19ECE349 - RISC PROCESSOR DESIGN USING HDL

Assignment II - Sequential Circuits Modelling

Nigil M.R [CB.EN.U4ECE19136]

Question 1

Write synthesizable behavioural model of an 8-bit shift parallel to serial convertor shift register with a control input mode. The circuit shifts out data through a serial output serial_out. The circuit has the following modes.

mode 00 – No shifting

mode 01 – Shift left

mode 10 – Shift right

mode 11 – External load from an input *data_in*

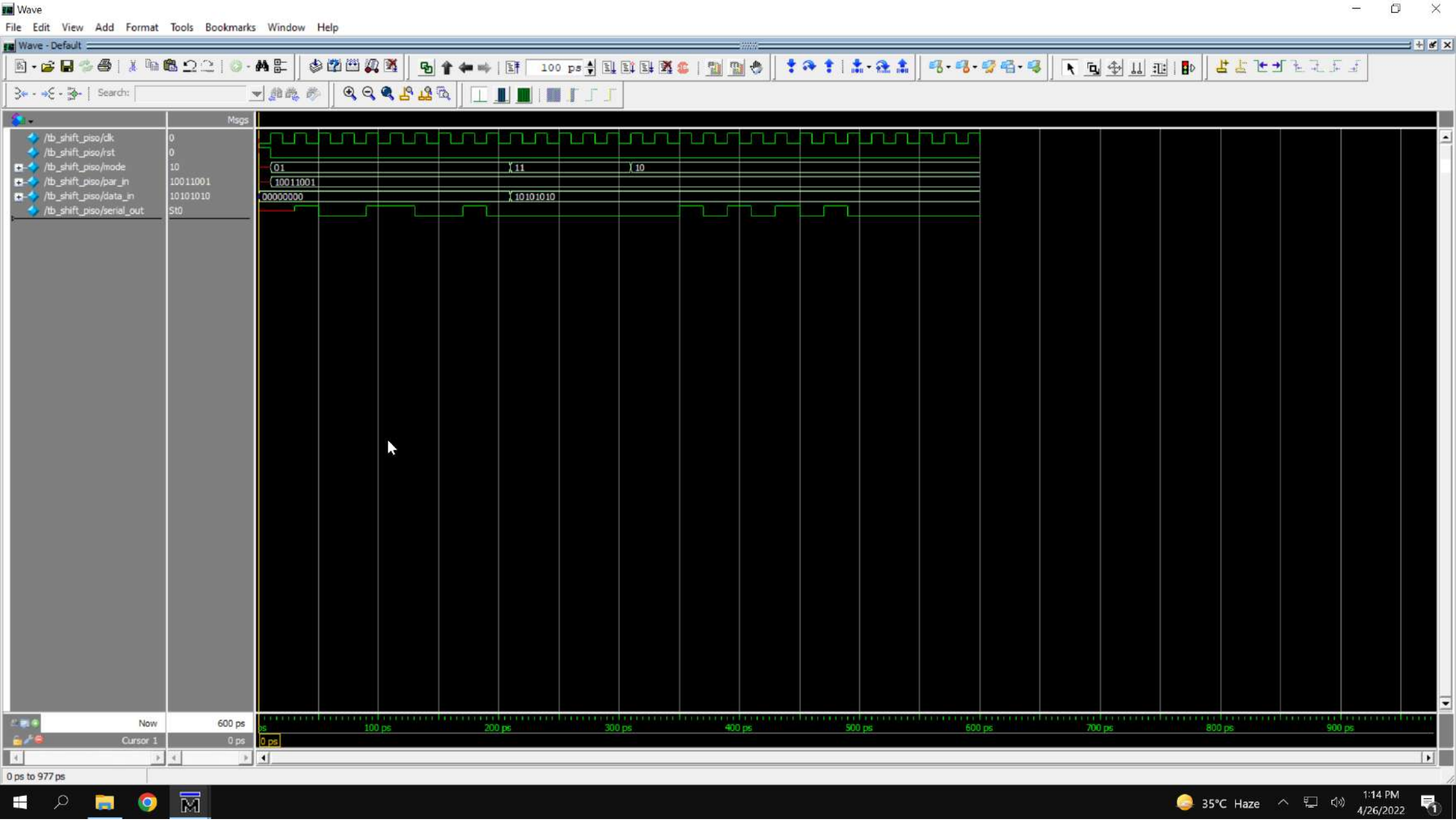
Source Code

```
module shift_piso (par_in, mode, data_in, rst, clk, serial_out);
input [1:0] mode;
input [7:0] data_in, par_in;
input rst, clk;
reg [7:0] sreg;
output reg serial_out;

always @ (posedge clk or posedge rst)
begin
        if (rst == 1'b1)
        begin
                sreg <= 8'd0;
        end
        else if (clk == 1'b1)
        sreg <= par_in;
        begin

                case (mode)
```

```
                2'b00 : sreg <= sreg; // No Shifting

                2'b01 :
                // Left Shifted Serial Outout (Little Endian)
                begin
                serial_out <= sreg[7];
                sreg <= {sreg[6:0], 1'b0};
                end
                // Right Shifted Serial Output (Big Endian)
                2'b10 :
                begin
                serial_out <= sreg[0];
                sreg <= {1'b0,sreg[7:1]};
                end

                2'b11 : sreg <= data_in; // Load

            endcase
        end
end
endmodule

module tb_shift_piso ();
wire serial_out;
reg [7:0] data_in, par_in;
reg [1:0] mode;
reg rst, clk;
```

```
always begin
        #10;
        clk = ~clk;
        end

initial
begin
        rst = 1'b1; clk = 1'b0; data_in = 8'd0; #10;

        rst = 1'b0; clk = 1'b1;

        par_in = 8'b10011001;
        mode <= 01; #200; // Left Shift

        mode <= 11; data_in <= 8'b10101010; #100; // External Load


        par_in = 8'b10011001;
        mode <= 10; #200; // Right Shift on Load

end

shift_piso piso_1 (par_in, mode, data_in, rst, clk, serial_out);

endmodule
```

Output Waveform


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_shift_piso/clk
/tb_shift_piso/rst
/tb_shift_piso/mode
/tb_shift_piso/par_in
/tb_shift_piso/data_in
/tb_shift_piso/serial_out
0
0
10
10011001
10101010
St0
01
11
10
10011001
00000000
10101010
Now
600 ps
Cursor 1
0 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
0 ps to 977 ps
35°C Haze
1:14 PM
4/26/2022

Question 2

Write the synthesizable- behavioral model of a BCD up/down counter which is controlled by a 2-bit mode input. It exhibits the following behavior.

mode 00 – No shifting

mode 01 – Shift left

mode 10 – Shift right

mode 11 – External load from an external BCD input *ext_in*

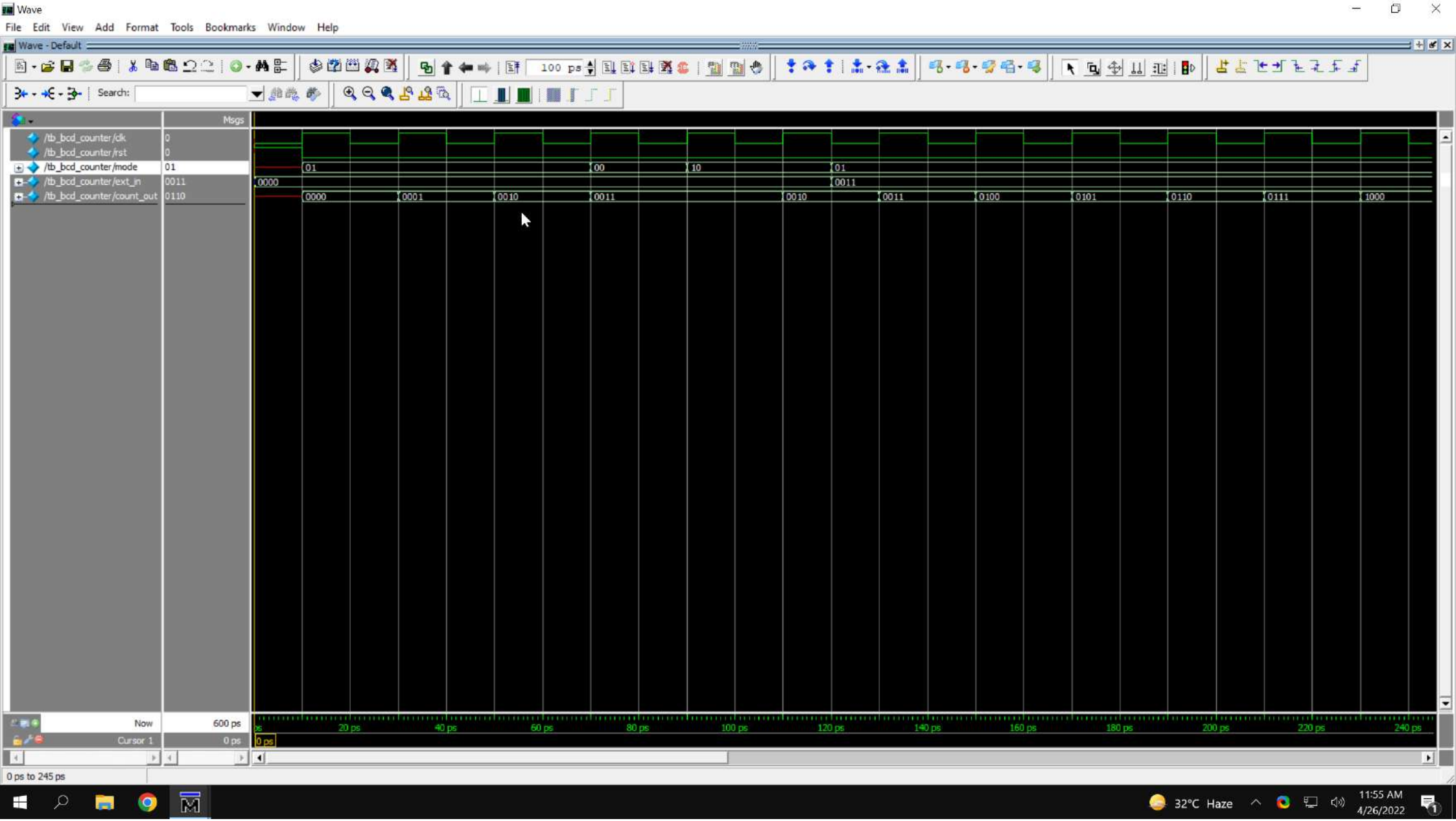
Source Code

```
module bcd_counter (mode, rst, clk, ext_in, count_out);
input [1:0] mode;
input [3:0] ext_in; // BCD Counter
input rst, clk;
reg [3:0] count;
output reg [3:0] count_out;

always @ (posedge clk or posedge rst)
begin
        if (rst == 1'b1)
                count <= 4'd0;
        else if (clk == 1'b1)
        begin

                case (mode)

                        2'b00 : count <= count; // No Change in Count

                        2'b01 :
                        // BCD Up Counter
                        begin
                        if (count < 9)
```

```
                begin
                        count = count + 1'b1;
                end
                else
                        count = 4'b0000;
                end

                2'b10 :
                // BCD Down Counter
                begin
                if (count < 9)
                begin
                        count = count - 1'b1;
                end
                else
                        count = 4'b0000;
                end

                2'b11 : count <= ext_in; // Load

        endcase
        assign count_out = count;
    end
end
endmodule

module tb_bcd_counter ();
wire [3:0] count_out;
reg [3:0] ext_in;
reg [1:0] mode;
```

```
reg rst, clk;



always begin
        #10;
        clk = ~clk;
        end

initial
begin
        rst = 1'b1; clk = 1'b0; ext_in = 8'd0; #10;

        rst = 1'b0; clk = 1'b1;
        mode <= 01; #60;
        mode <= 00; #20;
        mode <= 10; #30;
        mode <= 11; ext_in <= 4'b011;
        mode <= 01; #10;
end

bcd_counter bcc1 (mode, rst, clk, ext_in, count_out);

endmodule
```

Output Waveform


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_bcd_counter/clk
/tb_bcd_counter/rst
/tb_bcd_counter/mode
/tb_bcd_counter/ext_in
/tb_bcd_counter/count_out
0
0
01
0011
0110
01
00
10
01
0000
0011
0000
0001
0010
0011
0010
0011
0100
0101
0110
0111
1000
Now
600 ps
Cursor 1
0 ps
0 ps
20 ps
40 ps
60 ps
80 ps
100 ps
120 ps
140 ps
160 ps
180 ps
200 ps
220 ps
240 ps
0 ps to 245 ps
32°C Haze
11:55 AM
4/26/2022

Question 3

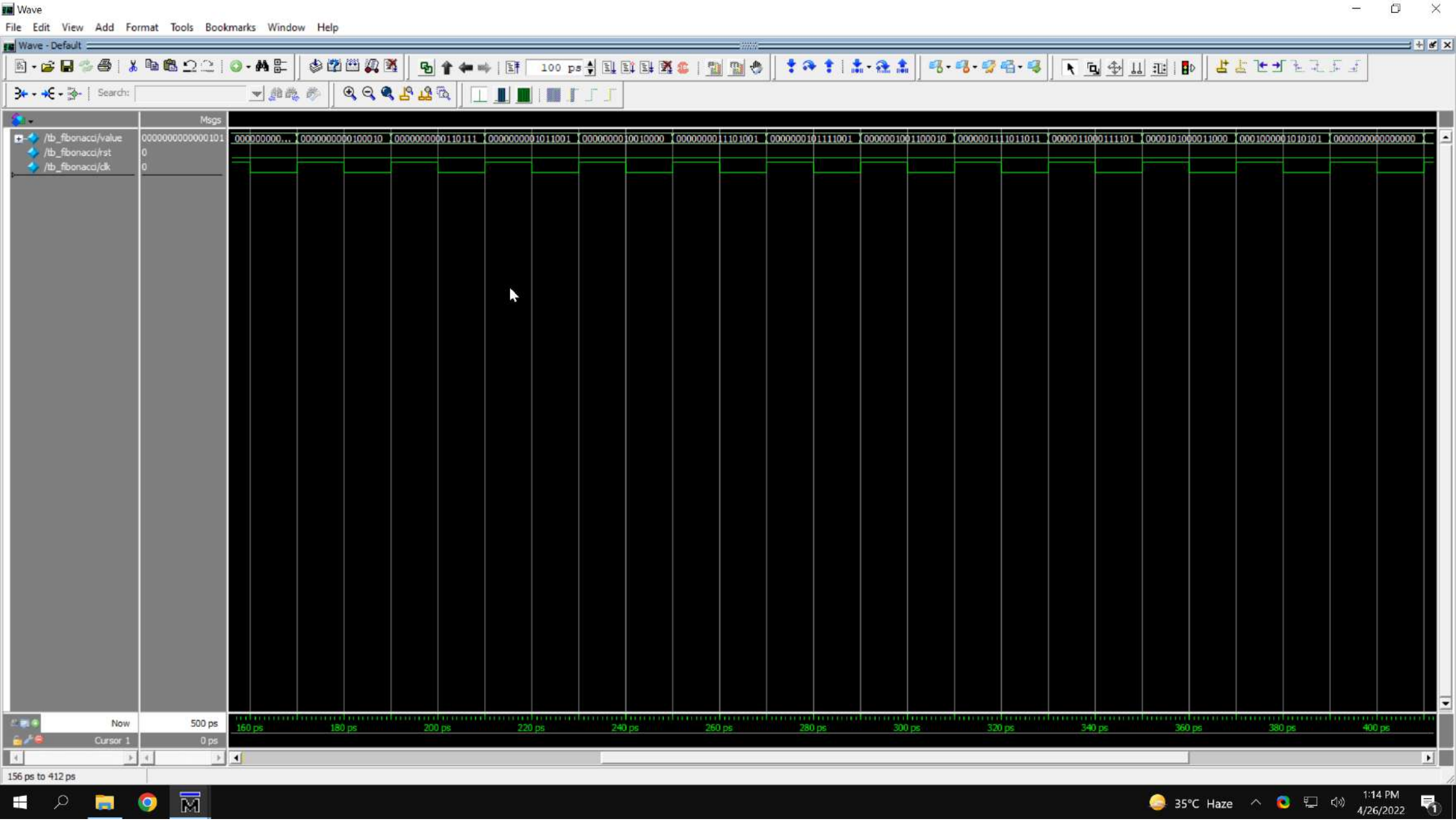
Write the synthesizable behavioural model of a circuit that generates the first 20 values of the Fibonacci series 0, 1, 1, 2, 3, 5, 8, 13, 21…..

Source Code

```
module fibonacci (clk, rst, value);
input rst, clk;
output reg [15:0] value;
reg [15:0] previous, current;
reg [5:0] count;

always @ (posedge clk or posedge rst)
begin
	if (rst == 1'b1)
	begin
		value <= 16'd0;
		previous <= 16'd0;
		current <= 16'd1;
		count <= 6'd1;
	end
	else if (clk == 1'b1)
	begin

		count = count + 1;
		current = current + previous;
		previous = current - previous;
	end
	if (count == 21)
	begin
		value <= 16'd0;
		previous <= 16'd0;
```

```
                current <= 16'd1;
                count <= 6'd1;
        end
        assign value = previous;
end

endmodule

module tb_fibonacci ();
wire [15:0] value;
reg rst, clk;

always begin
        #10;
        clk = ~clk;
        end

initial
begin
        rst = 1'b1; clk = 1'b0; #10;

        rst = 1'b0; clk = 1'b1; #500;
end

fibonacci f1 (clk, rst, value);
endmodule
```

Output Waveform


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_fibonacci/value
/tb_fibonacci/rst
/tb_fibonacci/clk
0000000000000101
0
0
Now
500 ps
Cursor 1
0 ps
160 ps
180 ps
200 ps
220 ps
240 ps
260 ps
280 ps
300 ps
320 ps
340 ps
360 ps
380 ps
400 ps
156 ps to 412 ps
35°C Haze
1:14 PM
4/26/2022

Question 4

Write the synthesizable behavioural model of a Moore machine that detects the sequence 111 with overlaps allowed

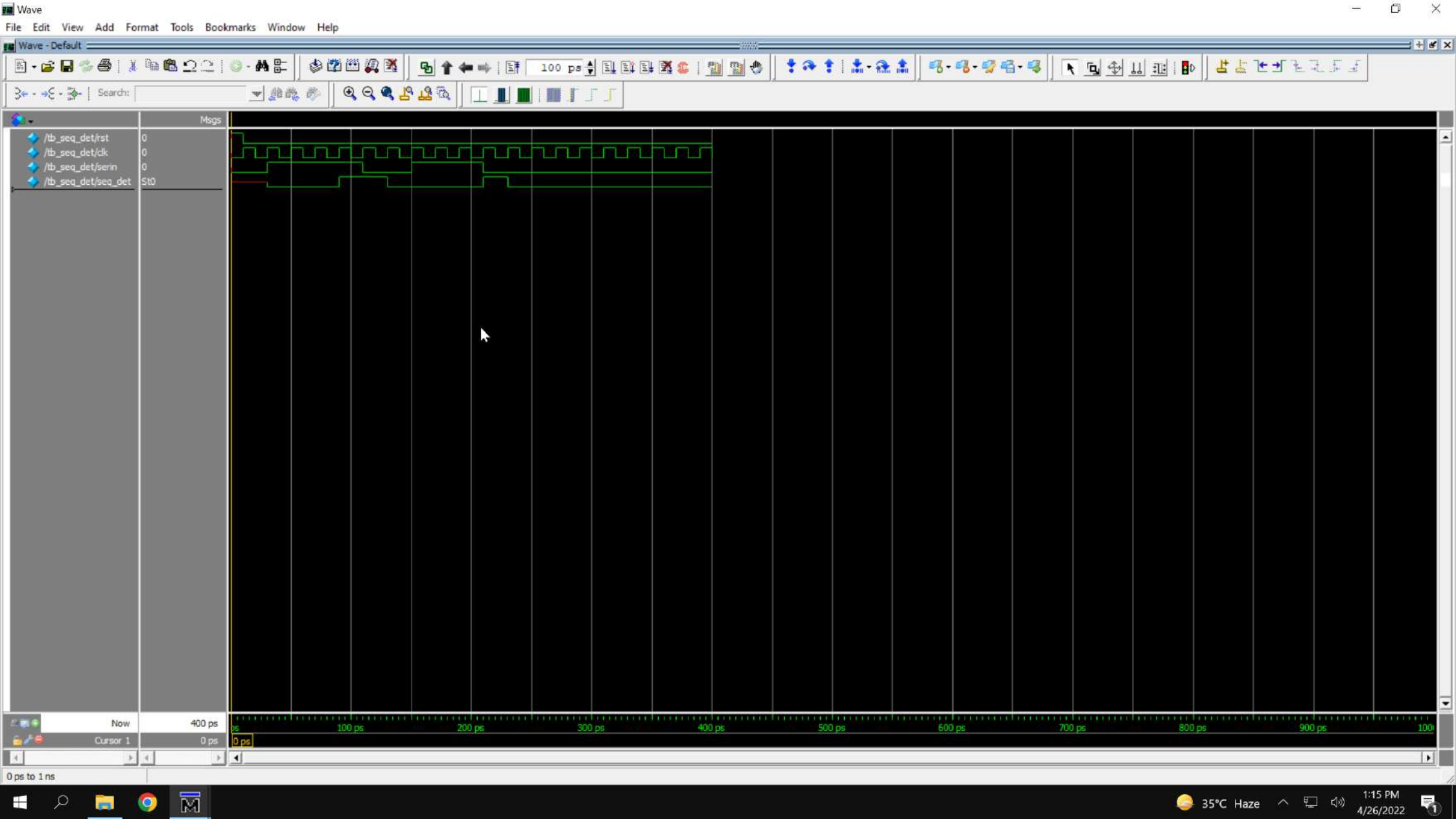
Source Code

```
module seq_det (rst, clk, serin, seq_det);
input rst, clk, serin;
output reg seq_det;
reg [1:0] present = 2'b00, next;

always @ (posedge clk or posedge rst)
begin
	if(rst)
		present <= 2'b00; // Ideal
	else if (clk)
	begin
		present <= next;
	if (next == 2'b11)
		seq_det <= 1'b1;
	else
		seq_det <= 1'b0;
	end
end

always @ (present or serin)
	begin
		case(present)
		2'b00:
		begin
		if (serin == 1'b1)
			next = 2'b01;
```

```
                else
                        next = 2'b00;
                end
                2'b01:
                begin
                if (serin == 1'b1)
                        next = 2'b10;
                else
                        next = 2'b00;
                end
                2'b10:
                begin
                if (serin == 1'b1)
                        next = 2'b11;
                else
                        next = 2'b00;
                end
                2'b11:
                begin
                if (serin == 1'b1)
                        next = 2'b11;
                else
                        next = 2'b00;
                end
                endcase
        end
endmodule

module tb_seq_det();
wire seq_det;
```

```
reg rst, clk, serin;
always
begin
        clk = ~clk;
        #10;
end
initial
begin
        rst <= 1'b1; clk <= 1'b0; serin <= 1'b0; #10;

        rst <= 1'b0; clk <= 1'b1;
        serin <= 1'b0; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b0; #20;
        serin <= 1'b0; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b0; #20;

end

seq_det sqn1 (rst, clk, serin, seq_det);
endmodule
```

Output Waveform


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_seq_det/rst
/tb_seq_det/clk
/tb_seq_det/serin
/tb_seq_det/seq_det
0
0
0
St0
Now
400 ps
Cursor 1
0 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
0 ps to 1 ns
35°C Haze
1:15 PM
4/26/2022

Question 5

Write the synthesizable behavioural model of a Moore machine that detects the sequence 111 with no overlaps allowed.

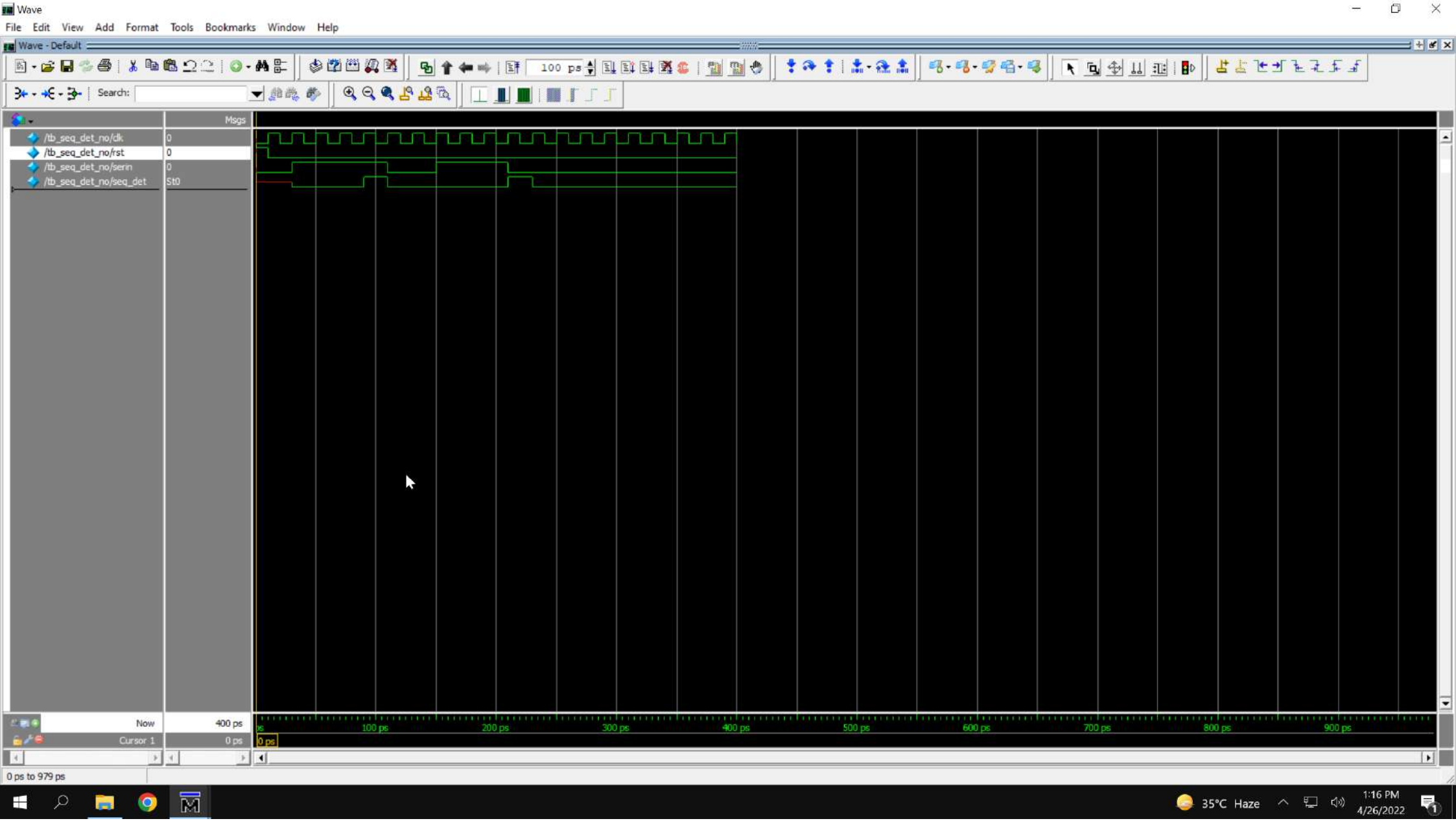
Source Code

```
module seq_det_no (rst, clk, serin, seq_det);
input rst, clk, serin;
output reg seq_det;
reg [1:0] present, next;

always @ (posedge clk or posedge rst)
begin
	if(rst)
		present <= 2'b00; // Ideal
	else if (clk)
	begin
		present <= next;
	if (next == 2'b11)
		seq_det <= 1'b1;
	else
		seq_det <= 1'b0;
	end
end

always @ (present or serin)
	begin
		case(present)

		2'b00:
		begin
		if (serin == 1'b1)
```

```
                next = 2'b01;
            else
                next = 2'b00;
            end
            2'b01:
            begin
            if (serin == 1'b1)
                next = 2'b10;
            else
                next = 2'b00;
            end
            2'b10:
            begin
            if (serin == 1'b1)
                next = 2'b11;
            else
                next = 2'b00;
            end
            2'b11:
            begin
            if (serin == 1'b1)
                next = 2'b01;
            else
                next = 2'b00;
            end
            endcase
        end
endmodule
```

```
module tb_seq_det_no();
wire seq_det;
reg rst, clk, serin;
always
begin
        clk = ~clk;
        #10;
end
initial
begin
        rst <= 1'b1; clk <= 1'b0; serin <= 1'b0; #10;

        rst <= 1'b0; clk <= 1'b1;
        serin <= 1'b0; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b0; #20;
        serin <= 1'b0; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b1; #20;
        serin <= 1'b0; #20;
end
seq_det_no sqno1 (rst, clk, serin, seq_det);
endmodule
```

Output Waveform


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_seq_det_no/clk
/tb_seq_det_no/rst
/tb_seq_det_no/serin
/tb_seq_det_no/seq_det
0
0
0
St0
Now
400 ps
Cursor 1
0 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
0 ps to 979 ps
35°C Haze
1:16 PM
4/26/2022

Question 6

Write the synthesizable behavioural model of a Moore machine that detects a valid 0 to 1 transition in a serial input *serin*

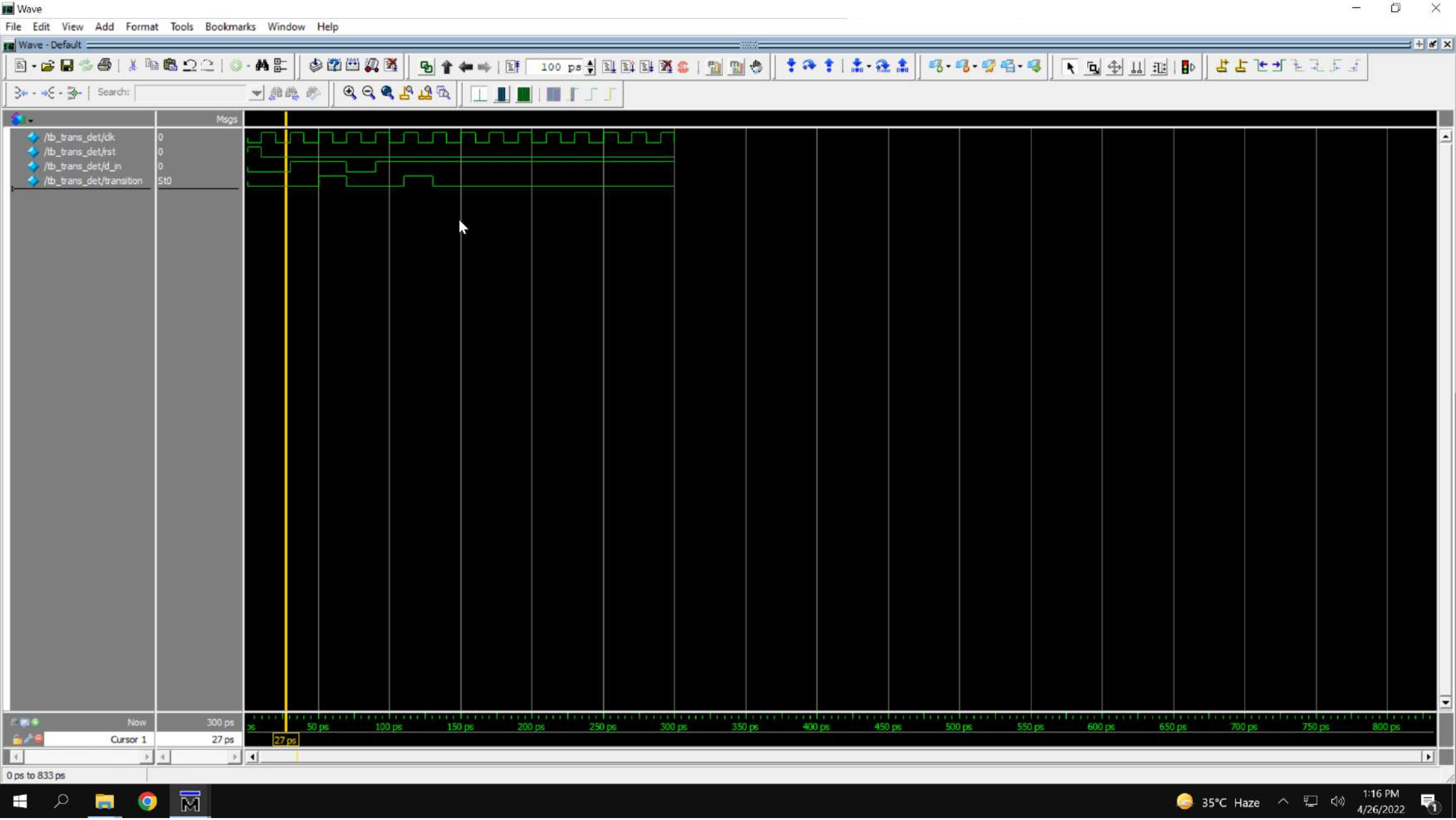
Source Code

```
module trans_det (rst, clk, d_in, transition);


input rst, clk, d_in;
output reg transition;
reg [1:0] present = 2'b00, next;



always @ (posedge clk or posedge rst)
begin
	if(rst)
		present <= 1'b0; // Ideal
	begin
		present <= next;
	if (next == 2'b11)
	begin
		transition <= 1'b1;
		present <= 2'b00;
	end
	else
		transition <= 1'b0;
	end
end

always @ (present or d_in)
	begin
		case(present)
```

```
                2'b00:
                begin
                if (d_in == 1'b0)
                        next = 2'b01;
                else
                        next = 2'b00;
                end

                2'b01:
                begin
                if (d_in == 1'b1)
                        next = 2'b11;
                else
                        next = 2'b01;
                end
                endcase
        end
endmodule


module tb_trans_det();
wire transition;
reg rst, clk, d_in;

always
begin
        clk = ~clk;
        #10;
end
```

```
initial
begin
        rst <= 1'b1; clk <= 1'b0; d_in <= 1'b0; #10;


        rst <= 1'b0; clk <= 1'b1;
        d_in <= 1'b0; #20;
        d_in <= 1'b1; #20;
        d_in <= 1'b1; #20;
        d_in <= 1'b0; #20;
        d_in <= 1'b1; #20;




end

trans_det td1 (rst, clk, d_in, transition);

endmodule
```

Output Waveform


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Msgs
/tb_trans_det/clk
/tb_trans_det/rst
/tb_trans_det/d_in
/tb_trans_det/transition
0
0
0
St0
Now
300 ps
Cursor 1
27 ps
50 ps
100 ps
150 ps
200 ps
250 ps
300 ps
350 ps
400 ps
450 ps
500 ps
550 ps
600 ps
650 ps
700 ps
750 ps
800 ps
0 ps to 833 ps
35°C Haze
1:16 PM
4/26/2022