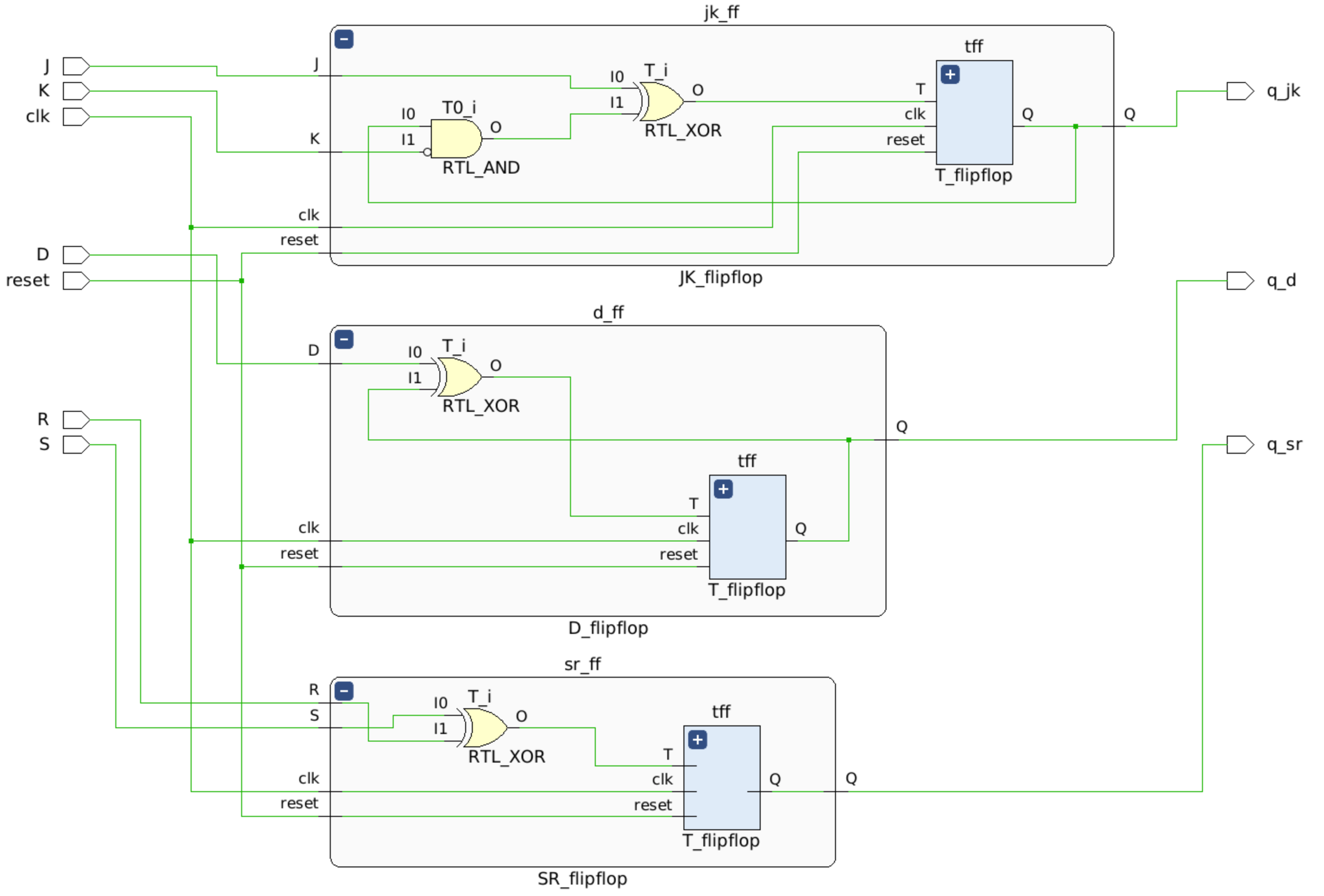
14 Cells 10 I/O Ports 38 Nets



ns ui-PlanAhead

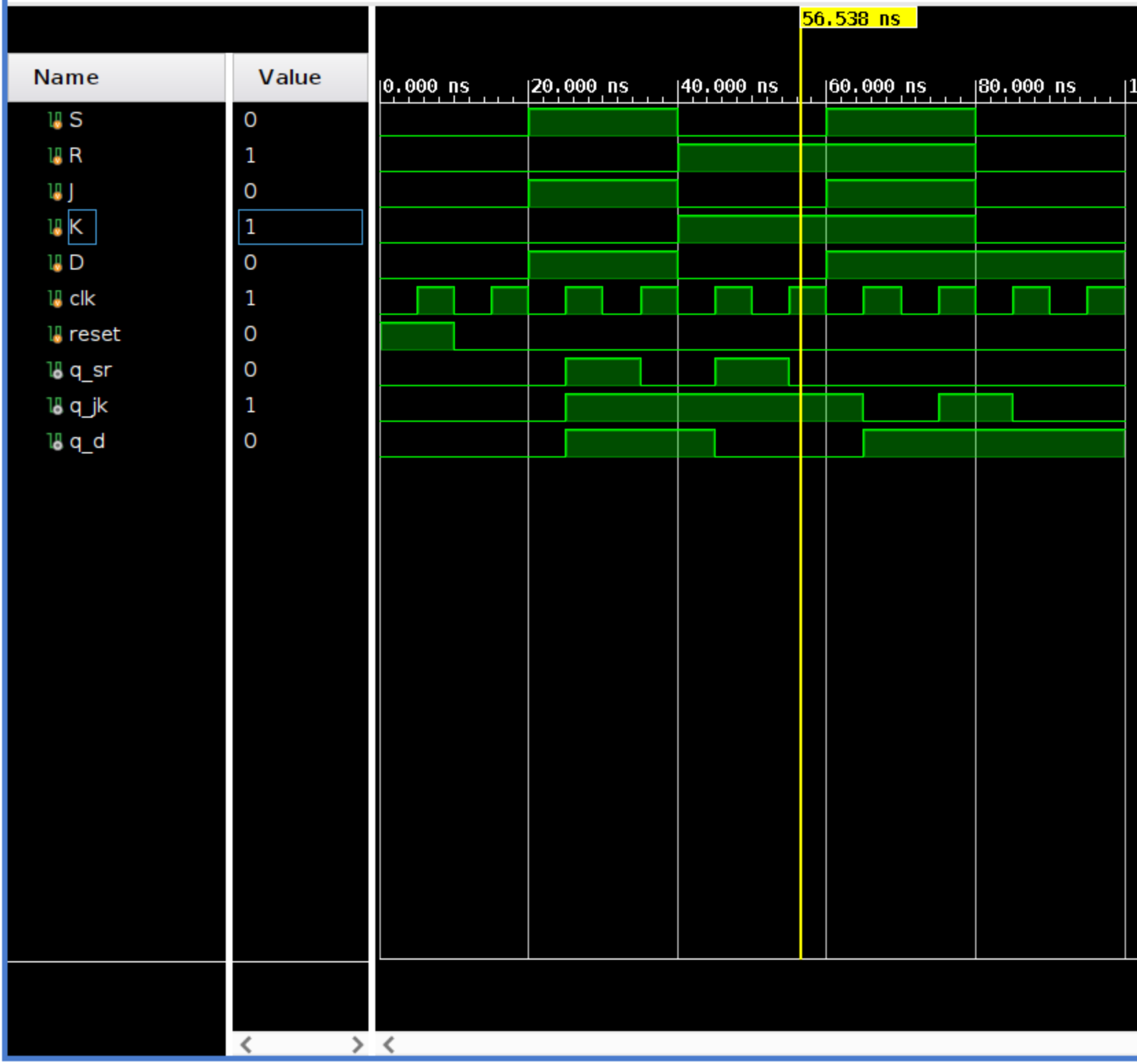
project_1 - [/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/pro

Window Layout View Run Help Quick Access

100 ms

MULATION - Behavioral Simulation - Functional - sim_1 - T_conversion_tb

T_conversion.v × T_flipflop.v × D_flipflop.v × JK_flipflop.v × SR_flipflop.v × T_conver



Tcl Console Messages Log

T_conversion.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/project_1.srcs/sources_1/new/T_conversion.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Engineer: Anjan Prasad
4  // Create Date: 11/14/2024 04:07:54 AM
5  // Module Name: T_conversion
6  //////////////////////////////////////////////////////////////////////////////////
7
8  module T_conversion (
9      input S, R, J, K, D, clk, reset,
10     output q_sr, q_jk, q_d
11 );
12     // Instantiate SR, JK, and D flip-flops using T flip-flop
13     SR_flipflop sr_ff (.S(S), .R(R), .clk(clk), .reset(reset), .Q(q_sr));
14     JK_flipflop jk_ff (.J(J), .K(K), .clk(clk), .reset(reset), .Q(q_jk));
15     D_flipflop d_ff (.D(D), .clk(clk), .reset(reset), .Q(q_d));
16 endmodule
17
18
```

T_conversion_tb.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/project_1.srcs/sim_1/new/T_conversion_tb.v

```
2  //////////////////////////////////////////////////////////////////////////////////
3  // Engineer: Anjan Prasad
4  // Create Date: 11/14/2024 04:18:18 AM
5  // Module Name: T_conversion_tb
6  //////////////////////////////////////////////////////////////////////////////////
7
8  module T_conversion_tb;
9      reg S, R, J, K, D, clk, reset;
10     wire q_sr, q_jk, q_d;
11
12     T_conversion uut (.S(S), .R(R), .J(J), .K(K), .D(D),.clk(clk), .reset(reset),.q_sr(q_sr), .q_jk(q_jk), .q_d(q_d));
13     initial begin
14         clk = 0;
15         forever #5 clk = ~clk;
16     end
17     initial begin
18         reset = 1;
19         #10 reset = 0; // Release reset after 10 ns
20     end
21     initial begin
22         S = 0; R = 0;
23         #20 S = 1; R = 0;  // Set condition
24         #20 S = 0; R = 1;  // Reset condition
25         #20 S = 1; R = 1;  // Invalid condition
26         #20 S = 0; R = 0;  // Hold condition
27     end
28     initial begin
29         J = 0; K = 0;
30         #20 J = 1; K = 0;  // Set condition
31         #20 J = 0; K = 1;  // Reset condition
32         #20 J = 1; K = 1;  // Toggle condition
33         #20 J = 0; K = 0;  // Hold condition
34     end
35
36     initial begin
37         D = 0;
38         #20 D = 1;         // Set D to 1
39         #20 D = 0;         // Set D to 0
40         #20 D = 1;         // Set D to 1 again
41     end
42     initial begin
43         $monitor("Time = %0d | S = %b, R = %b, q_sr = %b | J = %b, K = %b, q_jk = %b | D = %b, q_d = %b",
44                  $time, S, R, q_sr, J, K, q_jk, D, q_d);
45     end
46     initial begin
47         #100 $stop;
48     end
49 endmodule
50
51
```

SR_flipflop.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/project_1.srcs/sources_1/new/SR_flipflop.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Engineer: Anjan Prasad
4  // Create Date: 11/14/2024 04:09:24 AM
5  // Module Name: SR_flipflop
6  //////////////////////////////////////////////////////////////////////////////////
7
8  module SR_flipflop (
9      input S, R, clk, reset,
10     output Q
11 );
12     wire T;
13     assign T = S ^ R; // T is high when S and R are different
14     T_flipflop tff (.T(T), .clk(clk), .reset(reset), .Q(Q));
15 endmodule
16
17
```

JK_flipflop.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/project_1.srcs/sources_1/new/JK_flipflop.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Engineer: Anjan Prasad
4  // Create Date: 11/14/2024 04:09:24 AM
5  // Module Name: JK_flipflop
6  //////////////////////////////////////////////////////////////////////////////////
7
8  module JK_flipflop (
9      input J, K, clk, reset,
10     output Q
11 );
12     wire T;
13     assign T = J ^ (Q & ~K); // T depends on JK inputs and current Q
14     T_flipflop tff (.T(T), .clk(clk), .reset(reset), .Q(Q));
15 endmodule
16
17
```

T_flipflop.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/project_1.srcs/sources_1/new/T_flipflop.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Engineer: Anjan Prasad
4  // Create Date: 11/14/2024 04:09:24 AM
5  // Module Name: T_flipflop
6  //////////////////////////////////////////////////////////////////////////////////
7
8  module T_flipflop (
9      input T, clk, reset,
10     output reg Q
11 );
12     always @(posedge clk or posedge reset) begin
13         if (reset)
14             Q <= 1'b0;
15         else if (T)
16             Q <= ~Q;
17     end
18 endmodule
19
```

D_flipflop.v

/home/itzzinfinity/Cozy Drive/100daysofRTL/day_054/project_1/project_1.srcs/sources_1/new/D_flipflop.v

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////////////////////////////////////
3  // Engineer: Anjan Prasad
4  // Create Date: 11/14/2024 04:09:24 AM
5  // Module Name: D_flipflop
6  //////////////////////////////////////////////////////////////////////////////////
7
8  module D_flipflop (
9      input D, clk, reset,
10     output Q
11 );
12     wire T;
13     assign T = D ^ Q; // T flips only when D is different from current Q
14     T_flipflop tff (.T(T), .clk(clk), .reset(reset), .Q(Q));
15 endmodule
16
17
```