CLOCK DIVIDER USING COUNTER

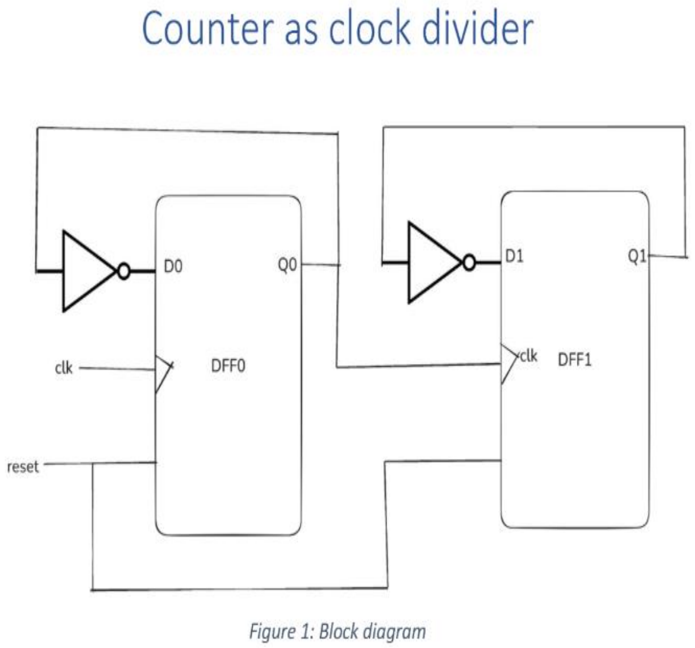
Verilog code:

```verilog
module clock_div (
  input wire clk,
  input wire rst,
  output reg divided_clk
);

  reg [2:0] counter;

  always @(posedge clk or posedge rst) begin
    if (rst) begin
      counter <= 3'b000;
      divided_clk <= 1'b0;
    end else begin
      if (counter == 3'b111) begin
        counter <= 3'b000;
        divided_clk <= ~divided_clk;
      end else begin
        counter <= counter + 1;
      end
    end
  end

endmodule
```

TESTBENCH:

```
module clock_div_tb

  reg clk;

   reg rst;

   wire divided_clk;

clock_div uut (

       .clk(clk),

       .rst(rst),

       .divided_clk(divided_clk)

   );

 initial begin

       clk = 0;

       forever #5 clk = ~clk;

   end



   initial begin


       rst = 1;

       #15;

       rst = 0;

#250

 $finish;

   end

initial begin

       $display("Time\tclk\trst\tdivided_clk");

       $monitor("%0dns\t%b\t%b\t%b", $time, clk, rst, divided_clk);

   end  endmodule
```

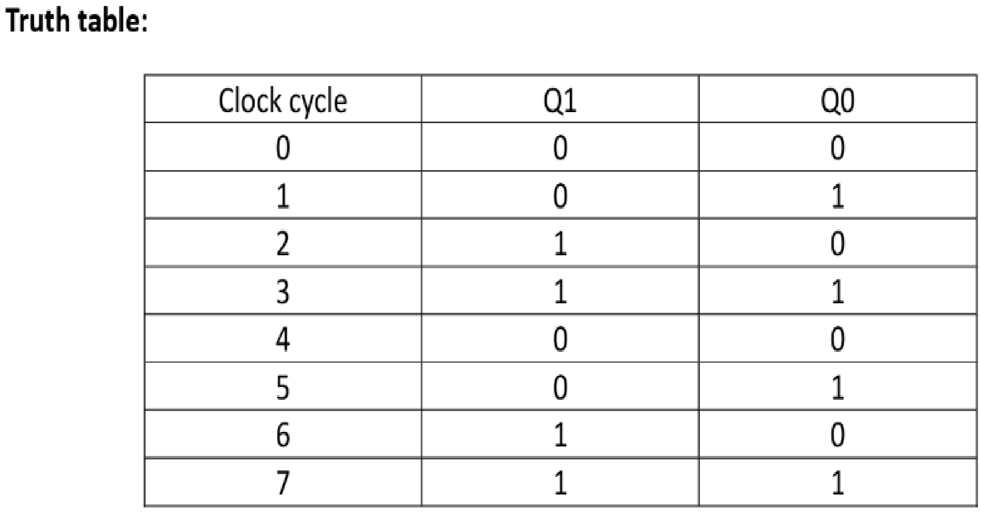
BLOCK DIAGRAM:



Counter as clock divider

Figure 1: Block diagram

Figure 1 BLOCK DIAGRAM



Truth table:

Clock cycle	Q1	Q0
0	0	0
1	0	1
2	1	0
3	1	1
4	0	0
5	0	1
6	1	0
7	1	1

Figure 2 TRUTTH TABLE

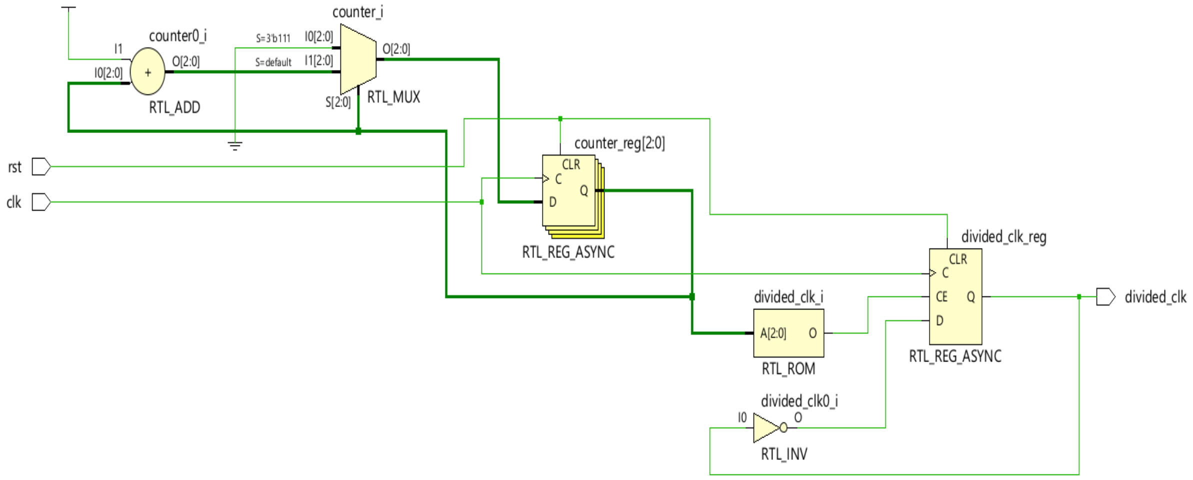


Figure 3 RTL SCHEMATIC

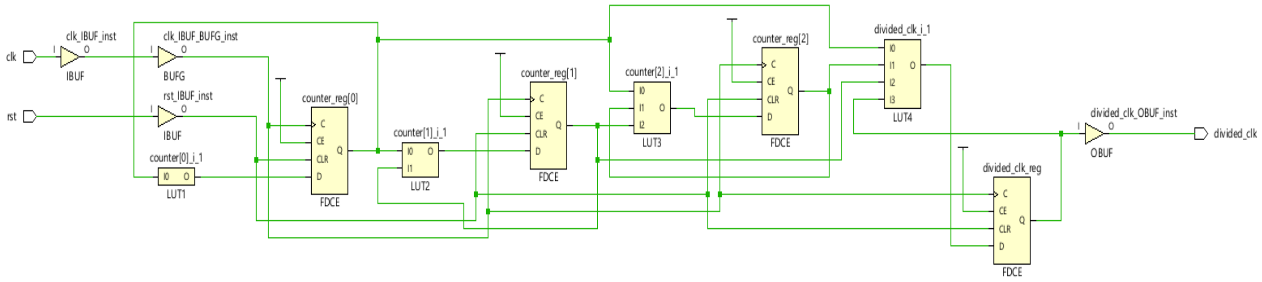


Figure 4 synthesis schematic

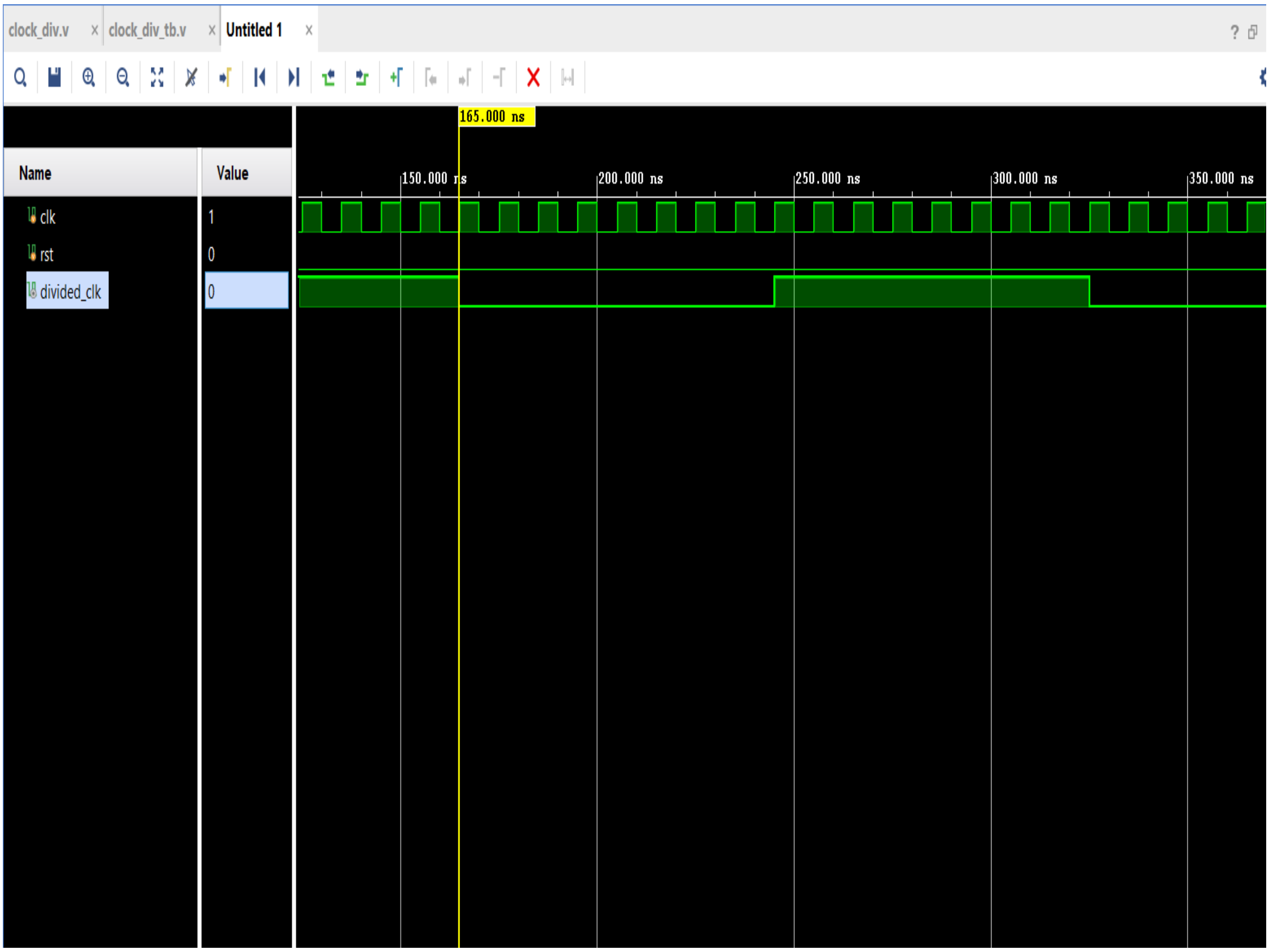


Figure 5 waveforms