

```

\timescale 1ns / 1ps
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
// Company:
// Engineer:
//
// Create Date: 07.02.2026 11:02:16
// Design Name:
// Module Name: modulecode
// Project Name:
// Target Devices:
// Tool Versions:
// Description:
//
// Dependencies:
//
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
//
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////

module buttoncontrol(
input clock,
input reset,
input button,
output reg valid_vote
);

reg [31:0] counter; // take a register to store the number of
valid votes polled

always @(posedge clock)
begin
if(reset)
counter <= 0;

```

```

else
begin
    if(button && counter < 11)
        counter <= counter + 1;
    else if( !button)
        counter <= 0;
end
end

```

```

always @(posedge clock)
begin
    if(reset)
valid_vote <= 1'b0;
    else
    begin
        if(counter == 10)
            valid_vote <= 1'b1;
        else
            valid_vote <= 1'b0;
        end
    end
endmodule

```

```

module modeControl(
input clock,
input reset,
input mode,
input valid_vote_casted,
input [7:0] candidate1_vote,
input [7:0] candidate2_vote,
input [7:0] candidate3_vote,
input [7:0] candidate4_vote,
input candidate1_button_press,
input candidate2_button_press,
input candidate3_button_press,
input candidate4_button_press,
output reg [7:0] leds

```

```
);
```

```
reg [31:0] counter;
```

```
always @(posedge clock)
```

```
begin
```

```
if(reset)
```

```
counter <= 0;          // whenever reset is pressed counter  
started from 0
```

```
else if(valid_vote_casted) // if a valid votes is casted counter  
becomes 1
```

```
counter <= counter +1;
```

```
else if( counter !=0 && counter < 10) // if counter is not 0 then  
increment it till 10
```

```
counter <= counter + 1;
```

```
else // once counter becomes 10 reset it to 0
```

```
counter <= 0;
```

```
end
```

```
always @(posedge clock)
```

```
begin
```

```
    if(reset)
```

```
        leds <=0;
```

```
    else
```

```
    begin
```

```
if(mode == 0 && counter >0 )
```

```
    leds <= 8'hFF;
```

```
else if( mode == 0)
```

```
    leds <= 8'h00;
```

```
else if (mode == 1) begin
```

```
    leds <= 8'h00;
```

```
    if (candidate1_button_press)
```

```
        leds <= candidate1_vote;
```

```
    else if (candidate2_button_press)
```

```
        leds <= candidate2_vote;
```

```
    else if (candidate3_button_press)
```

```
        leds <= candidate3_vote;
```

```

        else if (candidate4_button_press)
            leds <= candidate4_vote;
        end

    end
end

endmodule

```

```

module voteLogger(
input clock,
input reset,
input mode,
input cand1_vote_valid,
input cand2_vote_valid,
input cand3_vote_valid,
input cand4_vote_valid,
output reg [7:0] cand1_vote_recvd,
output reg [7:0] cand2_vote_recvd,
output reg [7:0] cand3_vote_recvd,
output reg [7:0] cand4_vote_recvd);

```

```

always @(posedge clock)
begin
    if(reset)
    begin
        cand1_vote_recvd <= 0;
        cand2_vote_recvd <= 0;
        cand3_vote_recvd <= 0;
        cand4_vote_recvd <= 0;
    end
    else
    begin
        if(cand1_vote_valid && mode == 0)
            cand1_vote_recvd <= cand1_vote_recvd + 1;
        else if(cand2_vote_valid && mode == 0)

```

```

cand2_vote_recvd <= cand2_vote_recvd + 1;
else if(cand3_vote_valid && mode == 0)
cand3_vote_recvd <= cand3_vote_recvd + 1;
else if(cand4_vote_valid && mode == 0)
cand4_vote_recvd <= cand4_vote_recvd + 1;
end
end
end

```

```

endmodule

```

```

module votingMachine(
input clock,
input reset,
input mode,
input button1,
input button2,
input button3,
input button4,
output [7:0] led
);

wire valid_vote_1;
wire valid_vote_2;
wire valid_vote_3;
wire valid_vote_4;
wire [7:0] cand1_vote_recvd;
wire [7:0] cand2_vote_recvd;
wire [7:0] cand3_vote_recvd;
wire [7:0] cand4_vote_recvd;
wire anyValidVote;

assign anyValidVote =
valid_vote_1|valid_vote_2|valid_vote_3|valid_vote_4;

buttoncontrol bc1(
.clock(clock),

```

```
.reset(reset),  
.button(button1),  
.valid_vote(valid_vote_1)  
);
```

```
buttoncontrol bc2(  
.clock(clock),  
.reset(reset),  
.button(button2),  
.valid_vote(valid_vote_2)  
);
```

```
buttoncontrol bc3(  
.clock(clock),  
.reset(reset),  
.button(button3),  
.valid_vote(valid_vote_3)  
);
```

```
buttoncontrol bc4(  
.clock(clock),  
.reset(reset),  
.button(button4),  
.valid_vote(valid_vote_4)  
);
```

```
voteLogger VL(  
.clock(clock),  
.reset(reset),  
.mode(mode),  
.cand1_vote_valid(valid_vote_1),  
.cand2_vote_valid(valid_vote_2),  
.cand3_vote_valid(valid_vote_3),  
.cand4_vote_valid(valid_vote_4),  
.cand1_vote_recvd(cand1_vote_recvd),  
.cand2_vote_recvd(cand2_vote_recvd),  
.cand3_vote_recvd(cand3_vote_recvd),
```

```

.cand4_vote_recvd(cand4_vote_recvd)
);

modeControl MC(
.clock(clock),
.reset(reset),
.mode(mode),
.valid_vote_casted(anyValidVote),
.candidate1_vote(cand1_vote_recvd),
.candidate2_vote(cand2_vote_recvd),
.candidate3_vote(cand3_vote_recvd),
.candidate4_vote(cand4_vote_recvd),
.candidate1_button_press(valid_vote_1),
.candidate2_button_press(valid_vote_2),
.candidate3_button_press(valid_vote_3),
.candidate4_button_press(valid_vote_4),
.leds(led)
);

endmodule

```