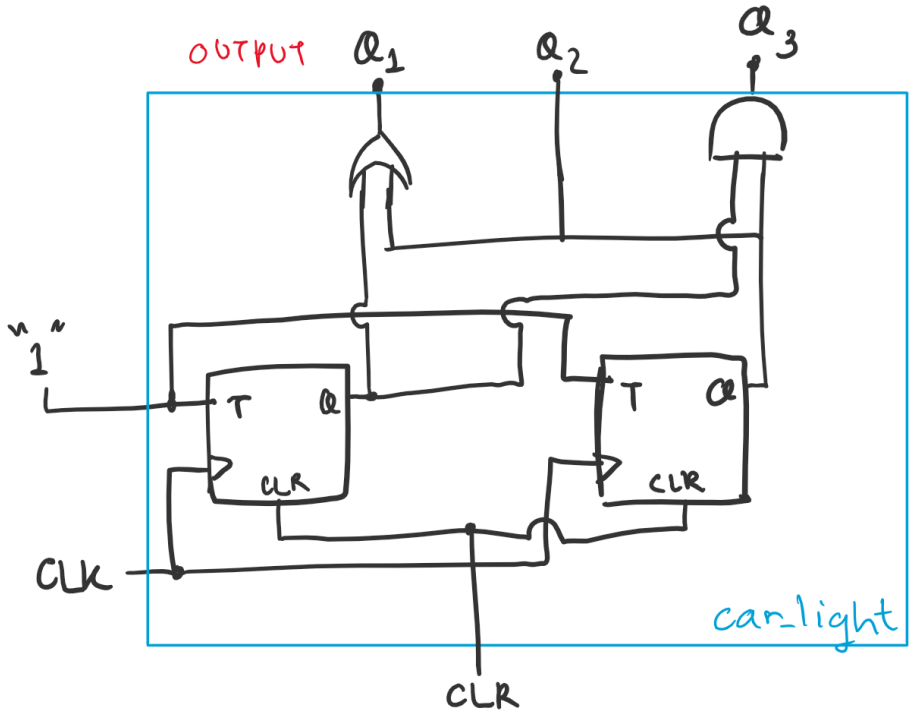
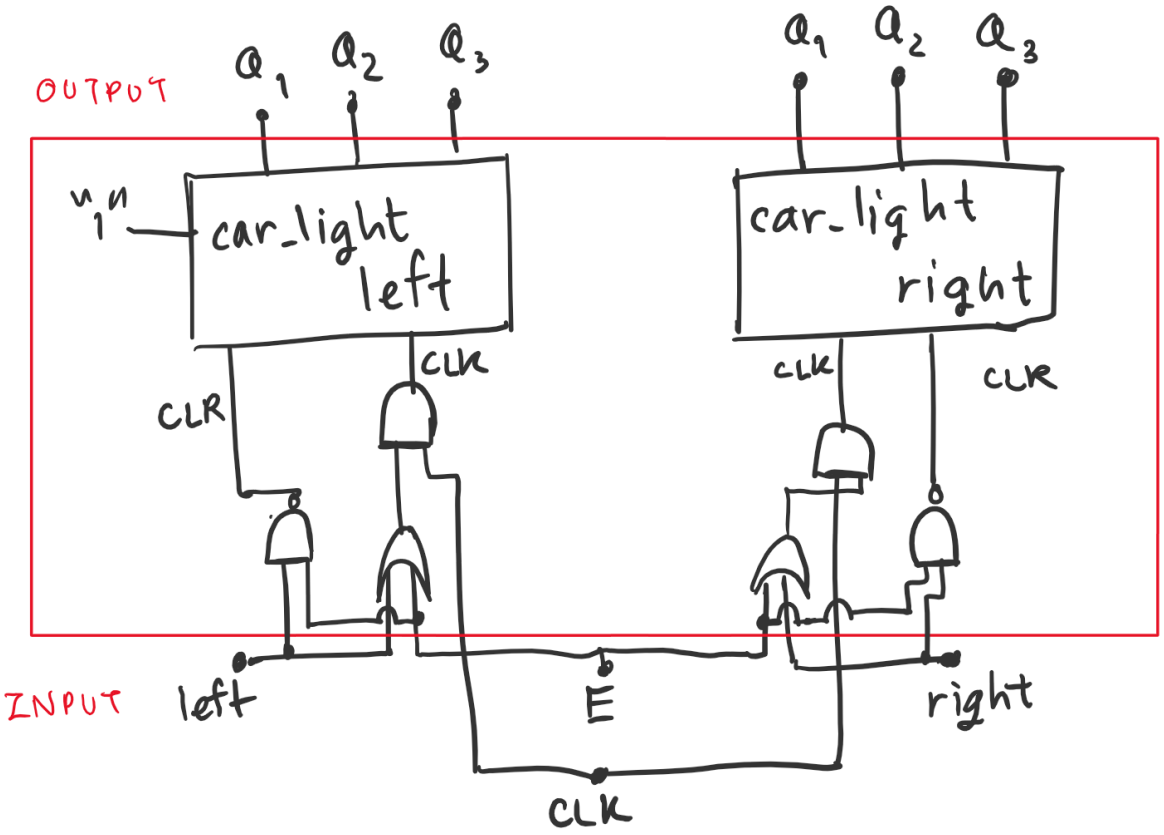
1. ไฟรถยนต์

a. Logic Gate Diagram





b. Verilog Code

----- car.v -----

```
module T_FF(q, t, clk, reset);
    output q;
    input t,clk, reset;
    reg q;

    initial
        begin
            q=1'b0;
        end

    always @ (posedge clk or posedge reset)
    if(reset)
        q <= 0;
    else
        begin
            q <= (t == 1) ? ~q : q;
        end
endmodule


module car_light(q, t, clk, reset);
    output [2:0] q;
    input t, clk, reset;

    wire q0, q1;


    T_FF t_ff1(q0, t, clk, reset);
    T_FF t_ff2(q1, q0, clk, reset);

    assign q[0] = q0 | q1 ;
    assign q[1] = q1;
    assign q[2] = q0 & q1;

endmodule
```

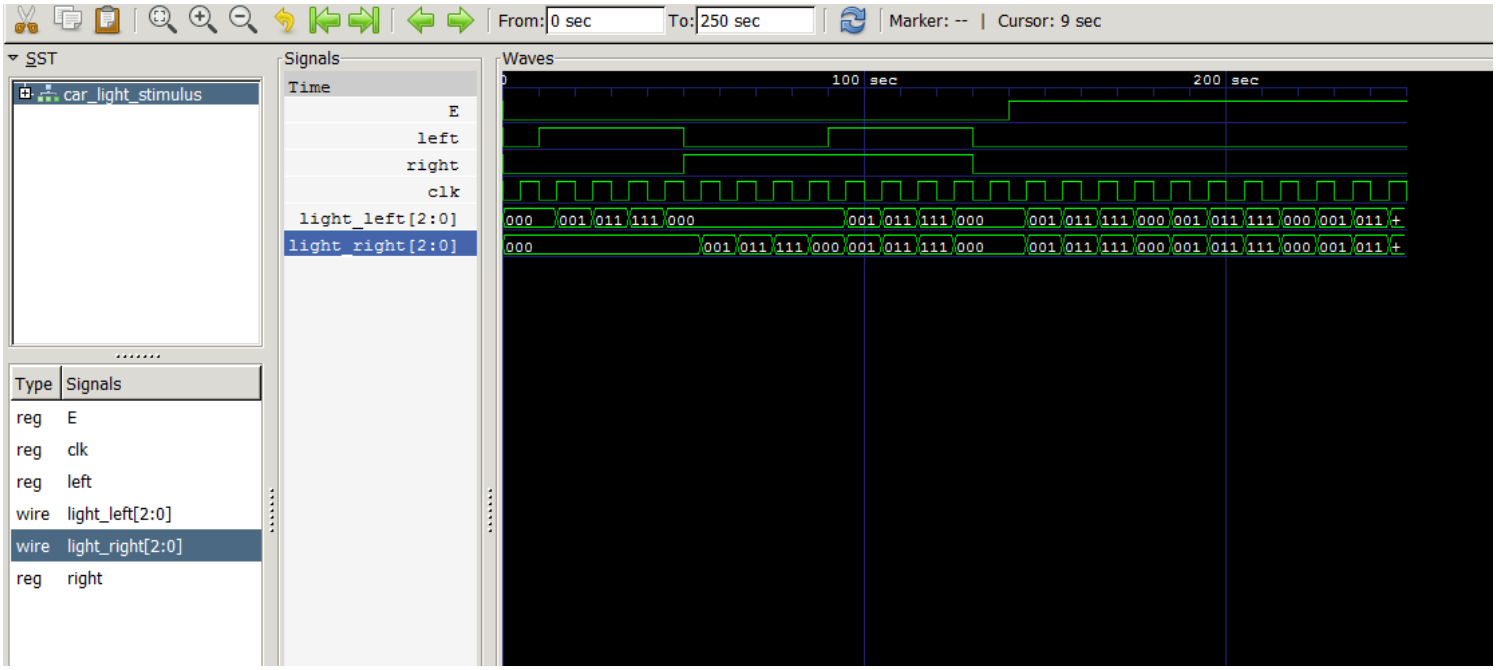
```
module car(light_left, light_right, left , right, E, clk);

    output [2:0] light_left, light_right;
    input left, right, E, clk;

    wire or_clk_out_1, re1, clk_left;
    wire or_clk_out_2, re2, clk_right;

    nor or_re_1(re1, left, E);
    or or_clk_1(or_clk_out_1, left, E);
    and and_clk_1(clk_left, or_clk_out_1, clk);
    car_light car_light_left(light_left, 1'b1, clk_left, re1);

    nor or_re_2(re2, right, E);
    or or_clk_2(or_clk_out_2, right, E);
    and and_clk_2(clk_right, or_clk_out_2, clk);
    car_light car_light_right(light_right, 1'b1, clk_right, re2);

endmodule
```

----- car_stimulus.v -----

```
module car_light_stimulus;
    reg left, right, E, clk;
    reg din_0, din_1;
    wire [2:0] light_left, light_right;

    car_light car1(light_left, light_right, left , right, E, clk);

    initial
        begin
            $dumpfile("TimeDiagram.vcd");
            $dumpvars(0,car_light_stimulus);
            left = 1'b0;
            right = 1'b0;
            E = 1'b0;
            clk = 1'b0;
        end

    always
        #5 clk = !clk;

    initial
        begin
          #10 left = !left;
```

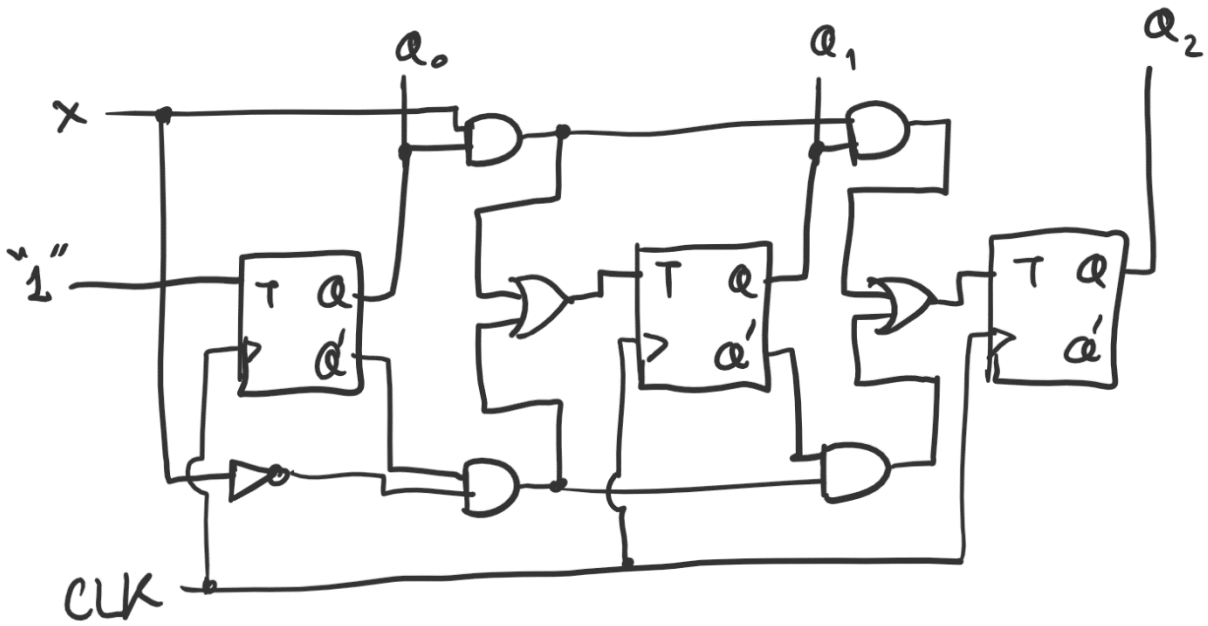
```
            #40 left = !left;
                right = !right;
            #40 left = !left;
            #40 left = !left;
                right = !right;
            #10 E = !E;
          end

    initial
    begin
        #250 $finish;
    end
    initial
        $monitor ($time, clk, light_left, light_right, E);
endmodule
```

C. Time diagram



2. Count Up and Down

Logic Gate Diagram

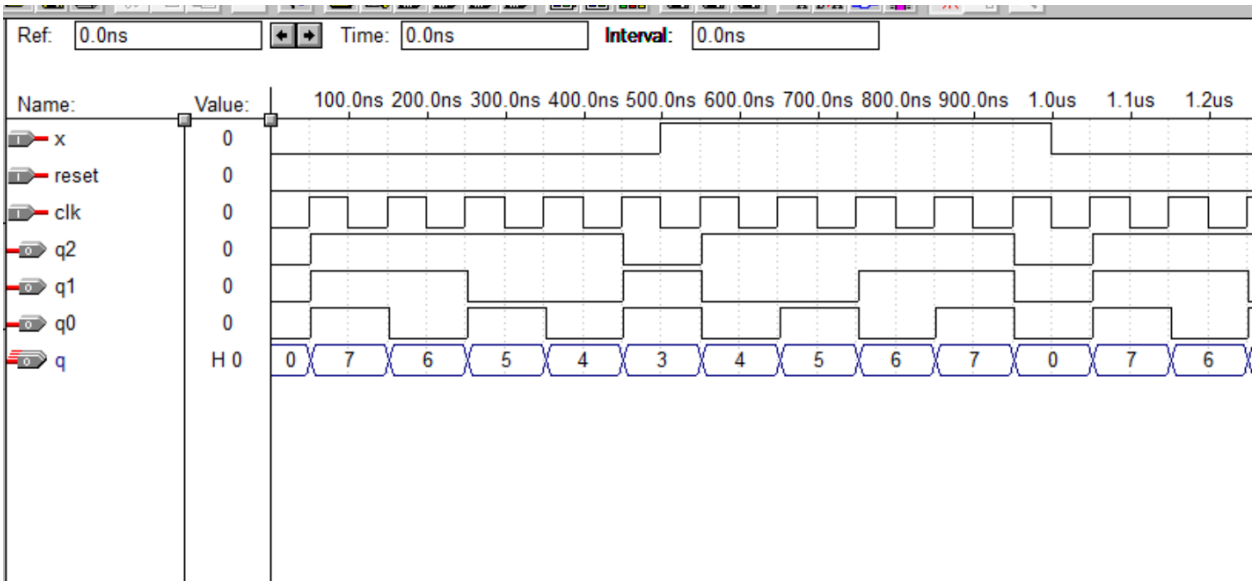


Verilog Code

```
module T_FF(q, t, clk, reset);
    output q;
    input t,clk, reset;
    reg q;

    initial
        begin
            q=1'b0;
        end

    always @ (posedge clk)
    if(reset)
        q <= 0;
    else
        begin
            q <= (t == 1) ? ~q : q;
        end
endmodule
```

```
module count_up_down(q, x, clk, reset);

output [2:0] q;
input x, clk, reset;

wire and_out1,and_out2,or_out1;
wire and2_out1,and2_out2,or_out2;
wire q1, q2, q3;

T_FF t_ff1(q1, 1'b1, clk, reset);

and and1_1(and_out1, !x, !q1);
and and1_2(and_out2, x, q1);
or or1_1(or_out1, and_out1, and_out2);

T_FF t_ff2(q2,  or_out1, clk, reset);

and and2_1(and2_out1, and_out1, !q2);
and and2_2(and2_out2, and_out2, q2);
or or2_1(or_out2, and2_out1, and2_out2);

T_FF t_ff3(q3, or_out2, clk, reset);

assign q[0] = q1;
assign q[1] = q2;
assign q[2] = q3;

endmodule
```

Time Diagram



3. <u>Gray Code</u>

Logic Gate Diagram