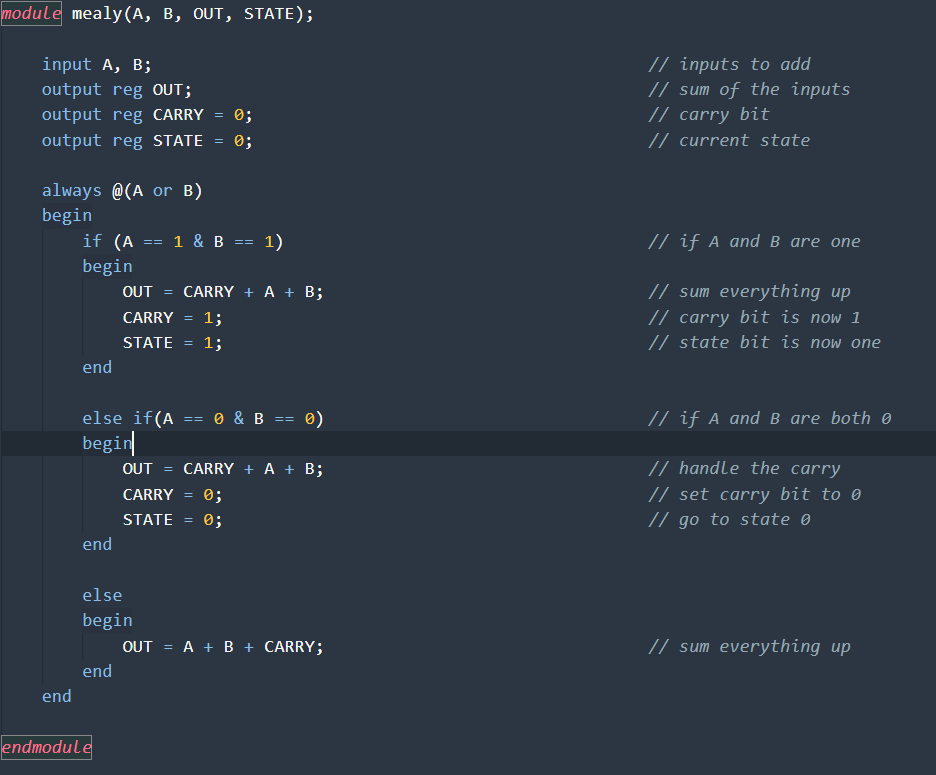
**Simulation 1 Verilog**



**Simulation 1 Test Bench**

`timescale 1s/100ms

`include "mealy.v"

*module* mealy\_tb();

wire OUT, STATE;

reg A, B, clk;

mealy uut

(

    .OUT(OUT),

    .STATE(STATE),

    .A(A),

    .B(B),

.clk(clk)

);

initial

begin

    $dumpfile("dump.vcd");

    $dumpvars(0, mealy\_tb);

    A = 0;                                      *// add 0 and 0*

    B = 0;

    #1;

    A = 1;                                      *// add 1 and 0*

    B = 0;

    #1;

    A = 0;                                      *// add 0 and 1*

    B = 1;

    #1;

    A = 1;                                      *// test transition to carry state*

    B = 1;

    #1;

    A = 1;                                      *// test staying in carry state*

    B = 0;

    #1;

    A = 1;

    B = 1;

    #1;

    A = 0;                                      *// test exiting from the carry state*

    B = 0;

    #1;

    A = 1;

    B = 0;

    #1;

    A = 0;

    B = 0;

    #1;

End

always

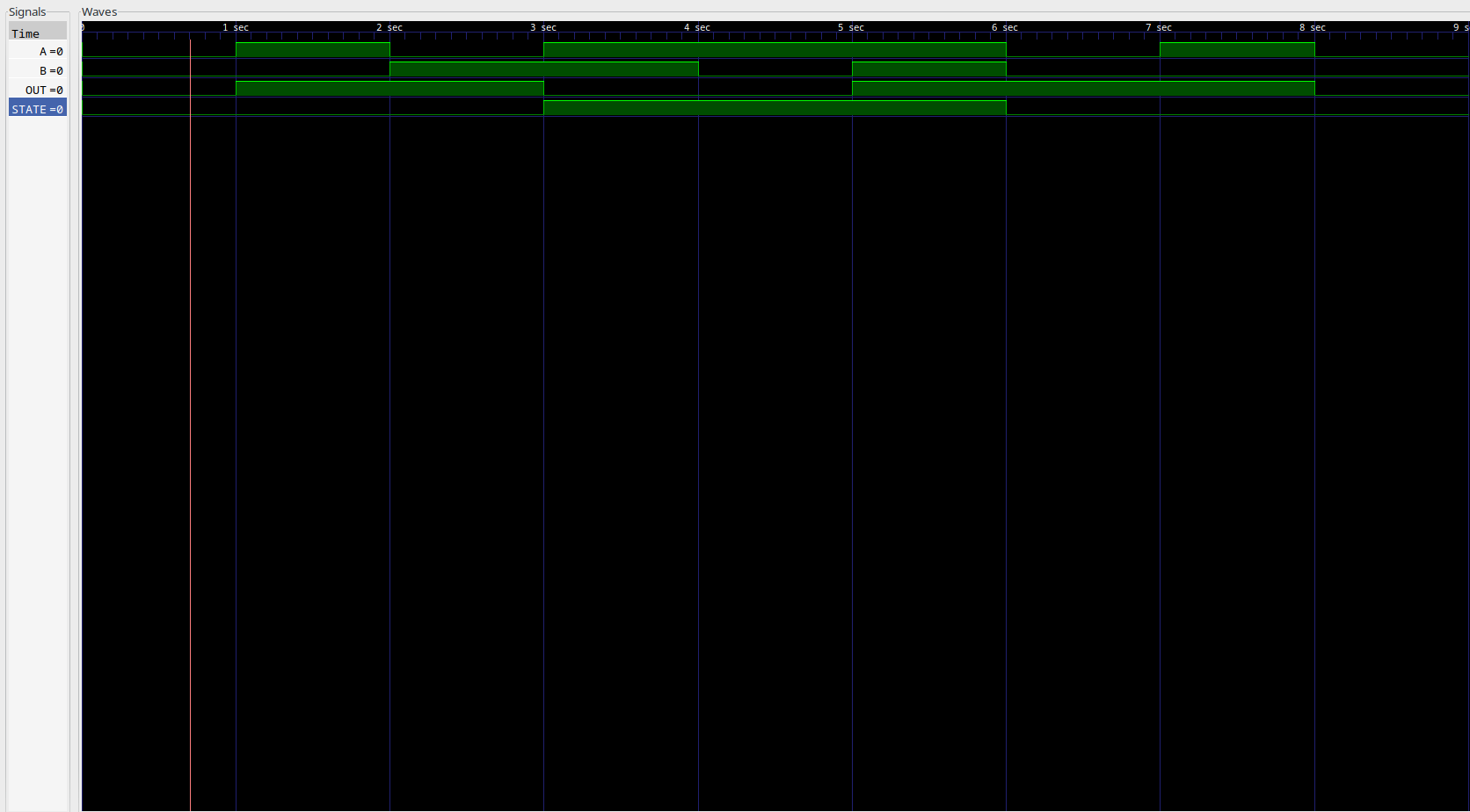
    #5  clk = ! clk;

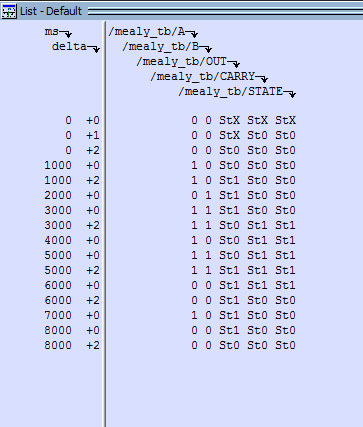
initial

    #100  $finish;

*endmodule*

**Simulation 1 Results**



****

**Simulation 2 Verilog**

*module* struct(OUT, A, B, clk);

  input A, B, clk;

  output OUT;

  wire cout, qt\_bar;

  FullAdder   C1(OUT, cout, A, B, qt);

  d\_ff        C2(qt, qt\_bar, cout, clk);

*endmodule*

*module* FullAdder(SUM, C\_OUT, A, B, C\_IN);

  output SUM, C\_OUT;

  input A, B, C\_IN;

  wire w1, w2, w3;

  xor   (SUM, A, B);

  and   (w2, A, B);

  and   (w1, w2, C\_IN);

  and   (w3, w2, A);

  or    (w2, w2, w3);

*endmodule*

*module* d\_ff(output qt, output qt\_bar, input d, input clk);

  wire qt, qt\_bar;

  wire d, clk;

  wire w1, w2, w3;

  nand    (w1, d, clk);

  not     (w2, d);

  nand    (w3, w2, clk);

  nand    (qt, w1, qt\_bar);

  nand    (qt\_bar, qt, w3);

*endmodule*

**Simulation 2 Testbench**

`timescale 1ns/100 ps

`include "struct.v"

*module* struct\_tb();

wire OUT;

reg A, B, clk = 0;

struct uut

(

    .OUT(OUT),

    .A(A),

    .B(B),

    .clk(clk)

);

initial

begin

    $dumpfile("dump.vcd");

    $dumpvars(0, struct\_tb);

    A = 0;                                      *// add 0 and 0*

    B = 0;

    #10;

    A = 1;                                      *// add 1 and 0*

    B = 0;

    #10;

    A = 0;                                      *// add 0 and 1*

    B = 1;

    #10;

    A = 1;                                      *// test transition to carry state*

    B = 1;

    #10;

    A = 1;                                      *// test staying in carry state*

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 0;                                      *// test exiting from the carry state*

    B = 0;

    #10;

    A = 1;

    B = 0;

    #10;

    A = 0;

    B = 0;

    #10;

end

always

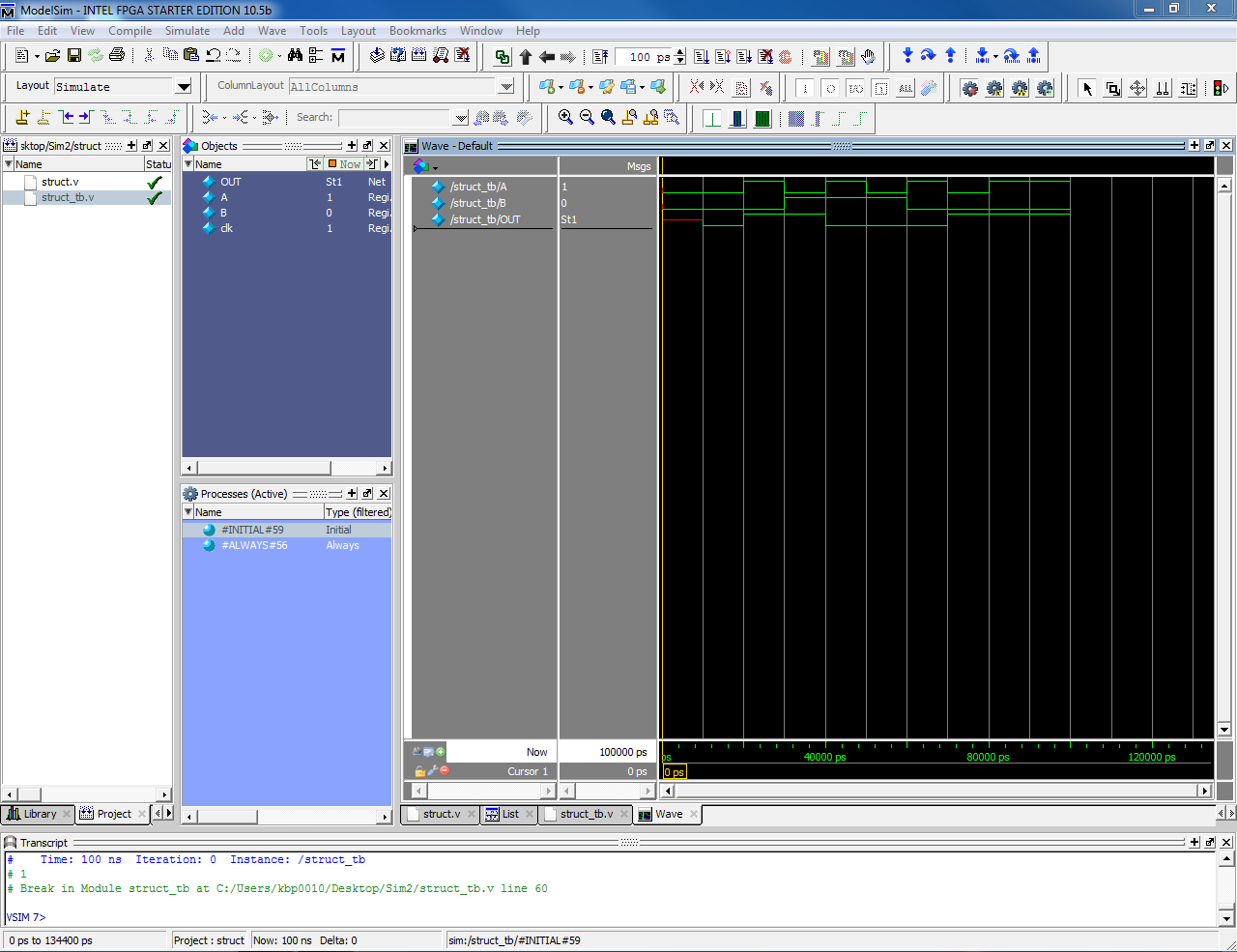
    #5  clk = ! clk;

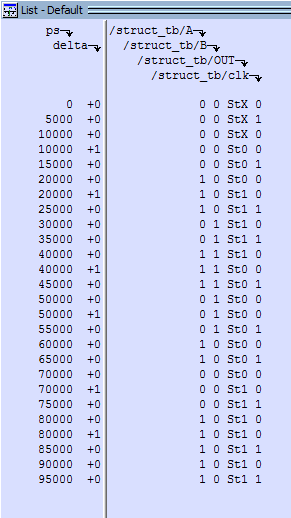
initial

    #100  $finish;

*endmodule*

**Simulation 2 Results**





**Simulation 3 Verilog**

*module* moore(A, B, OUT, STATE, clk);

    input       A, B, clk;                                  *// inputs to add*

    output reg  OUT;                                        *// sum of the inputs*

    parameter   CARRY = 0;                                  *// carry bit*

    output reg  [1:0] STATE = 0;                            *// current state*

    always @(posedge clk)

    begin

        if(STATE == 0)                                      *// in state 0*

        begin

            if(A ^ B)                                       *// if one of the inputs is one*

            begin

                STATE = 1;                                  *// transition to state 1*

            end

            else if(A & B)                                  *// if both inputs are 1*

            begin

                STATE = 2;                                  *// transition to state 2*

            end

        end

        else if (STATE == 1)

        begin

            if(!A & !B)                                     *// both inputs are 0*

            begin

                STATE = 0;

            end

            else if(A & B)                                  *// both inputs are 1*

            begin

                STATE = 2;                                  *// transition to state 2*

            end

        end

        else if (STATE == 2)                                *// if the current state is 2*

        begin

            if(!A & !B)                                     *// both inputs are 0*

            begin

                STATE = 1;                                  *// tranistion to state 1*

            end

            else if(A & B)                                  *// both inputs are 1*

            begin

                STATE = 3;                                  *// tranition to state 3*

            end

        end

        else if (STATE == 3)                                *// current state is 3*

        begin

            if(!A & !B)                                     *// if both inputs are 0*

            begin

                STATE = 1;                                  *// go to state 1*

            end

            if(A ^ B)                                       *// if one of the inputs is 1*

            begin

                STATE = 2;                                  *// go to state 2*

            end

        end

        if(STATE == 0)

            OUT = 0;

        else if(STATE == 1)

            OUT = 1;

        else if(STATE == 2)

            OUT = 0;

        else if(STATE == 3)

            OUT = 1;

    end

*endmodule*

**Simulation 3 Testbench**

`timescale 1ns/100 ps

`include "moore.v"

*module* moore\_tb();

wire OUT;

wire [1:0] STATE;

reg A, B, clk = 0;

moore uut

(

    .OUT(OUT),

    .STATE(STATE),

    .A(A),

    .B(B),

    .clk(clk)

);

initial

begin

    $dumpfile("dump.vcd");

    $dumpvars(0, moore\_tb);

    A = 0;

    B = 0;

    #10;

    A = 0;

    B = 1;

    #10;

    A = 1;

    B = 0;

    #10;

    A = 0;

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 0;

    B = 1;

    #10;

    A = 0;

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 1;

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 0;

    B = 0;

    #10;

end

always

    #5  clk = ! clk;

initial

    #150  $finish;

*endmodule*

**Simulation 3 Testbench**

`timescale 1ns/100 ps

`include "moore.v"

*module* moore\_tb();

wire OUT;

wire [1:0] STATE;

reg A, B, clk = 0;

moore uut

(

    .OUT(OUT),

    .STATE(STATE),

    .A(A),

    .B(B),

    .clk(clk)

);

initial

begin

    $dumpfile("dump.vcd");

    $dumpvars(0, moore\_tb);

    A = 0;

    B = 0;

    #10;

    A = 0;

    B = 1;

    #10;

    A = 1;

    B = 0;

    #10;

    A = 0;

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 0;

    B = 1;

    #10;

    A = 0;

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 1;

    B = 0;

    #10;

    A = 1;

    B = 1;

    #10;

    A = 0;

    B = 0;

    #10;

end

always

    #5  clk = ! clk;

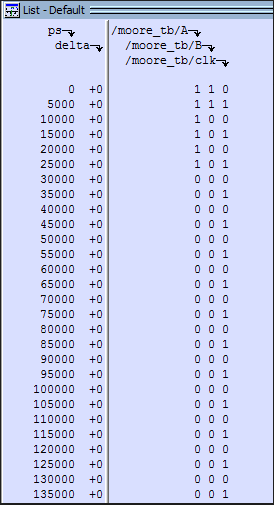
initial

    #150  $finish;

*endmodule*

**Simulation 3 Results**



****