



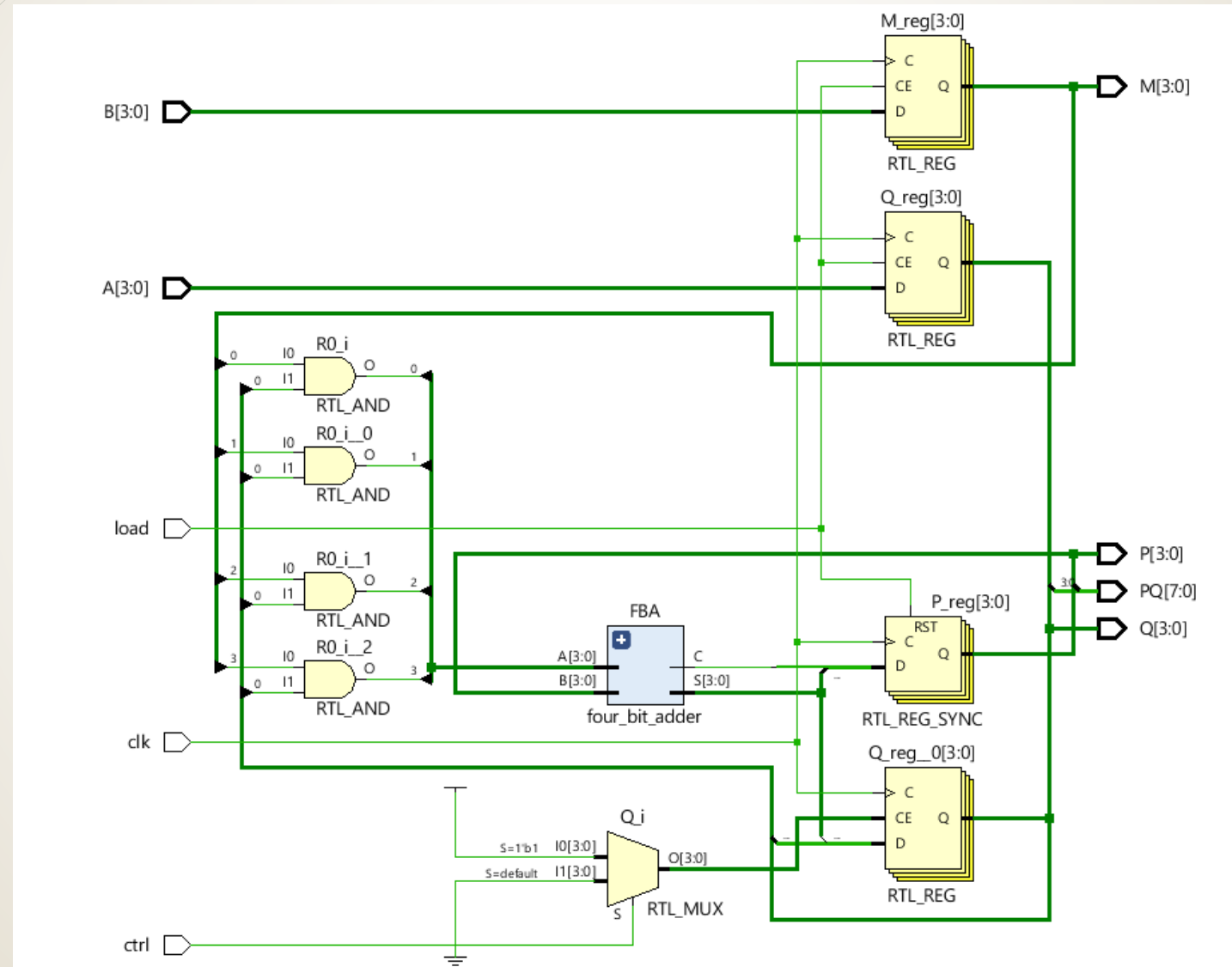
# **DIGITAL DESIGN**

## **LAB 7**

**DIVYAM PATEL**

**B20EE082**

# 1. Implement the Shift and Add Multiplier as a sequential circuit. Use parallel load for input data loading.



SCHEMATIC  
DIAGRAM

# CODE

```
module full_adder(A,B,C,Sum,Carry);
    input A,B,C;
    output Sum,Carry;
    assign Sum = A^B^C;
    assign Carry = (A&B) | (B&C) | (C&A);
endmodule
```

Full Adder

```
module four_bit_adder(A,B,S,C);
    input [3:0] A,B;
    output [3:0] S;
    output C;
    wire [3:1] D;
    full_adder FA1(A[0],B[0],0,S[0],D[1]);
    full_adder FA2(A[1],B[1],D[1],S[1],D[2]);
    full_adder FA3(A[2],B[2],D[2],S[2],D[3]);
    full_adder FA4(A[3],B[3],D[3],S[3],C);
endmodule
```

Four Bit  
Adder

```
module shift_and_add_multiplier(A,B,clk,ctrl,load,P,Q,M,PQ);
    input [3:0] A,B;
    input clk,ctrl,load;
    output reg [3:0] P,Q,M;
    output [7:0] PQ;
    wire [3:0] D1,D2,D3,R,S,U;
    wire V;
```

```
    assign D1[3] = load?A[3]:Q[3];
    assign D1[2] = load?A[2]:Q[2];
    assign D1[1] = load?A[1]:Q[1];
    assign D1[0] = load?A[0]:Q[0];
```

Multiplexer for  
Parallel Loading of  
Data/Input to Shift  
Registers

```
    assign D2[3] = load?B[3]:M[3];
    assign D2[2] = load?B[2]:M[2];
    assign D2[1] = load?B[1]:M[1];
    assign D2[0] = load?B[0]:M[0];
```

```
    assign D3[3] = load?0:V;
    assign D3[2] = load?0:U[3];
    assign D3[1] = load?0:U[2];
    assign D3[0] = load?0:U[1];
```

Multiplexer for  
Parallel Loading of  
Output of four-bit  
adder to Shift  
Register

```
always @(posedge clk)
begin
```

```
    Q[0] = D1[0];
    Q[1] = D1[1];
    Q[2] = D1[2];
    Q[3] = D1[3];
```

Parallel Loading of  
Data/Input to Shift  
Registers

```
    M[0] = D2[0];
    M[1] = D2[1];
    M[2] = D2[2];
    M[3] = D2[3];
```

```
end
```

```
    assign R[3] = M[3]&Q[0];
    assign R[2] = M[2]&Q[0];
    assign R[1] = M[1]&Q[0];
    assign R[0] = M[0]&Q[0];
```

AND operation of LSB of  
one number with all bits  
of other number

```
four_bit_adder FBA(R,P,U,V);
```

```
always @(posedge clk)
```

```
begin
```

```
    if(ctrl)
```

```
    begin
```

```
        Q[0] = Q[1];
```

```
        Q[1] = Q[2];
```

```
        Q[2] = Q[3];
```

```
        Q[3] = U[0];
```

```
    end
```

```
    P[0] = D3[0];
```

```
    P[1] = D3[1];
```

```
    P[2] = D3[2];
```

```
    P[3] = D3[3];
```

```
end
```

```
assign PQ[0] = Q[0];
```

```
assign PQ[1] = Q[1];
```

```
assign PQ[2] = Q[2];
```

```
assign PQ[3] = Q[3];
```

```
assign PQ[4] = P[0];
```

```
assign PQ[5] = P[1];
```


```
assign PQ[6] = P[2];
```

```
assign PQ[7] = P[3];
```

```
endmodule
```

Shifting  
Process

Display  
Final  
Output

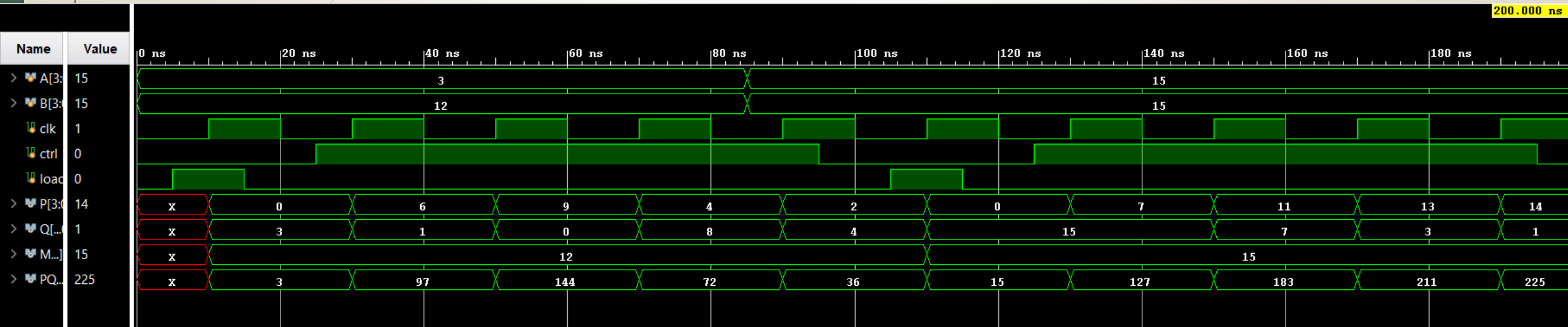


```
module test_shift_and_add_multiplier();
    reg [3:0] A,B;
    reg clk,ctrl,load;
    wire [3:0] P,Q,M;
    wire [7:0] PQ;
    shift_and_add_multiplier SAM(A,B,clk,ctrl,load,P,Q,M,PQ);
```

```
    initial
    fork
        A=4'd3;B=4'd12;clk=0;ctrl=0;load=0;
        #5 load=1;
        #10 clk=1; #15 load=0;
        #20 clk=0; #25 ctrl=1;
        #30 clk=1;
        #40 clk=0;
        #50 clk=1;
        #60 clk=0;
        #70 clk=1;
        #80 clk=0; #85 A=4'd15; #85 B=4'd15;
        #90 clk=1; #95 ctrl=0;
        #100 clk=0; #105 load=1;
        #110 clk=1; #115 load=0;
        #120 clk=0; #125 ctrl=1;
        #130 clk=1;
        #140 clk=0;
        #150 clk=1;
        #160 clk=0;
        #170 clk=1;
        #180 clk=0;
        #190 clk=1; #195 ctrl=0;
    join
    initial #200 $finish;
endmodule
```

# TESTBENCH

# SIMULATION



Load used to load the input to the Shift Registers.

As seen the input changes at 85 ns, but gets loaded to the S.R.s when load is high.

Start is used to begin the multiplication process.

The process requires 5 cycles of clock to complete (1 to load and 4 to do multiplication)



2. Write a program to implement an electronic voting machine.

a) There are two participants.

b) There is an Enable button (available with the administrator) to allow voting. The vote is counted only after the Admin allows it.

d) There should be a provision to cast the vote in a specified time after the admin has allowed it. Else, vote is not counted. Notify this to the admin.

e) Admin should have the controls of "**VOTING IN PROCESS**" and "**DISPLAY RESULTS**".

f) On **DISPLAY RESULT** : The number of votes casted for each candidate should be visible on a 7 segment Display.

```

module electronic_voting_machine(admin, C1, C2, reset, led1, led2, led, invalid, X1, Y1, Z1, X2, Y2, Z2);
    input admin, C1, C2, reset;
    output reg led1 = 0, led2 = 0, led = 0, invalid = 0;
    reg ctrl = 0;
    reg [7:0] CA = 0, CB = 0;
    output [3:0] X1, Y1, Z1, X2, Y2, Z2;
    always @(posedge admin)
    begin
        ctrl = 1;
        invalid = 0;

    end

    always @(posedge C1 or posedge C2)
    begin
        if(C1 & !C2 & ctrl)
        begin
            led1 = 1;
            ctrl = 0;
            led = 1;
            invalid = 0;
            CA = CA+1;
        end

        else if(C2 & !C1 & ctrl)
        begin
            led2 = 1;
            ctrl = 0;
            led = 1;
            invalid = 0;
            CB = CB+1;
        end

        else if((C1 & C2 & ctrl) | (!ctrl))
            invalid = 1;
    end
end

```

Admin  
Control

Counting and displaying  
the vote on pressing the  
respective button

# CODE

Displaying the  
count in BCD form  
XYZ

Reset the LEDs,  
control operation  
and invalid to 0

```

        assign Z1 = CA%(4'd10);
        assign Y1 = (CA%7'd100)/(4'd10);
        assign X1 = CA/(7'd100);
        assign Z2 = CB%(4'd10);
        assign Y2 = (CB%7'd100)/(4'd10);
        assign X2 = CB/(7'd100);

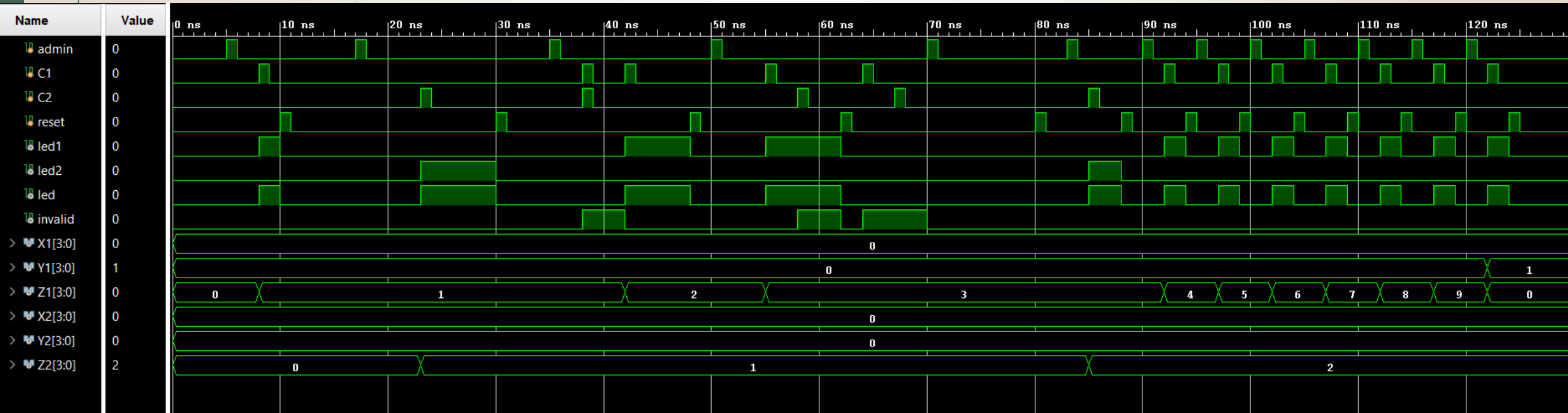
    always @(posedge reset)
    begin
        led1 = 0;
        led2 = 0;
        led = 0;
        invalid = 0;
        ctrl = 0;
    end
endmodule

```



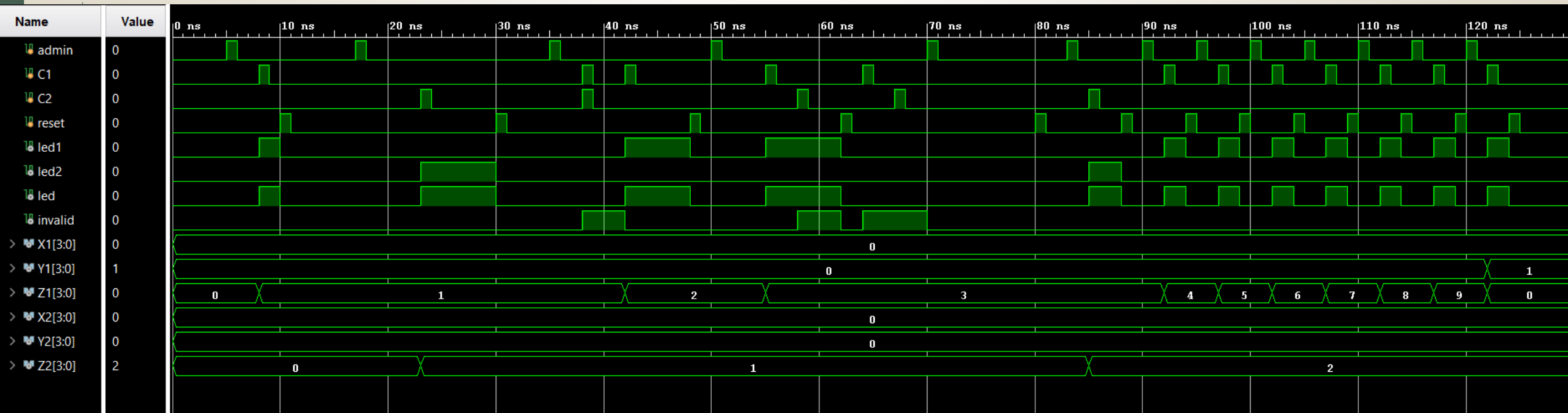


# SIMULATION



- Here, **admin** acts as the control after which the vote can be casted
- If someone tries to vote without the **admin control**, the vote will not be counted  
**Invalid LED** will glow and will also be notified to the **admin** [Around 65 ns]
- If someone tries to press both the buttons at same time then also the **invalid LED** will glow. [35-40 ns]

# SIMULATION



- If someone tries to cast the vote for 2<sup>nd</sup> time, his vote will not be counted and the admin will be notified by invalid LED. [55-60 ns]
- After giving the vote the LED of the corresponding candidate will glow and the admin will be notified with the admin LED.
- **reset** is used to turn off the LEDs.
- If one does not vote for long time the **admin LED** will not glow giving signal to **admin** that the voter is not giving the vote. [70-80 ns]



**THANK YOU**