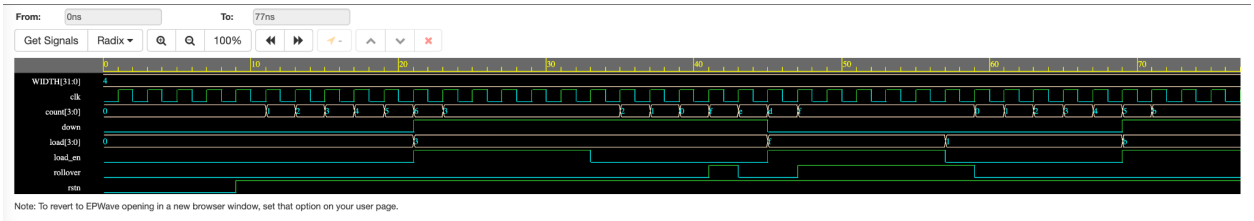
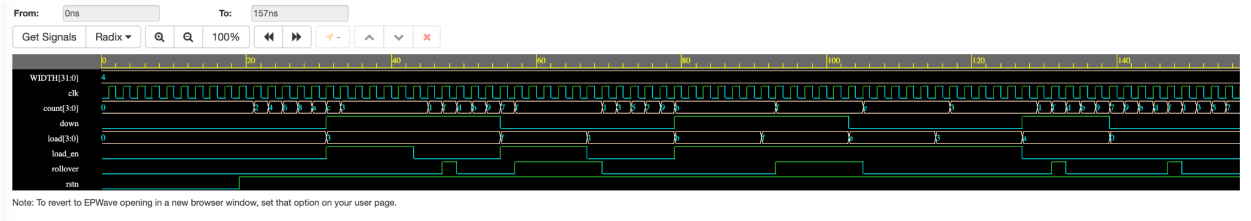
David Strube - dstrube3@gatech.edu
OMCS6603 - VIP - Lab 1

Here is a screenshot of the waveform viewer before I made any modifications



Here is a screenshot of the waveform viewer after I made some modifications



Here are the edits I made:
In the Verilog code, I changed this:

```
            if (down)
            count <= count - 1;
            else
            count <= count + 1;
```

to this:

```
            if (down)
            count <= count - 2;
            else
            count <= count + 2;
```

In the Testbench code, I changed the lines specifying the repeat count from 5 to 10. For example:

```
      repeat (10) @(posedge clk);
```

&

```
      for (int i = 0; i < 10; i++) begin
```

See below for the complete Verliog code and Testbench code:

design.sv:

```
module counter_ud
  #(parameter WIDTH = 4)
  (
```

```
  input                           clk,
  input                           rstn,
  input wire [WIDTH-1:0]   load,
  input                           load_en,
  input                           down,
  output                          rollover,
  output reg [WIDTH-1:0]   count
);


  always @ (posedge clk or negedge rstn) begin
    if (!rstn)
           count <= 0;
    else
      if (load_en)
        count <= load;
      else begin
           if (down)
           count <= count - 2;
           else
           count <= count + 2;
      end
  end

  assign rollover = &count;

endmodule
```

testbench.sv:

```
//------------------------------------------------------------------
// Author  : Admin, ChipVerify
// Website : www.chipverify.com
//------------------------------------------------------------------

interface cnt_if #(parameter WIDTH = 4) (input bit clk);
  logic               rstn;
  logic               load_en;
  logic [WIDTH-1:0] load;
  logic [WIDTH-1:0] count;
  logic               down;
  logic               rollover;
```

```
endinterface

module tb;
  reg clk;

  // TB Clock Generator used to provide the design
  // with a clock -> here half_period = 10ns => 50 MHz
  always #1 clk = ~clk;

  cnt_if     cnt_if0 (clk);
  counter_ud  c0 (    .clk       (cnt_if0.clk),
                      .rstn           (cnt_if0.rstn),
                      .load           (cnt_if0.load),
                      .load_en   (cnt_if0.load_en),
                      .down           (cnt_if0.down),
                      .rollover  (cnt_if0.rollover),
                      .count          (cnt_if0.count));

  initial begin
    bit load_en, down;
    bit [3:0] load;

    $monitor("[%0t] down=%0b load_en=%0b load=0x%0h count=0x%0h
rollover=%0b", $time, cnt_if0.down, cnt_if0.load_en, cnt_if0.load,
cnt_if0.count, cnt_if0.rollover);

    // Initialize testbench variables
    clk <= 0;
    cnt_if0.rstn <= 0;
    cnt_if0.load_en <= 0;
    cnt_if0.load <= 0;
    cnt_if0.down <= 0;

    // Drive design out of reset after 5 clocks
    repeat (10) @(posedge clk);
    cnt_if0.rstn <= 1;

    // Drive stimulus -> repeat 5 times
    for (int i = 0; i < 10; i++) begin
```

```
      // Drive inputs after some random delay
      int delay = $urandom_range (1,30);
      #(delay);

      // Randomize input values to be driven
      std::randomize(load, load_en, down);

      // Assign tb values to interface signals
      cnt_if0.load <= load;
      cnt_if0.load_en <= load_en;
      cnt_if0.down <= down;
    end

    // Wait for 5 clocks and finish simulation
    repeat(10) @ (posedge clk);
    $finish;
  end

  initial begin
    $dumpvars;
    $dumpfile("sth.vcd");
  end
endmodule

/*
Simulation Log:
---------------
ncsim> run
[0] down=0 load_en=0 load=0x0 count=0x0 rollover=0
[96] down=1 load_en=1 load=0x1 count=0x0 rollover=0
[102] down=0 load_en=0 load=0x9 count=0x0 rollover=0
[108] down=1 load_en=1 load=0x1 count=0x0 rollover=0
[110] down=1 load_en=1 load=0x1 count=0x1 rollover=0
[114] down=1 load_en=0 load=0xc count=0x1 rollover=0
[120] down=1 load_en=0 load=0x7 count=0x1 rollover=0
[130] down=1 load_en=0 load=0x7 count=0x0 rollover=0
[150] down=1 load_en=0 load=0x7 count=0xf rollover=1
[170] down=1 load_en=0 load=0x7 count=0xe rollover=0
[190] down=1 load_en=0 load=0x7 count=0xd rollover=0
Simulation complete via $finish(1) at time 210 NS + 0
```

```
./testbench.sv:66     $finish;
ncsim> exit
*/
```