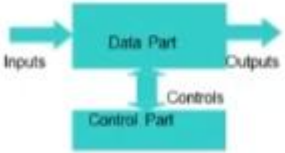
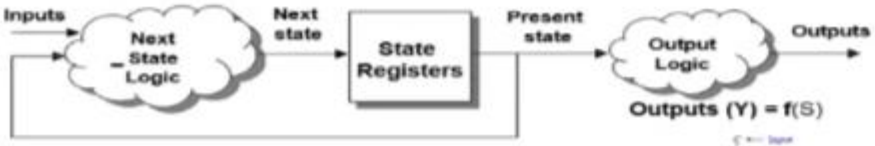
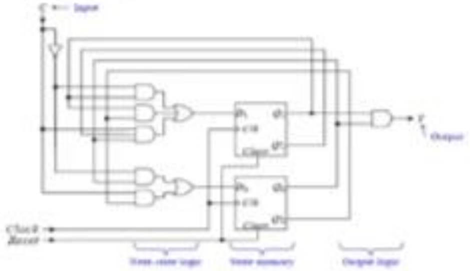
Need of FSM

- Any digital design consists of two parts:
  - Data part :
    - Responsible for the processing of data.
    - The processing is done through some blocks such as (full adder, digital filter, decoder,…)
  - Control part
    - Describes how and when these blocks will communicate with each other.
  - Control part is generally described using a FSM(Finite State Machine).



Moore state Machine





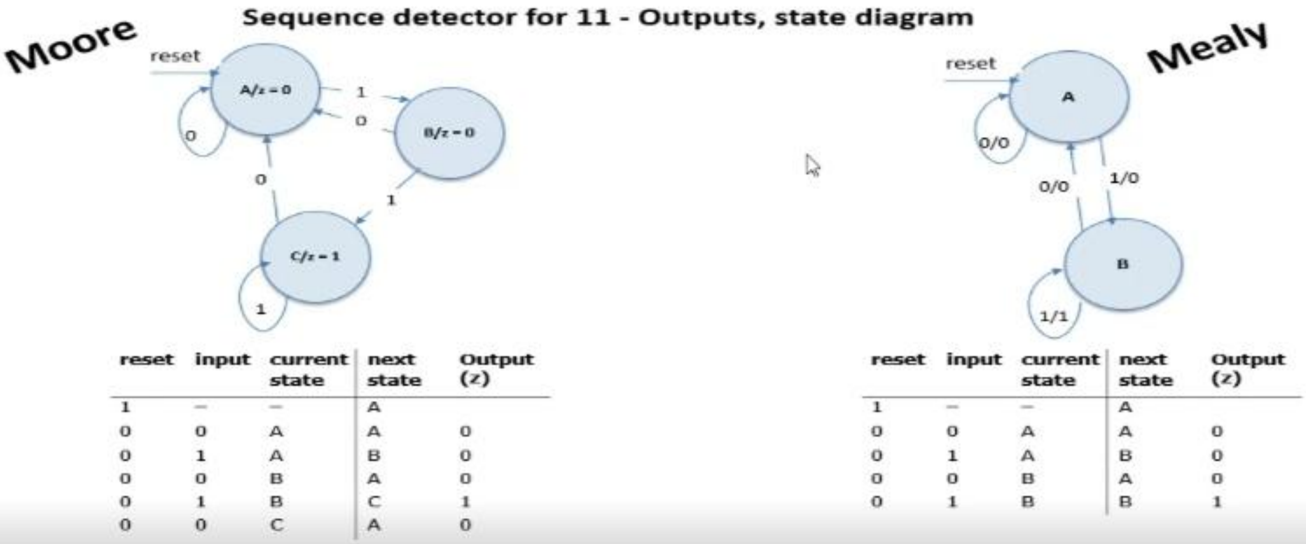
Features:

- State machine outputs are dependent only on the present state
- Output vector (Y) is function of the state vector (S)
- Outputs don't react immediately to input change.

Advantage:

- Moore machines effectively filter out transients.
- It can be used to eliminate race conditions when inputs are unfiltered.

Sequence-11

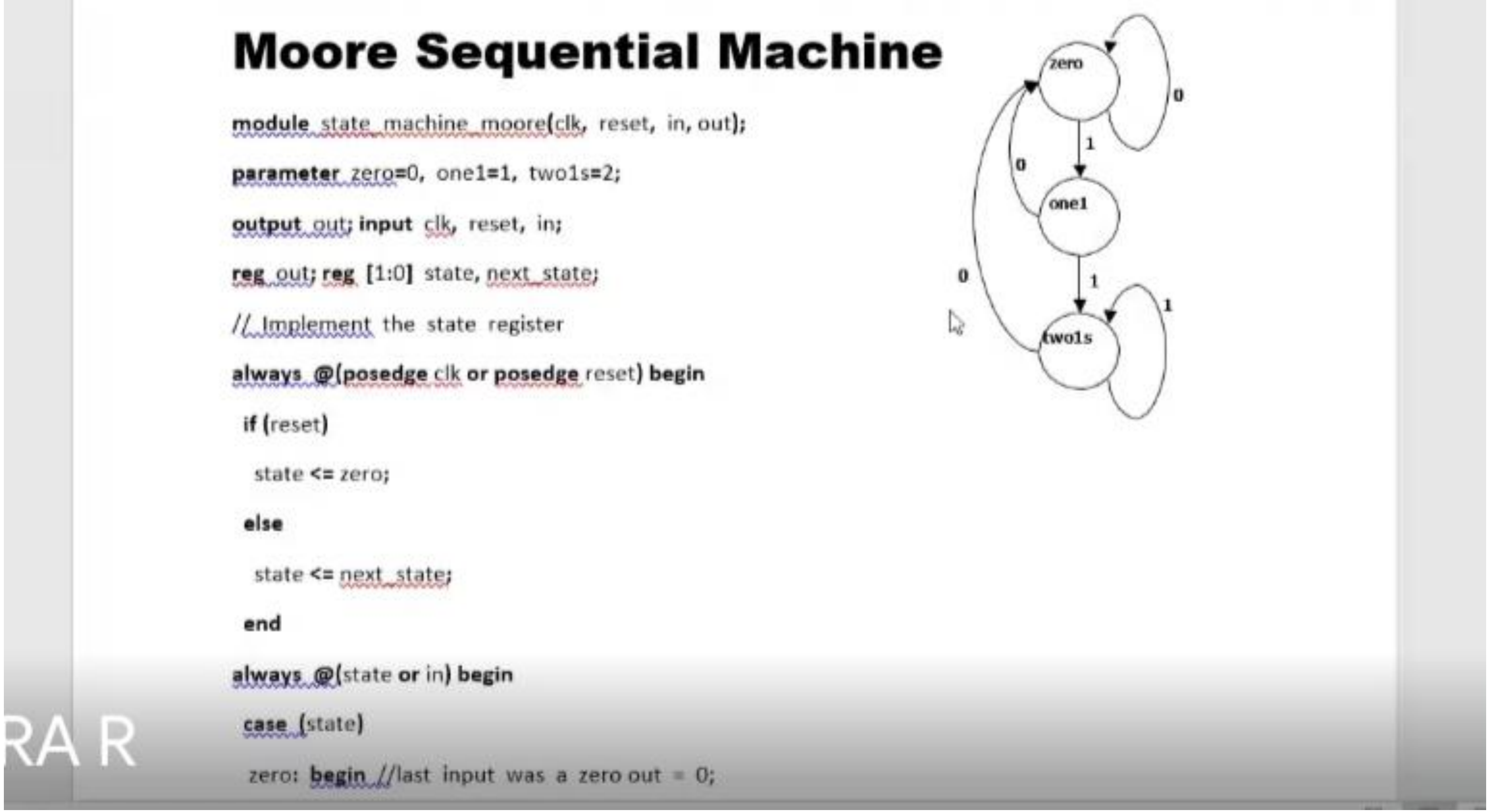


Sequence detector for 11 - Outputs, state diagram

reset	input	current state	next state	Output (z)
1	–	–	A	
0	0	A	A	0
0	1	A	B	0
0	0	B	A	0
0	1	B	C	1
0	0	C	A	0

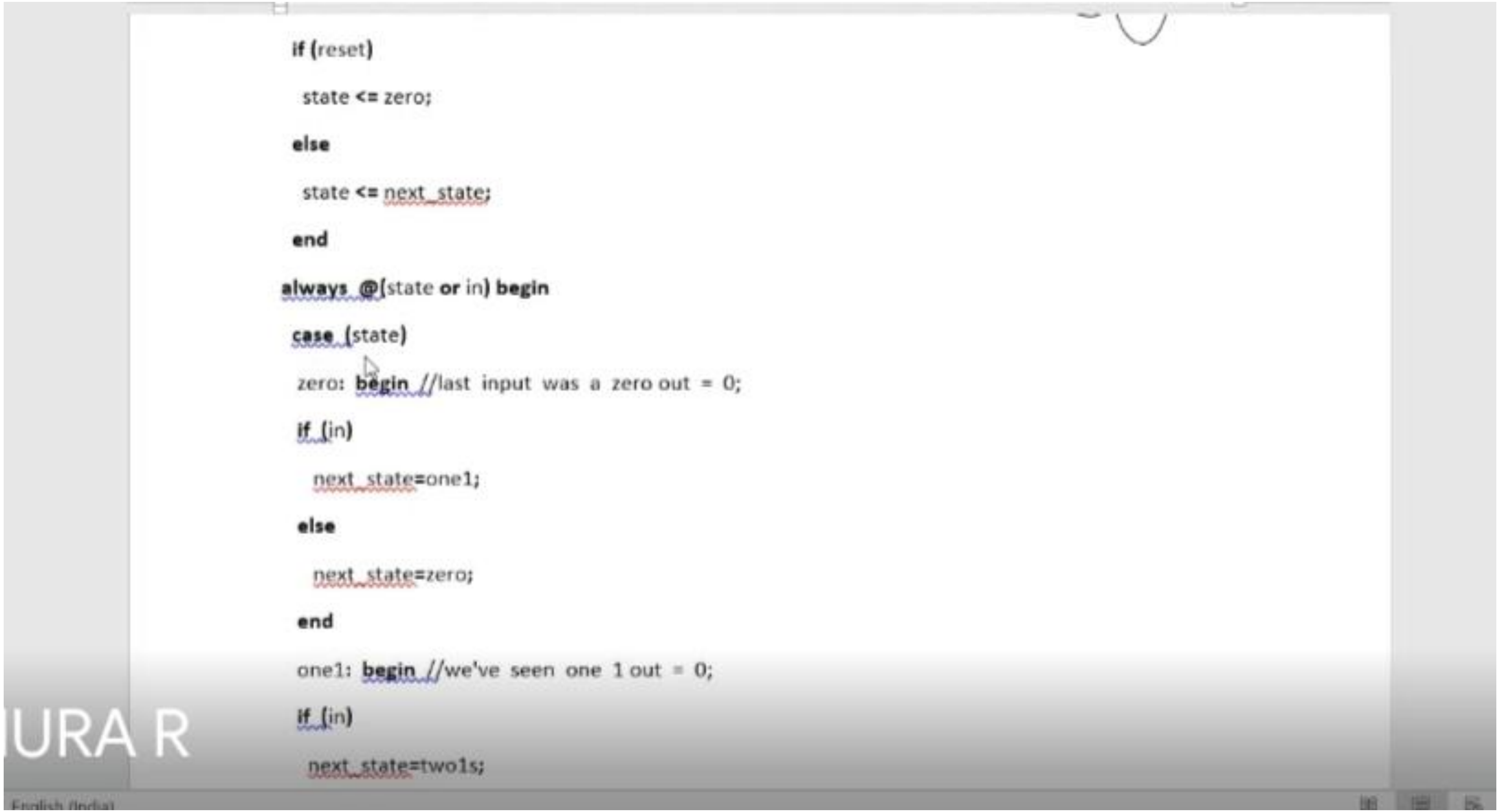
reset	input	current state	next state	Output (z)
1	–	–	A	
0	0	A	A	0
0	1	A	B	0
0	0	B	A	0
0	1	B	B	1

MADHURA R

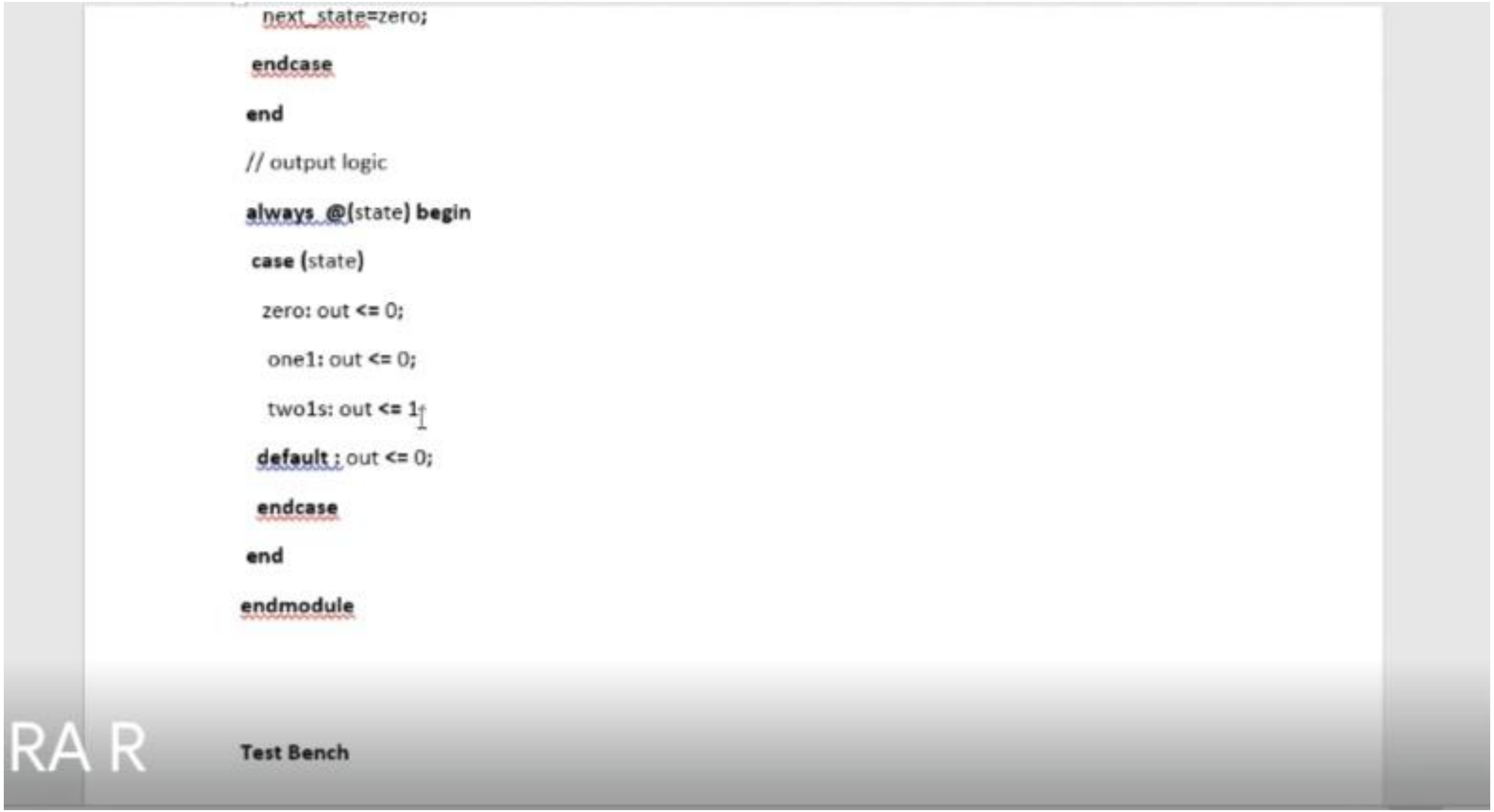


Moore Sequential Machine

```
module state_machine_moore(clk, reset, in, out);

parameter zero=0, one1=1, two1s=2;

output out; input clk, reset, in;

reg out; reg [1:0] state, next_state;

// Implement the state register

always @(posedge clk or posedge reset) begin

 if (reset)

  state <= zero;

 else

  state <= next_state;

 end

always @(state or in) begin

 case (state)

  zero: begin //last input was a zero out = 0;
```

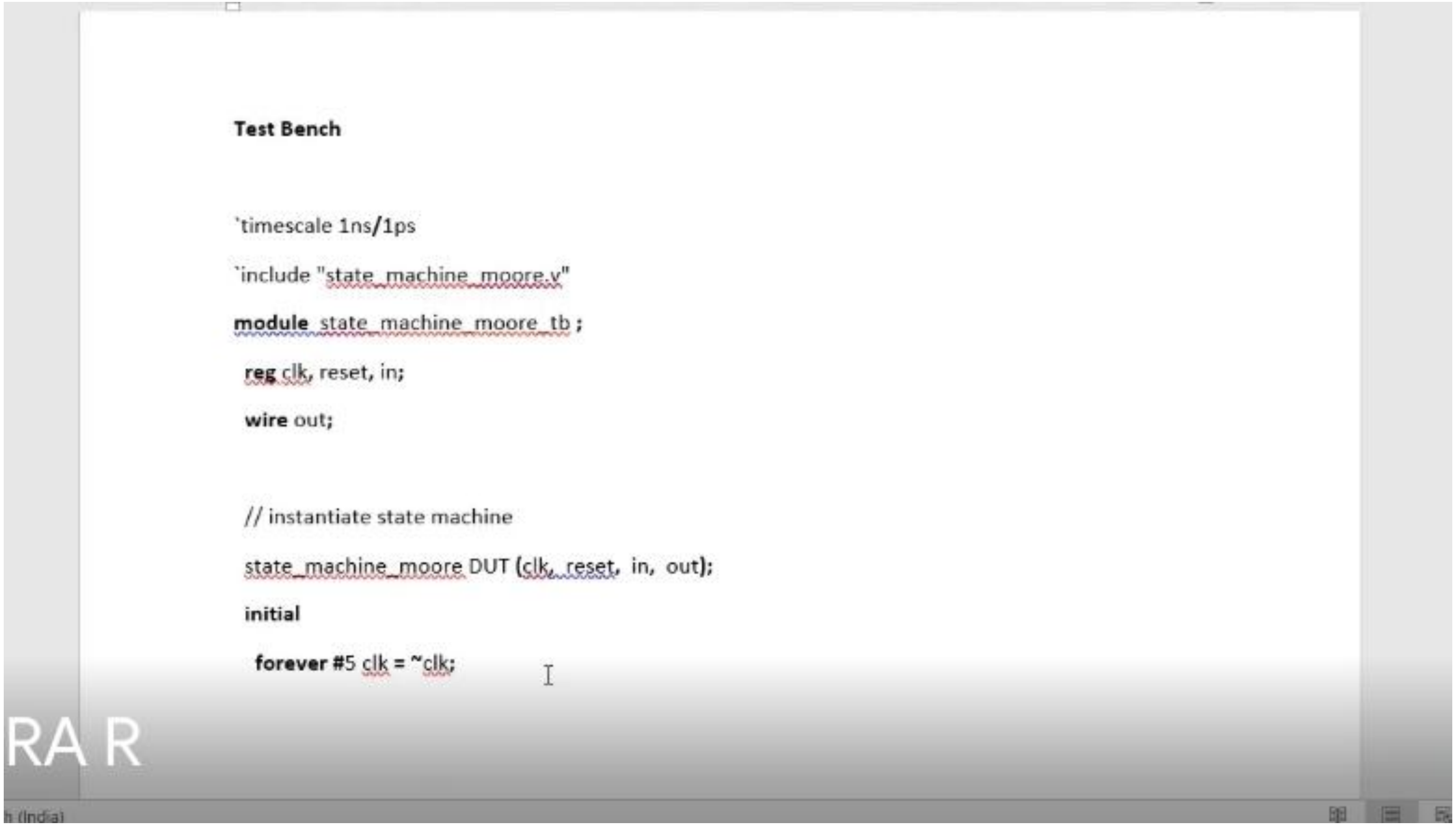


```
 if (reset)
  state <= zero;
 else
  state <= next_state;
 end
always @(state or in) begin
 case (state)
  zero: begin //last input was a zero out = 0;
  if (in)
   next_state=one1;
  else
   next_state=zero;
  end
  one1: begin //we've seen one 1 out = 0;
  if (in)
   next_state=two1s;
```



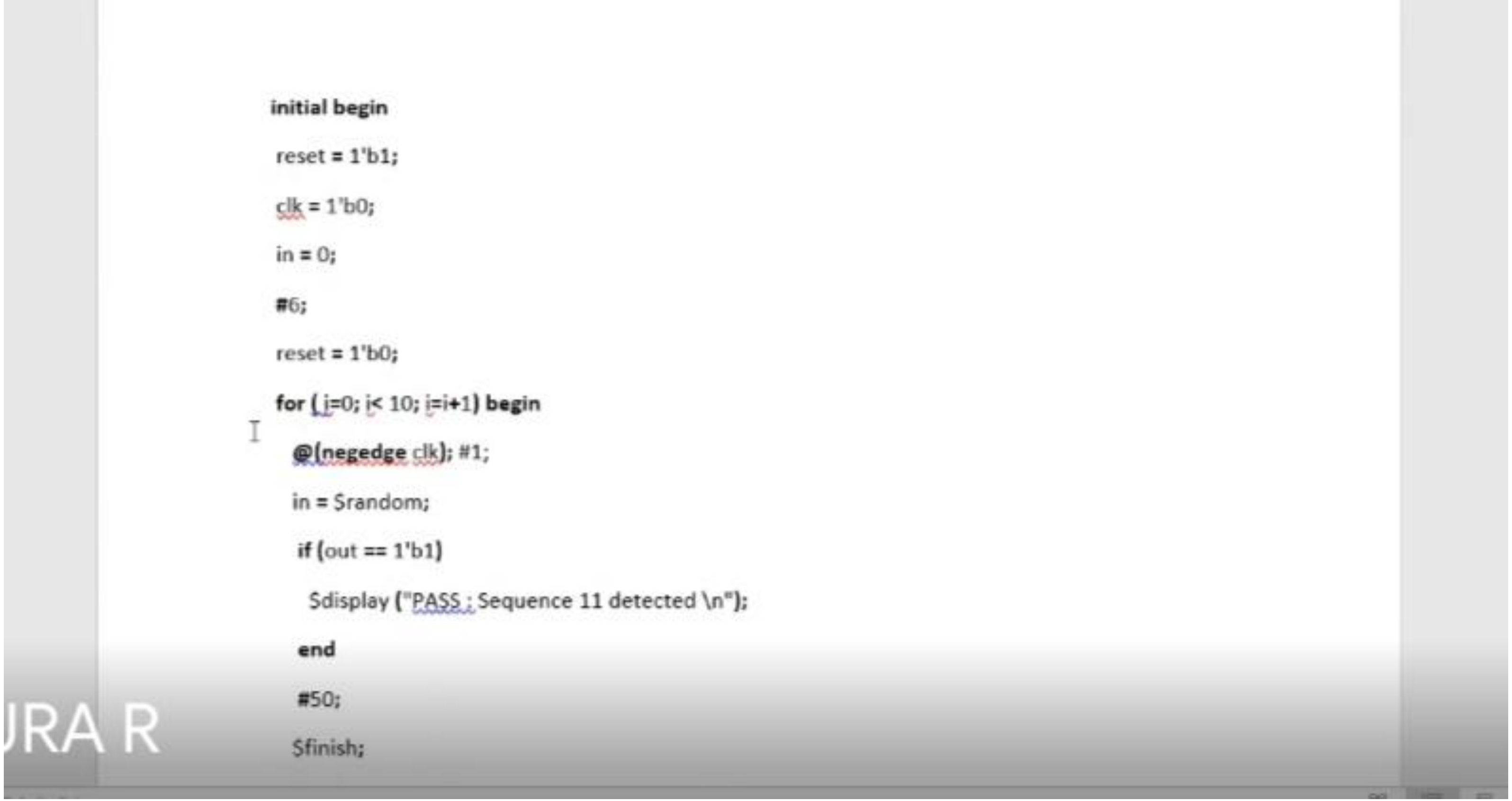
```
   next_state=zero;
  endcase
 end
 // output logic
 always  @(state) begin
  case (state)
   zero: out <= 0;
    one1: out <= 0;
    two1s: out <= 1
   default : out <= 0;
   endcase
 end
endmodule
```

Test Bench

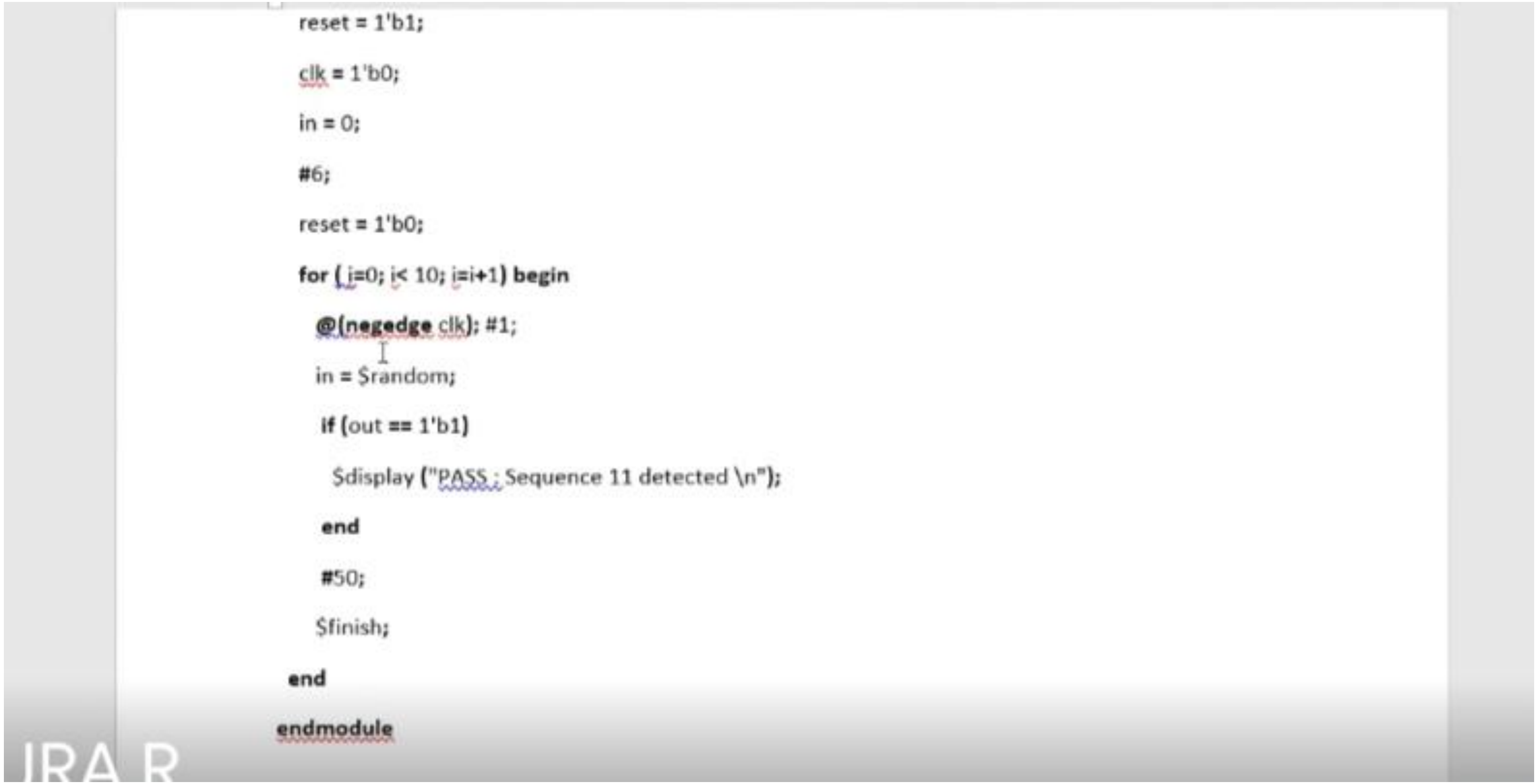


Test Bench

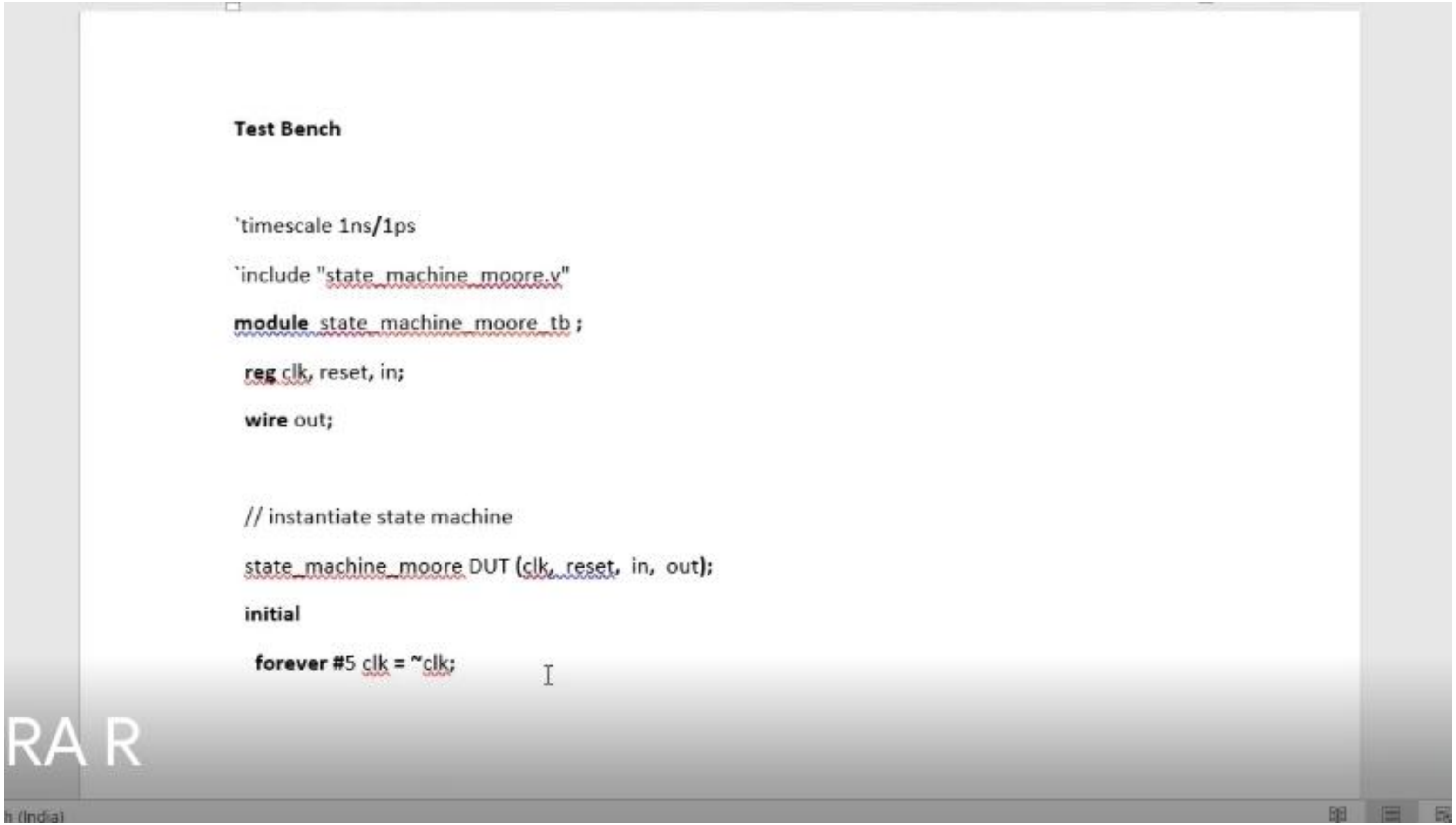
```
`timescale 1ns/1ps
`include "state_machine_moore.v"
module state_machine_moore_tb ;
 reg clk, reset, in;
 wire out;

 // instantiate state machine
 state_machine_moore DUT (clk, reset, in, out);
 initial
  forever #5 clk = ~clk;
```



```
initial begin
 reset = 1'b1;
 clk = 1'b0;
 in = 0;
 #6;
 reset = 1'b0;
 for ( i=0; i< 10; i=i+1) begin
   @(negedge clk); #1;
   in = $random;
    if (out == 1'b1)
      $display ("PASS : Sequence 11 detected \n");
    end
    #50;
   $finish;
```

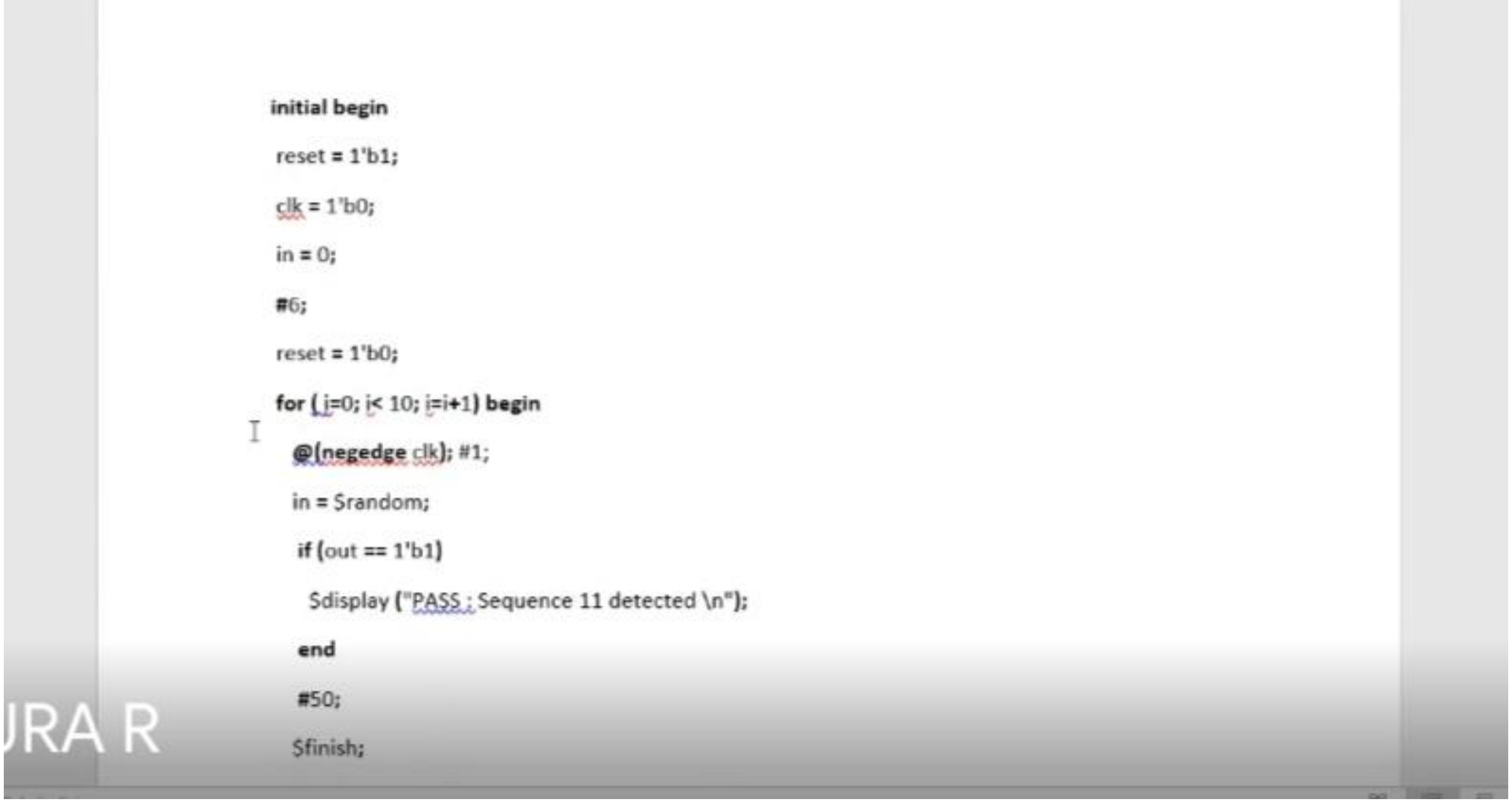


```
        reset = 1'b1;

        clk = 1'b0;

        in = 0;

        #6;

        reset = 1'b0;

        for ( i=0; i< 10; i=i+1) begin

          @(negedge clk); #1;

          in = $random;

           if (out == 1'b1)

            $display ("PASS : Sequence 11 detected \n");

           end

           #50;

          $finish;

      end

    endmodule
```

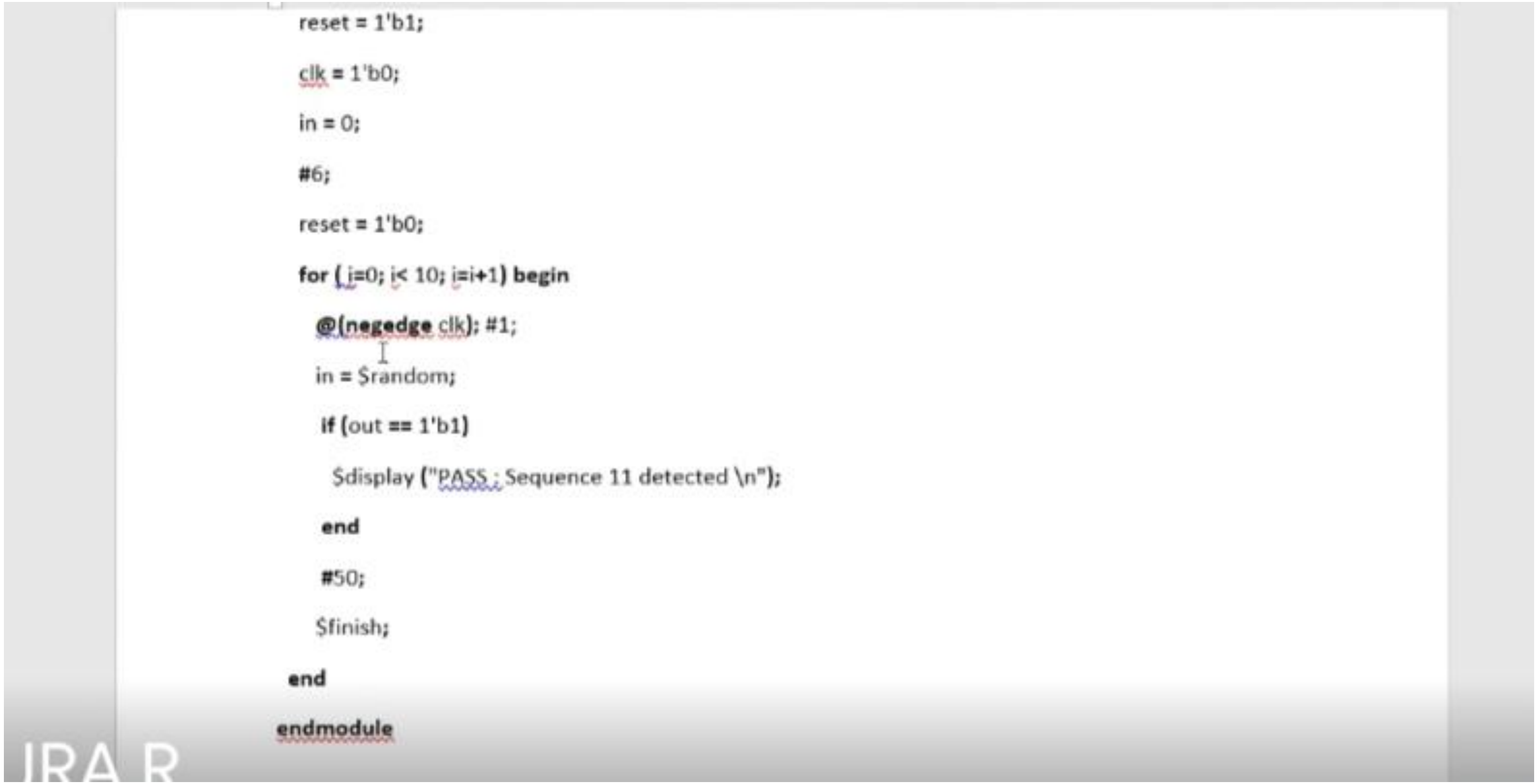


Test Bench

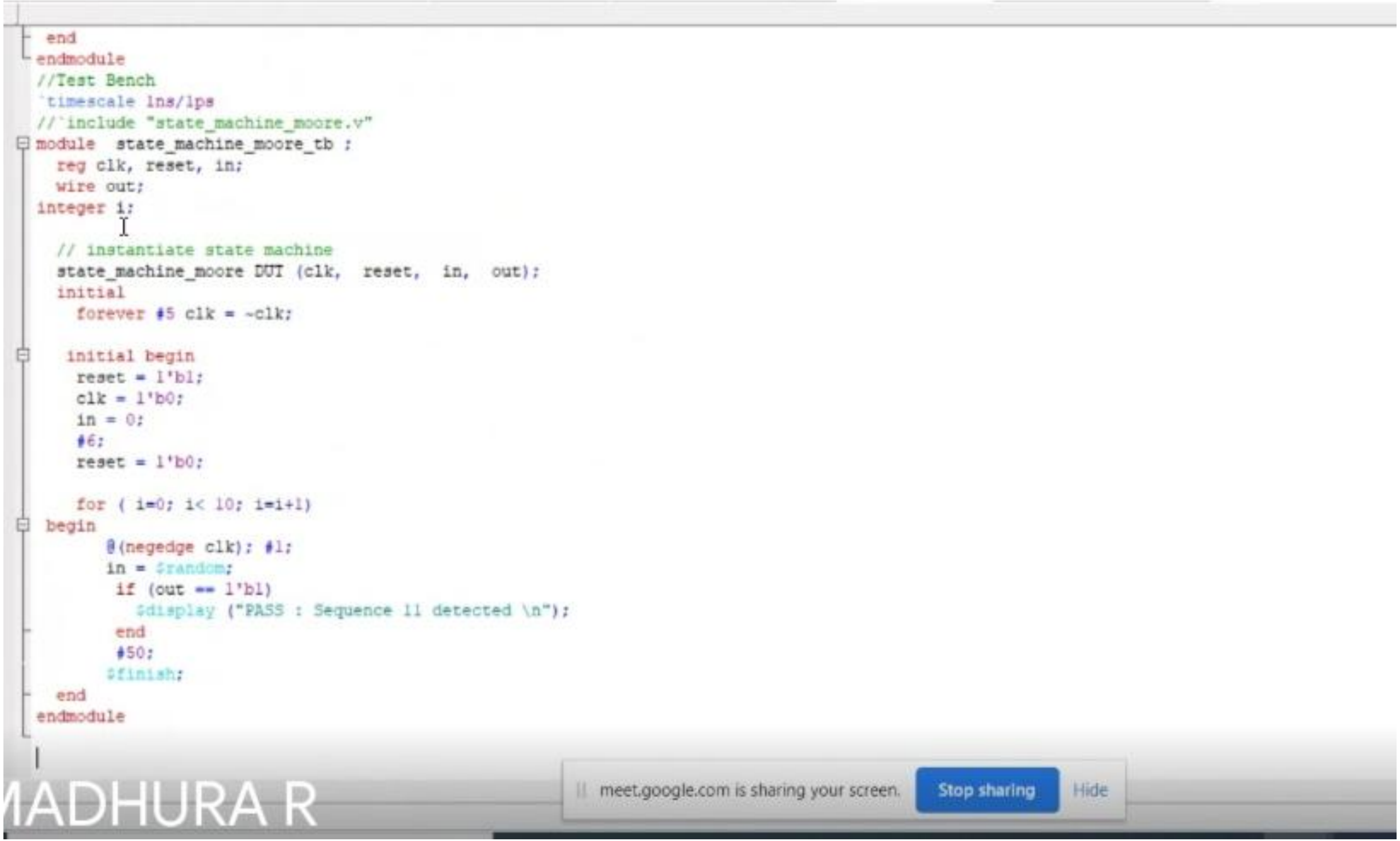
```
`timescale 1ns/1ps
`include "state_machine_moore.v"
module  state_machine_moore_tb ;
  reg clk, reset, in;
  wire out;

  // instantiate state machine
  state_machine_moore DUT (clk,  reset,  in,  out);
  initial
    forever #5 clk = ~clk;
```



```
initial begin
 reset = 1'b1;
 clk = 1'b0;
 in = 0;
 #6;
 reset = 1'b0;
 for ( i=0; i< 10; i=i+1) begin
   @(negedge clk); #1;
   in = $random;
    if (out == 1'b1)
      $display ("PASS : Sequence 11 detected \n");
    end
    #50;
   $finish;
```

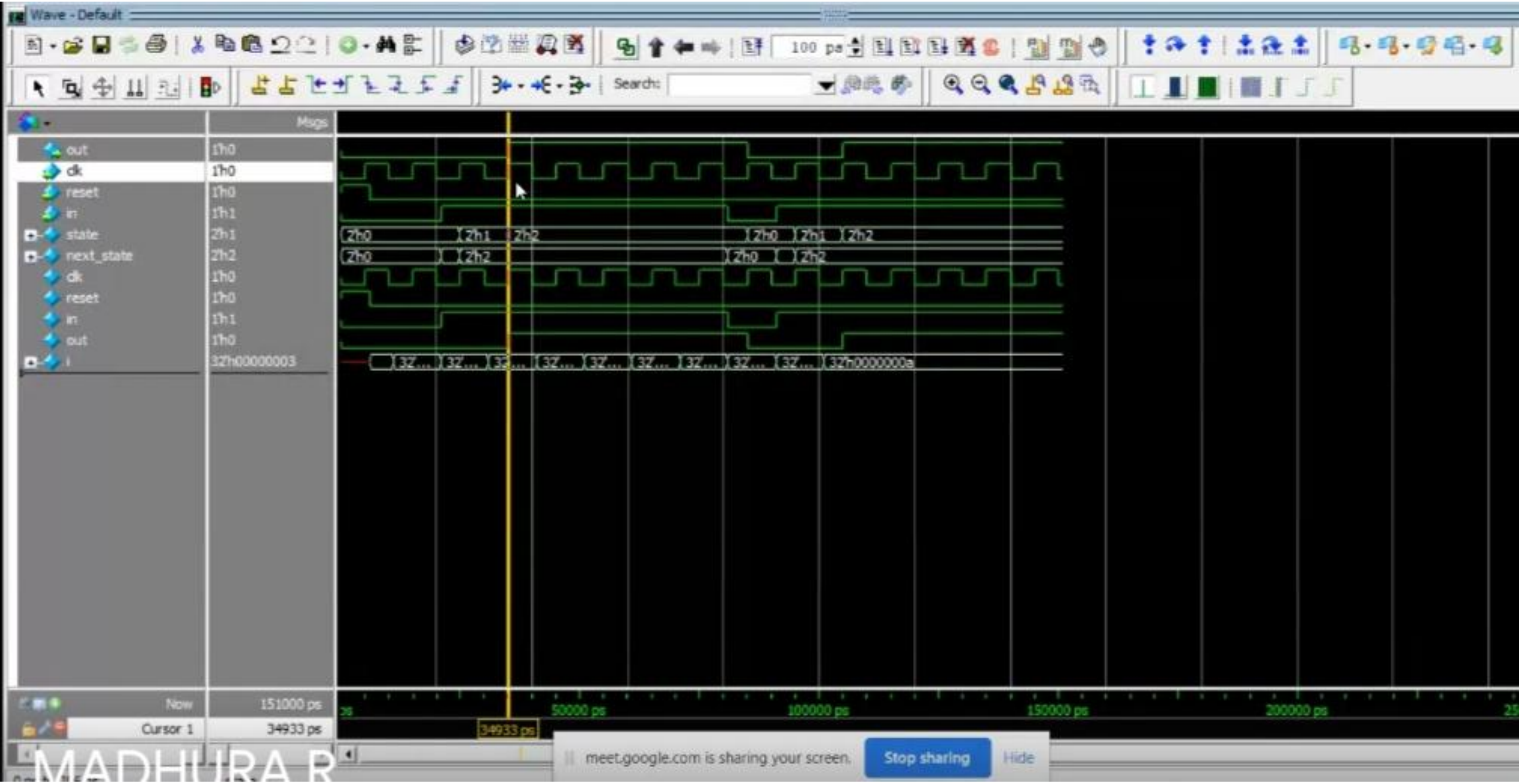


```
    reset = 1'b1;

    clk = 1'b0;

    in = 0;

    #6;

    reset = 1'b0;

    for ( i=0; i< 10; i=i+1) begin

      @(negedge clk); #1;

      in = $random;

       if (out == 1'b1)

         $display ("PASS : Sequence 11 detected \n");

       end

       #50;

      $finish;

  end

endmodule
```



```
 end
endmodule
//Test Bench
`timescale 1ns/1ps
//`include "state_machine_moore.v"
module  state_machine_moore_tb ;
  reg clk, reset, in;
  wire out;
integer i;

  // instantiate state machine
  state_machine_moore DUT (clk,  reset,  in,  out);
  initial
    forever #5 clk = ~clk;

   initial begin
    reset = 1'b1;
    clk = 1'b0;
    in = 0;
    #6;
    reset = 1'b0;

    for ( i=0; i< 10; i=i+1)
 begin
       @(negedge clk); #1;
       in = $random;
        if (out == 1'b1)
          $display ("PASS : Sequence 11 detected \n");
        end
        #50;
       $finish;
  end
endmodule
```

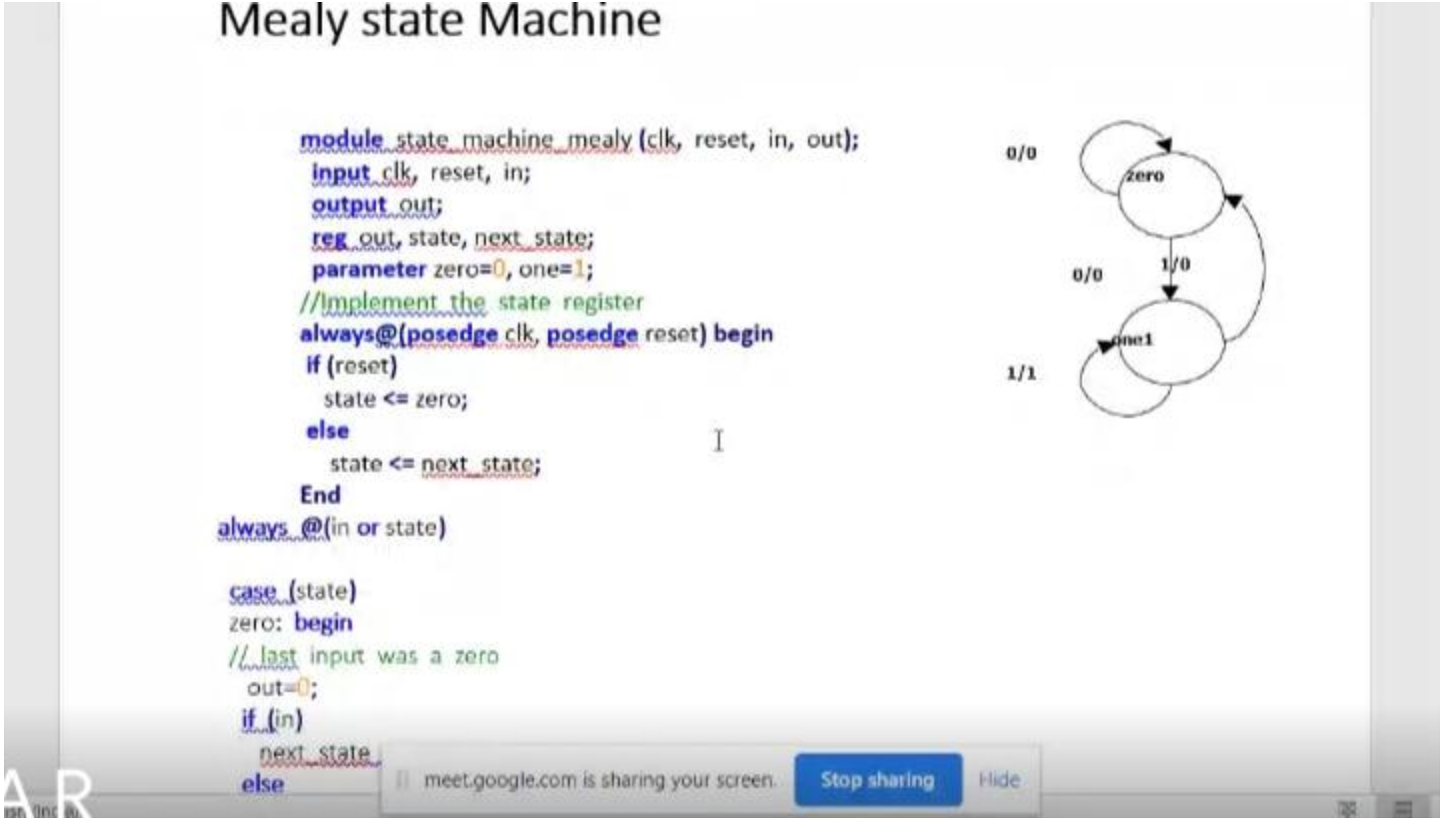
meet.google.com is sharing your screen.



	Msgs
out	1'h0
clk	1'h0
reset	1'h0
in	1'h1
state	2'h1
next_state	2'h2
clk	1'h0
reset	1'h0
in	1'h1
out	1'h0
i	32'h00000003

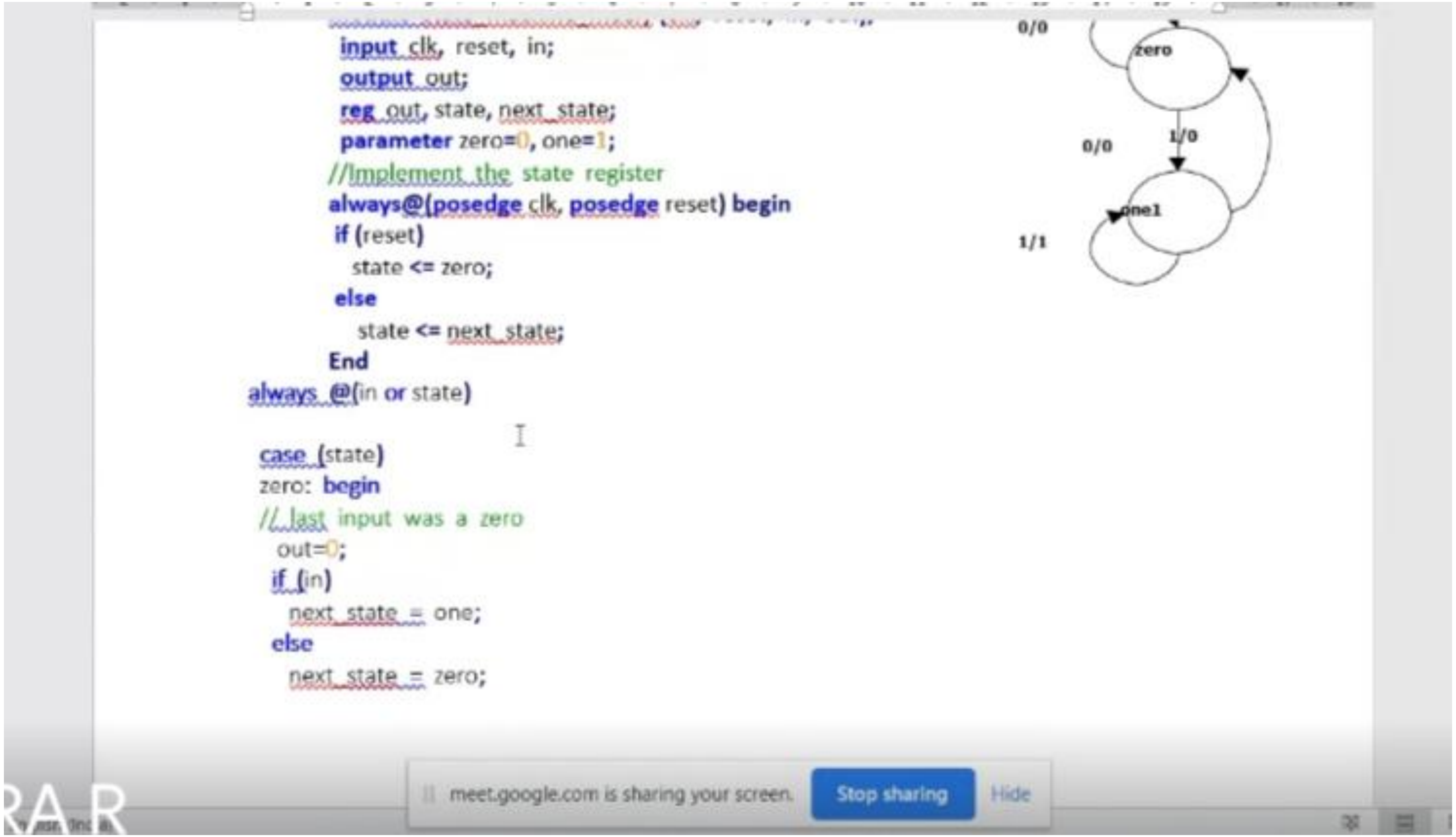
Now	151000 ps
Cursor 1	34933 ps

meet.google.com is sharing your screen.

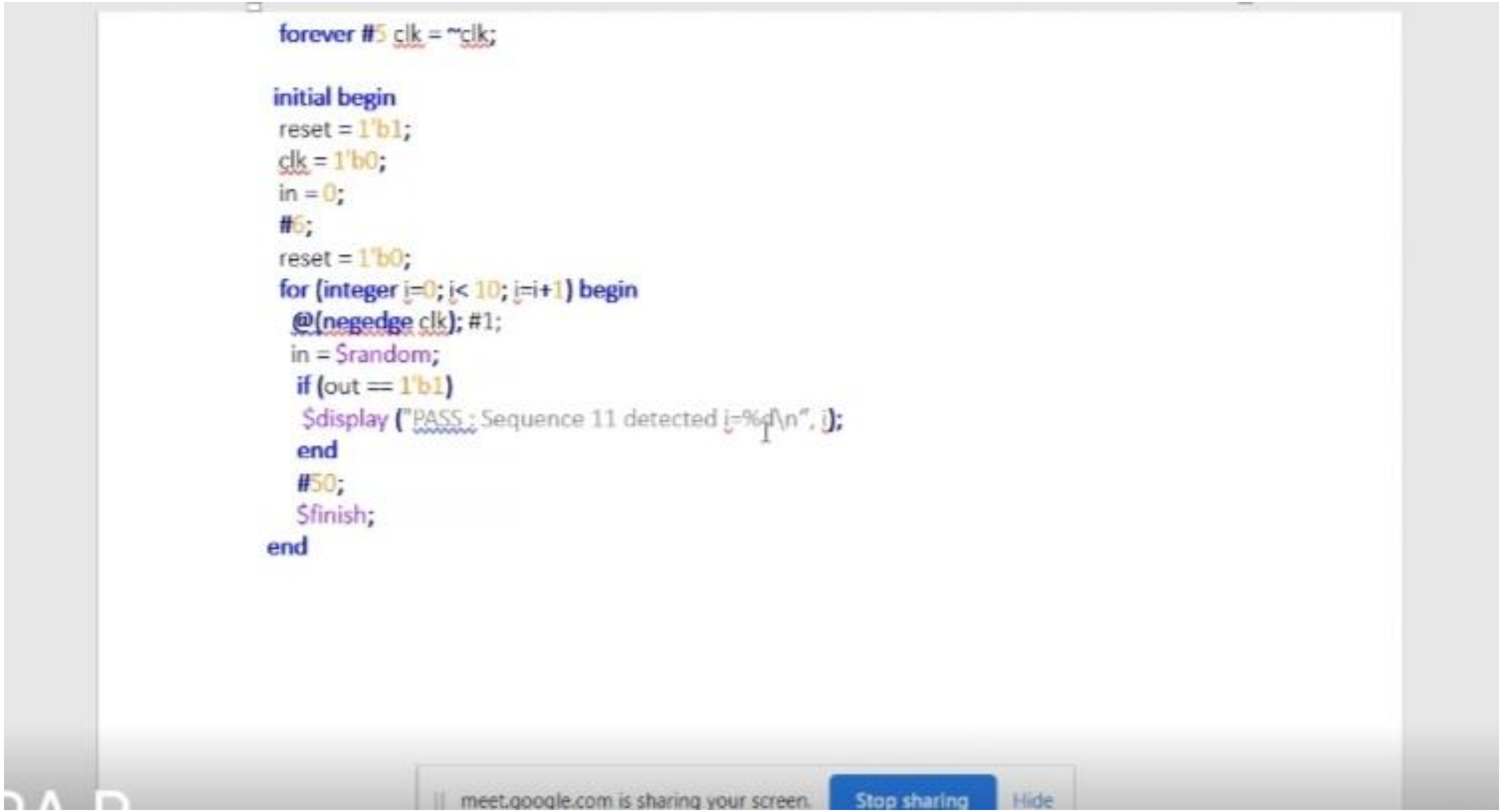


Mealy state Machine

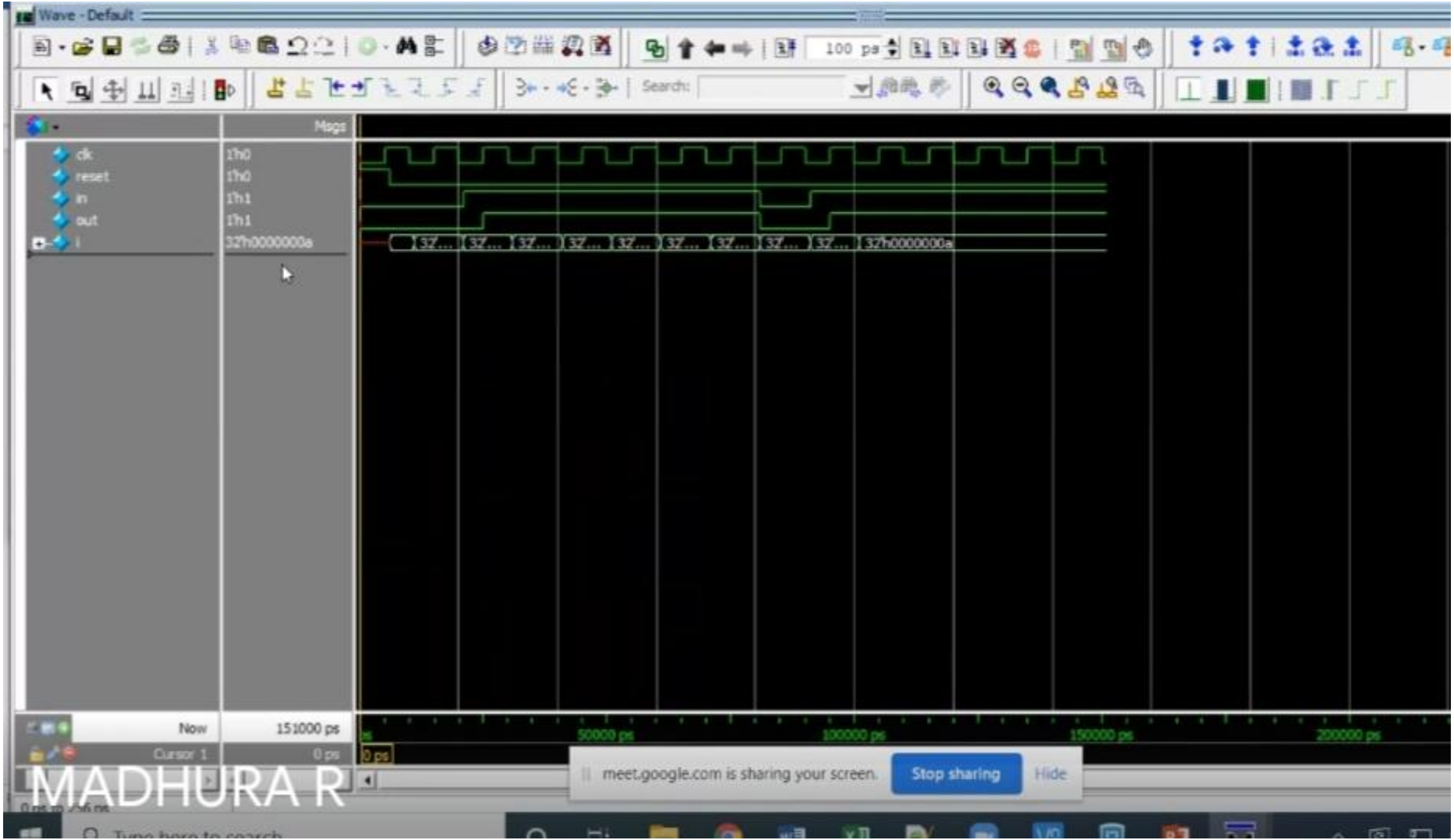
```
module state_machine_mealy (clk, reset, in, out);
 input clk, reset, in;
 output out;
 reg out, state, next_state;
 parameter zero=0, one=1;
//Implement the state register
always@(posedge clk, posedge reset) begin
 if (reset)
   state <= zero;
 else
   state <= next_state;
End
always @(in or state)

case (state)
zero: begin
// last input was a zero
  out=0;
 if (in)
   next_state
 else
```



```
         input clk, reset, in;
         output out;
         reg out, state, next_state;
         parameter zero=0, one=1;
        //Implement the state register
        always@(posedge clk, posedge reset) begin
         if (reset)
           state <= zero;
         else
            state <= next_state;
        End
always @(in or state)

 case (state)
 zero:  begin
 // last input was a zero
   out=0;
  if (in)
    next_state = one;
  else
    next_state = zero;
```



```
 forever #5 clk = ~clk;

initial begin
 reset = 1'b1;
 clk = 1'b0;
 in = 0;
 #6;
 reset = 1'b0;
 for (integer i=0; i< 10; i=i+1) begin
   @(negedge clk); #1;
  in = $random;
   if (out == 1'b1)
    $display ("PASS : Sequence 11 detected i=%d\n", i);
   end
   #50;
   $finish;
end
```



	Msgs
clk	1'h0
reset	1'h0
in	1'h1
out	1'h1
i	32'h0000000a

32'... 32'... 32'... 32'... 32'... 32'... 32'... 32'... 32'... 32'h0000000a

Now 151000 ps

Cursor 1 0 ps

meet.google.com is sharing your screen.

MADHURA R