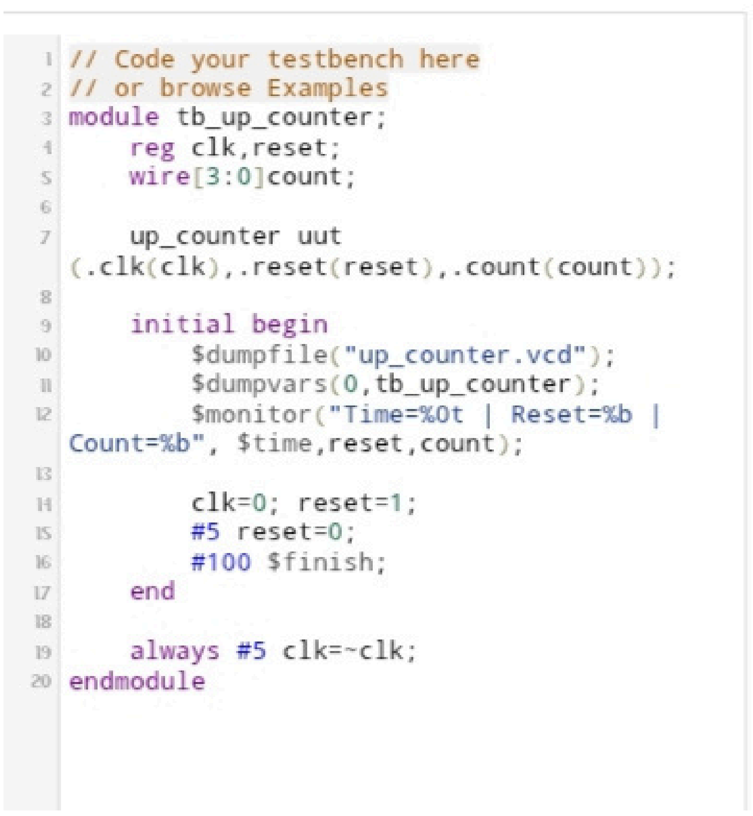
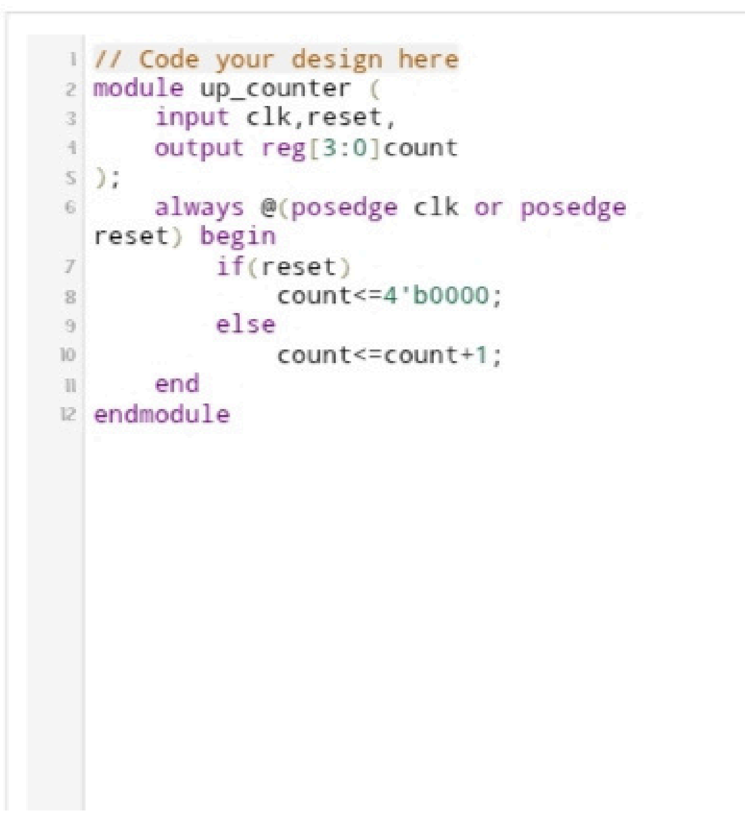
4-Bit Up counter



```
// Code your testbench here
// or browse Examples
module tb_up_counter;
    reg clk,reset;
    wire[3:0]count;

    up_counter uut (.clk(clk),.reset(reset),.count(count));

    initial begin
        $dumpfile("up_counter.vcd");
        $dumpvars(0,tb_up_counter);
        $monitor("Time=%0t | Reset=%b | Count=%b", $time,reset,count);

        clk=0; reset=1;
        #5 reset=0;
        #100 $finish;
    end

    always #5 clk=~clk;
endmodule
```

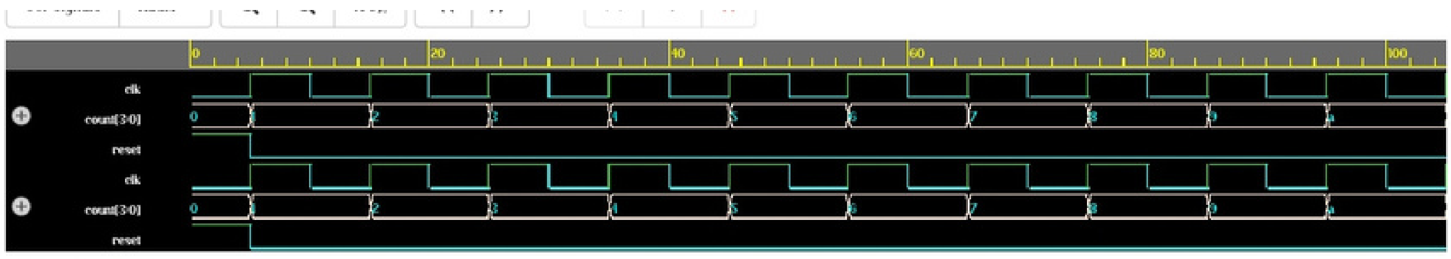


```
// Code your design here
module up_counter (
    input clk,reset,
    output reg[3:0]count
);
    always @(posedge clk or posedge reset) begin
        if(reset)
            count<=4'b0000;
        else
            count<=count+1;
    end
endmodule
```

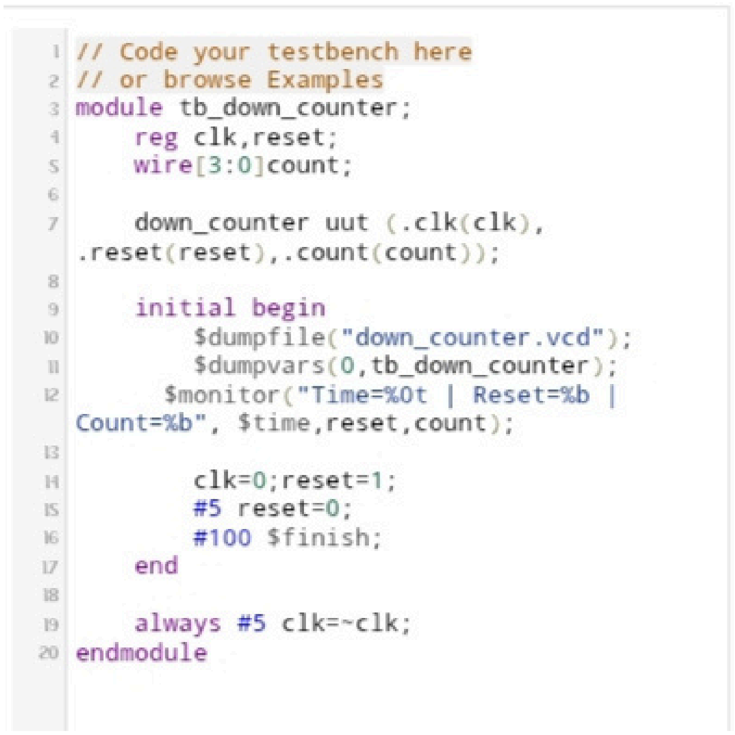
Output:

```
Time=0 | Reset=1 | Count=0000
Time=5 | Reset=0 | Count=0001
Time=15 | Reset=0 | Count=0010
Time=25 | Reset=0 | Count=0011
Time=35 | Reset=0 | Count=0100
Time=45 | Reset=0 | Count=0101
Time=55 | Reset=0 | Count=0110
Time=65 | Reset=0 | Count=0111
Time=75 | Reset=0 | Count=1000
Time=85 | Reset=0 | Count=1001
Time=95 | Reset=0 | Count=1010
```

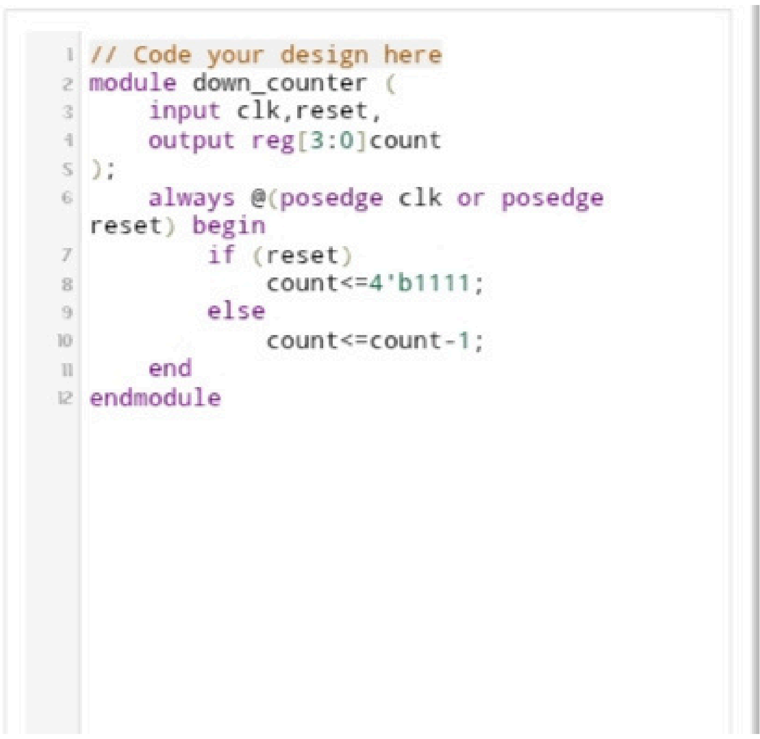
Waveform:



4-Bit Down counter



```
// Code your testbench here
// or browse Examples
module tb_down_counter;
    reg clk,reset;
    wire[3:0]count;

    down_counter uut (.clk(clk),
.reset(reset),.count(count));

    initial begin
        $dumpfile("down_counter.vcd");
        $dumpvars(0,tb_down_counter);
      $monitor("Time=%0t | Reset=%b |
Count=%b", $time,reset,count);

        clk=0;reset=1;
        #5 reset=0;
        #100 $finish;
    end

    always #5 clk=~clk;
endmodule
```

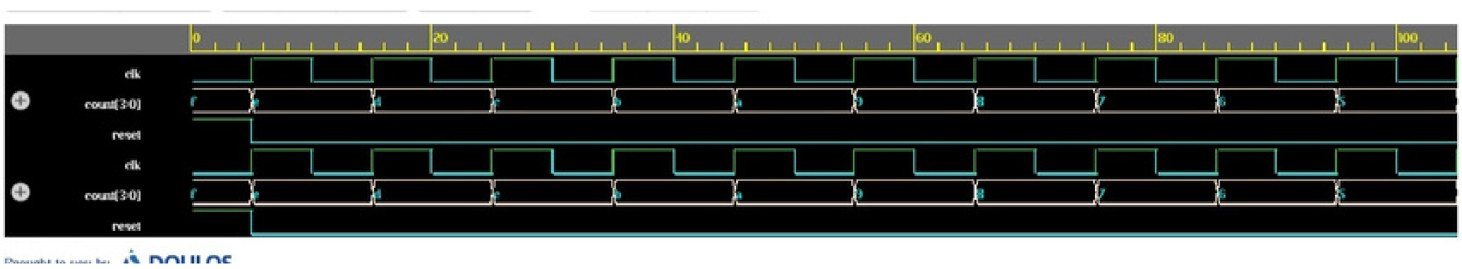


```
// Code your design here
module down_counter (
    input clk,reset,
    output reg[3:0]count
);
    always @(posedge clk or posedge
reset) begin
        if (reset)
            count<=4'b1111;
        else
            count<=count-1;
    end
endmodule
```

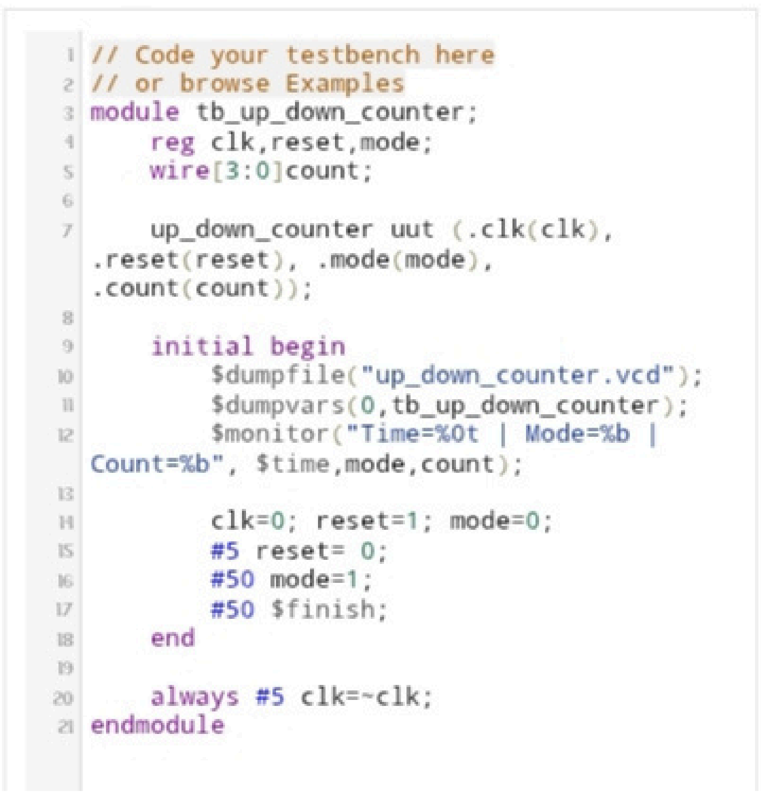
Output:

```
Time=0 | Reset=1 | Count=1111
Time=5 | Reset=0 | Count=1110
Time=15 | Reset=0 | Count=1101
Time=25 | Reset=0 | Count=1100
Time=35 | Reset=0 | Count=1011
Time=45 | Reset=0 | Count=1010
Time=55 | Reset=0 | Count=1001
Time=65 | Reset=0 | Count=1000
Time=75 | Reset=0 | Count=0111
Time=85 | Reset=0 | Count=0110
Time=95 | Reset=0 | Count=0101
```

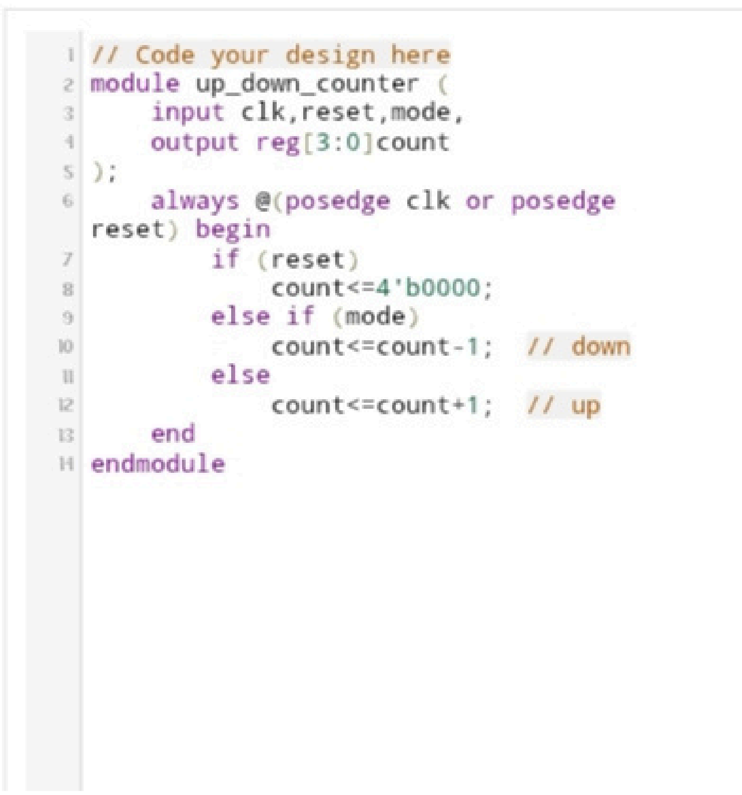
Waveform:



Up/down counter



```
// Code your testbench here
// or browse Examples
module tb_up_down_counter;
    reg clk,reset,mode;
    wire[3:0]count;

    up_down_counter uut (.clk(clk), .reset(reset), .mode(mode), .count(count));

    initial begin
        $dumpfile("up_down_counter.vcd");
        $dumpvars(0,tb_up_down_counter);
        $monitor("Time=%0t | Mode=%b | Count=%b", $time,mode,count);

        clk=0; reset=1; mode=0;
        #5 reset= 0;
        #50 mode=1;
        #50 $finish;
    end

    always #5 clk=~clk;
endmodule
```

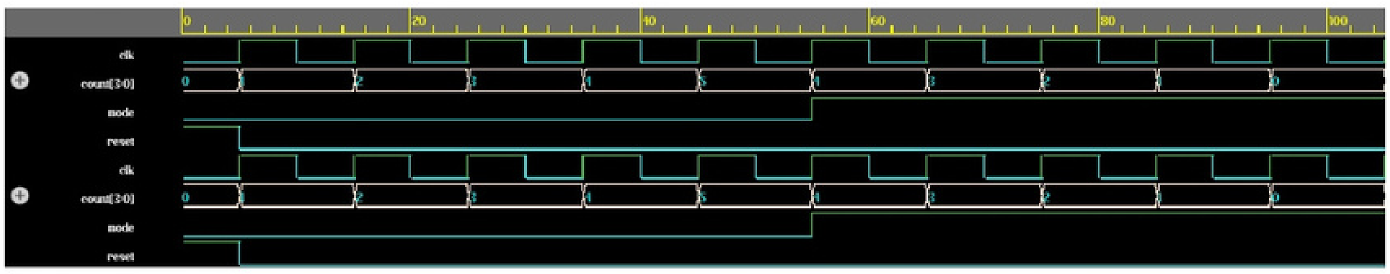


```
// Code your design here
module up_down_counter (
    input clk,reset,mode,
    output reg[3:0]count
);
    always @(posedge clk or posedge reset) begin
        if (reset)
            count<=4'b0000;
        else if (mode)
            count<=count-1;   // down
        else
            count<=count+1;   // up
    end
endmodule
```

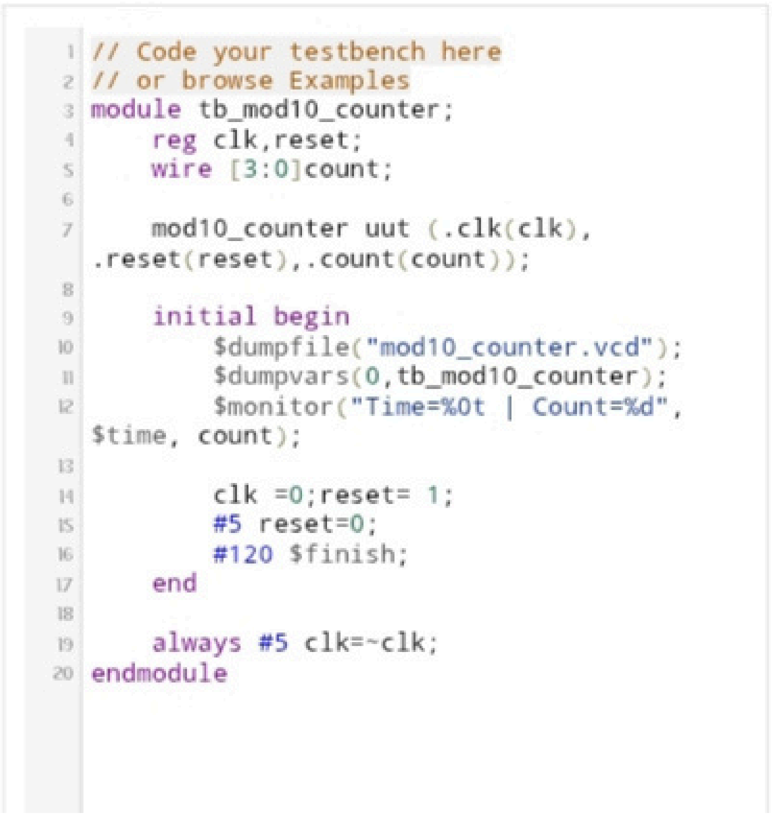
Output:

```
Time=0 | Mode=0 | Count=0000
Time=5 | Mode=0 | Count=0001
Time=15 | Mode=0 | Count=0010
Time=25 | Mode=0 | Count=0011
Time=35 | Mode=0 | Count=0100
Time=45 | Mode=0 | Count=0101
Time=55 | Mode=1 | Count=0100
Time=65 | Mode=1 | Count=0011
Time=75 | Mode=1 | Count=0010
Time=85 | Mode=1 | Count=0001
Time=95 | Mode=1 | Count=0000
```

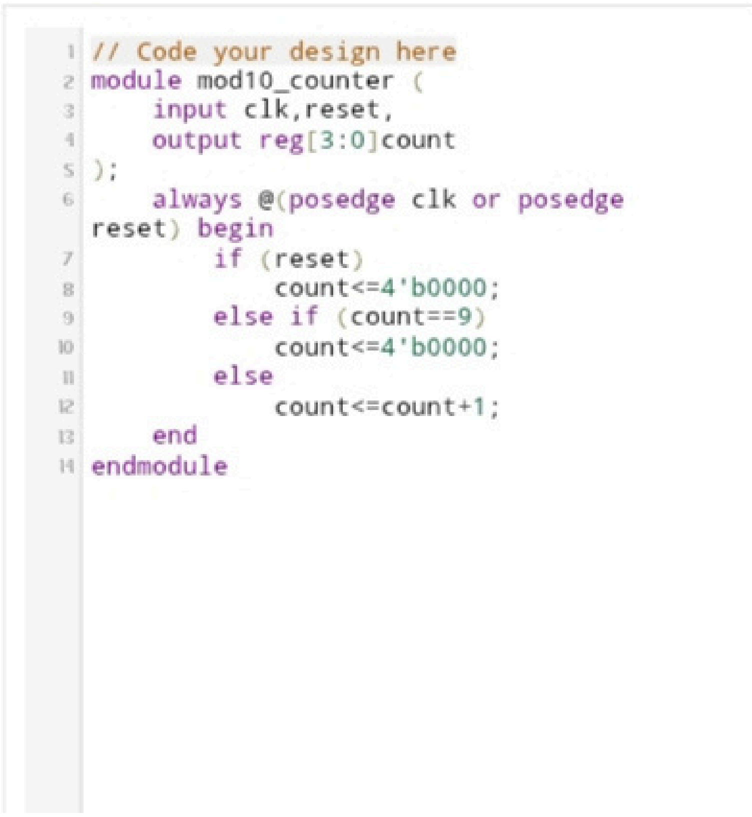
Waveform:



Modulo 10 counter



```verilog
// Code your testbench here
// or browse Examples
module tb_mod10_counter;
    reg clk,reset;
    wire [3:0]count;

    mod10_counter uut (.clk(clk), .reset(reset),.count(count));

    initial begin
        $dumpfile("mod10_counter.vcd");
        $dumpvars(0,tb_mod10_counter);
        $monitor("Time=%0t | Count=%d", $time, count);

        clk =0;reset= 1;
        #5 reset=0;
        #120 $finish;
    end

    always #5 clk=~clk;
endmodule
```

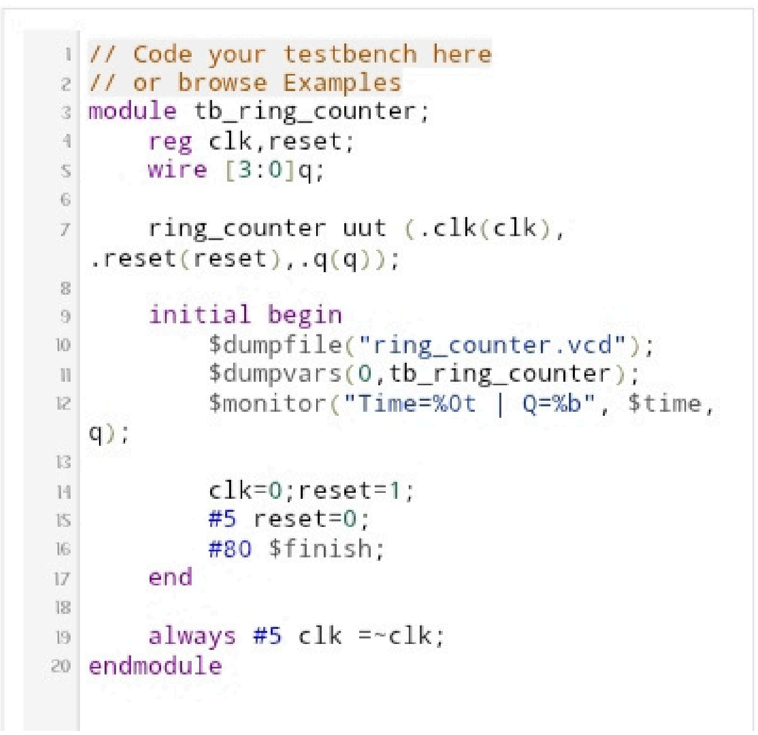


```verilog
// Code your design here
module mod10_counter (
    input clk,reset,
    output reg[3:0]count
);
    always @(posedge clk or posedge reset) begin
        if (reset)
            count<=4'b0000;
        else if (count==9)
            count<=4'b0000;
        else
            count<=count+1;
    end
endmodule
```

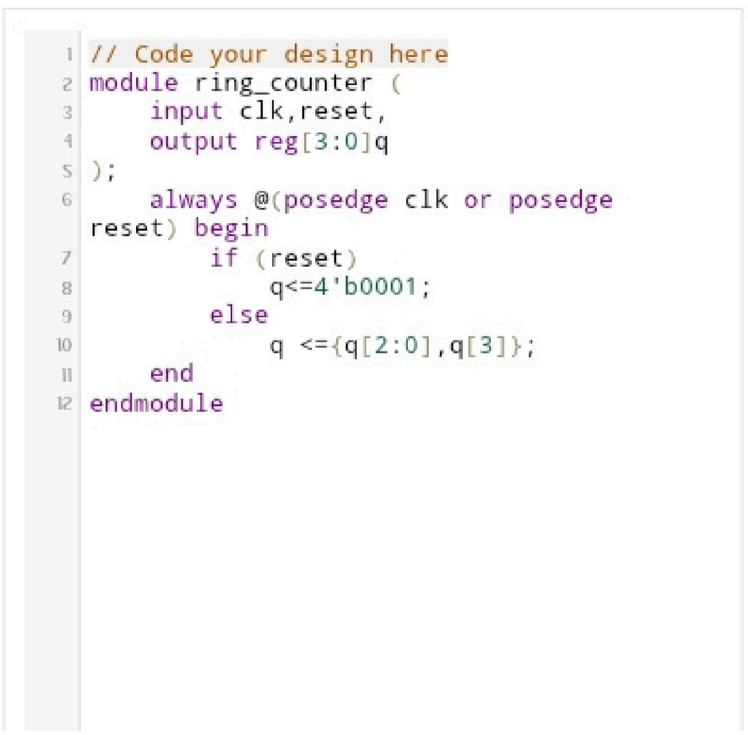
Output:

```
Time=0 | Count= 0
Time=5 | Count= 1
Time=15 | Count= 2
Time=25 | Count= 3
Time=35 | Count= 4
Time=45 | Count= 5
Time=55 | Count= 6
Time=65 | Count= 7
Time=75 | Count= 8
Time=85 | Count= 9
Time=95 | Count= 0
Time=105 | Count= 1
Time=115 | Count= 2
```

Ring counter



```
// Code your testbench here
// or browse Examples
module tb_ring_counter;
    reg clk,reset;
    wire [3:0]q;

    ring_counter uut (.clk(clk),
.reset(reset),.q(q));

    initial begin
        $dumpfile("ring_counter.vcd");
        $dumpvars(0,tb_ring_counter);
        $monitor("Time=%0t | Q=%b", $time,
q);

        clk=0;reset=1;
        #5 reset=0;
        #80 $finish;
    end

    always #5 clk =~clk;
endmodule
```

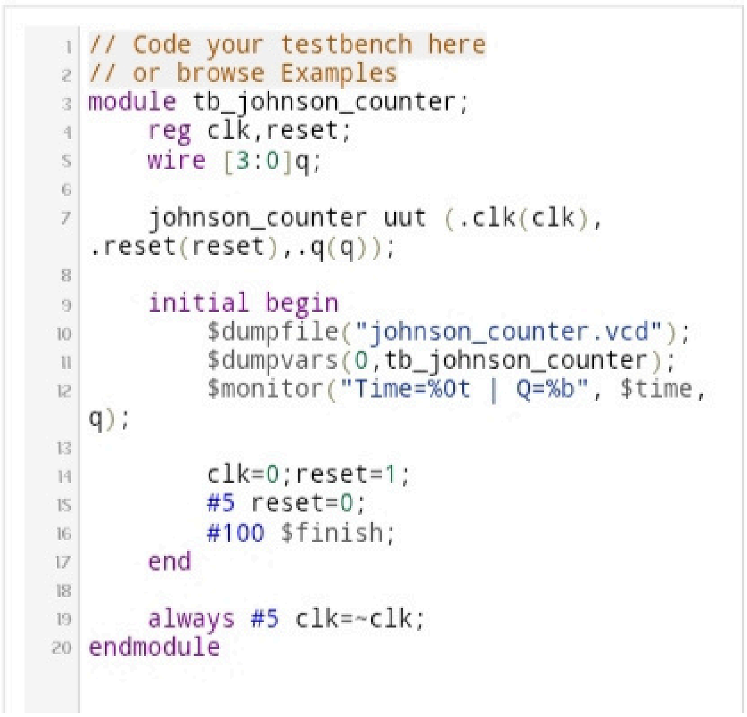


```
// Code your design here
module ring_counter (
    input clk,reset,
    output reg[3:0]q
);
    always @(posedge clk or posedge
reset) begin
        if (reset)
            q<=4'b0001;
        else
            q <={q[2:0],q[3]};
    end
endmodule
```

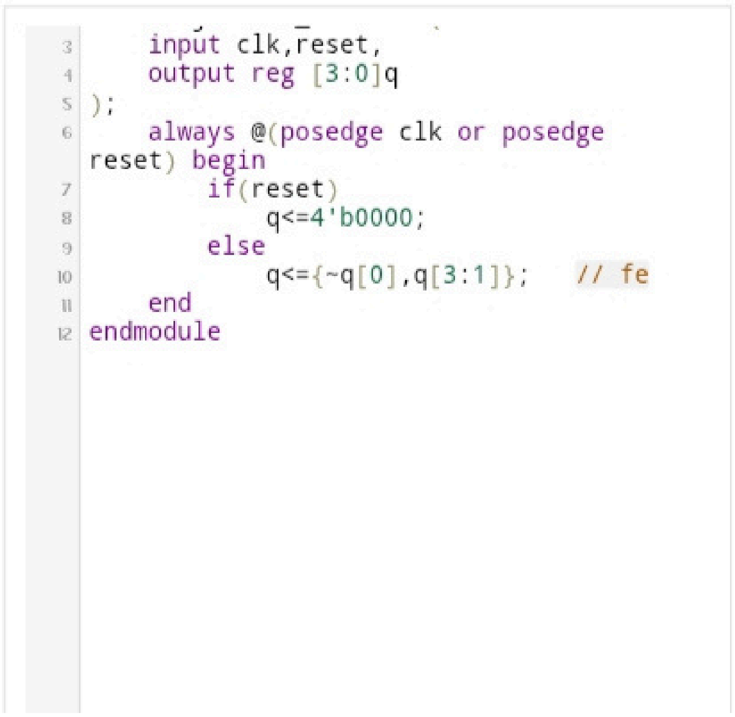
Output:

```
Time=0 | Q=0001
Time=5 | Q=0010
Time=15 | Q=0100
Time=25 | Q=1000
Time=35 | Q=0001
Time=45 | Q=0010
Time=55 | Q=0100
Time=65 | Q=1000
Time=75 | Q=0001
```

Johnson counter



```
// Code your testbench here
// or browse Examples
module tb_johnson_counter;
    reg clk,reset;
    wire [3:0]q;

    johnson_counter uut (.clk(clk),.reset(reset),.q(q));

    initial begin
        $dumpfile("johnson_counter.vcd");
        $dumpvars(0,tb_johnson_counter);
        $monitor("Time=%0t | Q=%b", $time, q);

        clk=0;reset=1;
        #5 reset=0;
        #100 $finish;
    end

    always #5 clk=~clk;
endmodule
```

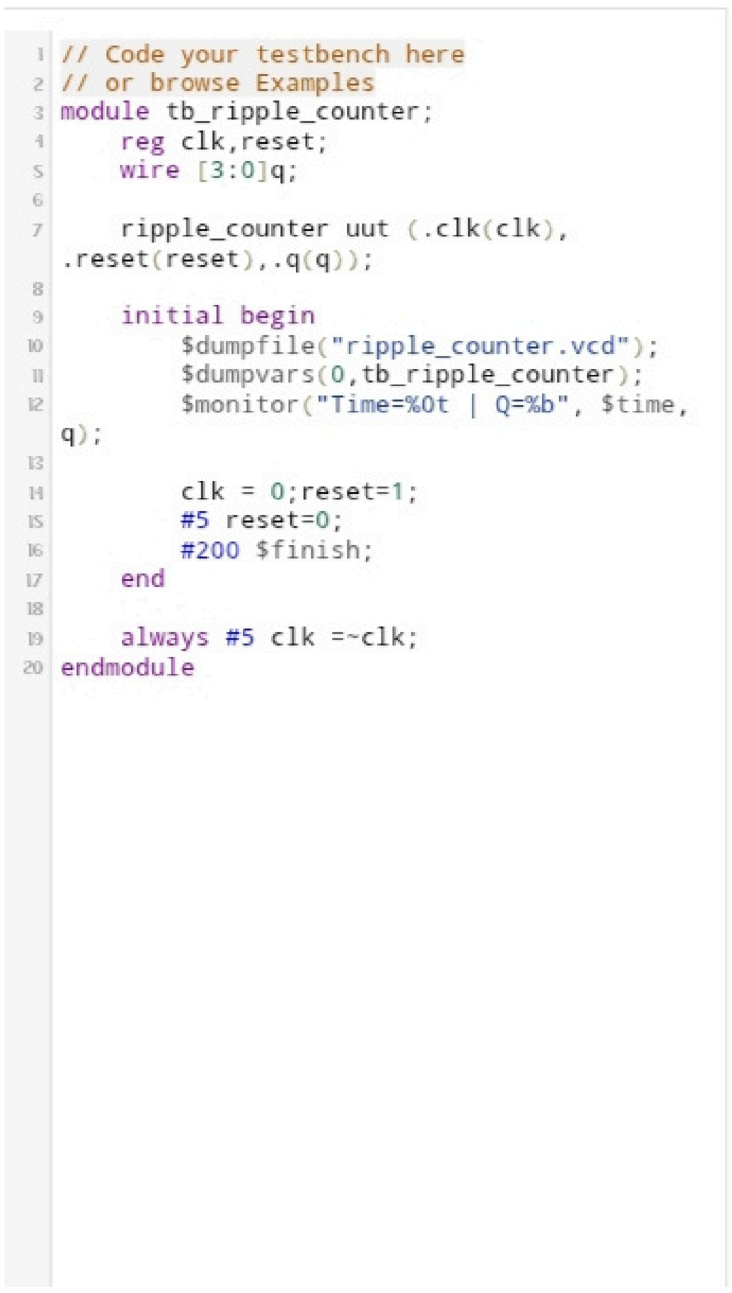


```
    input clk,reset,
    output reg [3:0]q
);
    always @(posedge clk or posedge reset) begin
        if(reset)
            q<=4'b0000;
        else
            q<={~q[0],q[3:1]};    // fe
    end
endmodule
```

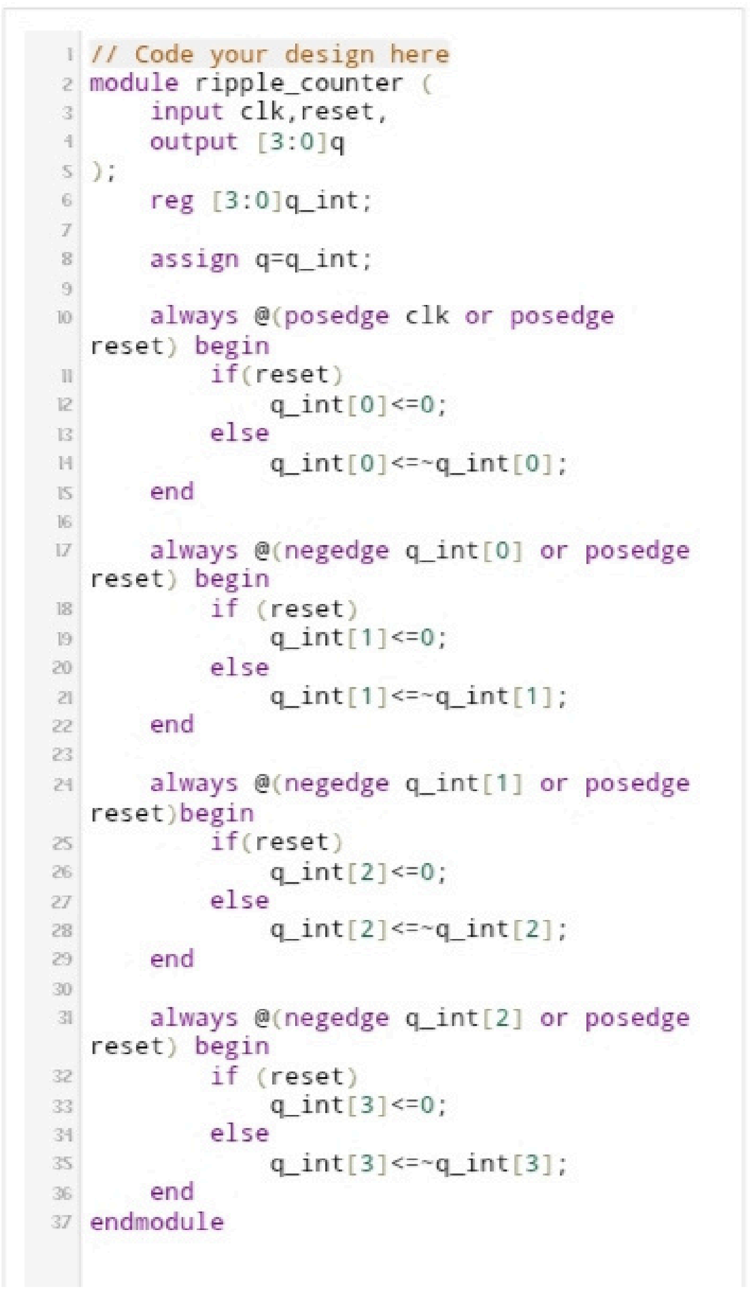
Output:

```
Time=0 | Q=0000
Time=5 | Q=1000
Time=15 | Q=1100
Time=25 | Q=1110
Time=35 | Q=1111
Time=45 | Q=0111
Time=55 | Q=0011
Time=65 | Q=0001
Time=75 | Q=0000
Time=85 | Q=1000
Time=95 | Q=1100
```

Ripple counter

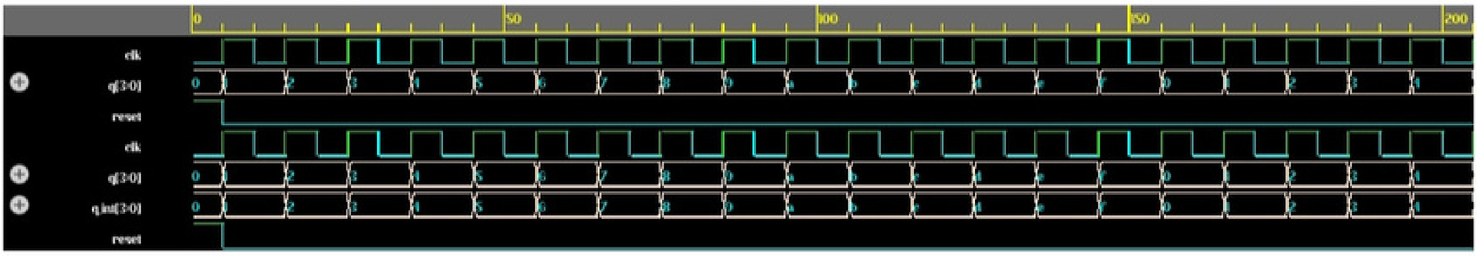


```verilog
// Code your testbench here
// or browse Examples
module tb_ripple_counter;
    reg clk,reset;
    wire [3:0]q;

    ripple_counter uut (.clk(clk), .reset(reset),.q(q));

    initial begin
        $dumpfile("ripple_counter.vcd");
        $dumpvars(0,tb_ripple_counter);
        $monitor("Time=%0t | Q=%b", $time, q);

        clk = 0;reset=1;
        #5 reset=0;
        #200 $finish;
    end

    always #5 clk =~clk;
endmodule
```



```verilog
// Code your design here
module ripple_counter (
    input clk,reset,
    output [3:0]q
);
    reg [3:0]q_int;

    assign q=q_int;

    always @(posedge clk or posedge reset) begin
        if(reset)
            q_int[0]<=0;
        else
            q_int[0]<=~q_int[0];
    end

    always @(negedge q_int[0] or posedge reset) begin
        if (reset)
            q_int[1]<=0;
        else
            q_int[1]<=~q_int[1];
    end

    always @(negedge q_int[1] or posedge reset)begin
        if(reset)
            q_int[2]<=0;
        else
            q_int[2]<=~q_int[2];
    end

    always @(negedge q_int[2] or posedge reset) begin
        if (reset)
            q_int[3]<=0;
        else
            q_int[3]<=~q_int[3];
    end
endmodule
```

Waveform:



Output:

```
Time=45 | Q=0101
Time=55 | Q=0110
Time=65 | Q=0111
Time=75 | Q=1000
Time=85 | Q=1001
Time=95 | Q=1010
Time=105 | Q=1011
Time=115 | Q=1100
Time=125 | Q=1101
Time=135 | Q=1110
Time=145 | Q=1111
Time=155 | Q=0000
Time=165 | Q=0001
Time=175 | Q=0010
Time=185 | Q=0011
Time=195 | Q=0100
```