19ECE349 - RISC Processor Design Using HDL

Nigil M.R [CB.EN.U4ECE19136]

Assignment 1 - Combinational Circuits Modelling using Verilog HDL

1. Write the synthesizable behavioural model of full adder. Use the full adder as a component to realize the structural model of an n-bit parametrizable ripple adder. Verify the ripple adder for the case where the word-size is 16 for five pairs of inputs.

Full Adder - Behavioural Model Code

```
module fulladder (c_in, x, y, s_out, c_out);
input wire c_in, x, y;
output reg s_out, c_out;

always @ (x or y or c_in)
begin
s_out = x ^ y ^ c_in;
c_out = (x & y ) | (x & c_in) | (y & c_in);
end
endmodule

// TEST BENCH
module tb_fulladder;
wire s_out,c_out;
reg x,y,c_in;

initial
begin
x = 1'b0; y = 1'b0; c_in = 1'b0;
#100; // Sum 0, Cout 0
x = 1'b1; y = 1'b0; c_in = 1'b0;
#100; // Sum 1, Cout 0
x = 1'b1; y = 1'b1; c_in = 1'b1;
#100; // Sum 1, Cout 1
x = 1'b0; y = 1'b1; c_in = 1'b1;
#100; // Sum 0, Cout 1
x = 1'b0; y = 1'b1; c_in = 1'b0;
#100; // Sum 1, Cout 0

end
fulladder fa1 (c_in, x, y, s_out, c_out);
endmodule
```

Code & Output Waveform Screenshot

C:/Users/Nigil/Documents/Model Sim/FA_Beh.v

File Edit View Tools Bookmarks Window Help

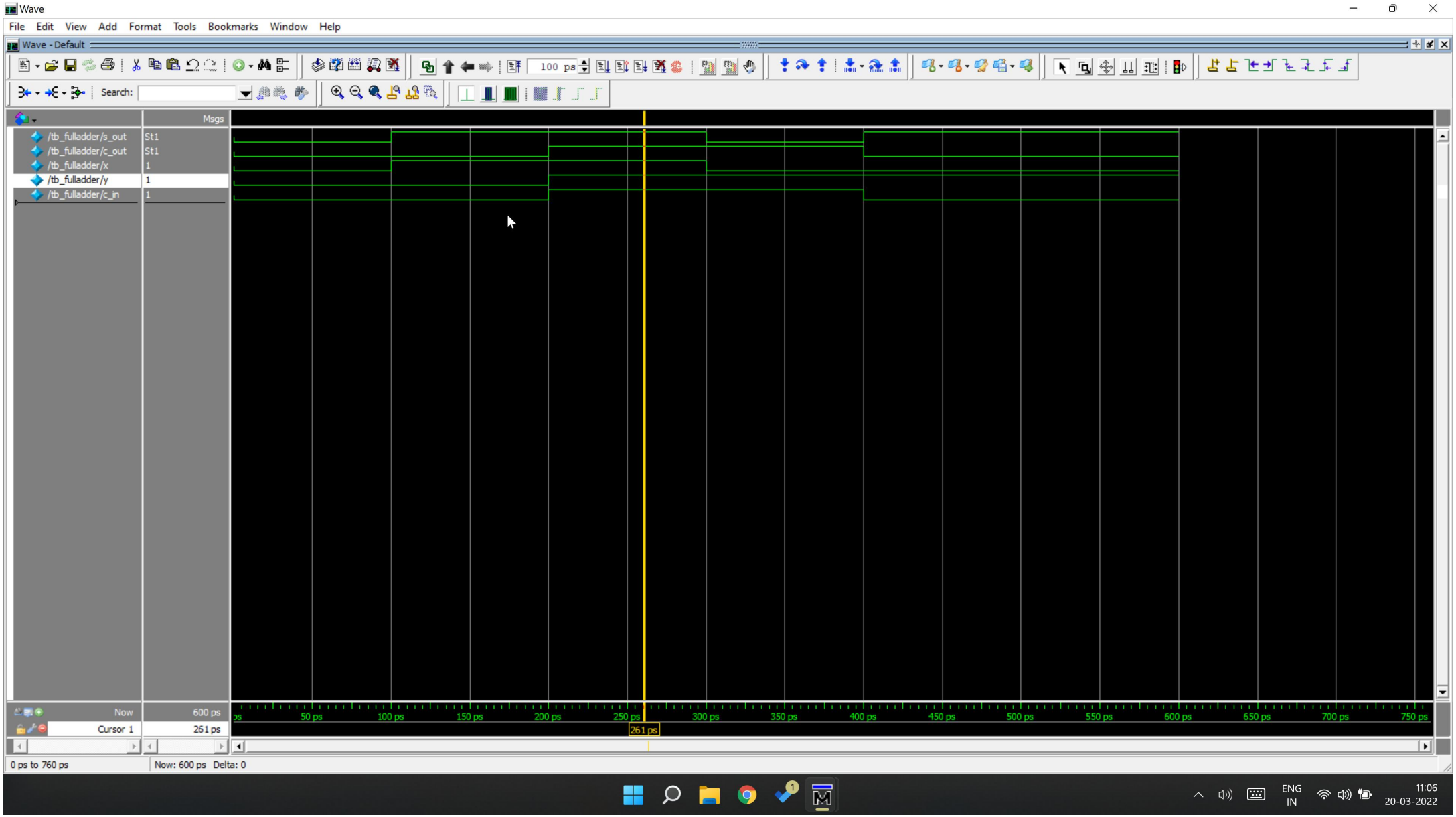
C:/Users/Nigil/Documents/Model Sim/FA_Beh.v - Default

Ln#

```
1  // FULL ADDER
2
3  module fulladder (c_in, x, y, s_out, c_out);
4
5  input wire c_in, x, y;
6  output reg s_out, c_out;
7
8  always @ (x or y or c_in)
9  begin
10
11 s_out = x ^ y ^ c_in;
12
13 c_out = (x & y ) | (x & c_in) | (y & c_in);
14
15 end
16 endmodule
17
18 module tb_fulladder;
19 wire s_out,c_out;
20 reg x,y,c_in;
21
22 initial
23 begin
24 x = 1'b0; y = 1'b0; c_in = 1'b0;
25 #100;
26 x = 1'b1; y = 1'b0; c_in = 1'b0;
27 #100;
28 x = 1'b1; y = 1'b1; c_in = 1'b1;
29 #100;
30 x = 1'b0; y = 1'b1; c_in = 1'b1;
31 #100;
32 x = 1'b0; y = 1'b1; c_in = 1'b0;
33 #100;
34
35 end
36
37 fulladder fal (c_in, x, y, s_out, c_out);
38
39 endmodule
40
```

Ln: 24 Col: 32

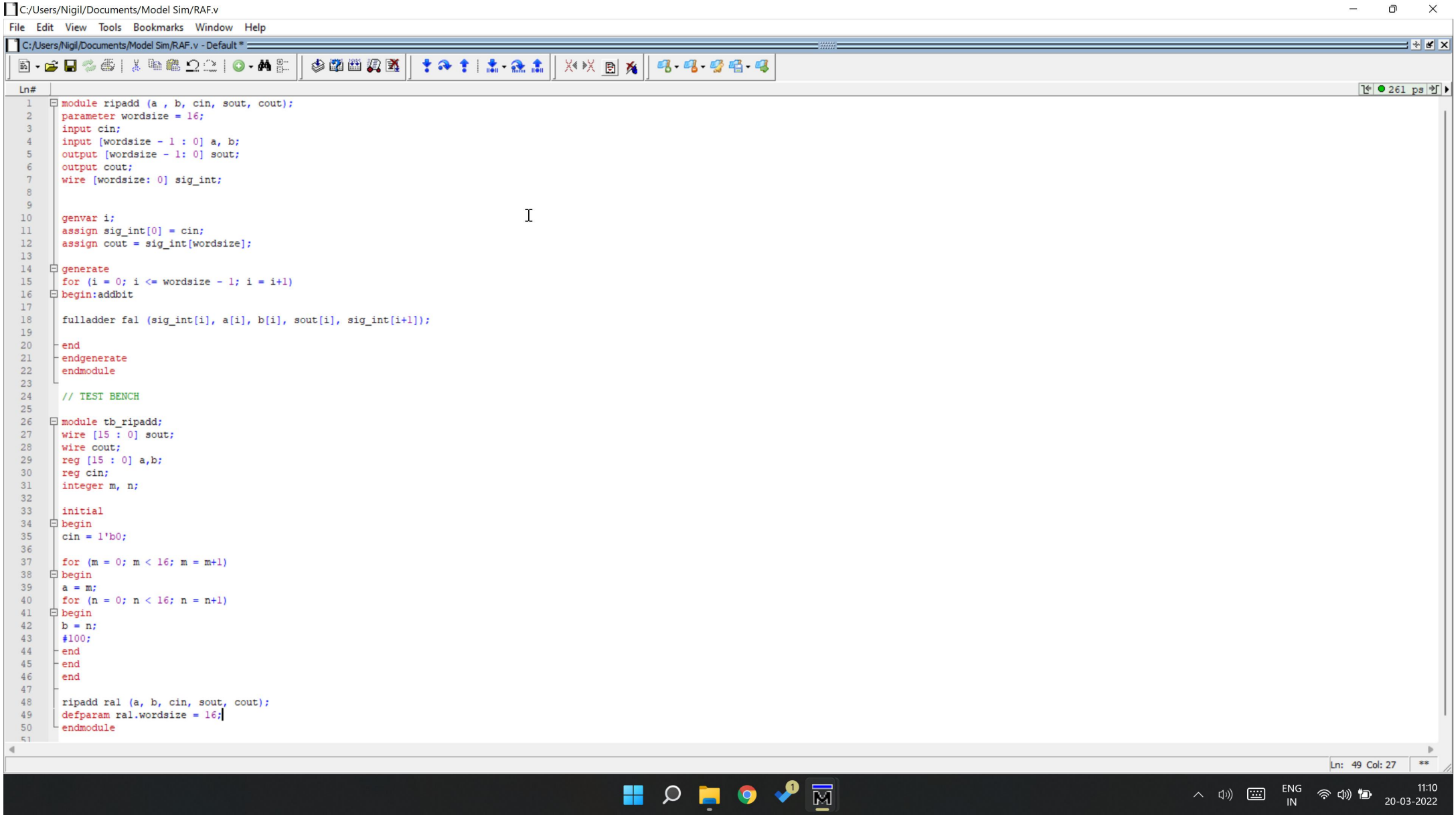
ENG IN 11:05 20-03-2022

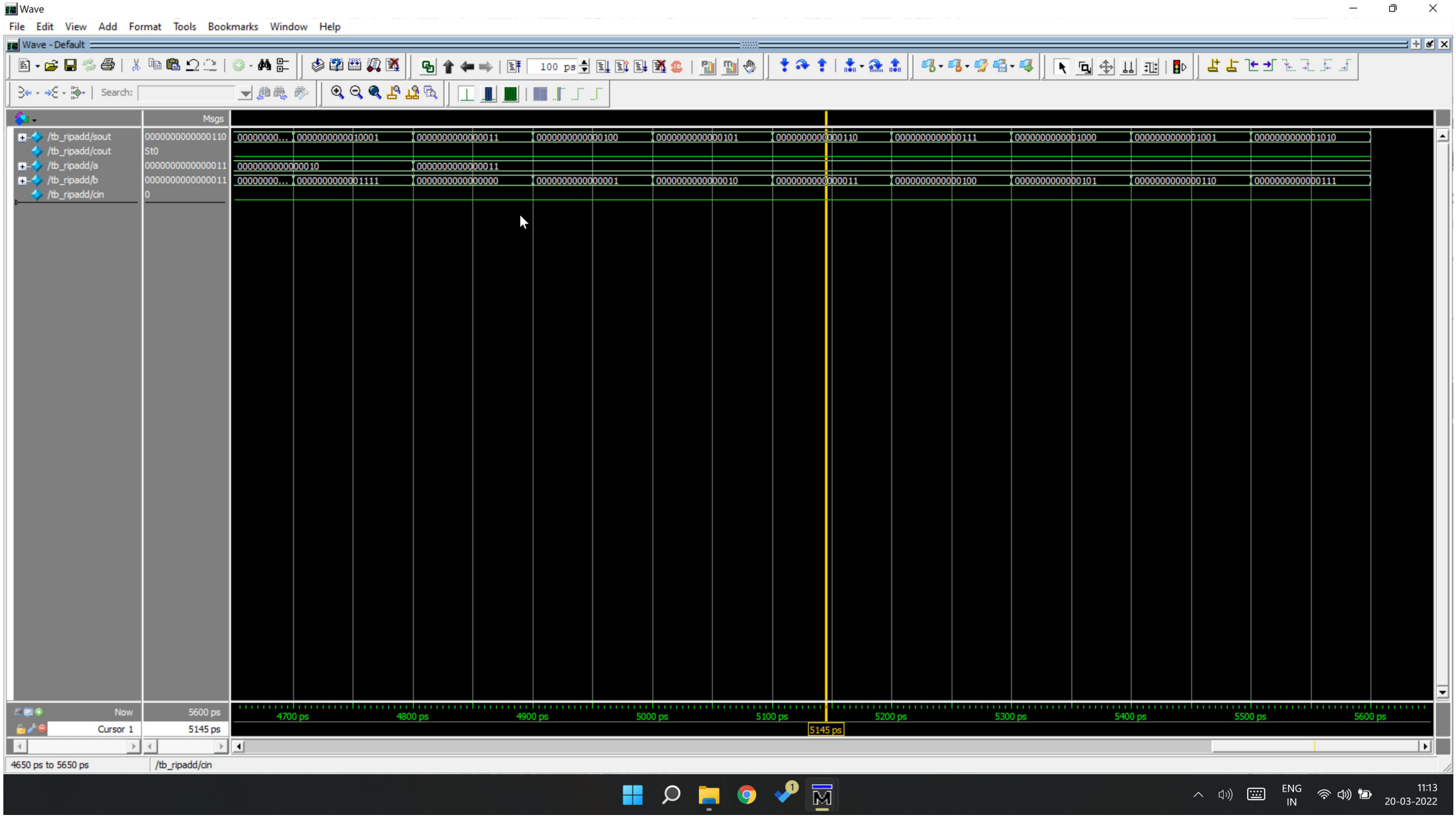

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_fulladder/s_out
/tb_fulladder/c_out
/tb_fulladder/x
/tb_fulladder/y
/tb_fulladder/c_in
St1
St1
1
1
1
Now
600 ps
Cursor 1
261 ps
50 ps
100 ps
150 ps
200 ps
250 ps
300 ps
350 ps
400 ps
450 ps
500 ps
550 ps
600 ps
650 ps
700 ps
750 ps
261 ps
0 ps to 760 ps
Now: 600 ps Delta: 0
ENG
IN
11:06
20-03-2022

Ripple Carry Adder - Structural Model Code

```
module ripadd (a , b, cin, sout, cout);
parameter wordsize = 16;
input cin;
input [wordsize - 1 : 0] a, b;
output [wordsize - 1: 0] sout;
output cout;
wire [wordsize: 0] sig_int;

genvar i;
assign sig_int[0] = cin;
assign cout = sig_int[wordsize];

generate
for (i = 0; i <= wordsize - 1; i = i+1)
begin:addbit

fulladder fa1 (sig_int[i], a[i], b[i], sout[i],
sig_int[i+1]);

end
endgenerate
endmodule

//  TEST BENCH
module tb_ripadd;
wire [15 : 0] sout;
wire cout;
reg [15 : 0] a,b;
reg cin;
integer m, n;
initial
begin
cin = 1'b0;

for (m = 0; m < 16; m = m+1)
begin
a = m;
for (n = 0; n < 16; n = n+1)
begin
b = n;
#100;

end
end
end

ripadd ra1 (a, b, cin, sout, cout);
defparam ra1.wordsize = 16;

endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/RAF.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/RAF.v - Default *
Ln#
261 ps
module ripadd (a , b, cin, sout, cout);
parameter wordsize = 16;
input cin;
input [wordsize - 1 : 0] a, b;
output [wordsize - 1: 0] sout;
output cout;
wire [wordsize: 0] sig_int;
genvar i;
assign sig_int[0] = cin;
assign cout = sig_int[wordsize];
generate
for (i = 0; i <= wordsize - 1; i = i+1)
begin:addbit
fulladder fal (sig_int[i], a[i], b[i], sout[i], sig_int[i+1]);
end
endgenerate
endmodule
// TEST BENCH
module tb_ripadd;
wire [15 : 0] sout;
wire cout;
reg [15 : 0] a,b;
reg cin;
integer m, n;
initial
begin
cin = 1'b0;
for (m = 0; m < 16; m = m+1)
begin
a = m;
for (n = 0; n < 16; n = n+1)
begin
b = n;
#100;
end
end
end
ripadd ral (a, b, cin, sout, cout);
defparam ral.wordsize = 16;
endmodule
Ln: 49 Col: 27
ENG
IN
11:10
20-03-2022

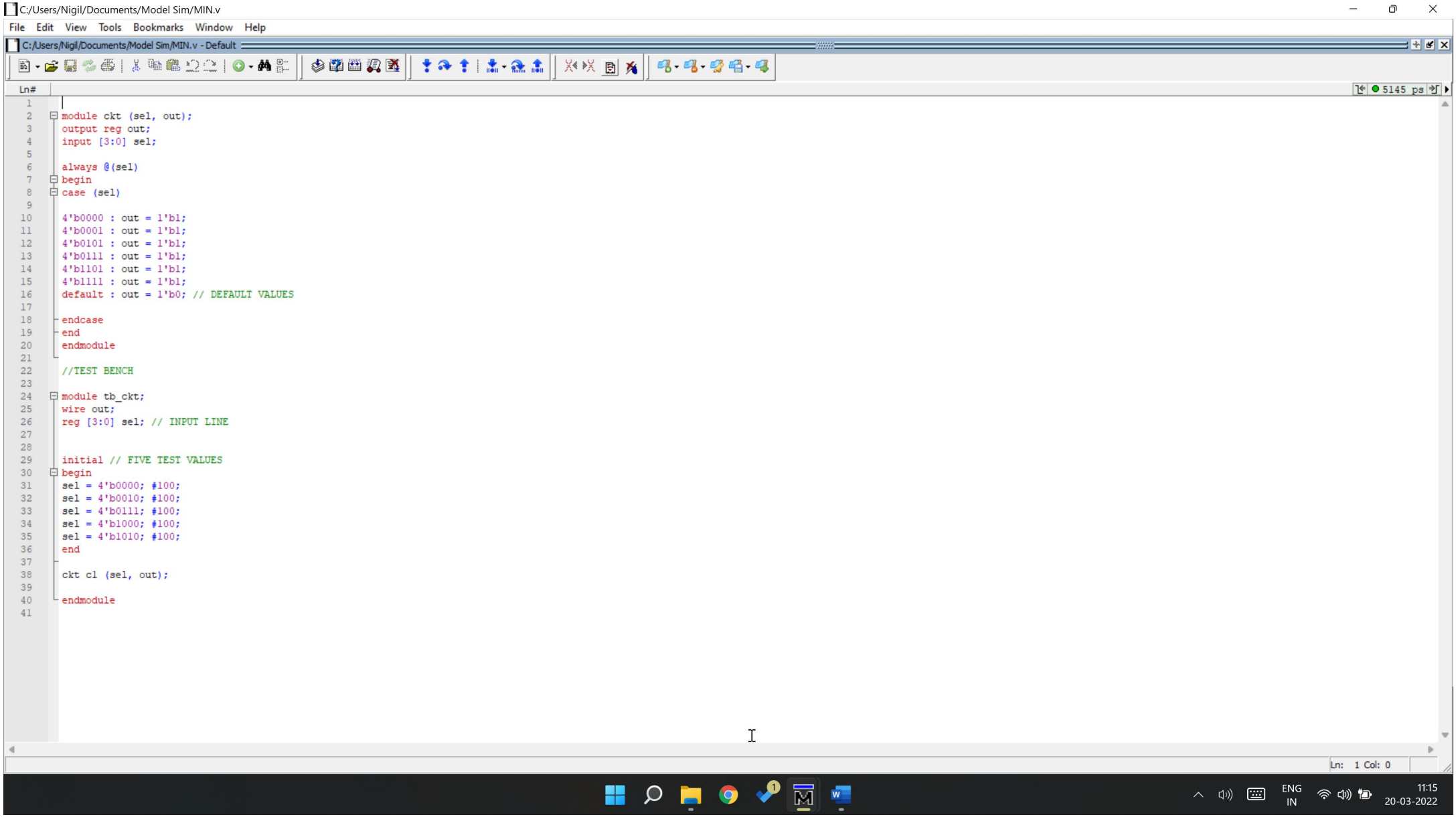

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_ripadd/sout
/tb_ripadd/cout
/tb_ripadd/a
/tb_ripadd/b
/tb_ripadd/cin
0000000000000110
St0
0000000000000011
0000000000000011
0
00000000... 0000000000010001 0000000000000011 0000000000000100 0000000000000101 0000000000000110 0000000000000111 0000000000001000 0000000000001001 0000000000001010
0000000000000010 0000000000000011
00000000... 0000000000001111 0000000000000000 0000000000000001 0000000000000010 0000000000000011 0000000000000100 0000000000000101 0000000000000110 0000000000000111
Now
5600 ps
Cursor 1
5145 ps
4700 ps 4800 ps 4900 ps 5000 ps 5100 ps 5200 ps 5300 ps 5400 ps 5500 ps 5600 ps
5145 ps
4650 ps to 5650 ps
/tb_ripadd/cin
ENG IN
11:13
20-03-2022

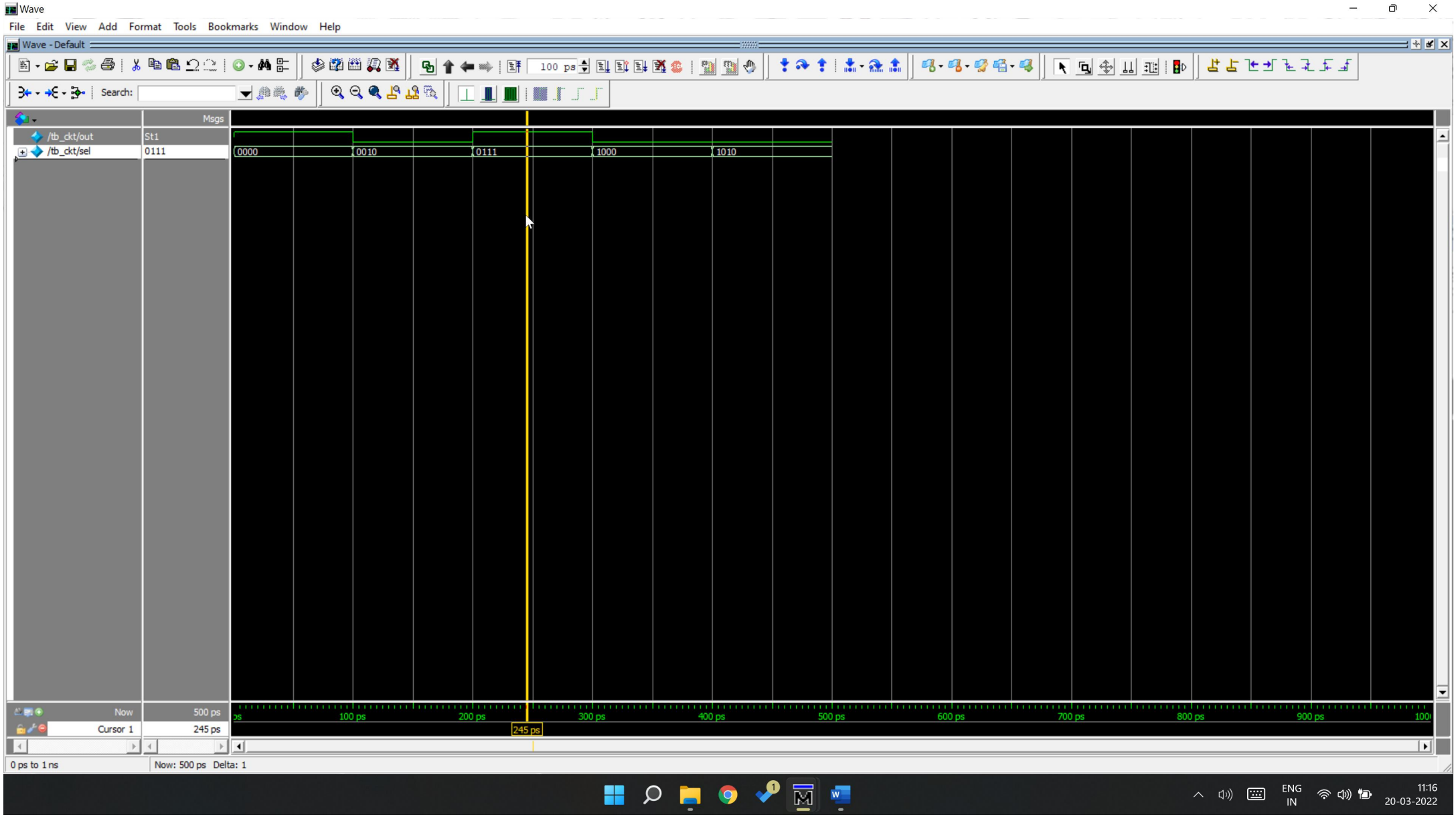
2. Using case statement write a synthesizable behavioural model for the following function
   F(A,B,C,D) = Σm(0,1,7,9,13,15)

Behavioural Model Code - Case Statement

```
module ckt (sel, out);
output reg out;
input [3:0] sel;

always @(sel)
begin
case (sel)

4'b0000 : out = 1'b1;
4'b0001 : out = 1'b1;
4'b0101 : out = 1'b1;
4'b0111 : out = 1'b1;
4'b1101 : out = 1'b1;
4'b1111 : out = 1'b1;
default : out = 1'b0; // DEFAULT VALUES

endcase
end
endmodule

// TEST BENCH
module tb_ckt;
wire out;
reg [3:0] sel; // INPUT LINE

initial
begin
sel = 4'b0000; #100; // out = 1
sel = 4'b0010; #100; // out = 0
sel = 4'b0111; #100; // out = 1
sel = 4'b1000; #100; // out = 0
sel = 4'b1010; #100; // out = 0
end

ckt c1 (sel, out);

endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/MIN.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/MIN.v - Default
Ln#
5145 ps
module ckt (sel, out);
output reg out;
input [3:0] sel;
always @(sel)
begin
case (sel)
4'b0000 : out = 1'b1;
4'b0001 : out = 1'b1;
4'b0101 : out = 1'b1;
4'b0111 : out = 1'b1;
4'b1101 : out = 1'b1;
4'b1111 : out = 1'b1;
default : out = 1'b0; // DEFAULT VALUES
endcase
end
endmodule
//TEST BENCH
module tb_ckt;
wire out;
reg [3:0] sel; // INPUT LINE
initial // FIVE TEST VALUES
begin
sel = 4'b0000; #100;
sel = 4'b0010; #100;
sel = 4'b0111; #100;
sel = 4'b1000; #100;
sel = 4'b1010; #100;
end
ckt c1 (sel, out);
endmodule
Ln: 1 Col: 0
ENG
IN
11:15
20-03-2022


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_ckt/out
/tb_ckt/sel
St1
0111
0000
0010
0111
1000
1010
Now
500 ps
Cursor 1
245 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
245 ps
0 ps to 1 ns
Now: 500 ps Delta: 1
ENG
IN
11:16
20-03-2022

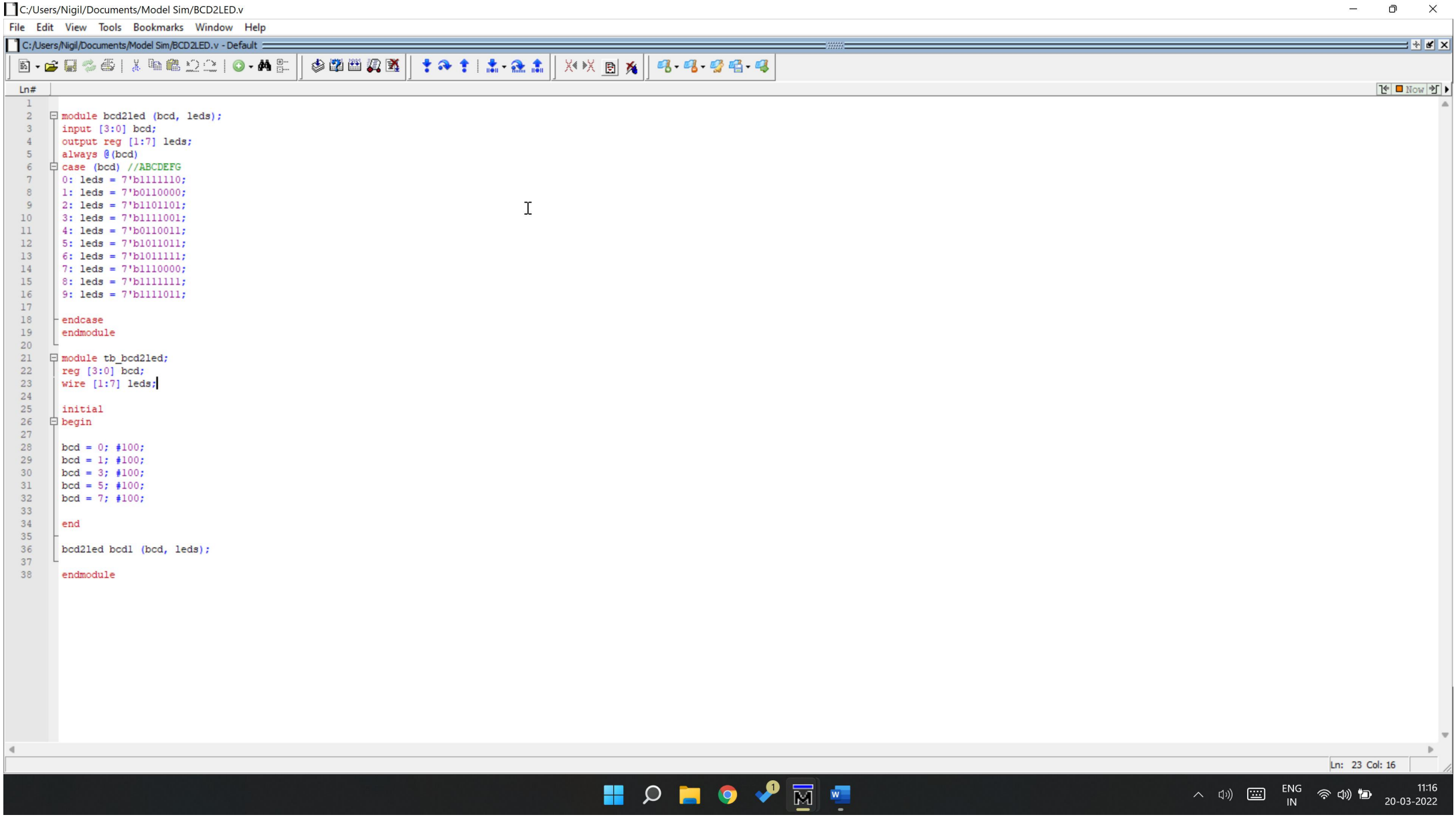
3. Write a synthesizable behavioural model for a BCD to seven segments decode.

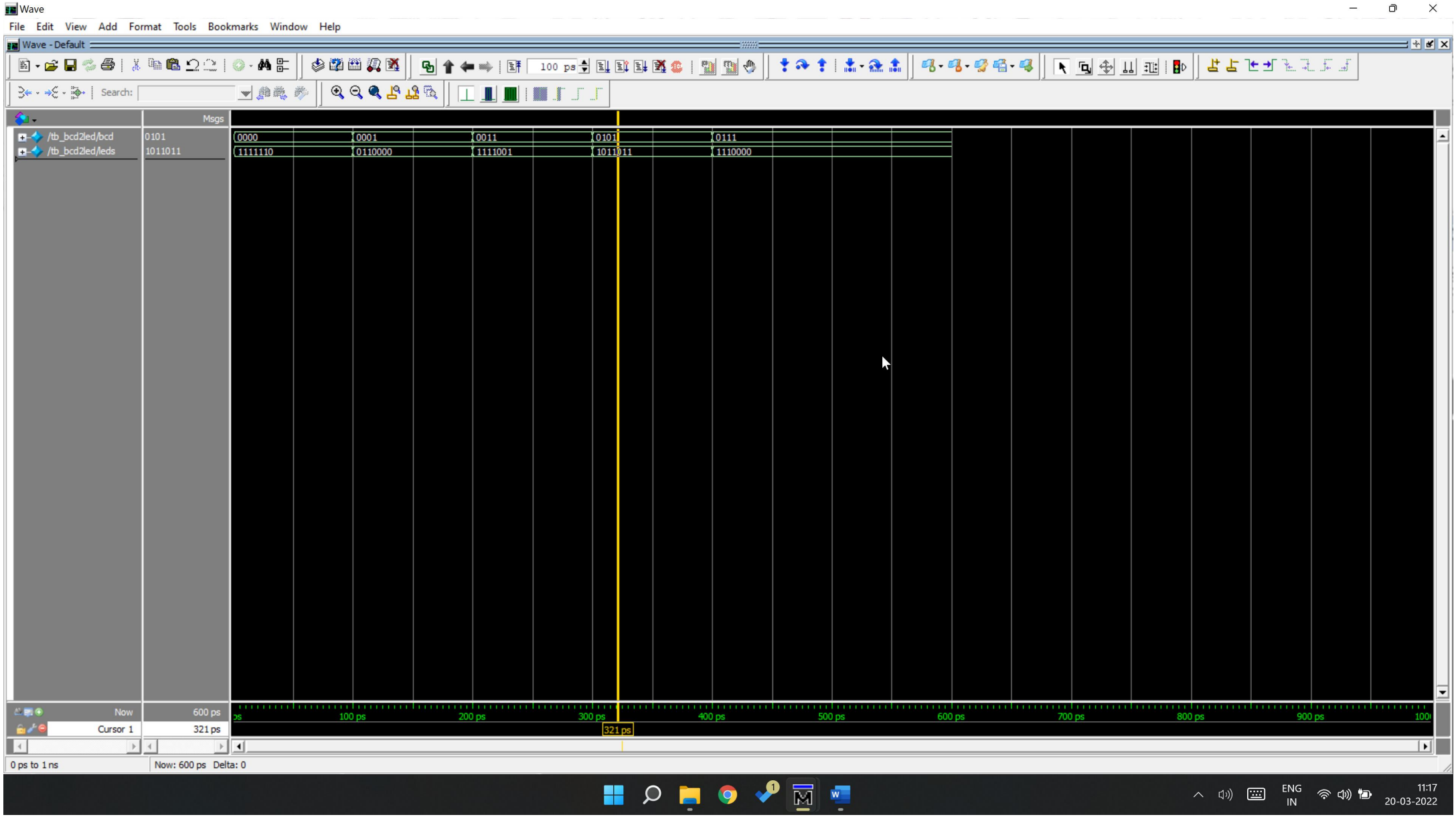
BCD-Seven Segment Display - Behavioural Model Code

```
module bcd2led (bcd, leds);
input [3:0] bcd;
output reg [1:7] leds;

always @(bcd)
case (bcd) //ABCDEFG
0: leds = 7'b1111110;
1: leds = 7'b0110000;
2: leds = 7'b1101101;
3: leds = 7'b1111001;
4: leds = 7'b0110011;
5: leds = 7'b1011011;
6: leds = 7'b1011111;
7: leds = 7'b1110000;
8: leds = 7'b1111111;
9: leds = 7'b1111011;

endcase
endmodule
```

```
// TEST BENCH
module tb_bcd2led;
reg [3:0] bcd;
wire [1:7] leds;

initial
begin

bcd = 0; #100; // leds = 7'b0110000
bcd = 1; #100; // leds = 7'b1111110
bcd = 3; #100; // leds = 7'b1111001
bcd = 5; #100; // leds = 7'b1011011
bcd = 7; #100; // leds = 7'b1110000

end

bcd2led bcd1 (bcd, leds);

endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/BCD2LED.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/BCD2LED.v - Default
Ln#
module bcd2led (bcd, leds);
input [3:0] bcd;
output reg [1:7] leds;
always @(bcd)
case (bcd) //ABCDEFG
0: leds = 7'b1111110;
1: leds = 7'b0110000;
2: leds = 7'b1101101;
3: leds = 7'b1111001;
4: leds = 7'b0110011;
5: leds = 7'b1011011;
6: leds = 7'b1011111;
7: leds = 7'b1110000;
8: leds = 7'b1111111;
9: leds = 7'b1111011;
endcase
endmodule
module tb_bcd2led;
reg [3:0] bcd;
wire [1:7] leds;
initial
begin
bcd = 0; #100;
bcd = 1; #100;
bcd = 3; #100;
bcd = 5; #100;
bcd = 7; #100;
end
bcd2led bcd1 (bcd, leds);
endmodule
Ln: 23 Col: 16
ENG
IN
11:16
20-03-2022

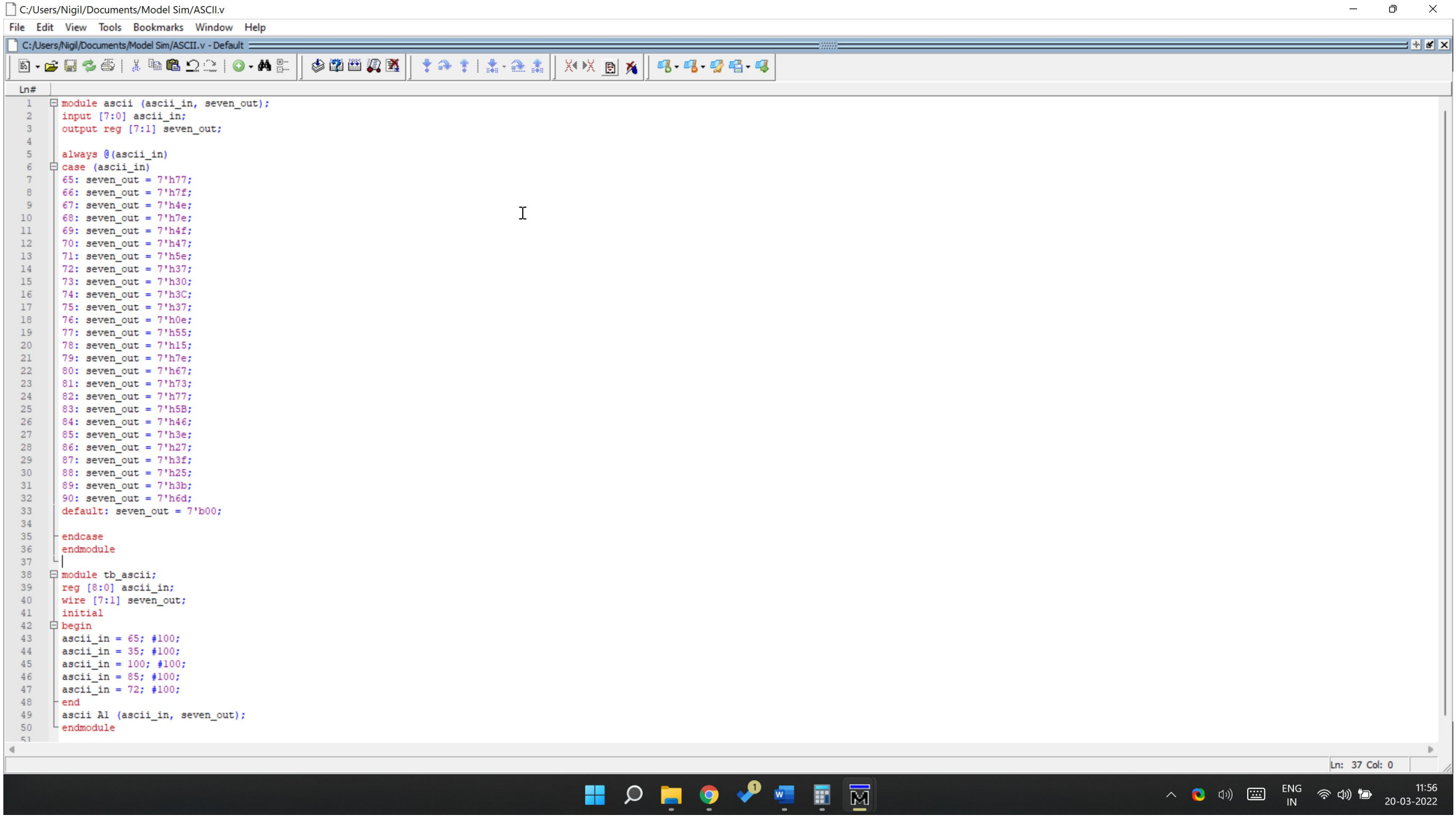

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_bcd2led/bcd
/tb_bcd2led/leds
0101
1011011
0000
0001
0011
0101
0111
1111110
0110000
1111001
1011011
1110000
Now
600 ps
Cursor 1
321 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
0 ps to 1 ns
Now: 600 ps Delta: 0
ENG
IN
11:17
20-03-2022

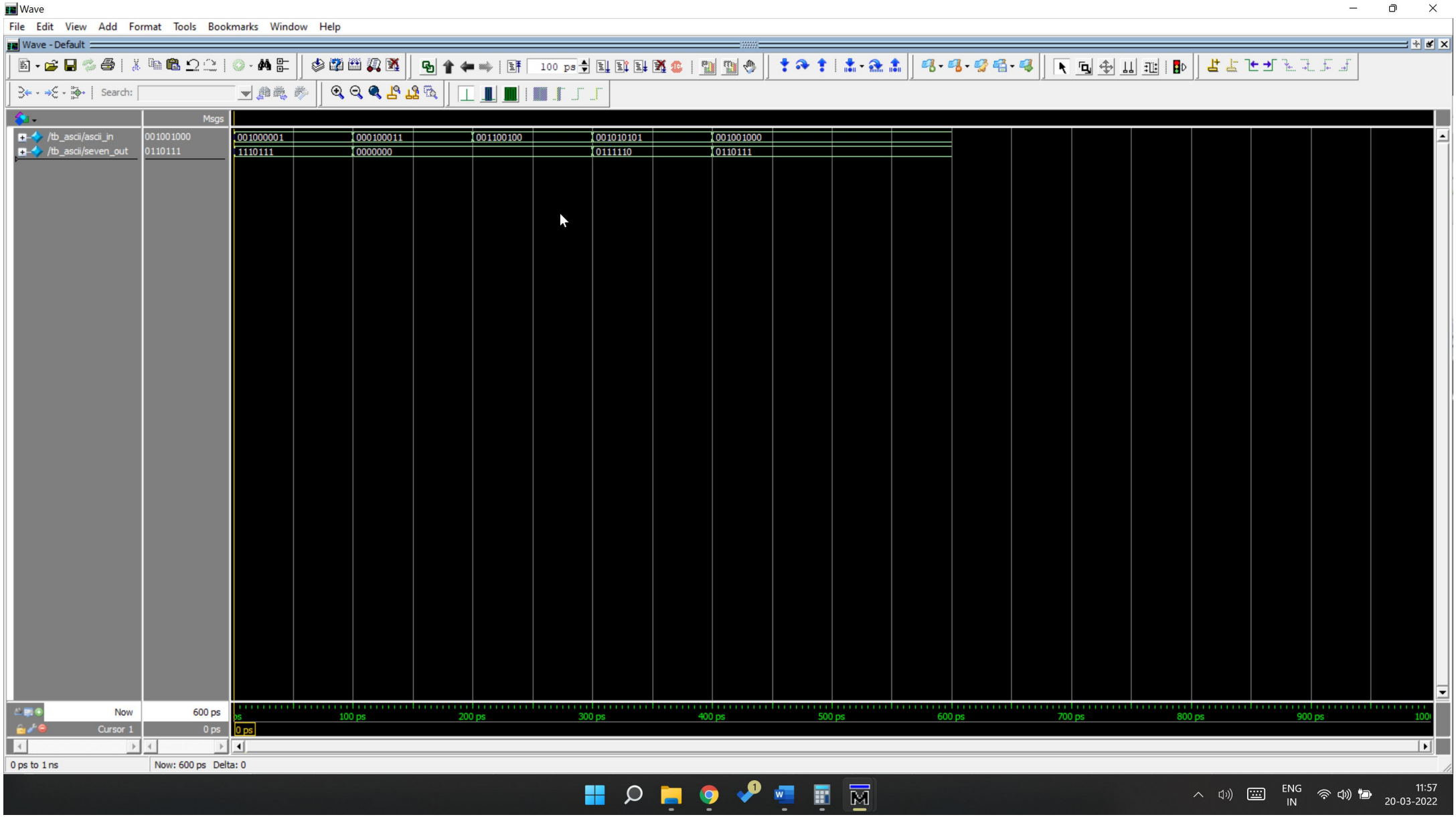
4. A circuit has an 8-bit ACII code as input. The circuit generates the outputs that drive a seven-segment display. If the ASCII input corresponds to an uppercase alphabet, the corresponding seven-segment code is generated. For any other ASCII code, the seven-segment display is not active. Write a synthesizable behavioural model for the circuit.

8-Bit ASCII - Seven Segment Display - Behavioural Model Code

```
module ascii (ascii_in, seven_out);
input [7:0] ascii_in;
output reg [7:1] seven_out;
always @(ascii_in)
case (ascii_in)
65: seven_out = 7'h77;
66: seven_out = 7'h7f;
67: seven_out = 7'h4e;
68: seven_out = 7’h7e;
69: seven_out = 7'h4f;
70: seven_out = 7'h47;
71: seven_out = 7'h5e;
72: seven_out = 7'h37;
73: seven_out = 7'h30;
74: seven_out = 8'h3C;
75: seven_out = 8'h37;
76: seven_out = 7'h0e;
77: seven_out = 7'h55;
78: seven_out = 7'h15;
79: seven_out = 7'h7e;
80: seven_out = 7'h67;
81: seven_out = 7'h73;
82: seven_out = 7'h77;
83: seven_out = 7'h5B;
84: seven_out = 7'h46;
85: seven_out = 7'h3e;
86: seven_out = 7'h27;
87: seven_out = 7'h3f;
88: seven_out = 7'h25;
89: seven_out = 7'h3b;
90: seven_out = 7'h6d;
default: seven_out = 7'b00;
endcase
endmodule
// TEST BENCH
module tb_ascii;
reg [8:0] ascii_in;
wire [7:1] seven_out;
initial
begin
ascii_in = 65; #100; // seven_out = 7'b111011
ascii_in = 35; #100; // seven_out = 7'b0000000
ascii_in = 100; #100; // seven_out = 7'h0000000
ascii_in = 85; #100; // seven_out = 7'h1000110
ascii_in = 72; #100; // seven_out = 7'h0110111
end
ascii A1 (ascii_in, seven_out);
endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/ASCII.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/ASCII.v - Default
Ln#
1 module ascii (ascii_in, seven_out);
2 input [7:0] ascii_in;
3 output reg [7:1] seven_out;
4
5 always @(ascii_in)
6 case (ascii_in)
7 65: seven_out = 7'h77;
8 66: seven_out = 7'h7f;
9 67: seven_out = 7'h4e;
10 68: seven_out = 7'h7e;
11 69: seven_out = 7'h4f;
12 70: seven_out = 7'h47;
13 71: seven_out = 7'h5e;
14 72: seven_out = 7'h37;
15 73: seven_out = 7'h30;
16 74: seven_out = 7'h3C;
17 75: seven_out = 7'h37;
18 76: seven_out = 7'h0e;
19 77: seven_out = 7'h55;
20 78: seven_out = 7'h15;
21 79: seven_out = 7'h7e;
22 80: seven_out = 7'h67;
23 81: seven_out = 7'h73;
24 82: seven_out = 7'h77;
25 83: seven_out = 7'h5B;
26 84: seven_out = 7'h46;
27 85: seven_out = 7'h3e;
28 86: seven_out = 7'h27;
29 87: seven_out = 7'h3f;
30 88: seven_out = 7'h25;
31 89: seven_out = 7'h3b;
32 90: seven_out = 7'h6d;
33 default: seven_out = 7'b00;
34
35 endcase
36 endmodule
37
38 module tb_ascii;
39 reg [8:0] ascii_in;
40 wire [7:1] seven_out;
41 initial
42 begin
43 ascii_in = 65; #100;
44 ascii_in = 35; #100;
45 ascii_in = 100; #100;
46 ascii_in = 85; #100;
47 ascii_in = 72; #100;
48 end
49 ascii A1 (ascii_in, seven_out);
50 endmodule
Ln: 37 Col: 0
ENG
IN
11:56
20-03-2022

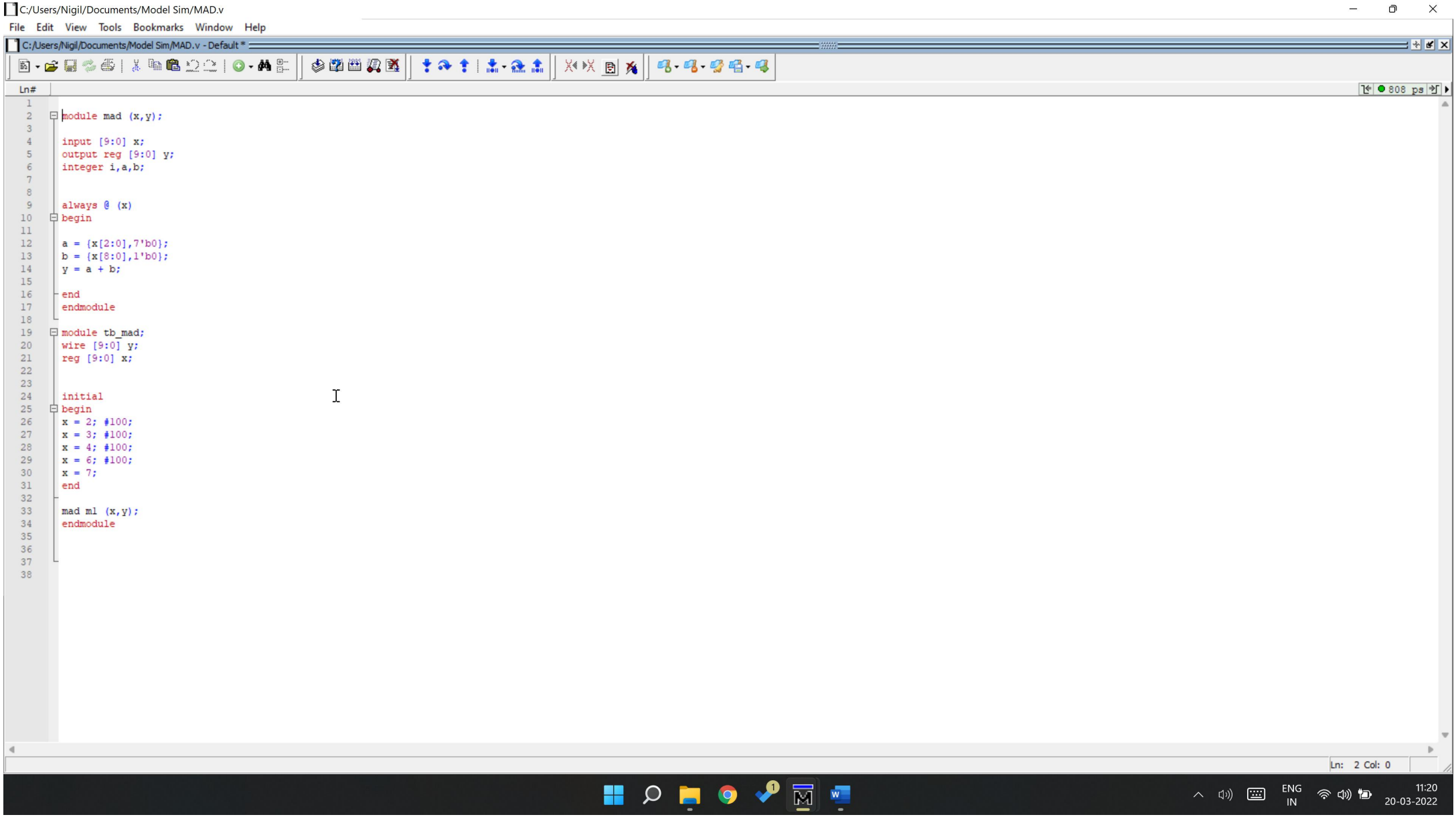

Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_ascii/ascii_in
/tb_ascii/seven_out
001001000
0110111
001000001
000100011
001100100
001010101
001001000
1110111
0000000
0111110
0110111
Now
600 ps
Cursor 1
0 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
0 ps to 1 ns
Now: 600 ps Delta: 0
ENG
IN
11:57
20-03-2022

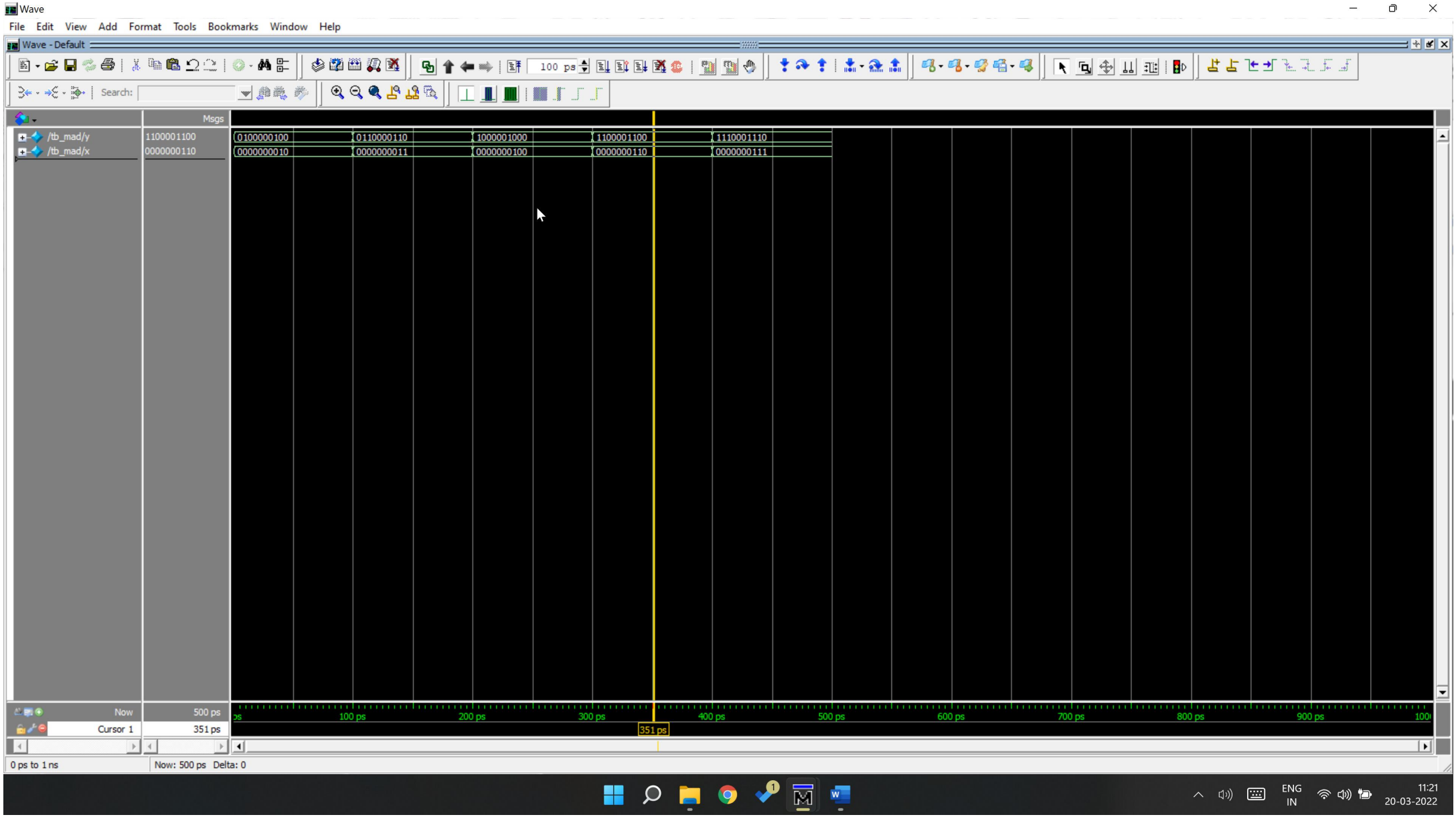
5. Write a synthesizable model of a combination circuit that computes y where y = 130x without using multiplication operator.

Barrel Shifter - Behavioural Model Code

```
module mad (x,y);

input [9:0] x;
output reg [9:0] y;
integer i,a,b;

always @ (x)
begin

a = {x[2:0],7'b0};
b = {x[8:0],1'b0};
y = a + b;

end
endmodule
// TEST BENCH
module tb_mad;
wire [9:0] y;
reg [9:0] x;

initial
begin
x = 2; #100;
// y = 10'b 1 0000 0100 (260)
x = 3; #100;
// y = 10'b 1 1000 0110 (390)
x = 4; #100;
// y = 10'b 10 0000 1000 (520)
x = 6; #100;
// y = 10'b 11 0000 1100 (780)
x = 7;
// y = 10'b 11 1000 1110 (910)
end

mad m1 (x,y);
endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/MAD.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/MAD.v - Default *
Ln#
808 ps
1 module mad (x,y);
2
3 input [9:0] x;
4 output reg [9:0] y;
5 integer i,a,b;
6
7
8 always @ (x)
9 begin
10
11 a = {x[2:0],7'b0};
12 b = {x[8:0],1'b0};
13 y = a + b;
14
15 end
16 endmodule
17
18 module tb_mad;
19 wire [9:0] y;
20 reg [9:0] x;
21
22
23 initial
24 begin
25 x = 2; #100;
26 x = 3; #100;
27 x = 4; #100;
28 x = 6; #100;
29 x = 7;
30 end
31
32 mad m1 (x,y);
33 endmodule
34
35
36
37
38
Ln: 2 Col: 0
ENG
IN
11:20
20-03-2022


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_mad/y
/tb_mad/x
1100001100
0000000110
0100000100
0110000110
1000001000
1100001100
1110001110
0000000010
0000000011
0000000100
0000000110
0000000111
Now
500 ps
Cursor 1
351 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
351 ps
0 ps to 1 ns
Now: 500 ps Delta: 0
ENG
IN
11:21
20-03-2022

6. A memory decoder having a 16-bit address input *addr* performs the following functions
   (a) It asserts an output *rom_en* for the ROM addresses 0000 to 00FF H
   (b) It asserts an output *ram1_en* for the RAM addresses 0100 to 03FF H
   (c) It asserts an output *ram2_en* for the RAM addresses 0400 to 0FFF H
   (d) It asserts an output *ram3_en* for the RAM addresses 1000 to FFFF H

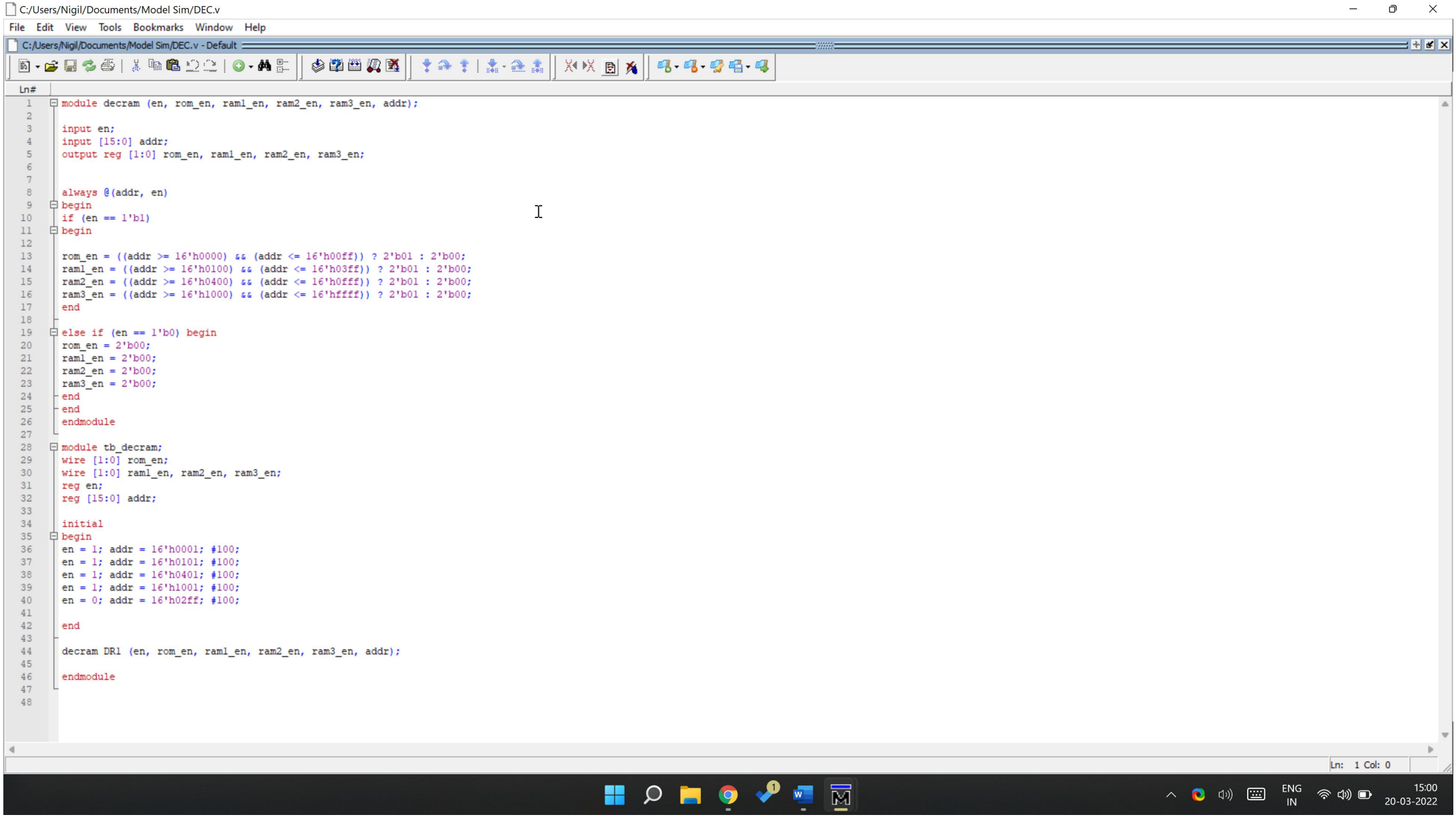
   Write a synthesizable model for the decoder

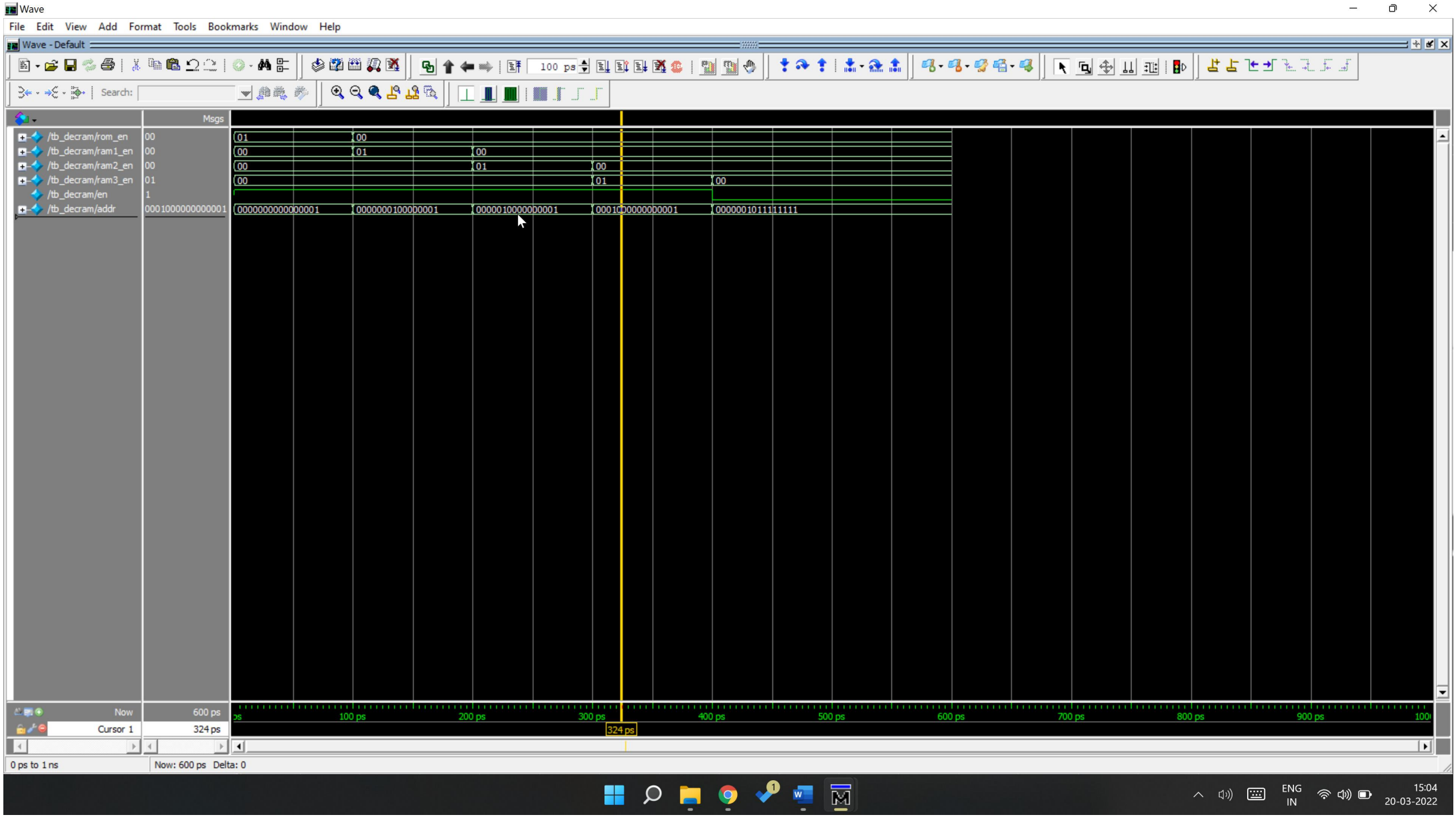
Memory Decoder - Behavioural Model Code

```
module decram (en, rom_en, ram1_en, ram2_en, ram3_en, addr);

input en;
input [15:0] addr;
output reg [1:0] rom_en, ram1_en, ram2_en, ram3_en;

always @(addr, en)
begin
if (en == 1'b1)
begin
rom_en = ((addr >= 16'h0000) && (addr <= 16'h00ff)) ? 2'b01 : 2'b00;
ram1_en = ((addr >= 16'h0100) && (addr <= 16'h03ff)) ? 2'b01 : 2'b00;
ram2_en = ((addr >= 16'h0400) && (addr <= 16'h0fff)) ? 2'b01 : 2'b00;
ram3_en = ((addr >= 16'h1000) && (addr <= 16'hffff)) ? 2'b01 : 2'b00;
end

else if (en == 1'b0) begin
rom_en = 2'b00;
ram1_en = 2'b00;
ram2_en = 2'b00;
ram3_en = 2'b00;
```

```
end
end
endmodule
// TEST BENCH
module tb_decram;
wire [1:0] rom_en;
wire [1:0] ram1_en, ram2_en, ram3_en;
reg en;
reg [15:0] addr;

initial
begin
en = 1; addr = 16'h0001; #100;  // rom_en = 1
en = 1; addr = 16'h0101; #100;  // ram1_en = 1
en = 1; addr = 16'h0401; #100; // ram2_en = 1
en = 1; addr = 16'h1001; #100; // ram3_en = 1
en = 0; addr = 16'h02ff; #100; // All Outputs De-Asserted

end

decram DR1 (en, rom_en, ram1_en, ram2_en, ram3_en, addr);
endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/DEC.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/DEC.v - Default
Ln#
1 module decram (en, rom_en, ram1_en, ram2_en, ram3_en, addr);
2
3 input en;
4 input [15:0] addr;
5 output reg [1:0] rom_en, ram1_en, ram2_en, ram3_en;
6
7
8 always @(addr, en)
9 begin
10 if (en == 1'b1)
11 begin
12
13 rom_en = ((addr >= 16'h0000) && (addr <= 16'h00ff)) ? 2'b01 : 2'b00;
14 ram1_en = ((addr >= 16'h0100) && (addr <= 16'h03ff)) ? 2'b01 : 2'b00;
15 ram2_en = ((addr >= 16'h0400) && (addr <= 16'h0fff)) ? 2'b01 : 2'b00;
16 ram3_en = ((addr >= 16'h1000) && (addr <= 16'hffff)) ? 2'b01 : 2'b00;
17 end
18
19 else if (en == 1'b0) begin
20 rom_en = 2'b00;
21 ram1_en = 2'b00;
22 ram2_en = 2'b00;
23 ram3_en = 2'b00;
24 end
25 end
26 endmodule
27
28 module tb_decram;
29 wire [1:0] rom_en;
30 wire [1:0] ram1_en, ram2_en, ram3_en;
31 reg en;
32 reg [15:0] addr;
33
34 initial
35 begin
36 en = 1; addr = 16'h0001; #100;
37 en = 1; addr = 16'h0101; #100;
38 en = 1; addr = 16'h0401; #100;
39 en = 1; addr = 16'h1001; #100;
40 en = 0; addr = 16'h02ff; #100;
41
42 end
43
44 decram DR1 (en, rom_en, ram1_en, ram2_en, ram3_en, addr);
45
46 endmodule
47
48
Ln: 1 Col: 0
ENG
IN
15:00
20-03-2022


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_decram/rom_en
/tb_decram/ram1_en
/tb_decram/ram2_en
/tb_decram/ram3_en
/tb_decram/en
/tb_decram/addr
00
00
00
01
1
0001000000000001
01
00
00
01
00
00
01
00
00
0000000000000001
0000000100000001
0000010000000001
0000000011111111
Now
600 ps
Cursor 1
324 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
324 ps
0 ps to 1 ns
Now: 600 ps Delta: 0
ENG
IN
15:04
20-03-2022

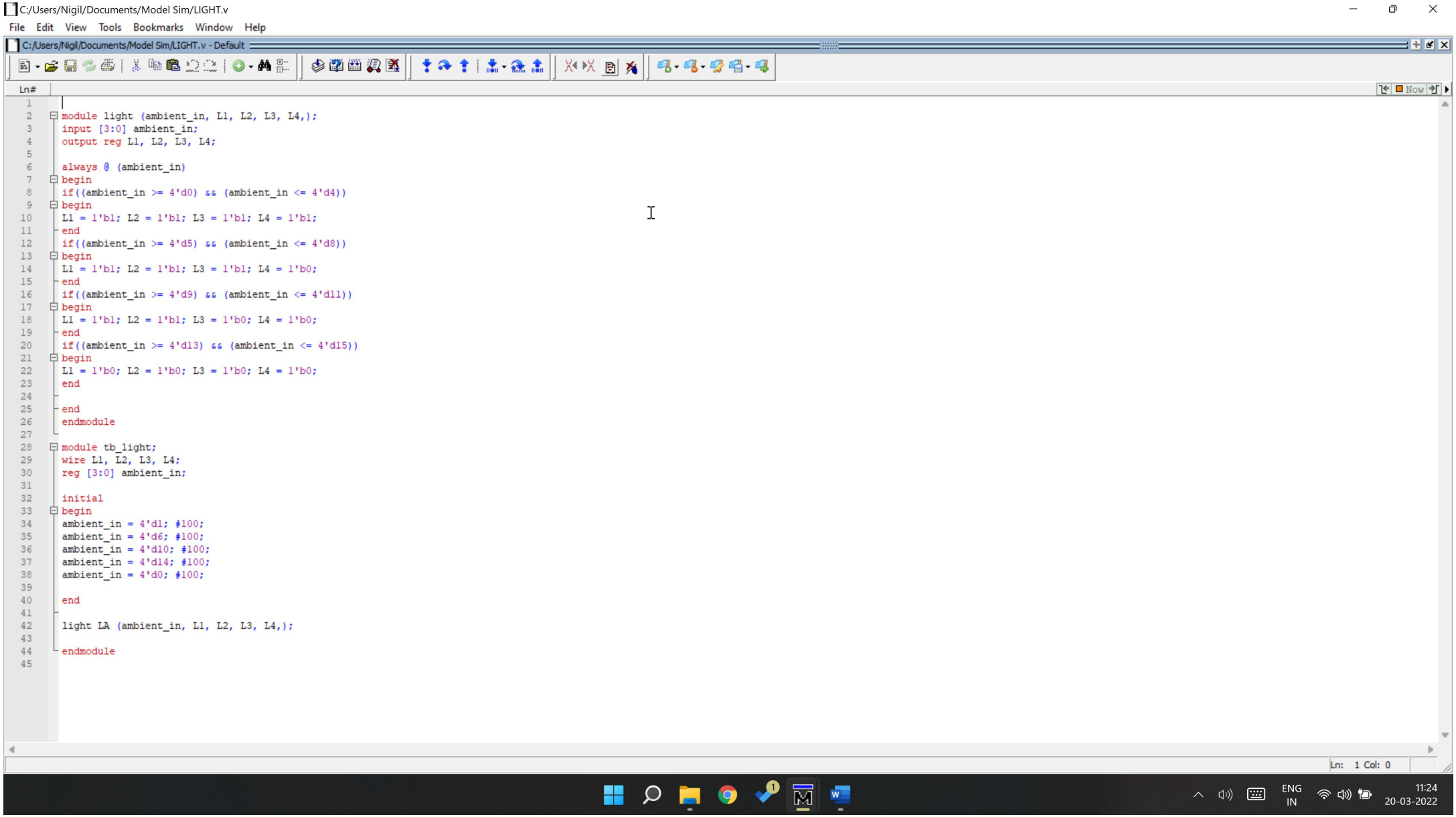
7. A street light controller has a 4-bit input *ambient_in* which provides the ambient illumination in encoded form. It drives four lights viz L1, L2. L3 and L4. The outputs are asserted as follows:
   (a) When *ambient_in* is between 0 and 4 all the outputs are asserted
   (b) When *ambient_in* is between 5 and 8 L1, L2 and L3 are asserted
   (c) When *ambient_in* is between 9 and 12 only L1 and L2 are asserted
   (d) When *ambient_in* is between 13 and 15 all the outputs are de-asserted
   Write a synthesizable behavioural model if the above circuit

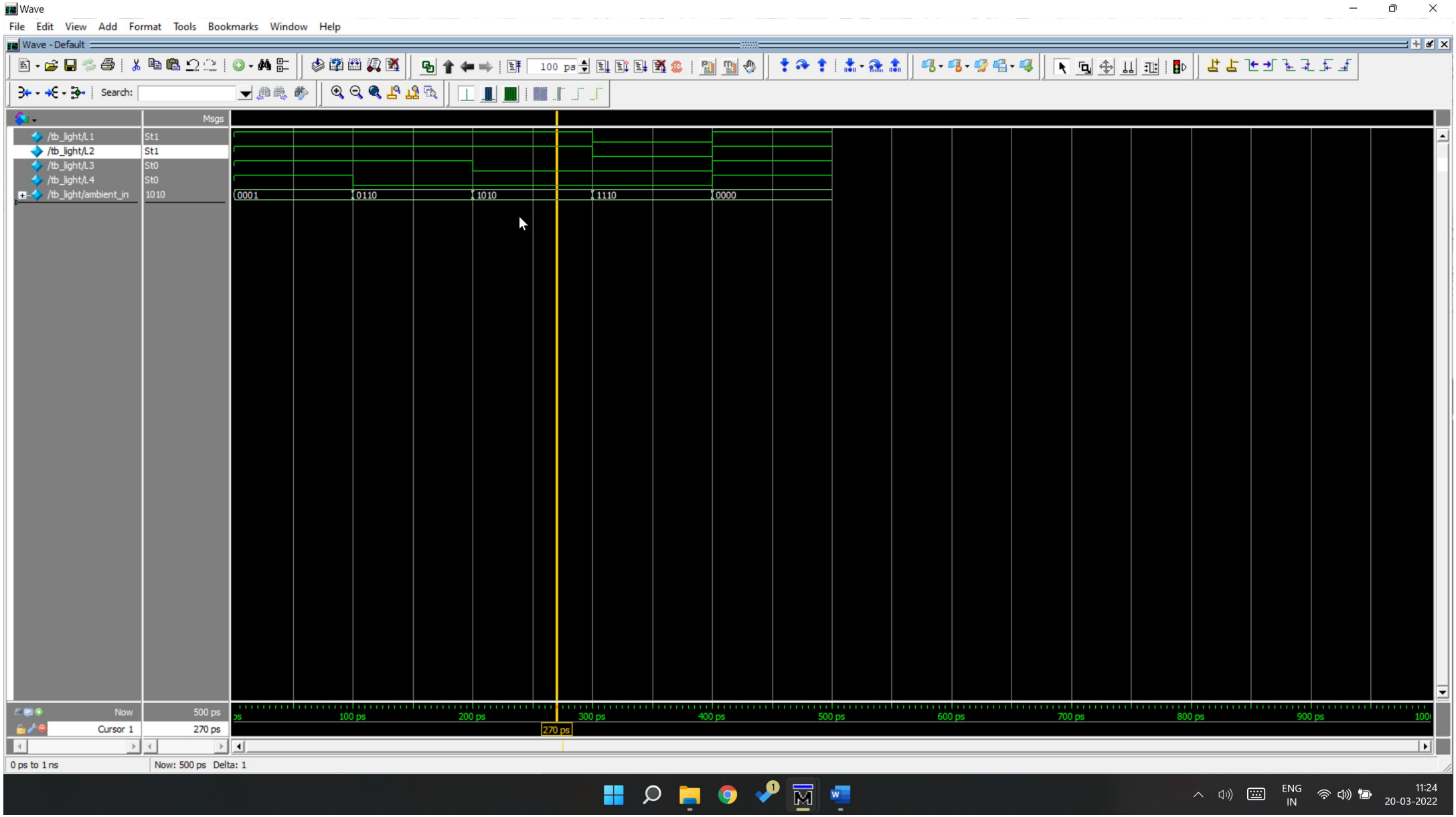
Ambient Light Control - Behavioural Model Code

```
module light (ambient_in, L1, L2, L3, L4,);
input [3:0] ambient_in;
output reg L1, L2, L3, L4;

always @ (ambient_in)
begin
if((ambient_in >= 4'd0) && (ambient_in <= 4'd4))
begin
L1 = 1'b1; L2 = 1'b1; L3 = 1'b1; L4 = 1'b1;
end
if((ambient_in >= 4'd5) && (ambient_in <= 4'd8))
begin
L1 = 1'b1; L2 = 1'b1; L3 = 1'b1; L4 = 1'b0;
end
if((ambient_in >= 4'd9) && (ambient_in <= 4'd11))
begin
L1 = 1'b1; L2 = 1'b1; L3 = 1'b0; L4 = 1'b0;
end
if((ambient_in >= 4'd13) && (ambient_in <= 4'd15))
begin
L1 = 1'b0; L2 = 1'b0; L3 = 1'b0; L4 = 1'b0;
end
```

```
end
endmodule
// TEST BENCH
module tb_light;
wire L1, L2, L3, L4;
reg [3:0] ambient_in;

initial
begin
ambient_in = 4'd1; #100; // L1 = 1, L2 = 1, L3 = 1, L4 = 1
ambient_in = 4'd6; #100; // L1 = 1, L2 = 1, L3 = 1, L4 = 0
ambient_in = 4'd10; #100;  // L1 = 1, L2 = 1, L3 = 0, L4 = 0
ambient_in = 4'd14; #100; // L1 = 0, L2 = 0, L3 = 0, L4 = 0
ambient_in = 4'd0; #100; // L1 = 1, L2 = 1, L3 = 1, L4 = 1

end

light LA (ambient_in, L1, L2, L3, L4,);

endmodule
```

Code & Output Waveform Screenshot


C:/Users/Nigil/Documents/Model Sim/LIGHT.v
File Edit View Tools Bookmarks Window Help
C:/Users/Nigil/Documents/Model Sim/LIGHT.v - Default
Ln#
module light (ambient_in, L1, L2, L3, L4,);
input [3:0] ambient_in;
output reg L1, L2, L3, L4;
always @ (ambient_in)
begin
if((ambient_in >= 4'd0) && (ambient_in <= 4'd4))
begin
L1 = 1'b1; L2 = 1'b1; L3 = 1'b1; L4 = 1'b1;
end
if((ambient_in >= 4'd5) && (ambient_in <= 4'd8))
begin
L1 = 1'b1; L2 = 1'b1; L3 = 1'b1; L4 = 1'b0;
end
if((ambient_in >= 4'd9) && (ambient_in <= 4'd11))
begin
L1 = 1'b1; L2 = 1'b1; L3 = 1'b0; L4 = 1'b0;
end
if((ambient_in >= 4'd13) && (ambient_in <= 4'd15))
begin
L1 = 1'b0; L2 = 1'b0; L3 = 1'b0; L4 = 1'b0;
end
end
endmodule
module tb_light;
wire L1, L2, L3, L4;
reg [3:0] ambient_in;
initial
begin
ambient_in = 4'd1; #100;
ambient_in = 4'd6; #100;
ambient_in = 4'd10; #100;
ambient_in = 4'd14; #100;
ambient_in = 4'd0; #100;
end
light LA (ambient_in, L1, L2, L3, L4,);
endmodule
Ln: 1 Col: 0
ENG
IN
11:24
20-03-2022


Wave
File Edit View Add Format Tools Bookmarks Window Help
Wave - Default
Search:
Msgs
/tb_light/L1
/tb_light/L2
/tb_light/L3
/tb_light/L4
/tb_light/ambient_in
St1
St1
St0
St0
1010
0001
0110
1010
1110
0000
Now
500 ps
Cursor 1
270 ps
100 ps
200 ps
300 ps
400 ps
500 ps
600 ps
700 ps
800 ps
900 ps
270 ps
0 ps to 1 ns
Now: 500 ps Delta: 1
ENG
IN
11:24
20-03-2022