30 Days of RTL Coding

-By T Tushar Shenoy

Day 4

Problem Statement: Implementing 0 to 255 Up/Down Counter based on the control Signal "ud" in Behavioural Style.

Theory:

A **Counter** is a device which stores (and sometimes displays) the number of times a particular event or process has occurred, often in relationship to a clock signal. Counters are used in digital electronics for counting purpose, they can count specific event happening in the circuit. For example, in UP counter a counter increases count for every rising edge of clock. Not only counting, a counter can follow the certain sequence based on our design like any random sequence 0,1,3,2… They can also be designed with the help of flip flops. They are used as frequency dividers where the frequency of given pulse waveform is divided. Counters are sequential circuit that count the number of pulses can be either in binary code or BCD form. The main properties of a counter are timing, sequencing, and counting. Counter works in two modes up counter and down counter

UP/DOWN Counter

Up counter and down counter is combined together to obtain an UP/DOWN counter. A mode control (ud) input is also provided to select either up or down mode. A combinational circuit is required to be designed and used between each pair of flip-flop in order to achieve the up/down operation.

Verilog Code:

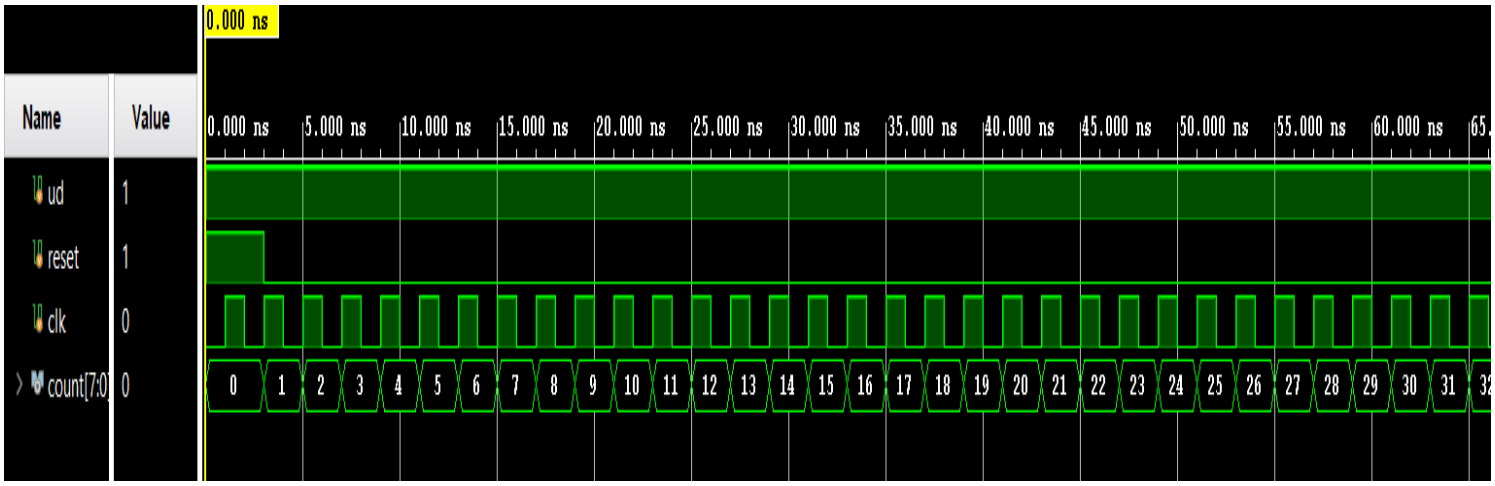
```
module Counter_up_down(ud,reset,clk,count);

input ud,reset,clk;
output reg[7:0]count;

always @(posedge clk,posedge reset)
  begin
    if(reset)
      begin
        count=1'b0;
      end
    else if(ud)
      begin
      count=count+1;
      end
    else
      count=count-1;
  end
endmodule
```

Testbench Code:

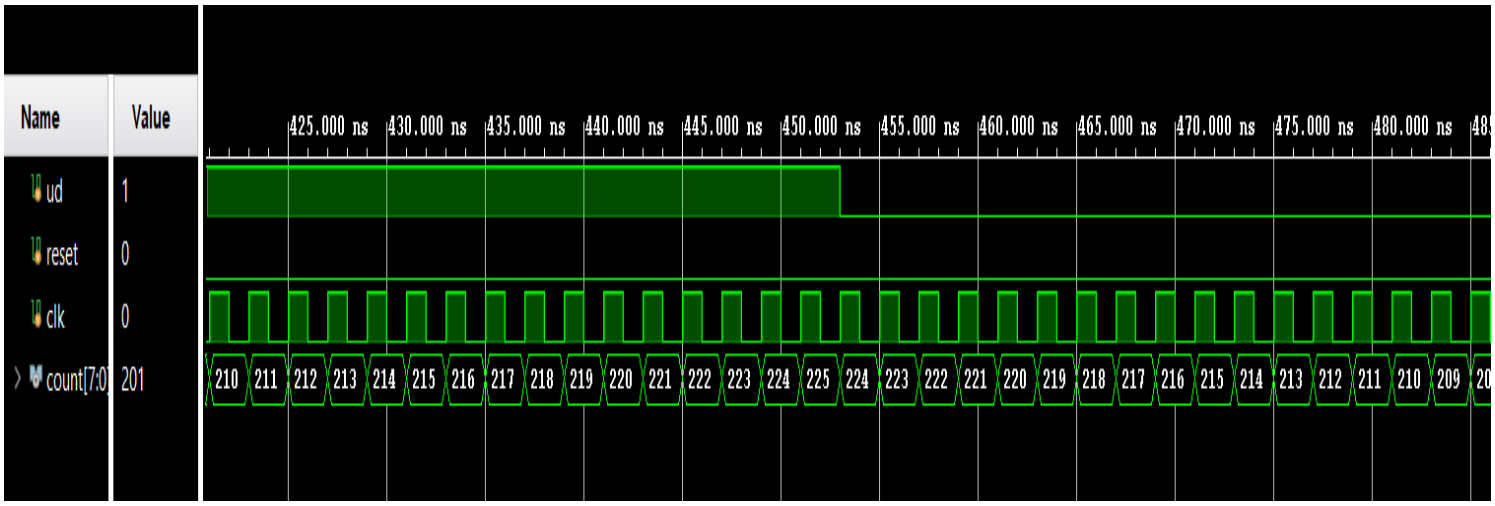
```
module Counter_up_down_tb();

reg ud,reset,clk;
wire [7:0]count;

Counter_up_down dut(.ud(ud),.reset(reset),.clk(clk),.count(count));

initial begin
ud=1'b1; //Up Counting
clk=1'b0;
reset=1'b1;
#3 reset=1'b0;
#450 ud=1'b0; //Down Counting
#450 $finish;
end
//The Above Code can be manipulated to obtain Up count from 0 to 255 and Down count from 255 to 0
always #1 clk=~clk;
endmodule
```

Simulation Output:

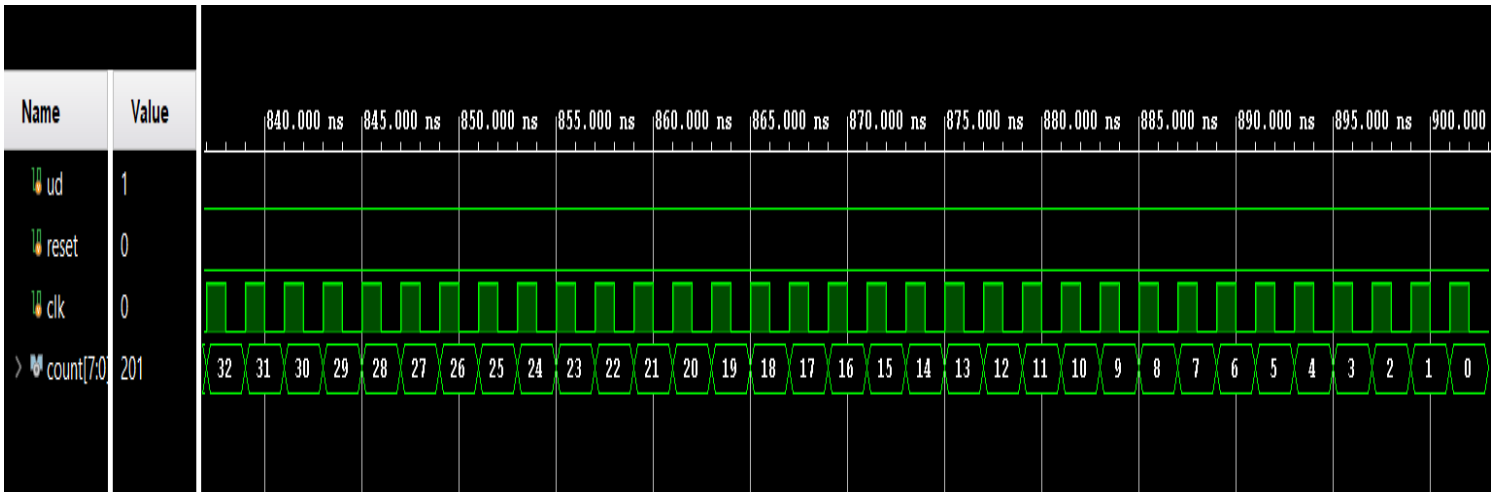
- **UP** Counting Sequence, ud=1



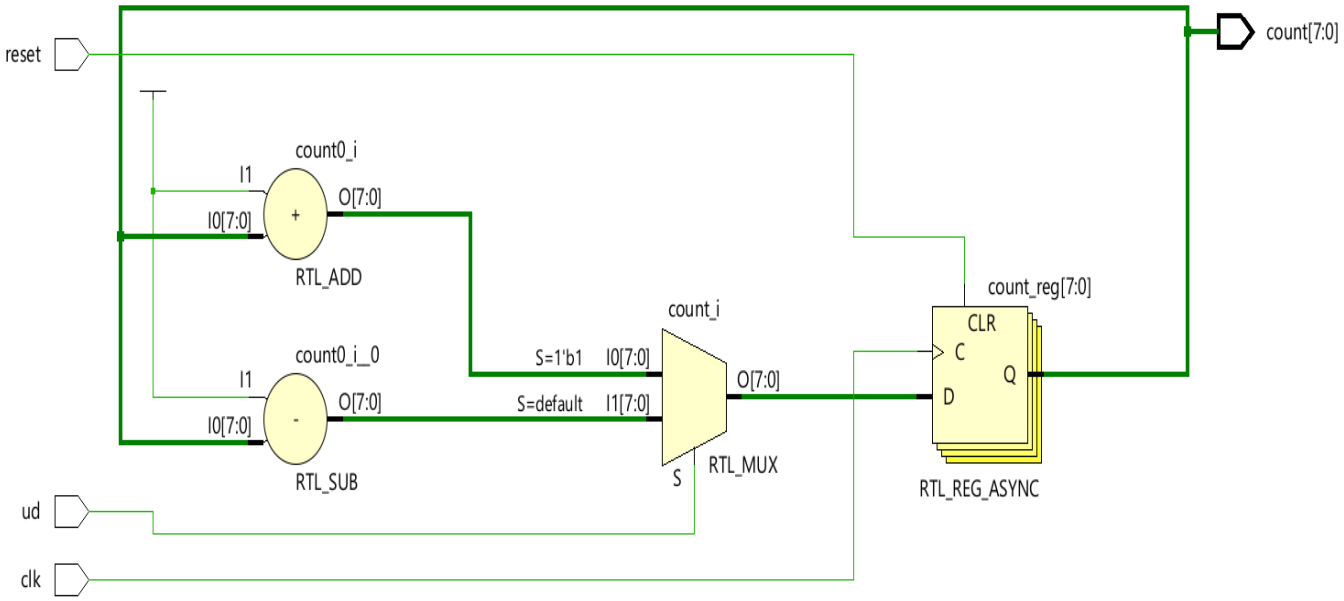
- Control signal ud changes to 0,Down Counting starts



- **DOWN** Counting Sequence, ud=0



Schematic:



GitHub Repository URL: - https://github.com/tusharshenoy/RTL-Day-4-Up-Down-Counter