**FLIP FLOPS AND**

**LATCHES IN**

**VERILOG**

**LAB # 0****9**

**Fall 2023**

**CSE-304L**

**Computer Organization & Architecture Lab**

Submitted by: **AIMAL KHAN**

Registration No.: **21PWCSE1996**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”



Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Dr. Bilal Habib**

Friday, December 29, 2023

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

**ASSESSMENT RUBRICS COA LABS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **LAB REPORT ASSESSMENT** | | | | |
| **Criteria** | **Excellent** | **Average** | **Nill** | **Marks Obtained** |
| 1. **Objectives of Lab** | All objectives of lab are properly covered  [Marks 10] | Objectives of lab are partially covered  [Marks 5] | Objectives of lab are not shown  [Marks 0] |  |
| 1. **MIPS instructions with**   **Comments and proper indentations.** | All the instructions are well written with comments explaining the code and properly indented  [Marks 20] | Some instructions are missing are poorly commented code  [Marks 10] | The instructions are not properly written  [Marks 0] |  |
| 1. **Simulation run without error and warnings** | The code is running in the simulator without any error and warnings  [Marks 10] | The code is running but with some warnings or errors.  [Marks 5] | The code is written but not running due to errors  [Marks 0] |  |
| 1. **Procedure** | All the instructions are written with proper procedure  [Marks 20] | Some steps are missing  [Marks 10] | steps are totally missing  [Marks 0] |  |
| 1. **OUTPUT** | Proper output of the code written in assembly  [Marks 20] | Some of the outputs are missing  [Marks 10] | No or wrong output  [Marks 0] |  |
| 1. **Conclusion** | Conclusion about the lab is shown and written  [Marks 20] | Conclusion about the lab is partially shown  [Marks 10] | Conclusion about the lab is not shown [Marks0]  [Marks 0] |  |
| 1. **Cheating** |  |  | Any kind of cheating will lead to 0 Marks |  |
| Total Marks Obtained: \_\_\_\_\_\_\_\_\_\_  Instructor Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | | | | |

**Flip Flop and Latches in Verilog**

Objectives:

* Write and implement latches in Verilog
* Write and implement flipflops in Verilog
* Understand working and design of sequential circuits

Tasks:

**Task 1**: Write a Verilog code to implement S R latch.

**DUT Code:**

module SR\_latch(S, R, Q, Qbar);

input S;

input R;

output Q;

output Qbar;

assign Q = ~(S & Qbar);

assign Qbar = ~(R & Q);

endmodule

**Test Code:**

module SR\_latch\_tb();

reg S;

reg R;

wire Q;

wire Qbar;

SR\_latch latch(S, R, Q, Qbar);

initial begin

S = 0;

R = 1;

#10

$display("%b %b %b %b", S, R, Q, Qbar);

S = 1;

R = 1;

#10

$display("%b %b %b %b", S, R, Q, Qbar);

S = 1;

R = 0;

#10

$display("%b %b %b %b", S, R, Q, Qbar);

S = 1;

R = 1;

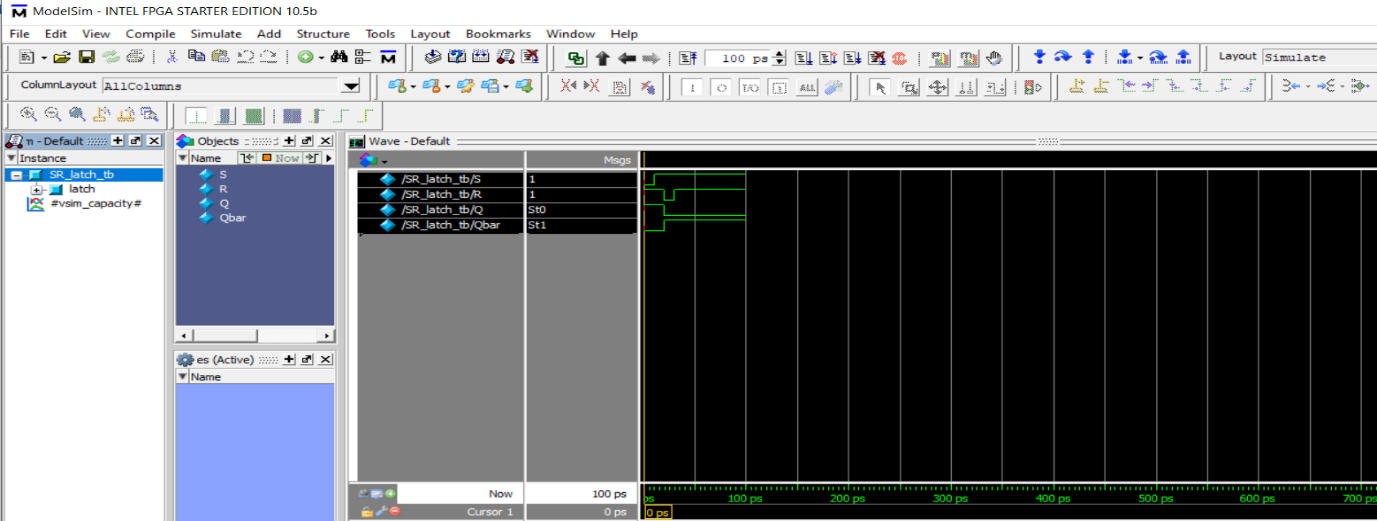
#10

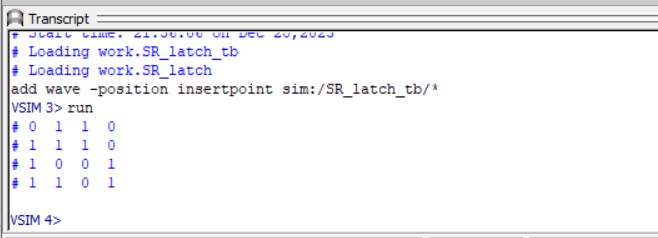
$display("%b %b %b %b", S, R, Q, Qbar);

end

endmodule;

**Output:**

****

****

**Task 2**: Implement SR flip flop in Verilog

**DUT Code:**

module SR\_flipflop(S, R, clk, Q, Qbar);

input S;

input R;

input clk;

output Q;

output Qbar;

wire nan1;

wire nan2;

assign nan1 = ~(S & clk);

assign nan2 = ~(R & clk);

assign Q = ~(nan1 & Qbar);

assign Qbar = ~(nan2 & Q);

endmodule;

**Test Code:**

module SR\_flipflop\_tb();

reg S;

reg R;

reg clk;

wire Q;

wire Qbar;

SR\_flipflop flipflop(S, R, clk, Q, Qbar);

initial begin

clk =1;

S = 0;

R = 1;

#10

$display("%b %b %b %b %b", S, R, clk, Q, Qbar);

clk =1;

S = 1;

R = 1;

#10

$display("%b %b %b %b %b", S, R, clk, Q, Qbar);

clk =1;

S = 1;

R = 0;

#10

$display("%b %b %b %b %b", S, R, clk, Q, Qbar);

clk =1;

S = 1;

R = 1;

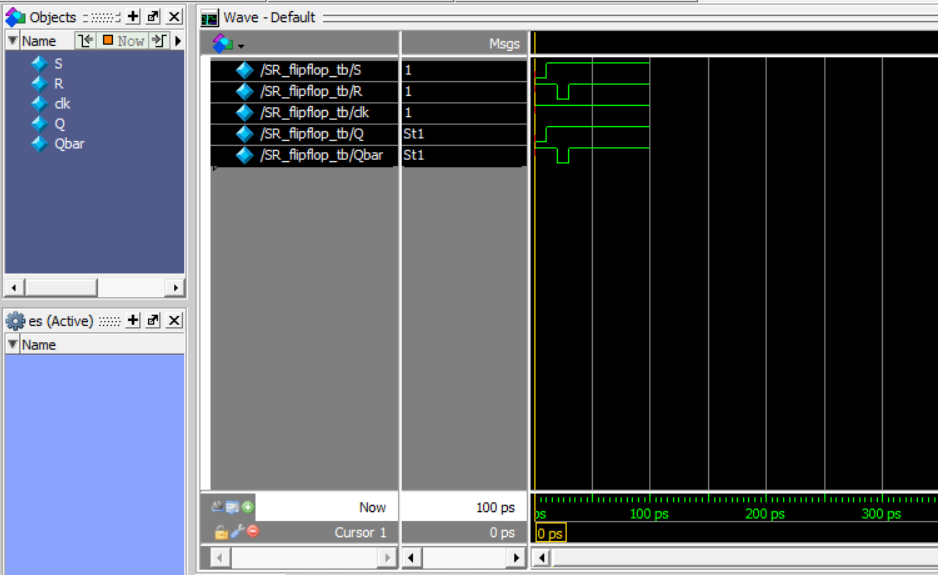
#10

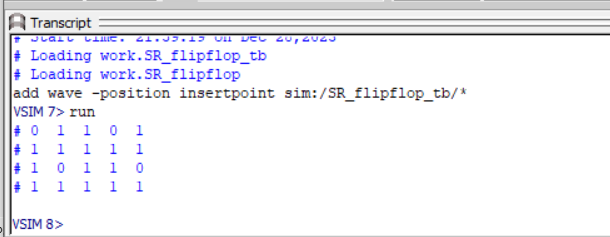
$display("%b %b %b %b %b", S, R, clk, Q, Qbar);

end

endmodule

**Output:**

****

****

**Task 3**: Implement JK Flip Flop in verilog

**DUT Code:**

module JK\_flipflop(J, K, clk, Q, Qbar);

input J;

input K;

input clk;

output Q;

output Qbar;

wire nan1;

wire nan2;

assign nan1 = ~(J & clk & Qbar);

assign nan2 = ~(K & clk & Q);

assign Q = ~(nan1 & Qbar);

assign Qbar = ~(nan2 & Q);

endmodule;

**Test Code:**

module JK\_flipflop\_tb();

reg J;

reg K;

reg clk;

wire Q;

wire Qbar;

JK\_flipflop flipflop(J, K, clk, Q, Qbar);

initial begin

clk =1;

J = 0;

K = 1;

#10

$display("%b %b %b %b %b", J, K, clk, Q, Qbar);

clk =1;

J = 1;

K = 1;

#10

$display("%b %b %b %b %b", J, K, clk, Q, Qbar);

clk =1;

J = 1;

K = 0;

#10

$display("%b %b %b %b %b", J, K, clk, Q, Qbar);

clk =1;

J = 1;

K = 1;

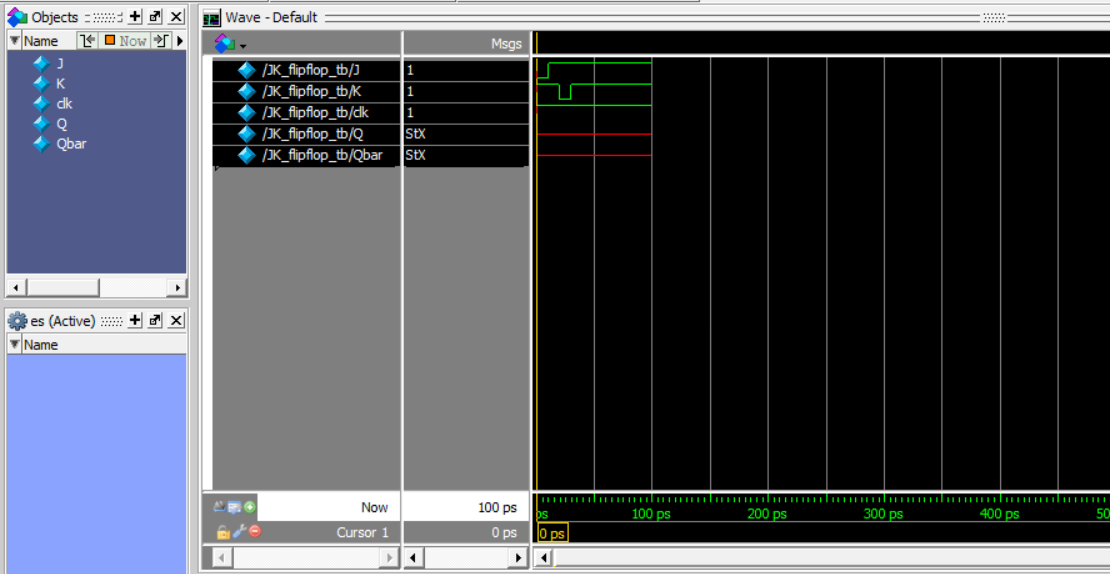
#10

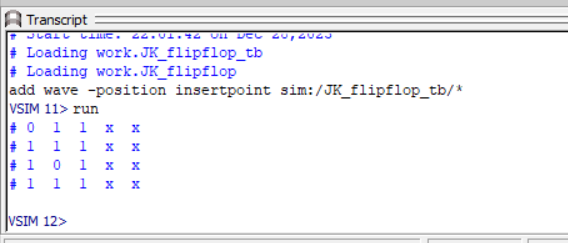
$display("%b %b %b %b %b", J, K, clk, Q, Qbar);

end

endmodule;

**Output:**

****

****

**Task 4**: Design D Flip Flop in Verilog

**DUT Code:**

module D\_flipflop(D, clk, Q, Qbar);

input D;

input clk;

output Q;

output Qbar;

wire nan1;

wire nan2;

wire nD;

not n1(nD, D);

assign nan1 = ~(D & clk);

assign nan2 = ~(nD & clk);

assign Q = ~(nan1 & Qbar);

assign Qbar = ~(nan2 & Q);

endmodule;

**Test Code:**

module D\_flipflop\_tb();

reg D;

reg clk;

wire Q;

wire Qbar;

D\_flipflop flipflop(D, clk, Q, Qbar);

initial begin

clk =0;

D = 0;

#10

$display("%b %b %b %b ", D, clk, Q, Qbar);

clk =1;

D = 0;

#10

$display("%b %b %b %b ", D, clk, Q, Qbar);

clk =1;

D = 1;

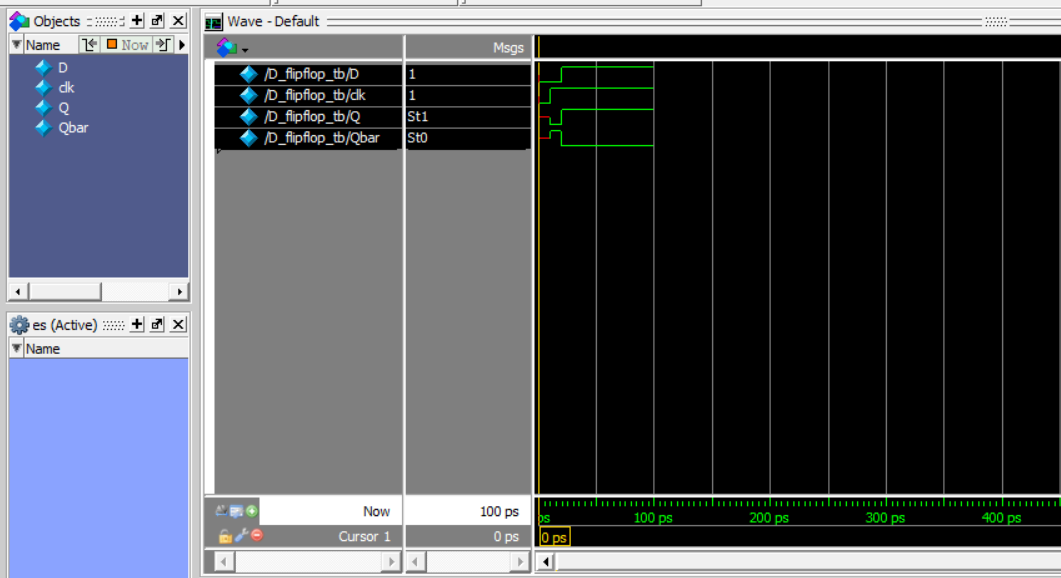
#10

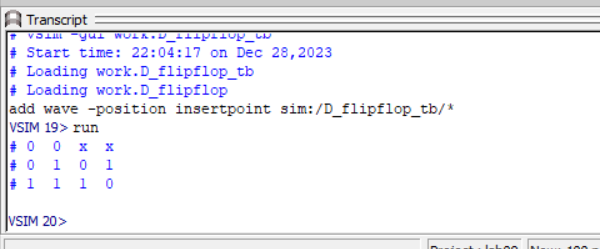
$display("%b %b %b %b", D, clk, Q, Qbar);

end

endmodule;

**Output:**

****

****

**Task 5**: Design T Flip Flop using D Flip Flop in Verilog.

**DUT Code:**

module T\_flipflop(T, clk, Q, Qbar);

input T;

input clk;

output Q;

output Qbar;

wire D;

wire Q;

xor x1(D, T, Q);

D\_flipflop d1(D, clk, Q, Qbar);

endmodule;

**Test Code:**

module T\_flipflop\_tb();

reg T;

reg clk;

wire Q;

wire Qbar;

T\_flipflop flipflop(T, clk, Q, Qbar);

initial begin

$display( "clk T Q Qbar ");

clk =0;

T = 0;

#10

$display("%b %b %b %b ", clk , T, Q, Qbar);

clk =1;

T = 0;

#10

$display("%b %b %b %b ", clk , T, Q, Qbar);

clk =1;

T = 1;

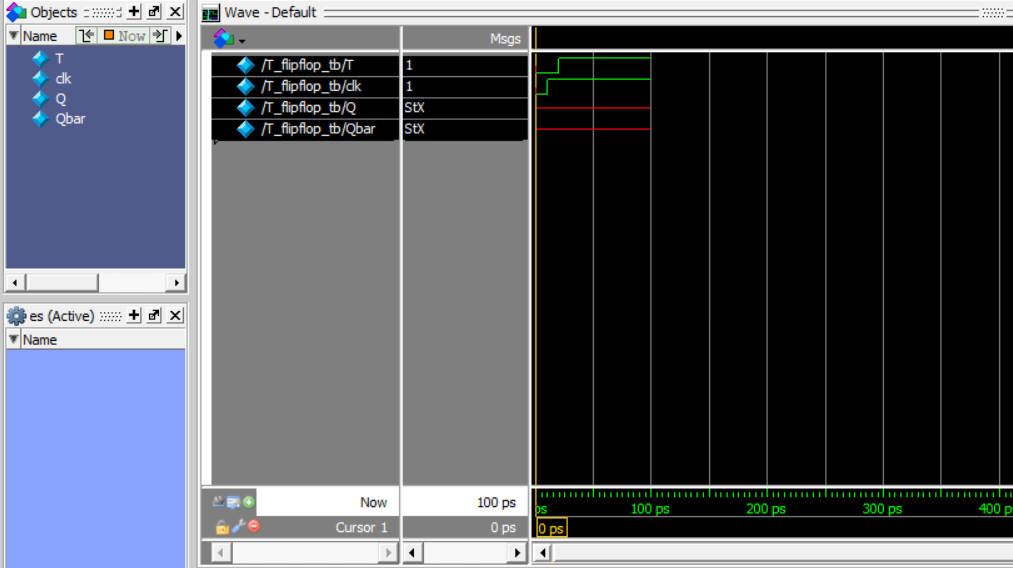
#10

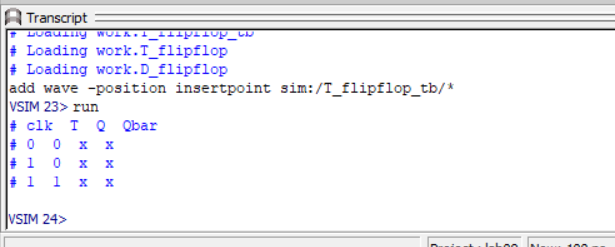
$display("%b %b %b %b", clk , T, Q, Qbar);

End

endmodule;

**Output:**

****

****

Reference:

To view my codes, please refer to my [GitHub Account.](https://github.com/aimalexe/DCSE/tree/main/semester_5_(fall-23)/computer_organization_and_architechure_lab/lab_reports/)

Conclusion:

In conclusion, I have learned how to work in Verilog to implement flip flops and latches of various types. Now I am able to use my founding’s in real world problems solving stratigies.

The End.