Verilog: Timescales

Sini Balakrishnan April 1, 2014 15 Comments

As we are aware, compiler directive ``timescale` in Verilog is a tricky topic and have many discussion around it.

Timescale specifies the time unit and time precision of a module that follow it. The simulation time and delay values are measured using time unit. The precision factor is needed to measure the degree of accuracy of the time unit, in other words how delay values are rounded before being used in simulation.

Let's have a look into it and see how time units and precision are taken to calculate simulation time.

In the examples, following time scale /time precision combinations are used.

```
`timescale 1ps / 1ps     // module timescale_check1
`timescale 1 ns / 1 ps   // module timescale_check2;
`timescale 100ns / 1ns   // module timescale_check3;
`timescale 1ms / 1us     // module timescale_check4;
`timescale 10 ms / 10 ns // module timescale_check5;
```

The `rval` changes in different time-steps but due to the differences in timescale directives the simulation time varies.

```
          rval = 20;
   #10.566601 rval = 10;
   #10.980003 rval = 55;
   #15.674 rval = 0;
   #5.0000001 rval = 250;
   #5.67891224 rval = 100;
```

In the below code, timescale is `timescale 1ps / 1ps.

To find out number of digits taken after decimal, first divide time scale with time precision. The exponent number will be your result.

Here, 1ps/1ps = 1 = 10^0, as the result is 10^0, NO digit will be taken after decimal. So 10.566601 will be 11 and next time step 21.546604 will be just 22.

Check below result to see all simulation time for rval. Also note that simulation ends at 100 PS.

```
`timescale 1ps / 1ps
module timescale_check1;
  reg[31:0] rval;

  initial begin
  rval = 20;
  #10.566601 rval = 10;
  #10.980003 rval = 55;
  #15.674 rval = 0;
  #5.0000001 rval = 250;
  #5.67891224 rval = 100;
  end
 initial begin
    $monitor("TimeScale 1ps/1ps : Time=%0t,  rval = %d\n",$realtime,rval);
    #100 $finish;
   end
```

```
endmodule

/*

Simulation Result:

ncsim> run

TimeScale 1ps/1ps : Time=0,  rval =         20


TimeScale 1ps/1ps : Time=11,  rval =         10


TimeScale 1ps/1ps : Time=22,  rval =         55


TimeScale 1ps/1ps : Time=38,  rval =          0


TimeScale 1ps/1ps : Time=43,  rval =        250


TimeScale 1ps/1ps : Time=49,  rval =        100


Simulation complete via $finish(1) at time 100 PS + 0

./timescale.v:17     #100 $finish;

*/
```

Now, let us check one more example. In `module timescale_check2`, timescale is 1ns / 1ps.

Timescale/Time Precision = 1ns/1ps = 1000 = 10^3 So 3 digits after decimal will be used. In this case 10.566601 becomes 10567 and 21.546604 becomes 21547.

```
`timescale 1 ns / 1 ps

module timescale_check2;

  reg[31:0] rval;

  initial begin

  rval = 20;

  #10.566601 rval = 10;

  #10.980003 rval = 55;
  #15.674 rval = 0;

  #5.0000001 rval = 250;

  #5.67891224 rval = 100;

  end


 initial begin

    $monitor("TimeScale 1ns/1ps : Time=%0t,  rval = %d\n",$realtime,rval);

    #100 $finish;

   end


endmodule

/*

Simulation Result

ncsim> run

TimeScale 1ns/1ps : Time=0,  rval =         20


TimeScale 1ns/1ps : Time=10567,  rval =         10


TimeScale 1ns/1ps : Time=21547,  rval =         55


TimeScale 1ns/1ps : Time=37221,  rval =          0


TimeScale 1ns/1ps : Time=42221,  rval =        250


TimeScale 1ns/1ps : Time=47900,  rval =        100


Simulation complete via $finish(1) at time 100 NS + 0

./timescale.v:56     #100 $finish;

*/
```

Below examples show simulation results for timescales 100ns/1ns, 1ms / 1us and 10 ms / 10 ns.

```
`timescale 100ns / 1ns
module timescale_check3;
  reg[31:0] rval;
  initial begin
  rval = 20;
  #10.566601 rval = 10;
  #10.980003 rval = 55;
  #15.674 rval = 0;
  #5.0000001 rval = 250;
  #5.67891224 rval = 100;
  end

  initial begin
    $monitor("TimeScale 100 ns/1ns : Time=%0t,  rval = %d\n",$realtime,rval);
    #100 $finish;
   end
endmodule

/*
Simulation Result
ncsim> run
TimeScale 100 ns/1ns : Time=0,  rval =         20


TimeScale 100 ns/1ns : Time=1057,  rval =         10


TimeScale 100 ns/1ns : Time=2155,  rval =         55


TimeScale 100 ns/1ns : Time=3722,  rval =          0


TimeScale 100 ns/1ns : Time=4222,  rval =        250


TimeScale 100 ns/1ns : Time=4790,  rval =        100
```

```
Simulation complete via $finish(1) at time 10 US + 0

./timescale.v:94     #100 $finish;

*/
```

```
`timescale 1ms / 1us

module timescale_check4;

  reg[31:0] rval;

  initial begin

  rval = 20;

  #10.566601 rval = 10;

  #10.980003 rval = 55;

  #15.674 rval = 0;

  #5.0000001 rval = 250;

  #5.67891224 rval = 100;

  end


  initial begin

    $monitor("TimeScale 1ms/1us : Time=%0t,  rval = %d\n",$realtime,rval);

    #100 $finish;

   end


endmodule

/*

Simulation Result

ncsim> run

TimeScale 1ms/1us : Time=0,  rval =         20


TimeScale 1ms/1us : Time=10567,  rval =         10


TimeScale 1ms/1us : Time=21547,  rval =         55
TimeScale 1ms/1us : Time=37221,  rval =          0


TimeScale 1ms/1us : Time=42221,  rval =        250


TimeScale 1ms/1us : Time=47900,  rval =        100


Simulation complete via $finish(1) at time 100 MS + 0
./timescale.v:132     #100 $finish;
*/
```

```
`timescale 10 ms / 10 ns
module timescale_check5;
  reg[31:0] rval;
  initial begin
  rval = 20;
  #10.566601 rval = 10;
  #10.980003 rval = 55;
  #15.674 rval = 0;
  #5.0000001 rval = 250;
  #5.67891224 rval = 100;
  end

  initial begin
    $monitor("TimeScale 10 ms/10 ns : Time=%0t,  rval = %d\n",$realtime,rval);
    #100 $finish;
   end

endmodule
/*
Simulation Result
```

```
ncsim> run
TimeScale 10 ms/10 ns : Time=0,  rval =         20


TimeScale 10 ms/10 ns : Time=10566601,  rval =         10


TimeScale 10 ms/10 ns : Time=21546604,  rval =         55


TimeScale 10 ms/10 ns : Time=37220604,  rval =          0


TimeScale 10 ms/10 ns : Time=42220604,  rval =        250


TimeScale 10 ms/10 ns : Time=47899516,  rval =        100


Simulation complete via $finish(1) at time 1 S + 0
./timescale.v:170     #100 $finish;
*/
```

📂 Verilog 🏷 Verilog

SIMILAR ARTICLES

- » Verilog: Timing Controls
- » Verilog: Continuous & Procedural Assignments
- » Verilog: Task & Function
- » Verilog: Control Statements

ABOUT SINI BALAKRISHNAN

Sini has spent more than a dozen years in the semiconductor industry, focusing mostly on verification. She is an expert on Formal Verification and has written international papers and articles on related topics.

View all posts by Sini Balakrishnan →