Digital Design Lab Report

Name	Dev Goel
Roll No.	B20CS090
LAB Number	**VII**

Objective of LAB-7:

(i) Implement the Shift and Add Multiplier as a sequential circuit. Use parallel load for input data loading.

(ii) 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.
- **c.** 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.

EXPERIMENT (i): Implement the Shift and Add Multiplier as a sequential circuit. Use parallel load for input data loading.

- **Shift and Add Multiplier (Serial Multiplier):**

```
`timescale 1ns / 1ps
module shift_and_add_multiplier(inpA, inpB, clk, ctrl, ld, P, Q, M, PQ);
input [3:0] inpA, inpB;
input clk, ctrl, ld;
```

```
output reg [3:0] P, Q, M;
output [7:0] PQ;
wire V;
wire [3:0] D1, D2, D3, R, S, U;

assign D1[0] = ld?inpA[0]:Q[0];
assign D1[1] = ld?inpA[1]:Q[1];
assign D1[2] = ld?inpA[2]:Q[2];
assign D1[3] = ld?inpA[3]:Q[3];

assign D2[0]= ld?inpB[0]:M[0];
assign D2[1]= ld?inpB[1]:M[1];
assign D2[2]= ld?inpB[2]:M[2];
assign D2[3]= ld?inpB[3]:M[3];

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

always @(posedge clk)

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

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

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

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

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
```

Test Bench:

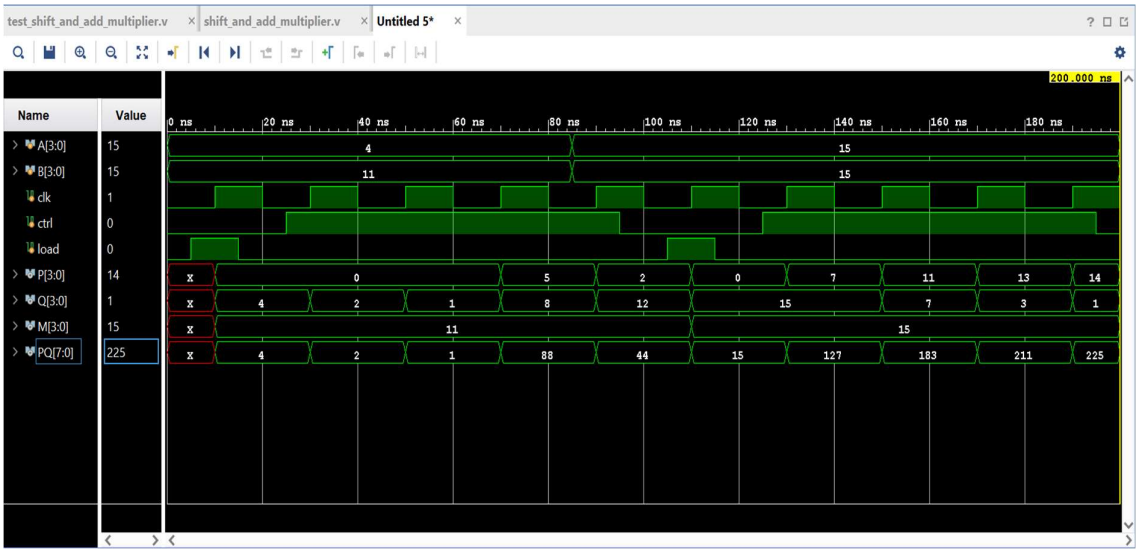
```
module test_shift_and_add_multiplier();
reg [3:0] A, B;
reg clk, ctrl, load;
```

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

always #10 clk=~clk;

initial
fork
   A=4'd4; B=4'd11; clk=0; ctrl=0; load=0;
   #5 load=1;
   #15 load=0;
   #25 ctrl=1;
   #85 A=4'd15; #85 B=4'd15;
   #95 ctrl=0;
   #105 load=1;
   #115 load=0;
   #125 ctrl=1;
   #195 ctrl=0;
join
initial #200 $finish;
endmodule
```

Waveform:



EXPERIMENT (ii): Write a program to implement an electronic voting machine.

- **Electronic Voting Machine:**

```
`timescale 1ns / 1ps
module electronic_voting_machine(admin, participant1, participant2, reset,
led1, led2, led, invalid, X1, Y1, Z1, X2, Y2, Z2);

input admin, participant1, participant2, 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 participant1 or posedge participant2)

begin

   //Participant 2 Votes First and Ctrl is high
   if(participant2 & !participant1 & ctrl)
   begin
      led2 = 1;
      ctrl = 0;
      led = 1;
      invalid = 0;
      CB = CB+1;
   end
```

```
    // Participant 1 Votes First and Ctrl is high
    else if(participant1 & !participant2 & ctrl)
    begin
        led1 = 1;
        ctrl = 0;
        led = 1;
        invalid = 0;
        CA = CA+1;
    end

    // Invalid State
    else if((participant1 & participant2 & ctrl) | (!ctrl))
        invalid = 1;
end


// For BCD Display
assign Z1 = CA%(4'd10);
assign Z2 = CB%(4'd10);

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

assign X1 = CA/(7'd100);
assign X2 = CB/(7'd100);

// Condition is reset is high
always @(posedge reset)
begin
    led1 = 0;
    led2 = 0;
    led = 0;
    invalid = 0;
    ctrl = 0;
end

endmodule
```

Test Bench:

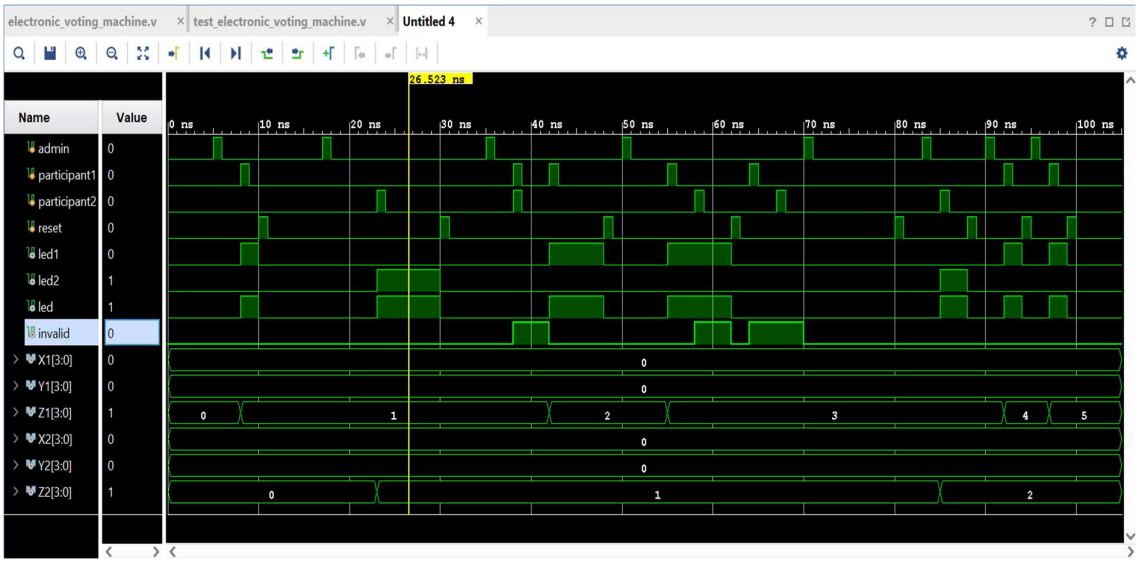
```
`timescale 1ns / 1ps
module test_electronic_voting_machine();

reg admin, participant1, participant2, reset;
wire led1, led2, led, invalid;
wire [3:0] X1, Y1, Z1, X2, Y2, Z2;
electronic_voting_machine EVM(admin, participant1, participant2, reset, led1,
led2, led, invalid, X1, Y1, Z1, X2, Y2, Z2);

initial
begin
   admin=0; participant1=0; participant2=0; reset=0; #5 admin=1; #1 admin=0;
   #2 participant1=1; #1 participant1=0;
   #1 reset=1; #1 reset=0;
   #6 admin=1; #1 admin=0; #5 participant2=1; #1 participant2=0;
   #6 reset=1; #1 reset=0;
   #4 admin=1; #1 admin=0;
   #2 participant1=1; participant2=1; #1 participant1=0; participant2=0;
   #3 participant1=1; #1 participant1=0;
   #5 reset=1; #1 reset=0;
   #1 admin=1; #1 admin=0; #4 participant1=1; #1 participant1=0;
   #2 participant2=1; #1 participant2=0;
   #3 reset=1; #1 reset=0;
   #1 participant1=1; #1 participant1=0; #2 participant2=1; #1 participant2=0;
   #2 admin=1; #1 admin=0; #9 reset=1; #1 reset=0;
   #2 admin=1; #1 admin=0; #1 participant2=1; #1 participant2=0;
   #2 reset=1; #1 reset=0;
   #1 admin=1; #1 admin=0; #1 participant1=1; #1 participant1=0; #1 reset=1;
#1 reset=0;
   admin=1; #1 admin=0; #1 participant1=1; #1 participant1=0; #1 reset=1; #1
reset=0;

end
initial #105 $finish;
endmodule
```

Waveform:


electronic_voting_machine.v
test_electronic_voting_machine.v
Untitled 4
26.523 ns
Name
Value
admin
participant1
participant2
reset
led1
led2
led
invalid
X1[3:0]
Y1[3:0]
Z1[3:0]
X2[3:0]
Y2[3:0]
Z2[3:0]
0 ns
10 ns
20 ns
30 ns
40 ns
50 ns
60 ns
70 ns
80 ns
90 ns
100 ns