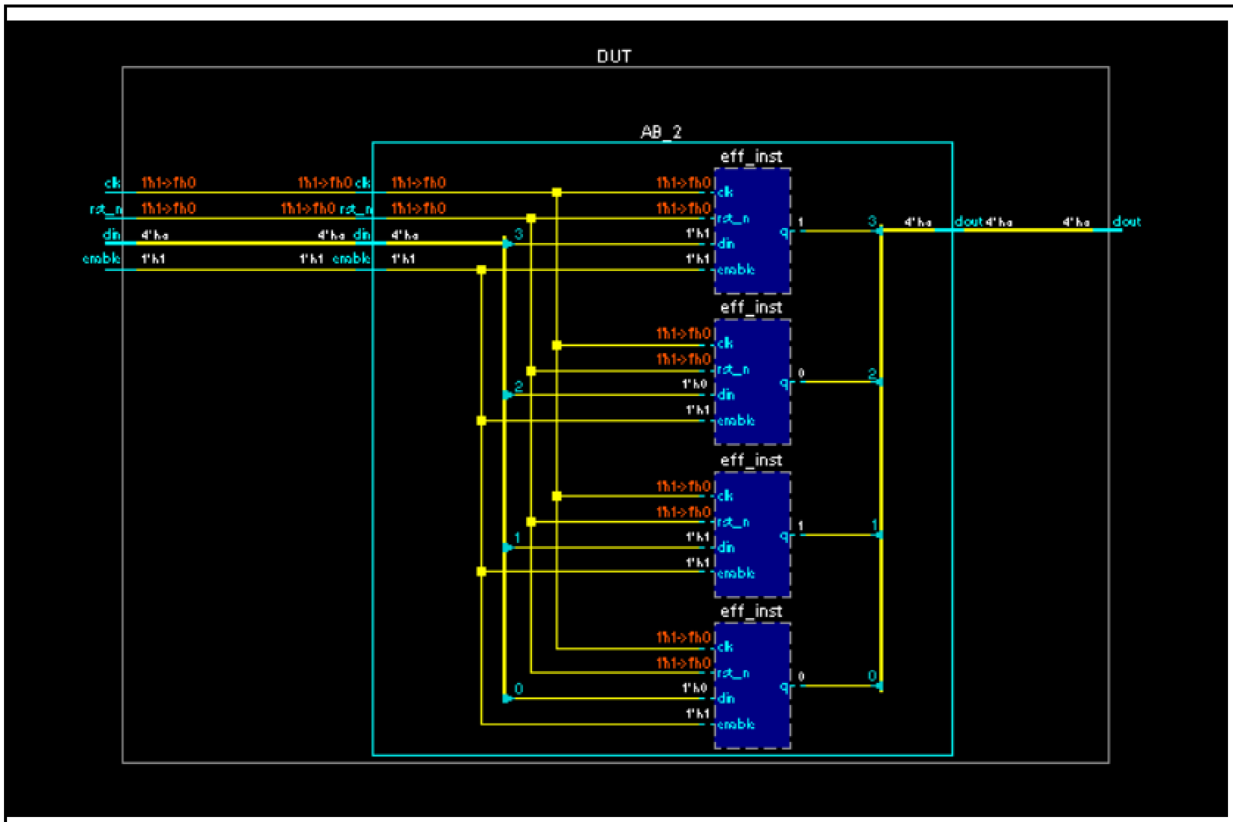
	`   endgenerate`
	`Purpose of I: It is used to generate the multiple instances of the main module which is effreg here`

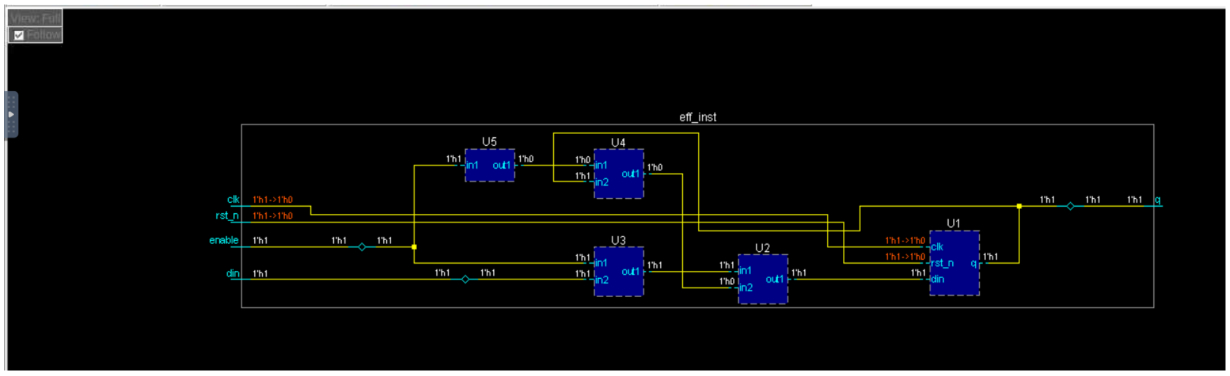
Design Hierarchy View	**Source Code**

 sim - Default
Instance
effreg_tb
DUT
eff_generate[0]
eff_inst
#ASSIGN#44
#ASSIGN#45
#ASSIGN#46
U1
U2
U3
U4
U5
Objects
Name
clk
rst_n
din
enable
q
n1
n2
n3
n4
n5
n6
n7
	`Name the file(s), and paste code that creates the selected module instance, the respective module and the items in the Object window.`
	
	`File X.sv:`
	`/homedir01/amiah25/DT2_2025/labs/lab2_effreg/input/effreg.sv`
	`(/effreg_tb/DUT/eff_generate[3]/eff_inst)`
	`pasted code here: module eff (input logic clk, rst_n, din, enable,`
	`           output logic q);`
	
	`   logic               n1, n2, n3, n4, n5, n6, n7;`
	
	`   assign n1 = enable;`
	`   assign n2 = din;`
	`   assign q = n7;`
	
	`   dff U1 (.clk(clk), .rst_n(rst_n), .din(n6), .q(n7));`
	`   or2 U2 (.in1(n4), .in2(n5), .out1(n6));`
	`   and2 U3 (.in1(n1), .in2(n2), .out1(n4));`
	`   and2 U4 (.in1(n3), .in2(n7), .out1(n5));`
	`   inv U5 (.in1(n1), .out1(n3));`
	
	`endmodule`

Q2. Schematic Generation (Ex 3.3)

Schematic View
Insert reports/eff_inst_schematic.png  DUT AB_2 clk rst_n enable din 1'h1->1'h0 1'h1->1'h0 1'h1->1'h0 1'h1->1'h0 1'h1 1'h1 4'ha 4'ha clk rst_n enable din eff_generate dout 4'ha 4'ha dout


DUT
AB_2
eff_inst
clk
rst_n
din
enable
q
dout
1'h1->1'h0
4'ha
1'h1
1'h0


View: Full
Follow
eff_inst
U5
U4
U3
U2
U1
in1
in2
out1
clk
rst_n
enable
din
q
1'h1->1'h0
1'h1
1'h0

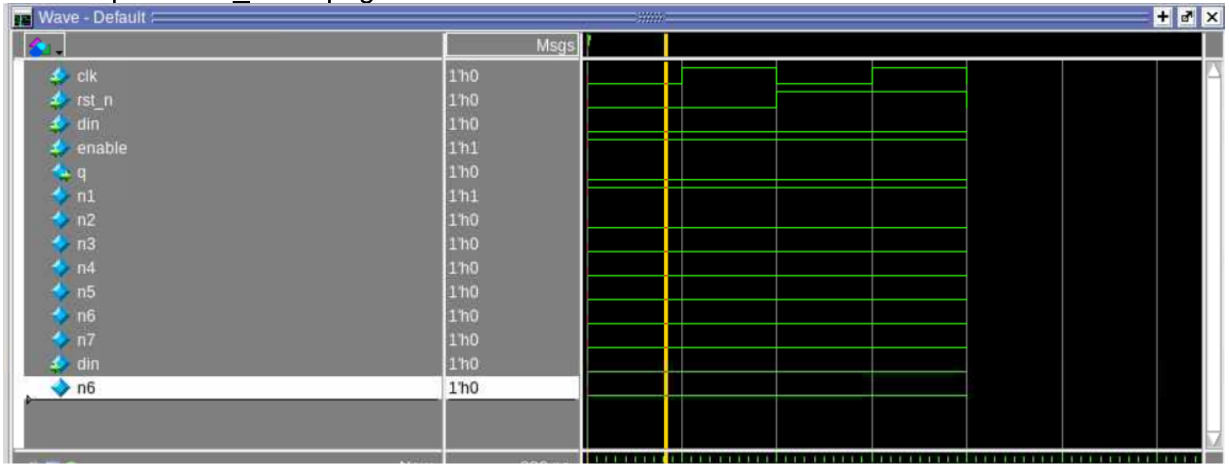
Source Code:

Name the file(s), and paste code that declares the interconnect wires and creates the module instances U1 - U2.
File X.sv: /homedir01/amiah25/DT2_2025/labs/lab2_effreg/input/effreg.sv (/effreg_tb/DUT/eff_generate[0]/eff_inst/U1)
Source code: **dff U1 (.clk(clk), .rst_n(rst_n), .din(n6), .q(n7));**
or2 U2 (.in1(n4), .in2(n5), .out1(n6));

Q3. Wave Window Usage (Ex 3.4)

Wave View

Insert reports/add_wave.png



clk	1'h0
rst_n	1'h0
din	1'h0
enable	1'h1
q	1'h0
n1	1'h1
n2	1'h0
n3	1'h0
n4	1'h0
n5	1'h0
n6	1'h0
n7	1'h0
din	1'h0
n6	1'h0

Source Code: /homedir01/amiah25/DT2_2025/labs/lab2_effreg/input/effreg.sv (/effreg_tb/DUT/eff_generate[0]/eff_inst)

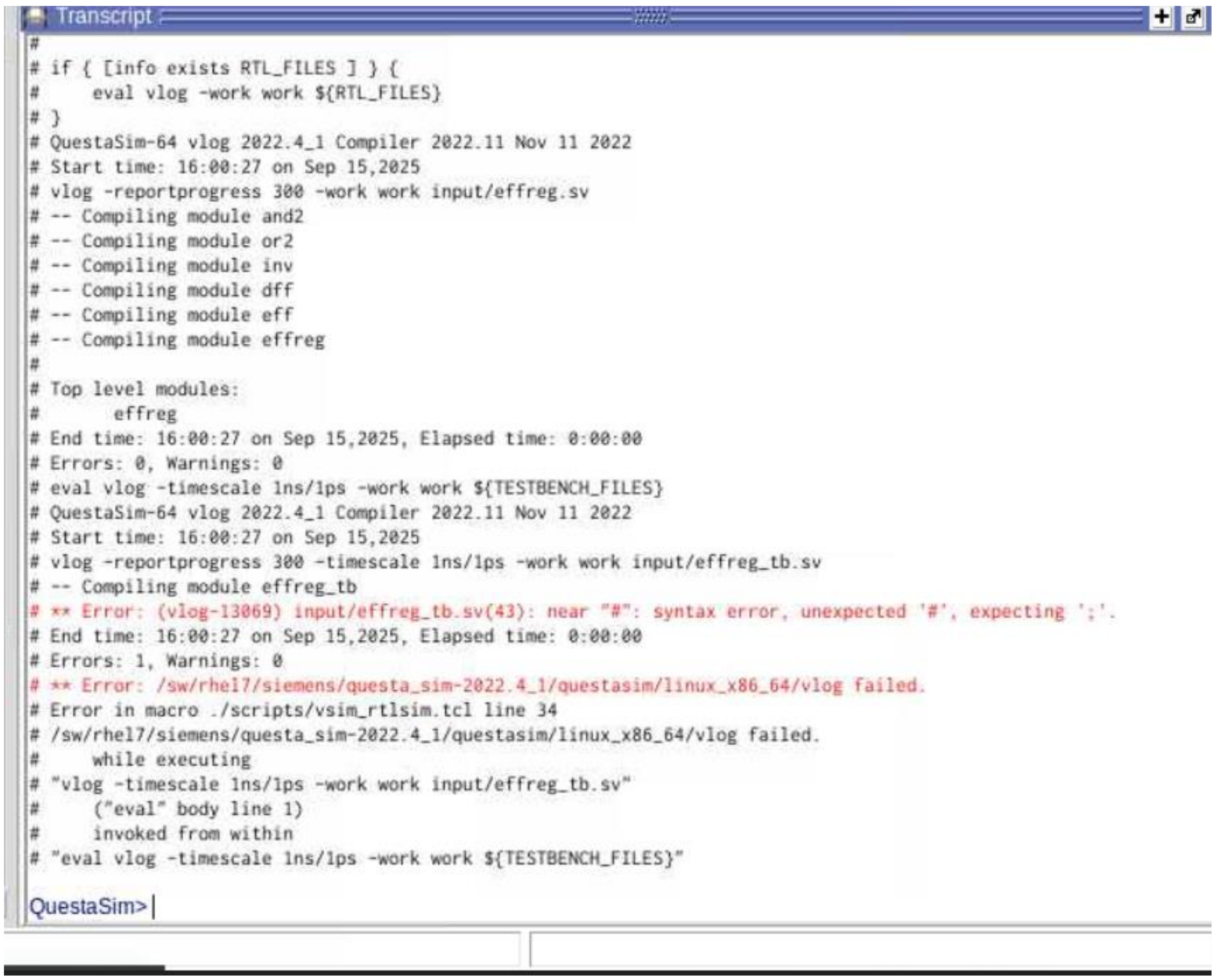
Paste the TCL (tool command language) statement from wave.do that added the net to the Wave window.

```
gedit workdir/wave.do &
```

Q4. Simulation - Debug (Ex 4)

Wave View with $info Message

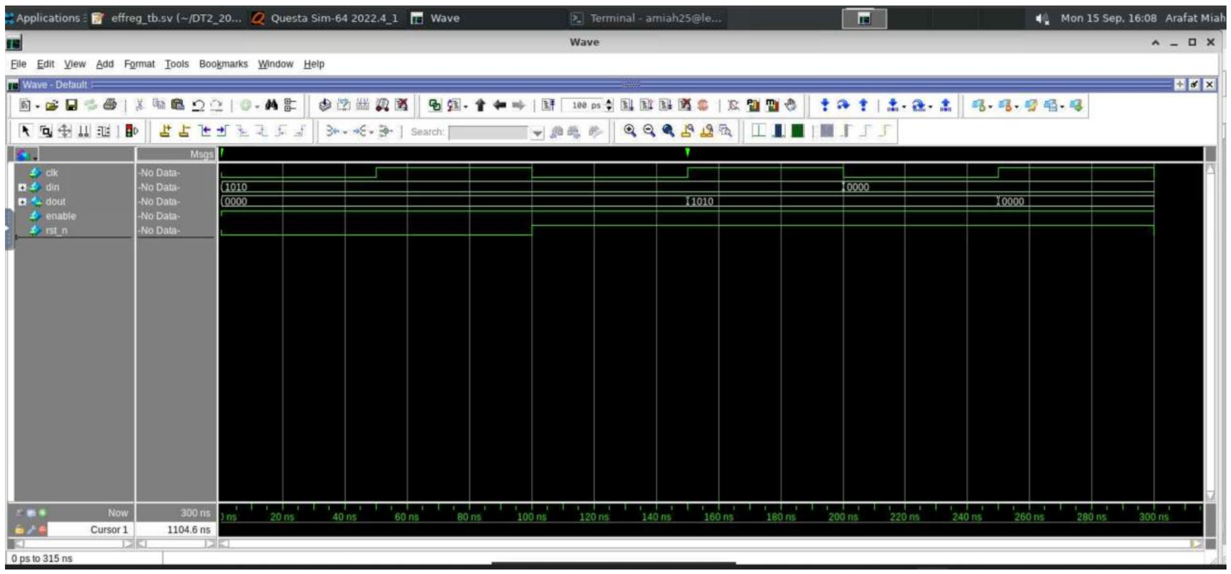
Insert reports/wave_with_info.png



```
#
# if { [info exists RTL_FILES ] } {
#     eval vlog -work work ${RTL_FILES}
# }
# QuestaSim-64 vlog 2022.4_1 Compiler 2022.11 Nov 11 2022
# Start time: 16:00:27 on Sep 15,2025
# vlog -reportprogress 300 -work work input/effreg.sv
# -- Compiling module and2
# -- Compiling module or2
# -- Compiling module inv
# -- Compiling module dff
# -- Compiling module eff
# -- Compiling module effreg
#
# Top level modules:
#       effreg
# End time: 16:00:27 on Sep 15,2025, Elapsed time: 0:00:00
# Errors: 0, Warnings: 0
# eval vlog -timescale 1ns/1ps -work work ${TESTBENCH_FILES}
# QuestaSim-64 vlog 2022.4_1 Compiler 2022.11 Nov 11 2022
# Start time: 16:00:27 on Sep 15,2025
# vlog -reportprogress 300 -timescale 1ns/1ps -work work input/effreg_tb.sv
# -- Compiling module effreg_tb
# ** Error: (vlog-13069) input/effreg_tb.sv(43): near "#": syntax error, unexpected '#', expecting ';'.
# End time: 16:00:27 on Sep 15,2025, Elapsed time: 0:00:00
# Errors: 1, Warnings: 0
# ** Error: /sw/rhel7/siemens/questa_sim-2022.4_1/questasim/linux_x86_64/vlog failed.
# Error in macro ./scripts/vsim_rtlsim.tcl line 34
# /sw/rhel7/siemens/questa_sim-2022.4_1/questasim/linux_x86_64/vlog failed.
#     while executing
# "vlog -timescale 1ns/1ps -work work input/effreg_tb.sv"
#     ("eval" body line 1)
#     invoked from within
# "eval vlog -timescale 1ns/1ps -work work ${TESTBENCH_FILES}"

QuestaSim>
```

Message Viewer: # ** Error: (vlog-13069) input/effreg_tb.sv(43): near "#": syntax error, unexpected '#', expecting ';'.

Insert reports/message_viewer.png


Applications
effreg_tb.sv (~/DT2_20...
Questa Sim-64 2022.4_1
Wave
Terminal - amiah25@le...
Mon 15 Sep, 16:08 Arafat Miah
Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
clk -No Data-
din -No Data-
dout -No Data-
enable -No Data-
rst_n -No Data-
1010
0000
1010
0000
0000
Now 300 ns
Cursor 1 1104.6 ns
0 ns 20 ns 40 ns 60 ns 80 ns 100 ns 120 ns 140 ns 160 ns 180 ns 200 ns 220 ns 240 ns 260 ns 280 ns 300 ns
0 ps to 315 ns

Lab 3. Combinational Logic Modeling

Ex1. Combinational Process Execution in Simulation

Q1. Explain why the process muxa, muxb and add **was** or **was not** executed at the time instances 0, 100, 200 and 300 ns in the simulation.

Time	muxa	muxb	add	Explanation

0	executed	executed	executed	sel_in = '0;
				a1_in = 1;
				a2_in = 2;
				b1_in = 3;
				b2_in = 4;
				#100ns;
				
				The above part is taken from muxadd_test and we can see that at the start selection pin is set to zero and mux a and mux b is executed, and the adder takes the the input coming from the muxa and muxb , as the input of adder is changed, so the adder is also operate and we will get an output.

100	Execute	Execute	Execute	sel_in = '1;
				a1_in = 1;
				a2_in = 2;
				b1_in = 3;
				b2_in = 4;
				#100ns;
				
				Now, this part means that the second part of the test will execute after 100ns. We see that the selection pin has changed, so muxa and muxb will be executed. As the muxes change their outputs, the adder will also execute.

200	Not execute	execute	execute	sel_in = '1;
				a1_in = 1;
				a2_in = 2;
				b1_in = 3;
				b2_in = 5;
				#100ns;
				After another 100ns, this part will execute, where we can see that the selection pin remains unchanged, so MUX A will not be executed since its input has not changed. However, MUX B has changed its input, so it will execute, and as a result, the adder will also execute.

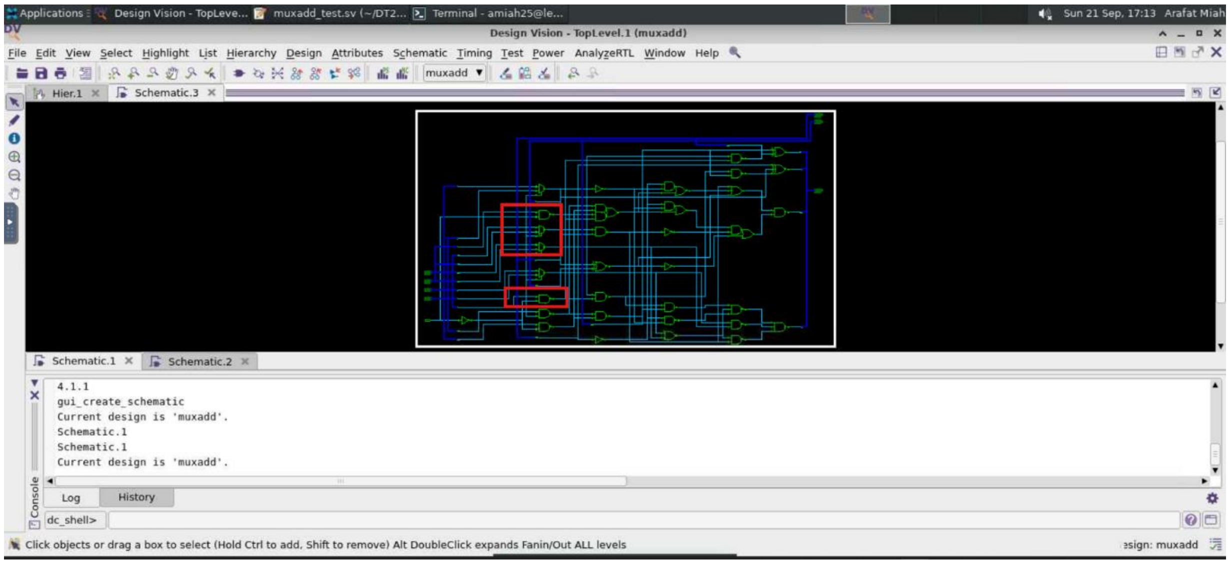
300	execute	execute	execute	sel_in = '0; #100ns; //all of them execute as the overall selection pin value changes

Ex2. Synthesis Results Examination

Q2. Insert annotated schematic form logic synthesis showing logic gates for the **muxa** procedure.

Gate-Level Schematic with muxa Components Highlighted

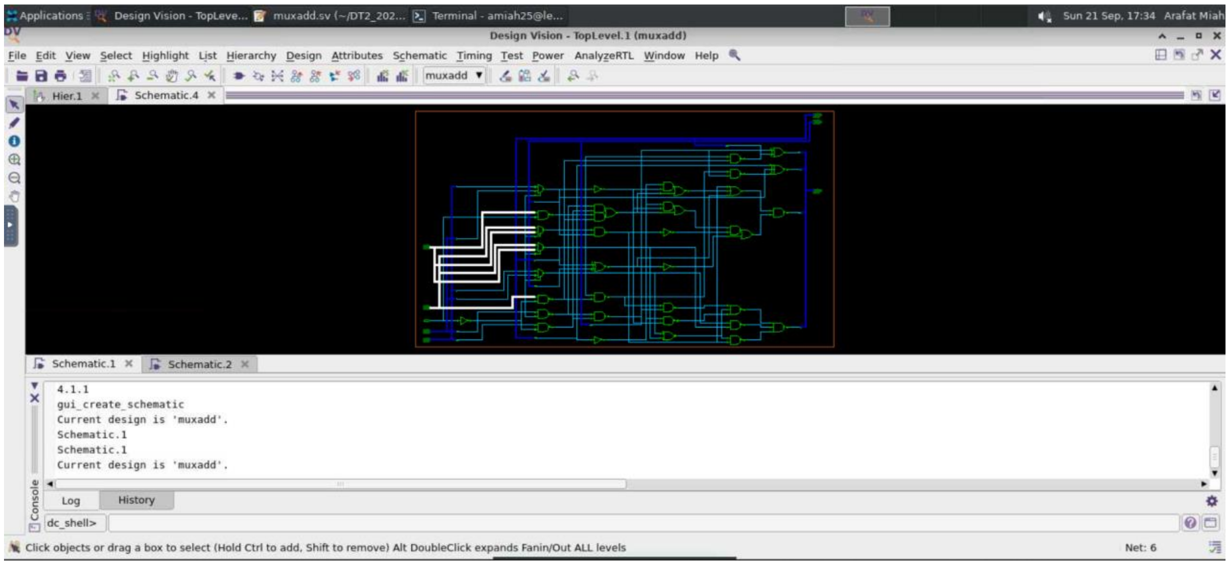
`Insert reports/muxadd_schematic.png`


Applications
Design Vision - TopLeve...
muxadd_test.sv (~/DT2...
Terminal - amiah25@le...
Sun 21 Sep, 17:13 Arafat Miah
Design Vision - TopLevel.1 (muxadd)
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
muxadd
Hier.1
Schematic.3
Schematic.1
Schematic.2
Log
History
Console

```
4.1.1
gui_create_schematic
Current design is 'muxadd'.
Schematic.1
Schematic.1
Current design is 'muxadd'.
```

```
dc_shell>
```

Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels   esign: muxadd


Applications
Design Vision - TopLeve...
muxadd.sv (~/DT2_202...
Terminal - amiah25@le...
Sun 21 Sep, 17:34 Arafat Miah
Design Vision - TopLevel.1 (muxadd)
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
muxadd
Hier.1
Schematic.4
Schematic.1
Schematic.2
Log
History
Console

```
4.1.1
gui_create_schematic
Current design is 'muxadd'.
Schematic.1
Schematic.1
Current design is 'muxadd'.
```

```
dc_shell>
```

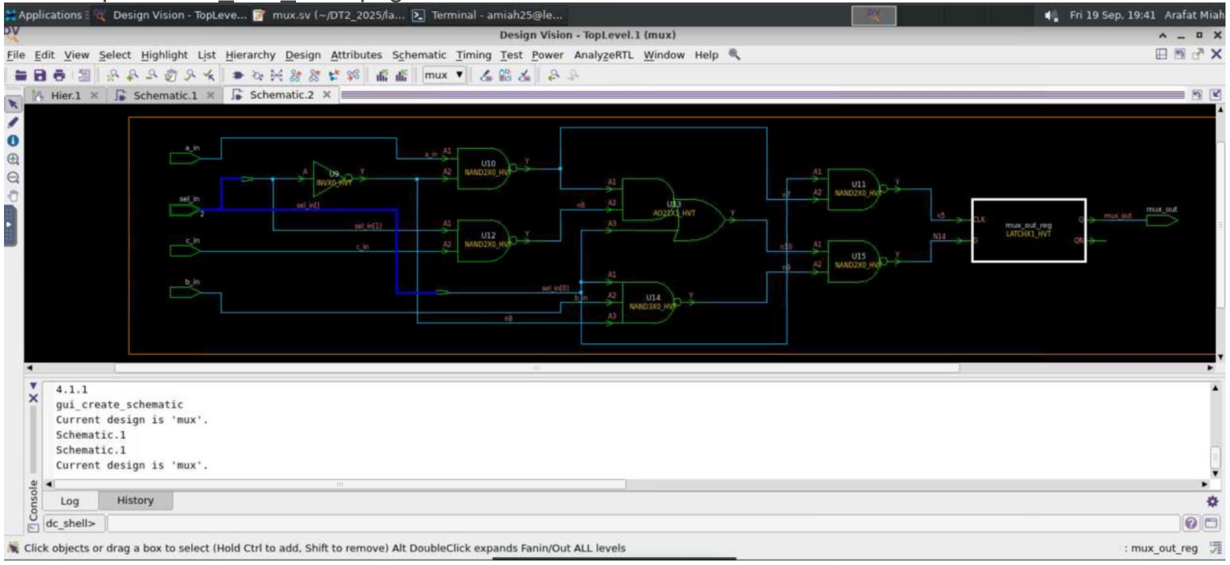
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels   Net: 6

Ex 3. Synthesizability Checking and Fixing

Q3. Present the evidence that the combinational logic model was incorrect.

Gate-Level Schematic

Insert reports/mux_with_latch.png



```
4.1.1
gui_create_schematic
Current design is 'mux'.
Schematic.1
Schematic.1
Current design is 'mux'.
```

Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels

Reference Report

Insert component reference listing from reports/mux.gatelevel.reference.txt and highlight the latch.
The references are:

1. AO21X1_HVT
2. INVX0_HVT
3. LATCHX1_HVT LATCHX1_HVT (This is the reference for latch)
4. NAND2X0_HVT
5. NAND3X0_HVT

Synthesis Log Warning

Present the latch warning and explain how the tool knew that a latch should nor have been synthesized.
Warning: ./input/mux.sv:8: Netlist for always_comb block contains a latch. (ELAB-974)

The tool knew that because all the inputs have not been assigned a value in all possible situations.

Q4. Explain what is a latch, and why did the synthesis program generate a latch in the mux logic, and why the incorrect code did not produce wrong simulation results.

A latch can store the previous data as it has two stable states. In this mux circuit, the latch is generated because in this combination block, all inputs have not been assigned under all possible conditions in the test bench.
The wrong code did not produce a wrong simulation because, from the Verilog language rule, if a variable is not assigned a new value in any procedural block, it will hold its earlier value.

Q5. What are some of the common coding principles for avoiding accidentally creating latches?

1. Always use always_comb
2. Assign all the possible new value for every procedural block
3. For all the output, Assign a default value

Q6. Present the fixed code.

```
module mux
  (input logic [1:0] sel_in,
   input logic      a_in,
   input logic      b_in,
   input logic      c_in,
   output logic mux_out);

  always_comb
    begin : mux_logic
       case (sel_in)
        2'b00: mux_out = a_in;
        2'b01: mux_out = b_in;
        2'b10: mux_out = c_in;
        2'b11: mux_out = c_in;
       endcase
    end : mux_logic

endmodule
```

Note: I also modified the test bench to see if my code is working perfectly in the simulation or not.

```
program mux_test
  (output logic [1:0] sel_in,
   output logic a_in,
   output logic b_in,
   output logic c_in,
   input logic      mux_out);

   initial
     begin
       sel_in = 2'b00;
       a_in = '0;
       b_in = '0;
       c_in = '0;

       #10ns;
         sel_in = 2'b01;
       a_in = '1;
       b_in = '1;
       c_in = '0;

       #10ns;
         sel_in = 2'b10;
       a_in = '0;
       b_in = '0;
         c_in = '1;
```

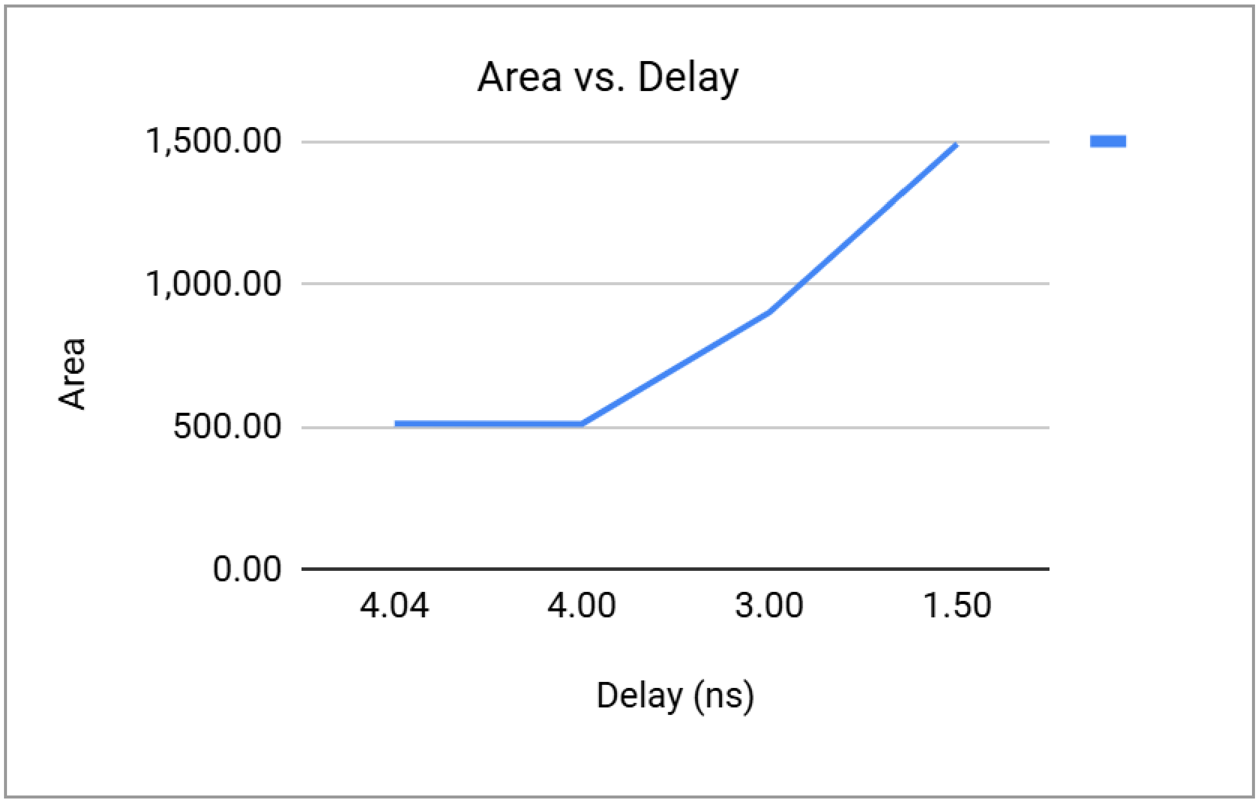
```
        #10ns;
        sel_in = 2'b11;


      #10ns;

      $finish;

     end
endprogram
```

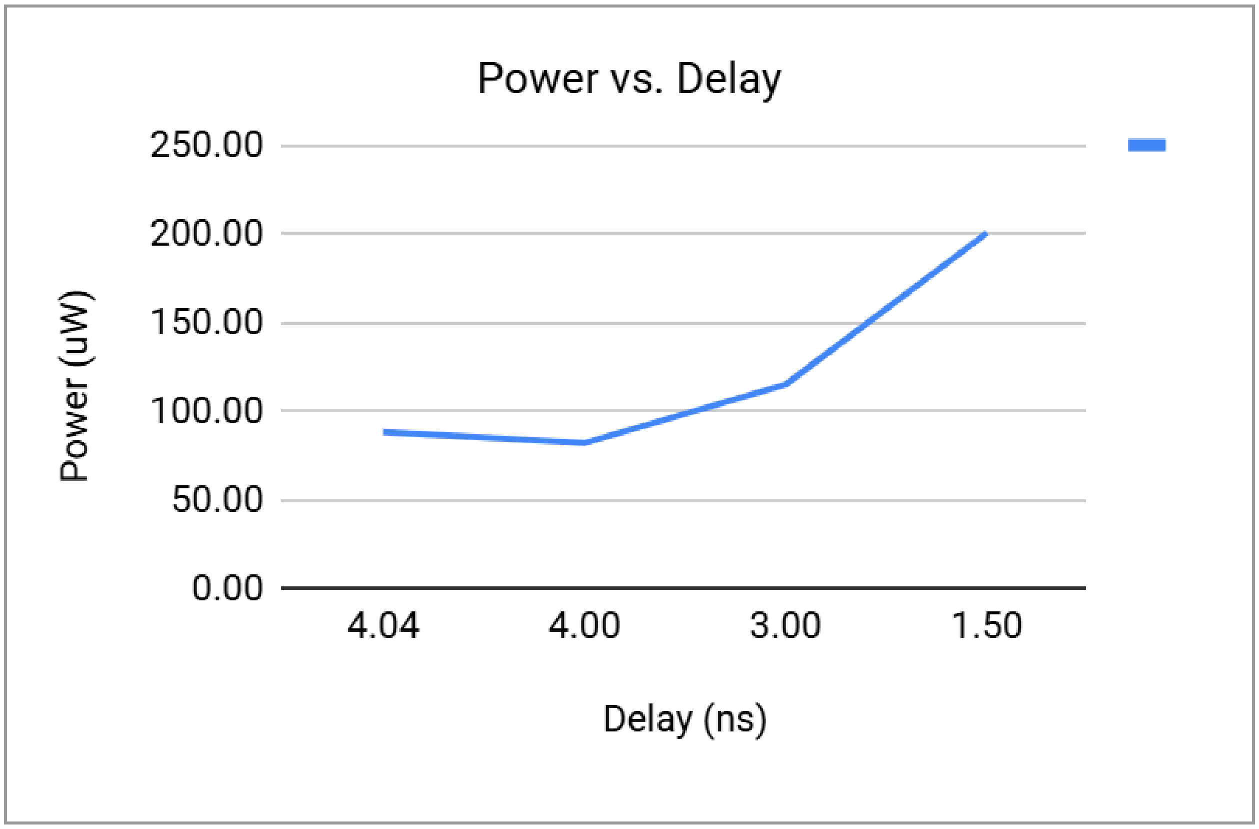
Q7. Present the reference report for the synthesized fixed code.

```
    1. INVX0_HVT
    2. INVX1_HVT
    3. NAND2X0_HVT
    4. NAND3X0_HVT
    5. OR2X1_HVT
```

Lab 4. Logic Synthesis

Q1. Insert your area-vs-delay graph for the **acs** design optimized with different timing constraints



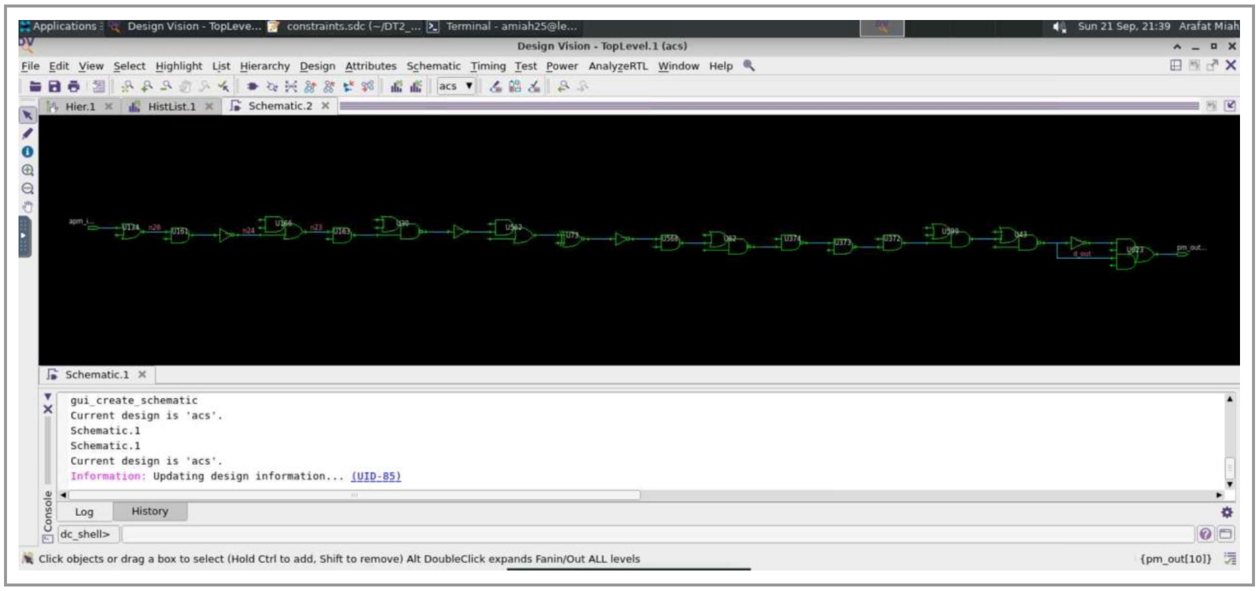


Q2. Explain the results!

A **low area** circuit is typically slow but low-power. To make a circuit **faster**, we must increase the area and power by using more complex, higher-performance gates.

Q3. How many components are on the critical path in the final (fastest) case?

```
Ans: 27
```


Applications
Design Vision - TopLeve...
constraints.sdc (~/DT2_...
Terminal - amiah25@le...
Sun 21 Sep, 21:39 Arafat Miah
Design Vision - TopLevel.1 (acs)
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
acs
Hier.1
HistList.1
Schematic.2
apm_i...
U134
n28
U161
n24
U166
n23
U163
U30
U562
U73
U568
U62
U374
U373
U372
U386
U43
d_out
U427
pm_out...
Schematic.1
Log
History
Console
{pm_out[10]}

```
gui_create_schematic
Current design is 'acs'.
Schematic.1
Schematic.1
Current design is 'acs'.
Information: Updating design information... (UID-85)
```

```
dc_shell>
```

Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels

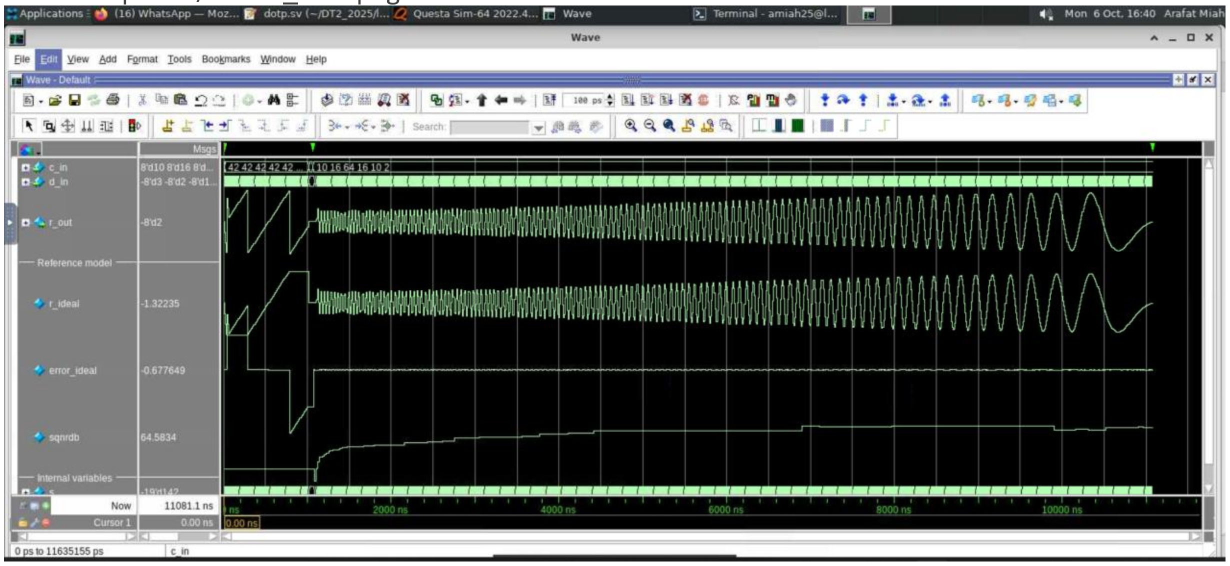
Lab 5. Digital Arithmetic

Ex1. Datapath Sizing

Q1. Present your parameter settings for the **dotp** exercise and justify them!

```
a) localparam MB =16; as we know Multiply bit= Bits in DB+Bits in SB

b) localparam SB = 19; as My last 3rd digit is 0 so my L=0 and parameter NTAPS= 6
       So, as we can see from the dataflow graph of the algorithm, there will be 6
       blocks, and each block's result is added to the next stage block. So, as we
       know from the hardware adding formula, the Result bit length is,
                       Result bit length= max(bit in DB, bits in CB)+1;
So after the first iteration of addition, my result will be,
                          S[0]= max (16,16)+1=17
c) localparam QB = 8;
```

Paste reports/waves_ex1.png here!



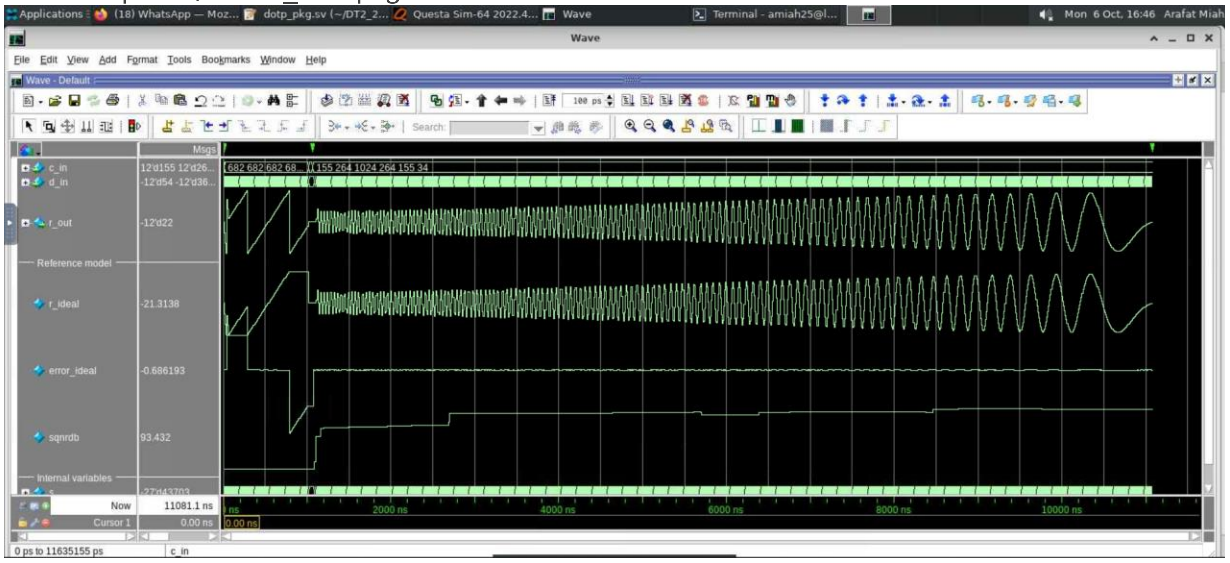
Ex2. Datapath bit-width optimization

Q1. Present your parameter settings for the **dotp** exercise and justify them!

```
a) localparam MB = 28;

b) localparam SB = 32;

c) localparam QB = 14;

d) localparam logic signed [QB-1:0] rmax = 14’b0111_1111_1111_11;
```

```
e) localparam logic signed [QB-1:0] rmin = 14’b1000_0000_0000_00;
```

```
Paste reports/waves_ex2.png here!
```



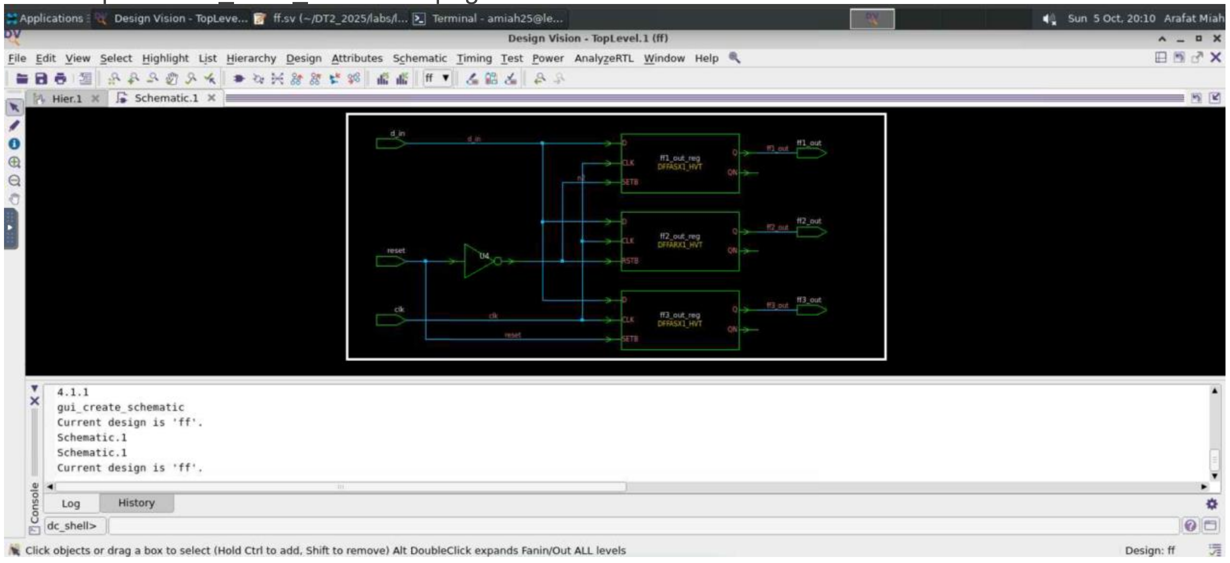
Lab 6: Flip-Flops and Registers

Ex 1. Flip-flop modeling

Q1. Explain the kind of reset defined for each flip-flop.

Flip-flop	Reset or Set?	Asynch or Synch?	Active Low or High?
ff1	Set	Asynch	High
ff2	Reset	Asynch	High
ff3	Set	Asynch	High

Q2. Present the gate-level schematic and explain how the reset was implemented

Gate-Level Schematic
Insert reports/lab6_reset_schematic.png 

Applications
Design Vision - TopLeve...
ff.sv (~/DT2_2025/labs/l...
Terminal - amiah25@le...
Sun 5 Oct, 20:10 Arafat Miah
Design Vision - TopLevel.1 (ff)
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
Hier.1
Schematic.1
d_in
reset
clk
U4
D
CLK
SETB
RSTB
Q
QN
ff1_out_reg DFFASX1_HVT
ff2_out_reg DFFARX1_HVT
ff3_out_reg DFFASX1_HVT
ff1_out
ff2_out
ff3_out
Console
Log
History
Design: ff

```
4.1.1
gui_create_schematic
Current design is 'ff'.
Schematic.1
Schematic.1
Current design is 'ff'.
```

dc_shell>

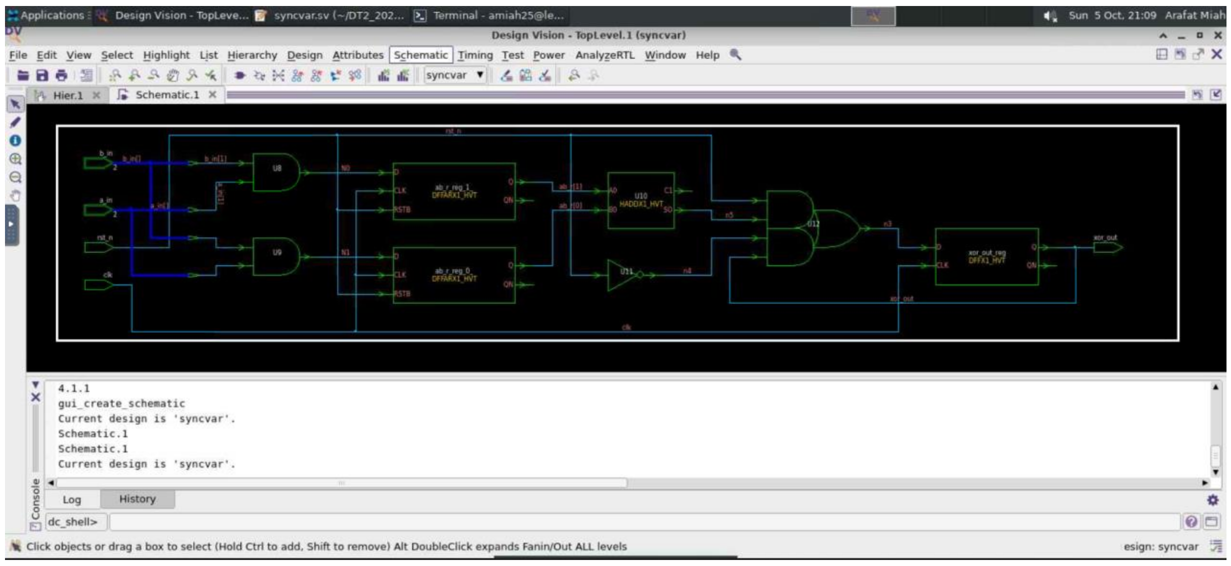
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels

Explanations
Asynchronous Reset:when reset and RSTB occur (active-low) the code instantly clear ff1_out_reg, ff2_out_reg, and ff3_out_reg and that is independent of clk.

Ex 2. Variable assignments in a sequential process

Q3. Present a gate-level schematic of the bad design and explain why flip-flops were generated or not generated for these variables/ports assigned inside an always procedure?.

Gate-Level Schematic
Insert reports/lab6_syncvar_bad.png


Design Vision - TopLevel.1 (syncvar)
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
Hier.1
Schematic.1
b_in
a_in
rst_n
clk
U8
U9
ab_r_reg_1
ab_r_reg_0
U10
HADDX1_HVT
U11
U12
xor_out_reg
DFFX1_HVT
xor_out
Log
History
dc_shell>

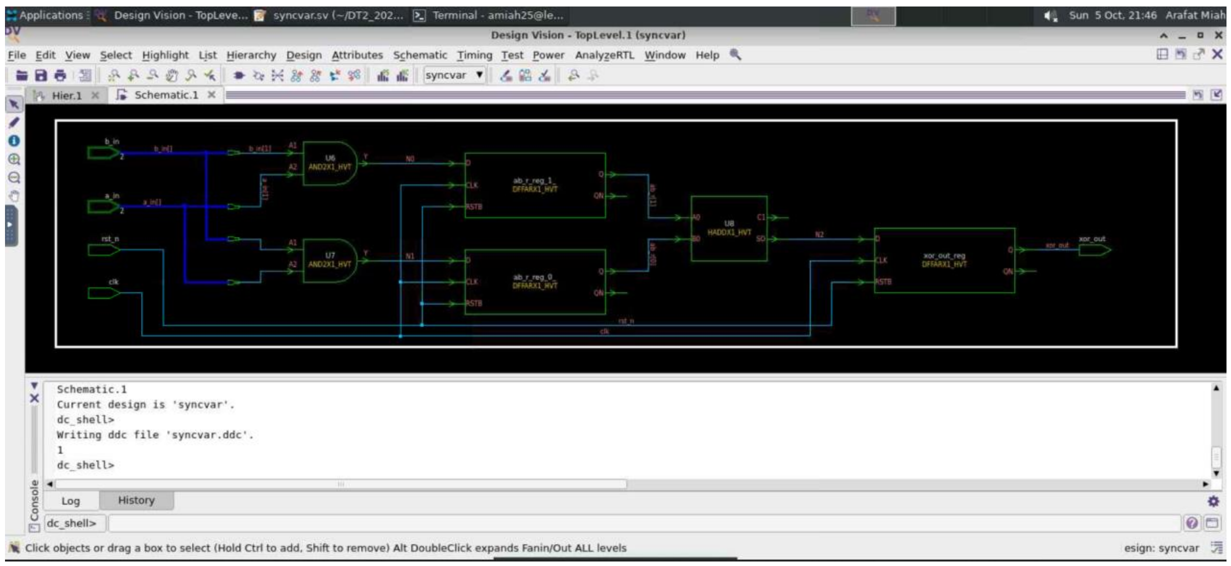
```
4.1.1
gui_create_schematic
Current design is 'syncvar'.
Schematic.1
Schematic.1
Current design is 'syncvar'.
```

Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels

Explanations
xor_out) It is working as a simple xor gate which is dependent on ab_r[0] ^ ab_r[1] and it is creating no flip flop ab_r) this is a regular logic ,however it will woking as flip flop when we use "<=", but will remain ok when I use "=" ab) It is working as an AND gate and no flip flop, and it works as a temporary holder of a_in and b_in

Q4. Present a gate-level schematic of the fixed design and explain the fix?.

Gate-Level Schematic
Insert reports/lab6_syncvar_good.png


Design Vision - TopLevel.1 (syncvar)
File Edit View Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
syncvar
Hier.1
Schematic.1
b_in
a_in
rst_n
clk
U6
AND2X1_HVT
U7
AND2X1_HVT
N0
N1
ab_r_reg_1_
DFFARX1_HVT
ab_r_reg_0_
DFFARX1_HVT
U8
HADDX1_HVT
N2
xor_out_reg
DFFARX1_HVT
xor_out
Log
History
dc_shell>
Click objects or drag a box to select (Hold Ctrl to add, Shift to remove) Alt DoubleClick expands Fanin/Out ALL levels
esign: syncvar

```
Schematic.1
Current design is 'syncvar'.
dc_shell>
Writing ddc file 'syncvar.ddc'.
1
dc_shell>
```

Explanation

The overall operation turns into a synchronous operation with terms to rising edge of the clk and it avoids glitches with using non blocking assignments and ensures accurate flip flop for ab_r and xor_out

Lab 7: Timing Analysis and Optimization

Ex 1. Sequential Circuit Synthesis

Q1) Show and explain the register inference report from the synthesis log (also available as reports/dc.log)

Register Inference Report

```
===============================================================================
|    Register Name     |   Type    | Width | Bus | MB | AR | AS | SR | SS | ST |
===============================================================================
|     state_r_reg      | Flip-flop |   4   |  Y  | N  | N  | N  | Y  | N  | N  |
===============================================================================
```

Explanation

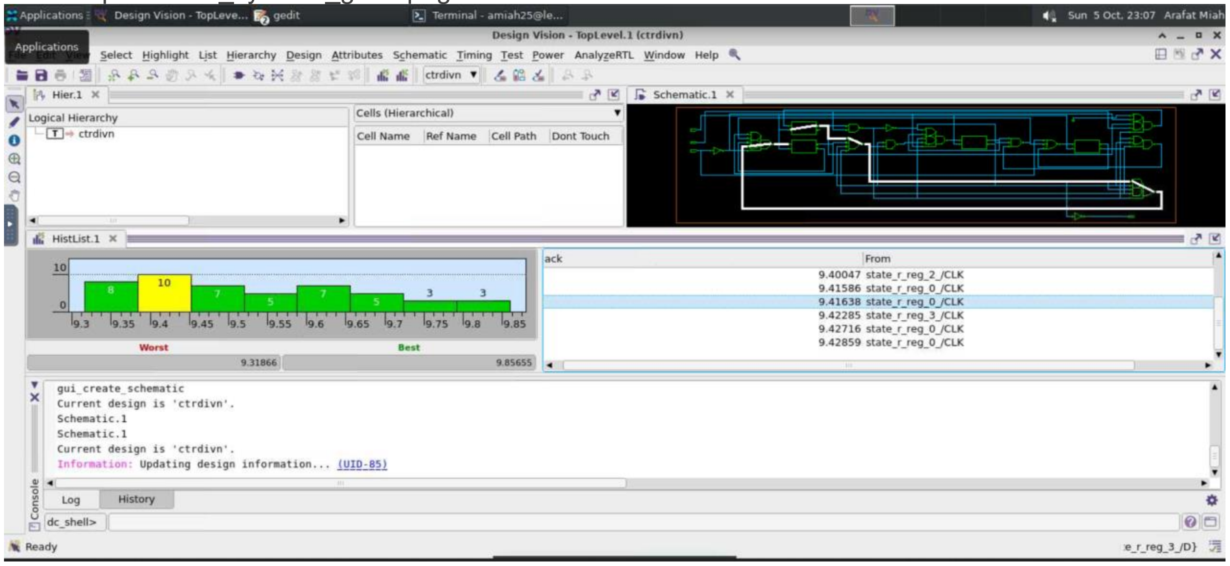
This indicates a **4-bit synchronous flip-flop** used for state control

Ex 2. Timing Analysis

Q2. Present a gate-level schematic with the critical path highlighted

Gate-Level Schematic

Insert reports/lab6_syncvar_good.png


Applications
Design Vision - TopLeve...
gedit
Terminal - amiah25@le...
Sun 5 Oct, 23:07 Arafat Miah
Design Vision - TopLevel.1 (ctrdivn)
Select Highlight List Hierarchy Design Attributes Schematic Timing Test Power AnalyzeRTL Window Help
Hier.1
Logical Hierarchy
ctrdivn
Cells (Hierarchical)
Cell Name Ref Name Cell Path Dont Touch
Schematic.1
HistList.1
Worst
Best
Console
Log
History
dc_shell>
Ready

```
ack                 From
          9.40047 state_r_reg_2_/CLK
          9.41586 state_r_reg_0_/CLK
          9.41638 state_r_reg_0_/CLK
          9.42285 state_r_reg_3_/CLK
          9.42716 state_r_reg_0_/CLK
          9.42859 state_r_reg_0_/CLK
```

Worst 9.31866 Best 9.85655

```
gui_create_schematic
Current design is 'ctrdivn'.
Schematic.1
Schematic.1
Current design is 'ctrdivn'.
Information: Updating design information... (UID-85)
```

Q3) Show the timing report (starting from the path listing), and explain its meaning.

Timing Report

```
Point                                       Incr       Path
  -----------------------------------------------------------
  clock clk (rise edge)                       0.00       0.00
  clock network delay (ideal)                 0.00       0.00
  state_r_reg_0_/CLK (DFFX1_HVT)           0.00       0.00 r
  state_r_reg_0_/Q (DFFX1_HVT)             0.14       0.14 r
  U29/Y (NAND2X0_HVT)                      0.13       0.27 f
  U31/Y (INVX0_HVT)                        0.08       0.35 r
  U34/Y (AND3X1_HVT)                       0.12       0.47 r
  U38/Y (OA221X1_HVT)                      0.13       0.60 r
  state_r_reg_3_/D (DFFX1_HVT)             0.01       0.61 r
  data arrival time                                   0.61

  clock clk (rise edge)                   10.00      10.00
  clock network delay (ideal)              0.00      10.00
  state_r_reg_3_/CLK (DFFX1_HVT)           0.00      10.00 r
  library setup time                      -0.07       9.93
  data required time                                  9.93
  -----------------------------------------------------------
  data required time                                  9.93
  data arrival time                                  -0.61
  -----------------------------------------------------------
  slack (MET)                                         9.32
```

Explanation

Startpoint: Output Q
Endpoint: Input D

Data arrival time: Together

Library setup time: Explain flip-flop setup time

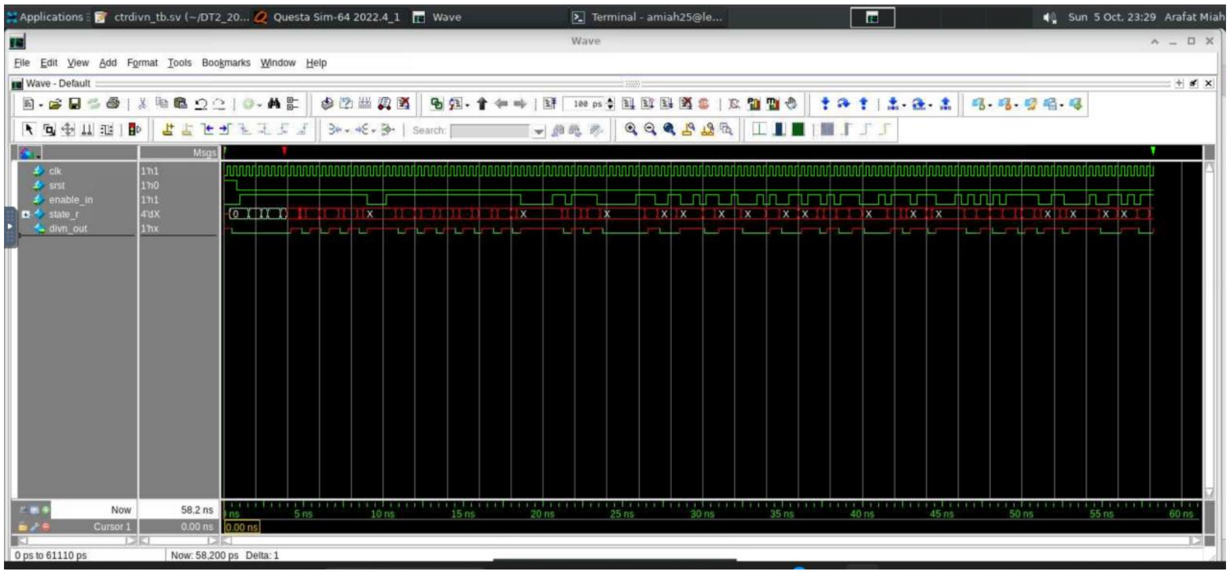
Data required time: Why is data required to arrive at this time?

Slack: What does this figure tell?

Q4) In gate-level simulation with the shortened clock period, timing violations (probably) occurred. Present the waveforms and explain the first timing error reported.

Gate Level Simulation Waveforms

reports/lab7_glsim_bad.png.



Error Message and Its Explanation

# ** Error: $setup(posedge D:3769 ps, posedge CLK:3821 ps, 72 ps);
#   Time: 3821 ps Iteration: 1 Process: /ctrdivn_tb/DUT/state_r_reg_1_/#Setuphold# File: /research/cas/public/DT2_2025/lib/verilog/saed32nm_hvt.v Line: 6422
# ** Error: $setup(posedge D:3769 ps, posedge CLK &&& D_DEFCHK:3821 ps, 72 ps);
#   Time: 3821 ps Iteration: 1 Process: /ctrdivn_tb/DUT/state_r_reg_1_/#Setuphold# File: /research/cas/public/DT2_2025/lib/verilog/saed32nm_hvt.v Line: 6432

The error message reports a **setup time violation** of for flip-flop `state_r_reg_1_`. The data () arrived at , which is **too late** to meet the setup requirement before the rising clock edge () at .

Ex 3. Timing Optimization

Q5) Present the timing optimization constraints you defined

Timing Constraints File

```
New clock setting from input/constraints.sdc

create_clock -name clk -period 0.48  clk

set_input_delay  -clock clk 0.0 srst
set_input_delay  -clock clk 0.0 enable_in
set_output_delay -clock clk 0.0 divn_out

set_driving_cell  -lib_cell INVX1_HVT  -library saed32hvt_tt1p05v25c  { enable_in }
set_load 0.1 divn_out

#Prevent usage of flip-flops with sync reset. This will add
```

```
#one extra AND-gate in the critical path :-)
set_dont_use [get_lib_cell saed32hvt_tt1p05v25c/DFFSSRX*]
```

Q6) Present the timing report following synthesis with the new constraints

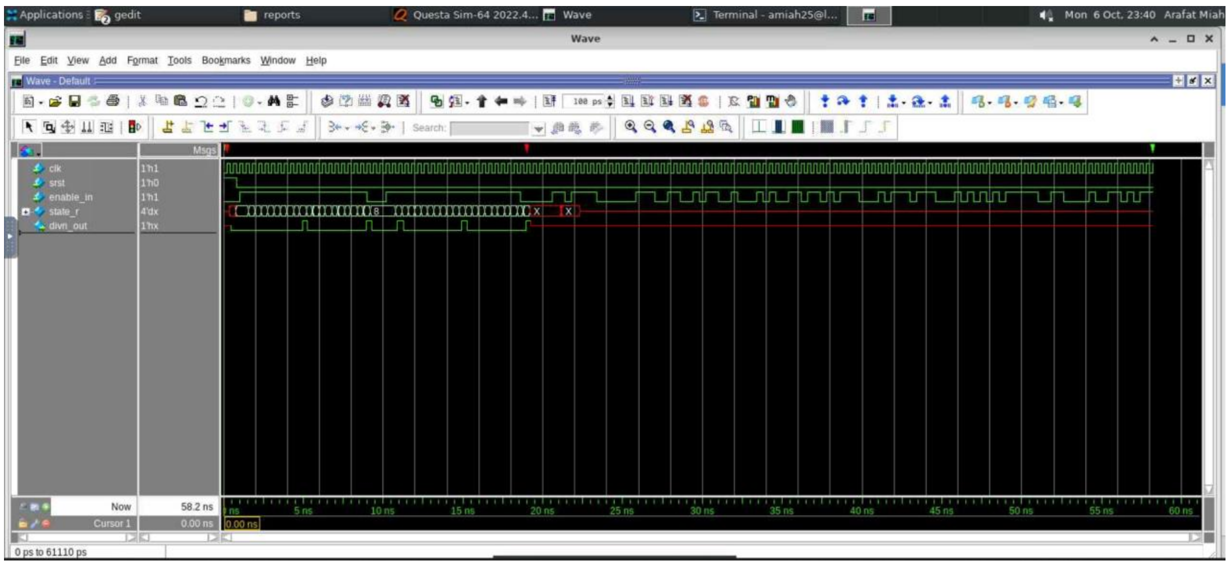
Timing Report

```
Point                                    Incr       Path
  ---------------------------------------------------------
  clock clk (rise edge)                    0.00       0.00
  clock network delay (ideal)              0.00       0.00
  state_r_reg[1]/CLK (DFFX1_HVT)           0.00       0.00 r
  state_r_reg[1]/QN (DFFX1_HVT)            0.13       0.13 f
  U28/Y (OR2X1_HVT)                        0.09       0.22 f
  U35/Y (NOR2X0_HVT)                       0.09       0.32 r
  U36/Y (OR2X1_HVT)                        0.07       0.38 r
  U40/Y (AND3X1_HVT)                       0.08       0.47 r
  state_r_reg[3]/D (DFFX1_HVT)             0.01       0.48 r
  data arrival time                                   0.48

  clock clk (rise edge)                    0.55       0.55
  clock network delay (ideal)              0.00       0.55
  state_r_reg[3]/CLK (DFFX1_HVT)           0.00       0.55 r
  library setup time                      -0.07       0.48
  data required time                                  0.48
  ---------------------------------------------------------
  data required time                                  0.48
  data arrival time                                  -0.48
  ---------------------------------------------------------
  slack (MET)                                         0.00
```

Q7) Present and comment on the results of the final gate-level simulation.

Gate Level Simulation Waveforms

reports/lab7_glsim_good.png.



Comments
The simulation confirms operation correctly and the timing constrain is in my case 0.55

Lab 8: Finite State Machine Design

Ex 1. ASM Chart Analysis

Q1) Present a state table for the functionality defined by the ASM chart presented in the assignment document.

State Table					
INPUTS			OUTPUTS		
Current State	enable	mode	Next State	ready	Pulse
Idle	0	0	Idle	1	0
Idle	0	0	Idle	1	0
Idle	1	0	Hold	1	0
Hold	1	1	End	0	1
End	1	1	Idle	0	1
Idle	1	1	Hold	1	0
Hold	0	1	Idle	0	0
Idle	0	1	Idle	1	0

Q2) Present the states of outputs ready and pulse.

Output Values by Clock Cycles		
Clock Edge	ready	pulse
1	1	0
2	1	0
3	1	0
4	0	1
5	0	1
6	1	0
7	0	0

8	1	0

Ex 2. RTL Code for ASM

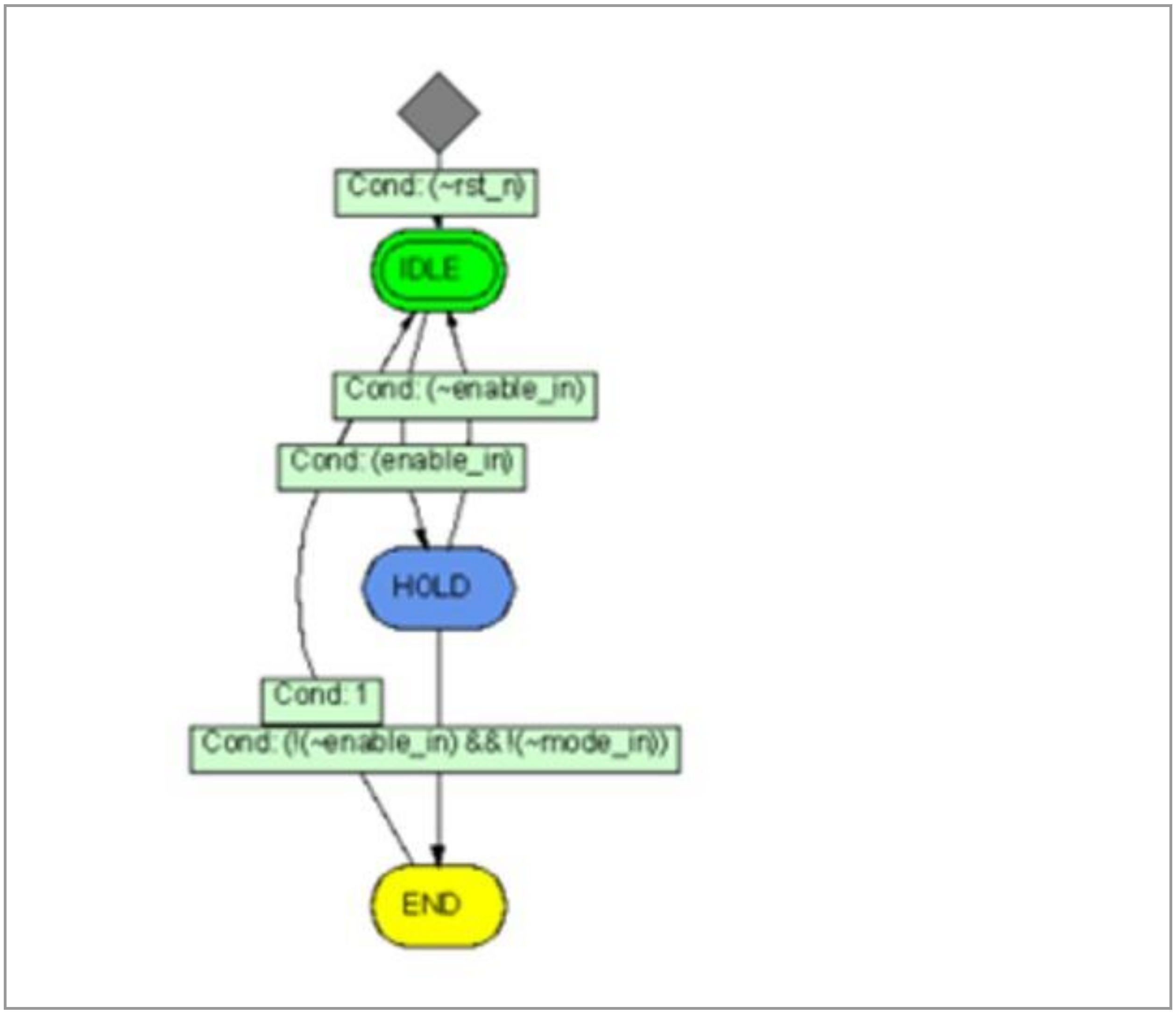
Q3. Describe the bug in the RTL code and how you fixed it.

Bug Description
The code was correct and runs perfectly.

Ex 3. FSM Simulation and Debug

Q4. Present the FSM state chart for the fixed version.

State Chart
Insert reports/lab8_fsm.png The diagram seems blur, I don’t know how this file lose it’s resolution, but when I took the screenshot it was fine.



Ex 4. State Encoding

Q3. What kind of state encoding is used in the RTL design?

Answer
That was a mealy type encoding

Lab 9: Testbench Techniques

Ex 1. Testbench Organization

Q1: Explain how the design hierarchy used in RTL simulation was defined in the code.

RTL Simulation Hierarchy			
Instance	**Design unit**	**Design unit type**	**File Name:** **Declaration (1-3 lines)** **Explanation**
ctr_tb	ctr_tb(fast)	Module	module ctr_tb; This is a top-level testbench which Instantiates DUT (ctr) and test program (ctr_test). In this part the clock and reset are generated.
#ALWAYS#34	ctr_tb(fast)	Process	Inside ctr_tb; This is the always block which will toggle the clk every 0.3ns.
DUT	ctr(fast)	Module	module ctr; The **Design Under Test**. It implements the counter logic and has parameters with inputs, outputs, and a sequential always_ff block.
TEST	ctr_test(fast)	Program	program ctr_test; The testbench stimulus generator. It sends data and mode signals, also observe data_out.
#INITIAL#15	ctr_test(fast)	Process	Inside ctr_test The initial block runs the test sequence (LOAD, UP/DOWN, etc.).
cb	ctr_test(fast)	ClockingBlock	Inside ctr_test:default clocking cb @(posedge clk); It defines timing and synchronization for driving and sampling signals.

Ex 4. Simulation with a Reference Model

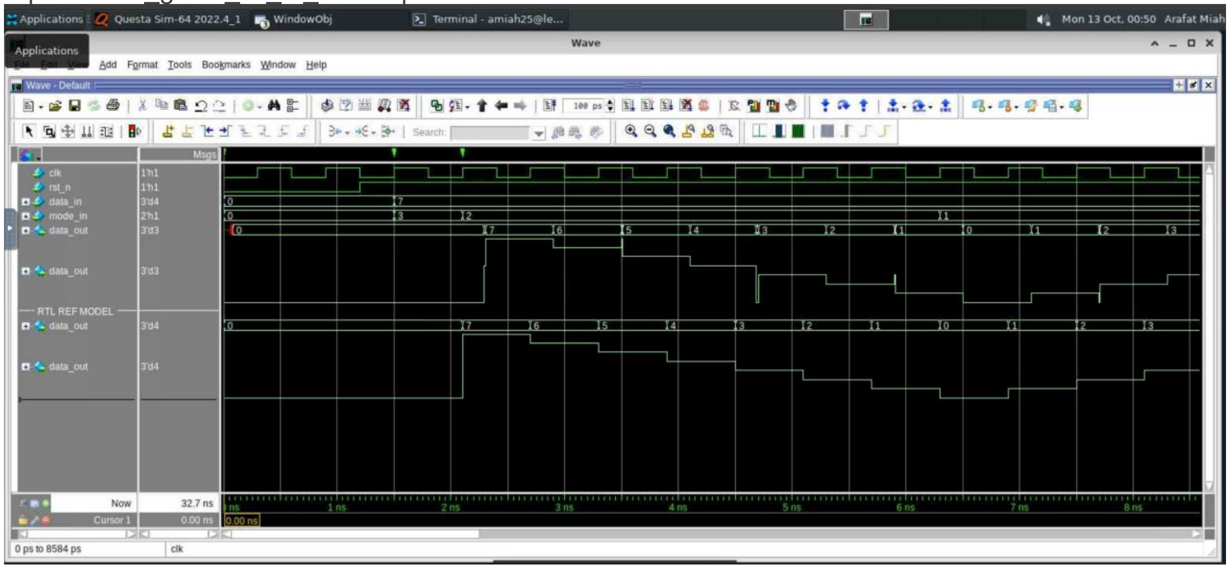
Q2. Explain how the design hierarchy used in Gate Level vs. RTLI simulation was defined in the code.

Gate Level vs. RTL Simulation Hierarchy			
Instance	**Design unit**	**Design unit type**	**File Name: Declaration (1-3 lines) Explanation**
ctr_tb	ctr_tb(fast)	Module	module ctr_tb; Top-level testbench, same as RTL. Instantiates both DUT (gate-level) and REF (RTL) counters.
DUT	ctr(fast)	Module	module ctr; This time, it is the **gate-level synthesized version** of the counter. Internally expanded into logic gates and flip-flops.
ctr_r_reg_0_	DFFARX1_HVT(fast)	Module	From synthesized netlist It represents one of the **flip-flops** implementing the register ctr_r[0]. Each bit of the counter becomes a DFF cell.
REF	ctr(fast)	Module	module ctr; The **reference RTL model** used for comparing results against the gate-level DUT.

Q3. Explanation of Differences in Timing and Values and the Causes. On which data was the changed timing based on?

Gate Level vs, RTL Simulation Waveforms

reports/lab9_gates_vs_rtl_waves.pnf.

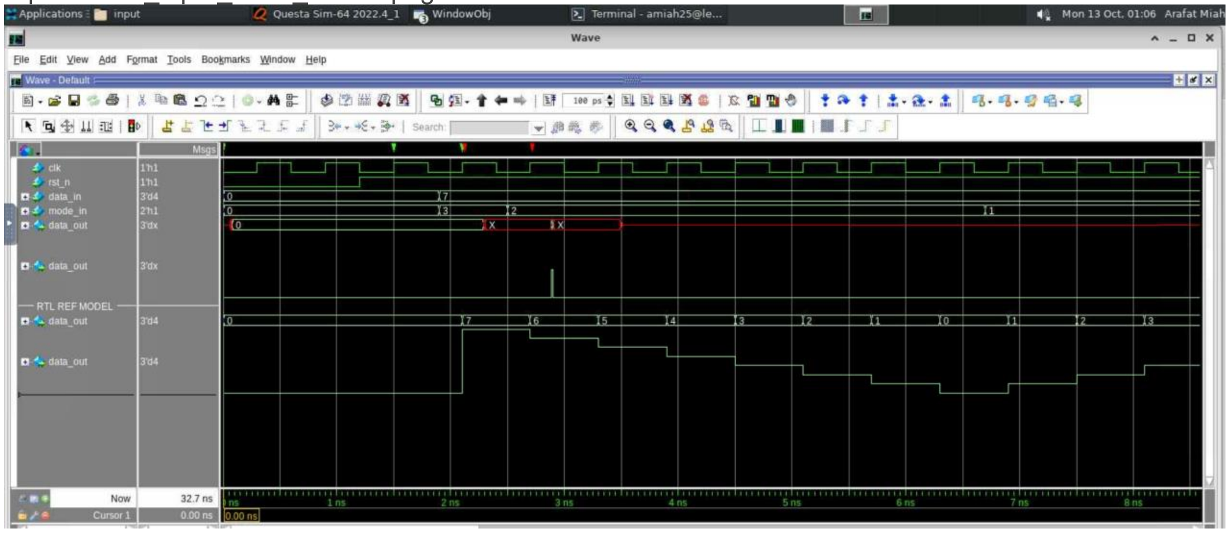


Explanations: the RTL implementation considers ideal timing, but the gates have their own

delay, which is considered in gate-level implementation,

Ex 5. Clocking block usage

Q4. Did the input skew setting have any effect on simulation timing? If timing errors appeared, what caused them?

Gate Level vs, RTL Simulation Waveforms with Input Skew
reports/lab9_input_skew_waves.png.  Applications input Questa Sim-64 2022.4_1 WindowObj Terminal - amiah25@le... Mon 13 Oct, 01:06 Arafat Miah Wave File Edit View Add Format Tools Bookmarks Window Help Wave - Default Msgs clk 1'h1 rst_n 1'h1 data_in 3'd4 mode_in 2'h1 data_out 3'dx data_out 3'dx RTL REF MODEL data_out 3'd4 data_out 3'd4 Now 32.7 ns Cursor 1 0.00 ns 1 ns 2 ns 3 ns 4 ns 5 ns 6 ns 7 ns 8 ns
Explanations
We have given it very small amount of time with respect to how much it needed, and it could not complete all the data propagation within the given time, and before completing all the tasks next clk cycle arrives.

Lab 10: Gate-Level Verification

Q1) What explains the differences in the operation of output comparators checker_1 and checker_2? Which of these would be more useful in practice?

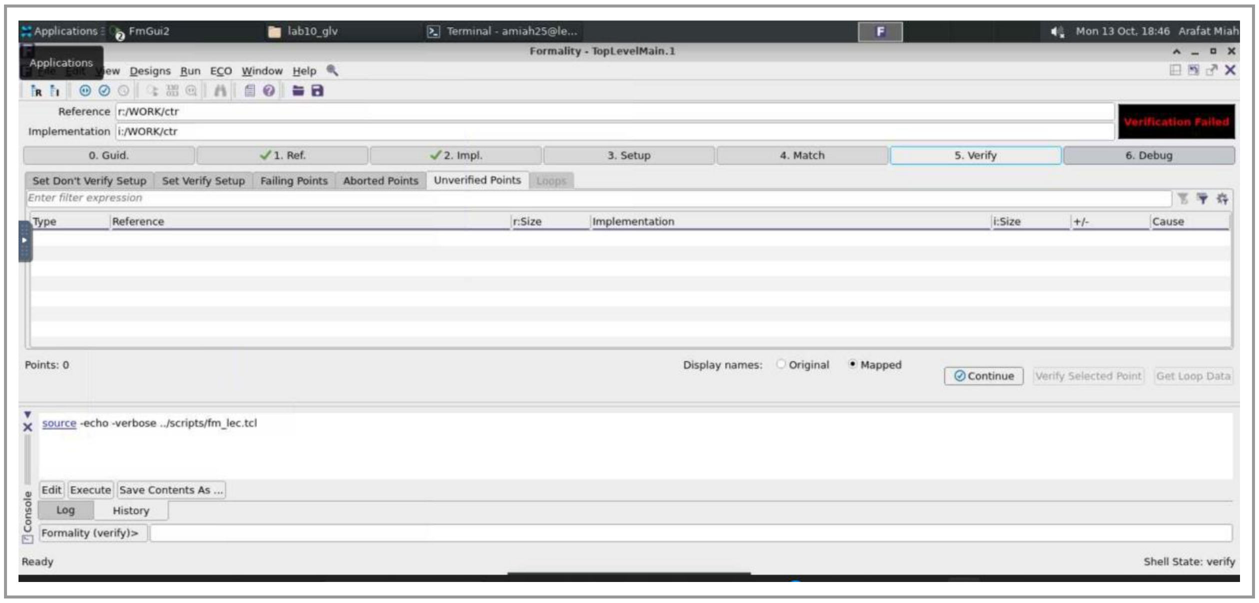
Answer
The difference occurs due to timing where**,** checker 1 compares outputs instantly on any change (including synthesis delays), while checker 2 compares them only synchronously at the clock edge when the sequential value is stable. I would suggest that checker 2 is more useful in practice, because it can verify the intended cycle-by-cycle functional correctness of the sequential logic.

Q2) Was the "self-inflicted" bug detected by Formality, and how was it reported? Also assess the ease of locating the reason for the verification failure from Formality's reports (including viewing logic cones).

Answer
We can see and observe the code that the code and simulation log do not show Formality output; they demonstrate a dynamic, assertion-based functional verification using a reference model comparison. The self-inflicted bug was not detected by Formality. As we know that, Formality will tell us a Non-Equivalence (FAIL) if the buggy RTL and the correct netlist were compared, or if the netlist did not preserve the logic of the buggy RTL. The self inflicted bug was detected by the tool identifyling the mismatched output/register and providing a counter-example vector. If we use formality then the bug can be found easily because Formality gives a precise counter-example and allows tracing the diverging logic.

Q3) Was the "self-inflicted" bug detected in simulation, and which of the tests T1 - T5 detected it, and why?

Answer
Yes, the self-inflicted bug was detected in the simulation (I provide a figure to prove this). T1 and T4 load successfully,T1 find the bug about T2 and T3, and then again T4 load successfully and it find the T5 bug



Reference r:/WORK/ctr
Implementation i:/WORK/ctr

Points: 0

```
source -echo -verbose ../scripts/fm_lec.tcl
```

Formality (verify)>

Shell State: verify

Lab 11: Finite-State Machine - Datapath Interaction

Ex 1. RTL Code Analysis

Q1. Explains the actions of each module in state S4. For control logic, describe the purpose of the control output values. For datapath logic modules, explain how they are controlled and what function the control activates. Present the relevant lines of code.

Activity in State S4		
Module	**Actions**	**Executed Code**

fsm	Here, the FSm first calculates the result of R1+R2 and then it writes this result to R4. And the next state is set to S5.
	The purpose of the control output which is:
	we_out=’1; enables one register and write: rsel_out=R4.
	asel_out and bsel_out select the R1 and R2 as ALU inputs.
	dsel_out write the ALU output.
	fsel_out = ADD; here it selects the addition function from ALU.
	done_out = '0;
		`I copied the code directly from the given example in linux:`
		`S4:`
		`begin`
		`next_state = S5;`
		`we_out = '1;`
		`rsel_out = R4;`
		`asel_out = R1;`
		`bsel_out = R2;`
		`dsel_out = ALU;`
		`fsel_out = ADD;`
		`done_out = '0;`
		`end`

rbank	rbank is controlled by we_in (this is coming from we_out) and rsel_in (which is coming from R4.
	
	The functionality of this block is: in the next rising edge clk the value of R1+R2 will be written inregister
		`always_ff @(posedge clk)`
		`begin : rbank`
		`if (srst)`
		`rb_r <= '0;`
		`else`
		`if (we_in)`
		`rb_r[rsel_in] <= busd;`
		`end : rbank`

muxa	It is controlled by asel_in
	And the functionality is that, it will select the value of R1 and N1 and then place them on busa
		`always_comb`
		`begin : muxa`
		`busa = rb_r[asel_in];`
		`end : muxa`

muxb	Similarly like muxa ,muxb will place the value on busb, however here it will select the value of R2 and N2	`always_comb`
		`begin : muxb`
		`busb = rb_r[bsel_in];`
		`end : muxb`

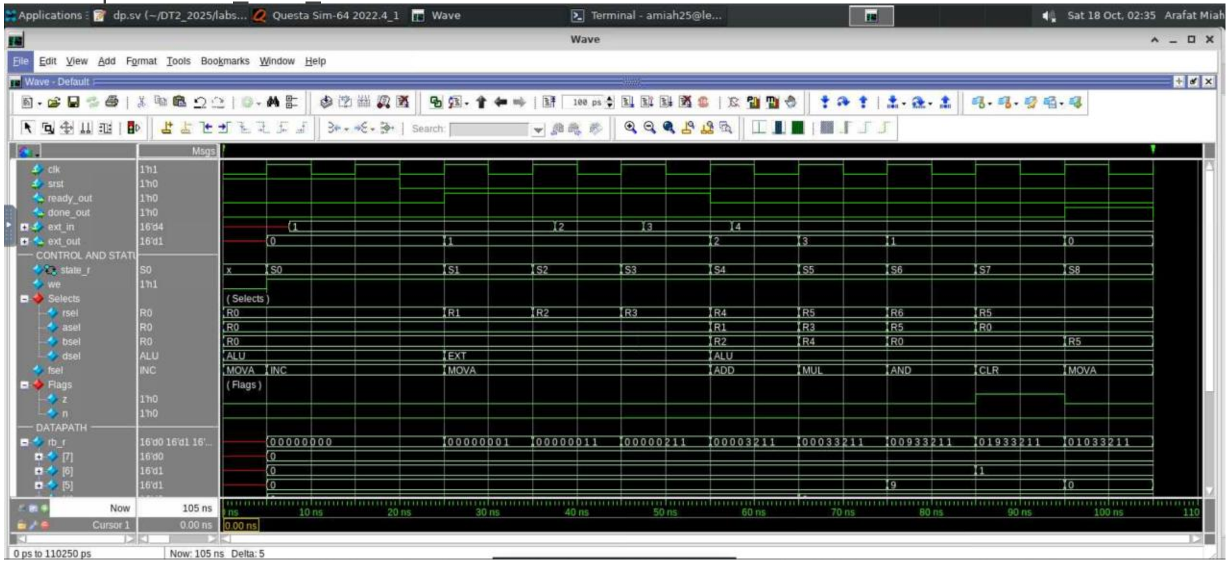
muxd	It will select the value of alu and then place it on busd	`always_comb`
		`begin : muxd`
		`if (dsel_in == ALU)`
		`busd = alu;`
		`else`
		`busd = $signed(ext_in);`

		`end : muxd`

ALU	ALU original functionality can be seen in the dp.sv file where it operate different kind of operations, however for S4 operation it just operating adding	`ADD:`
		`alu = busa + busb;`

Ex 2. Simulation

Q2. Present annotated waveform image.

Annotated Simulation Waveforms
Insert reports/lab11_fsmd_waves.* 

Q3. Explain the variable state changes and their order-

Events in State S4 at time 55ns			
Step	**Variable(s)**	**Old Value**	**New Value**
0	state_r	S3	S4
1	asel	R0	R1
2	bsel	R0	R2
2	rsel	R3	R4
2	bus	0	1
2	Busb	1	2
2	busd	3	2

3	alu	2	3
3	busd	2	3

Lab 12: FPGA Memories

Ex 1. Storage Resource Requirements Analysis

Q1. Present storage resource requirements as the number of bits for the variables in the RTL design.

Module	Variable	Bits Per Variable
echo	addr_ctr_r	18
	attn_in_r	12
	audio_in_r	16
	audio_out_r	16
	clkdiv_r	11
	dtime_in_r	18
	full_r	1
	read_data	Wire, so no bit required.
	tick	Wire, so no bit required.
	write_data	Wire, so no bit required.
mem	mem_r	Here, the bit is calculated with the NBITs* Delay_max_length= 24*8192= 196608
	rdata_r	24
Total Bits		196724

Q2. Present the storage resources that are available of XCA50T FPGA chips and comment on the feasibility of implementing the design using this specific chip type.

XC7A50T Storage Resources			
Resource	**Number of Resources Available**	**Number of Storage Bits Available**	**Is the Design Feasible with This Resource Type Only (YES/NO)**
CLB Flip-Flops	-	65,200	No
Maximum Distributed RAM (Kb)	600	600000	Yes
Total Block RAM (Kb)	2,700	2700000	Yes

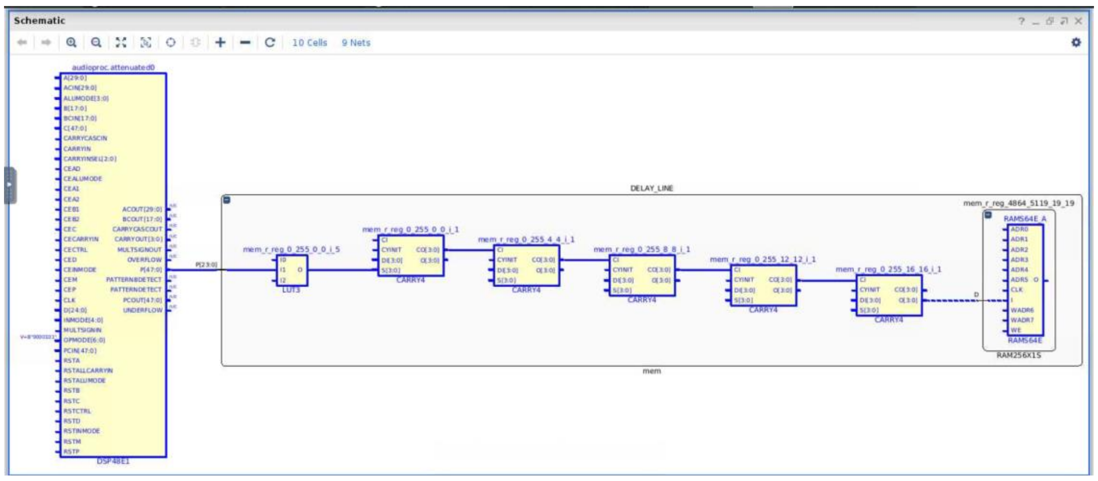
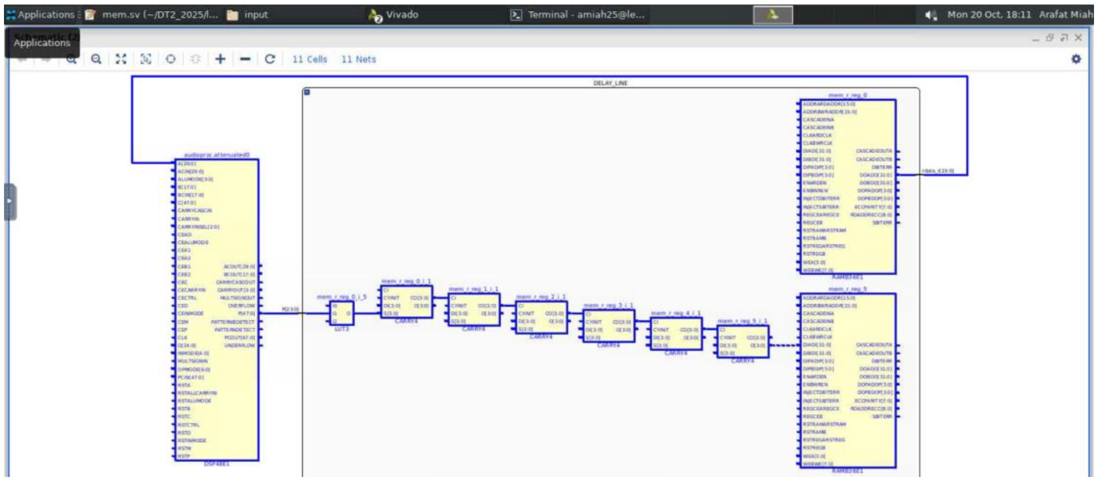
Ex 2: FPGA Implementation with User-Defined RAM Style

Q3. Present run status (success, fail … reason), Storage resource utilization and timing results.

FPGA Implementation Results				
Run #	**RAM Style**	**Status**	**Utilization % of Storage Resource**	**Slack**
1	`registers`	`fail`	`-`	
2	`distributed`	`successful`	`0.20%`	`0.139ns`
3	`block`	`successful`	`8%`	`-1.195ns`

Q4. Present path schematic

Critical Path Schematics	
Run #	**Schematic**
1	`-`

2	
3	

Q5. Compare the timing properties of the designs.

Comparison of Timing Properties of Different RAM Implementation Styles
We can observe that the timing slack for worst case for block RAM implementation is negative whereas the distributed RAM implementation has timing slack of positive, which clearly indicate that the distributed RAM implementation meets the timing requirements and block RAM implementation fails to meet the requirements.

The relevant text reports are:
echo_timing_summary_routed.rpt
echo_utilization_placed.rpt