Post-Synthesis Simulation

VITAL Models, SDF Files, Timing Simulation

Post-synthesis simulation

- Purpose:
  - Verify correctness of synthesized circuit
  - Verify synthesis tool delay/timing estimates
- Synthesis tool generates:
  - Gate-level netlist in Verilog (and/or VHDL**)
  - Standard Delay Format (SDF) file of estimated delays
- IBM_CMOS8HP technology directory:
  - /verilog gate-level Verilog models
  - /fvhdl gate-level functional VHDL models**
  - /vital VITAL-compliant VHDL models**
- Drive with same "do" file/testbench as for behavioral model

** VHDL models omitted from recent CMOS8HP package

Timing simulation of HDL netlists

- Synthesized netlists comprise cells from a library (CMOS8HP library, for example)
  - Each library cell has been *characterized* to determine delays, constraints, pin capacitance, area, etc.
  - Cell libraries represent timing information using the VITAL standard
    - VITAL = VHDL Initiative Toward ASIC Libraries (IEEE Standard 1076.4)
    - Verilog models can also be made VITAL-compatible
- Synthesis tools create a Standard Delay Format (SDF) file of estimated timing data for each cell in the design, for use with VITAL-compatible models
  - SDF = IEEE Standard 1497
  - Designed for "back-annotation" of netlists with delay data

Example

- The modulo 6 counter was synthesized using the CMOS8HP standard cell library
- The next slides show VITAL-compatible Verilog model of an XOR gate and a D flip flop from the library.
- Subsequent slides show a partial SDF file with the timing parameters extracted for one instance each of an XOR gate and D flip flop in the synthesized counter

Verilog netlist for cell modulo6 produced by Synopsys DC from CMOS8HP Cell Library

```
module modulo6 ( CLEARbar, L_Cbar, CLK, I, Q );
  input [2:0] I;
  output [2:0] Q;
  input CLEARbar, L_Cbar, CLK;
  wire   n8, n9, n10, n11, n12, n13, n14, n15, n16;

  DFFS_B Q_reg_0_ ( .D(n10), .CLK(CLK), .S(n11), .Q(n16), .QBAR(Q[0]) );
  DFFS_B Q_reg_2_ ( .D(n9), .CLK(CLK), .S(n11), .QBAR(Q[2]) );
  DFFS_B Q_reg_1_ ( .D(n8), .CLK(CLK), .S(n11), .QBAR(Q[1]) );
  AOI21_A U14 ( .A1(Q[2]), .A2(Q[0]), .B(L_Cbar), .Z(n15) );
  XOR2_A U15 ( .A(Q[0]), .B(Q[1]), .Z(n13) );
  INVERT_B U16 ( .A(CLEARbar), .Z(n11) );
  INVERT_B U17 ( .A(L_Cbar), .Z(n12) );
  AOI22_A U18 ( .A1(L_Cbar), .A2(I[0]), .B1(n16), .B2(n12), .Z(n10) );
  AOI22_A U19 ( .A1(L_Cbar), .A2(I[1]), .B1(n15), .B2(n13), .Z(n8) );
  AO21_B U20 ( .A1(Q[0]), .A2(Q[1]), .B(Q[2]), .Z(n14) );
  AOI22_A U21 ( .A1(L_Cbar), .A2(I[2]), .B1(n15), .B2(n14), .Z(n9) );

endmodule
```

Using the standard cell HDL library

- The CMOS8HP digital design kit contains HDL models for each of the standard cells:
  - /verilog   gate-level Verilog models
- The Verilog models have been compiled into library CMOS8HP.
- Add CMOS8HP to your Modelsim Library list:
  - *In Modelsim select: File > New > Library*
  - *Select: Create a map to an existing library*
  - *Enter Name: CMOS8HP*
  - *In box "Library Maps to", enter, or use the Browse button to select,*
    /class/ELEC6250/cmos8hp/std_cell/v.20130404/Verilog/CMOS8HP
  - *Click OK*
- If starting simulation from the menu, select CMOS8HP on the Libraries tab of the Start Simulation pane.
- If starting simulation with vsim command, use the -L switch:

```
vsim modulo6_1.v  –L  CMOS8HP
```

XOR2_A.v VITAL-compatible Verilog model

```
`timescale 1ns/1ps
`celldefine
module XOR2_A (Z, A, B);
output Z;
input A;
input B;

xor U0 (Z, B, A);        //logic function (Verilog primitive)

specify
specparam
tdelay_A_Z_01_0=0.01,
tdelay_A_Z_10_0=0.01,
tdelay_A_Z_01_1=0.01,
tdelay_A_Z_10_1=0.01;
tdelay_B_Z_01_0=0.01,
tdelay_B_Z_10_0=0.01,
tdelay_B_Z_01_1=0.01,
tdelay_B_Z_10_1=0.01;

(A -=> Z)=(tdelay_A_Z_01_0, tdelay_A_Z_10_0);
(B -=> Z)=(tdelay_B_Z_01_0, tdelay_B_Z_10_0);
endspecify
endmodule
`endcelldefine
```

Propagation delay parameters (with defaults)

Values to come from SDF file

Rise time

Fall time

DFFS_B.v   VITAL-Compatible Model

Logic and timing section of model:

delays

constraints

```
DFFS  i0 (Q,QBAR,CLK_dly,D_dly,S_dly,notifier);  \\logic function

specify    //timing information

(posedge CLK => (Q +: CLK)) = (0.01:0.01:0.01, 0.01:0.01:0.01);
(posedge S => (Q +: S)) = (0.01:0.01:0.01, 0.01:0.01:0.01);
(posedge CLK => (QBAR +: CLK)) = (0.01:0.01:0.01, 0.01:0.01:0.01);
(posedge S => (QBAR -: S)) = (0.01:0.01:0.01, 0.01:0.01:0.01);

$setuphold (posedge CLK &&& ~S_dly,posedge D,0.01,0.01,notifier, ,
                ,CLK_dly,D_dly);
$setuphold (posedge CLK &&& ~S_dly,negedge D,0.01,0.01,notifier, ,
                ,CLK_dly,D_dly);
$recrem (negedge S,posedge CLK,0.01,0.01,notifier, , ,S_dly,CLK_dly);
$width (negedge CLK &&& ~S_dly,0.01,0,notifier);
$width (posedge CLK &&& ~S_dly,0.01,0,notifier);
$width (posedge S,0.01,0,notifier);

endspecify
```

Standard Delay Format (SDF) File

- Produced by synthesis tool (Leonardo)
- Contains VITAL parameter values for all cells in the netlist
- Example (for synthesized counter and CMOS8HP library):

SDF Header – common to all cells in this synthesized netlist

```
(DELAYFILE
   (SDFVERSION "OVI 1.0")
   (DESIGN "modulo6")
   (DATE "Thu Sep 21 15:50:15 2017")
   (VENDOR "PnomV1p50T025_STD_CELL_8HP_12T")
   (PROGRAM "Synopsys Design Compiler cmos")
   (VERSION "L-2016.03-SP5-5")
   (DIVIDER /)
   (VOLTAGE 1.50:1.50:1.50)
   (PROCESS)
   (TEMPERATURE 25.00:25.00:25.00)
   (TIMESCALE 1ns)
```

Min: Typ: Max

Extracted SDF data for XOR instance U15

-- path delays only (no constraints)

```
(CELL
    (CELLTYPE “XOR2_A")
    (INSTANCE U15)
    (DELAY
        (ABSOLUTE     (   Min:Typical:Max )
            (IOPATH A Z (0.064:0.072:0.072) (0.042:0.082:0.082))
            (IOPATH B Z (0.058:0.079:0.079) (0.033:0.073:0.073))
        )                             Rise time            Fall time
    )
)
```

These will replace default values of delay parameters in XOR2_A.v

tdelay_A_Z_01_0   tdelay_A_Z_10_0
tdelay_B_Z_01_0   tdelay_B_Z_10_0

SDF data for DFFS_B instance Q_reg_1_

Select Min/Typ/Max at simulation setup

(Min: Typ: Max)

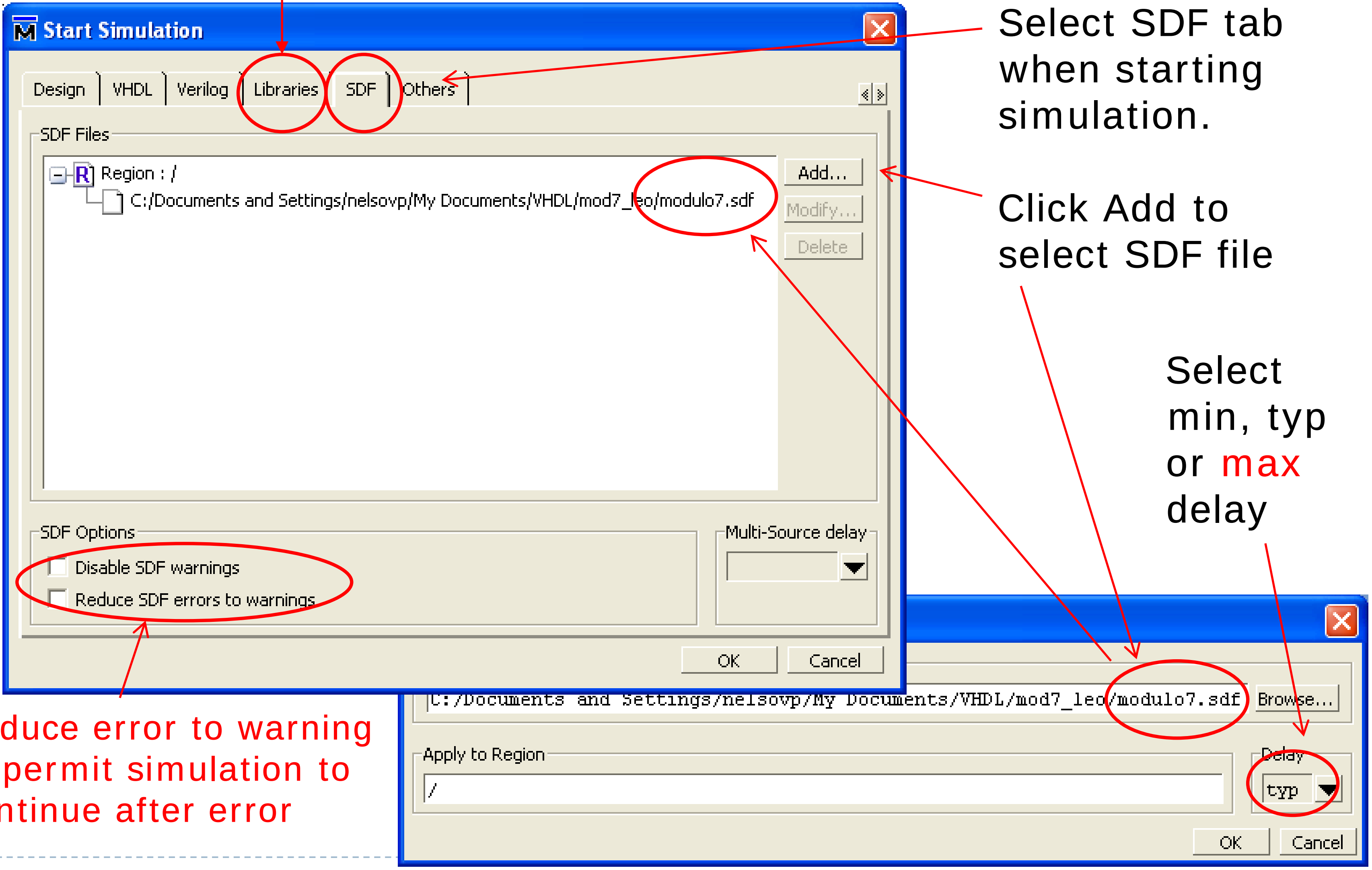
```
(CELL
    (CELLTYPE "DFFS_B")
    (INSTANCE Q_reg_1_)
    (DELAY
        (ABSOLUTE                            T_PLH           T_PHL
        (IOPATH CLK Q (0.132:0.132:0.132) (0.083:0.083:0.083))
        (IOPATH S Q (0.142:0.143:0.143) ())   S cannot force Q low
        (IOPATH CLK QBAR (0.069:0.073:0.073) (0.100:0.107:0.107))
        (IOPATH S QBAR () (0.118:0.124:0.124))    )  )
    (TIMINGCHECK                  S cannot force QBAR high
        (WIDTH (posedge CLK) (0.041:0.041:0.041))
        (WIDTH (negedge CLK) (0.072:0.072:0.072))
        (SETUP D (posedge CLK) (0.098:0.109:0.109))
        (HOLD  D (posedge CLK) (-0.073:-0.019:-0.019))
        (RECOVERY S (posedge CLK) (0.008:0.012:0.012))
        (HOLD  S (posedge CLK) (0.001:-0.003:-0.003))
        (WIDTH (posedge S) (0.086:0.086:0.086))
    )
)
```

delays

constraints

Timing simulation in Modelsim

Select CMOS8HP library



Start Simulation

Design | VHDL | Verilog | Libraries | SDF | Others

SDF Files

Region : /
C:/Documents and Settings/nelsovp/My Documents/VHDL/mod7_leo/modulo7.sdf

Add... | Modify... | Delete

SDF Options
- [ ] Disable SDF warnings
- [ ] Reduce SDF errors to warnings

Multi-Source delay

OK | Cancel

C:/Documents and Settings/nelsovp/My Documents/VHDL/mod7_leo/modulo7.sdf | Browse...

Apply to Region: /

Delay: typ

OK | Cancel

Select SDF tab when starting simulation.

Click Add to select SDF file

Select min, typ or max delay

Reduce error to warning to permit simulation to continue after error

Simulating with SDF & do file

vsim modulo6 –sdfmax modulo6_1.sdf -sdfnoerror -L CMOS8HP –t ps

<u>Compile options</u>

-sdfmax modulo6_1.sdf   May need full path to SDF file

Apply delays only to modulo6 netlist.
Can also use –sdfmin or –sdftyp.

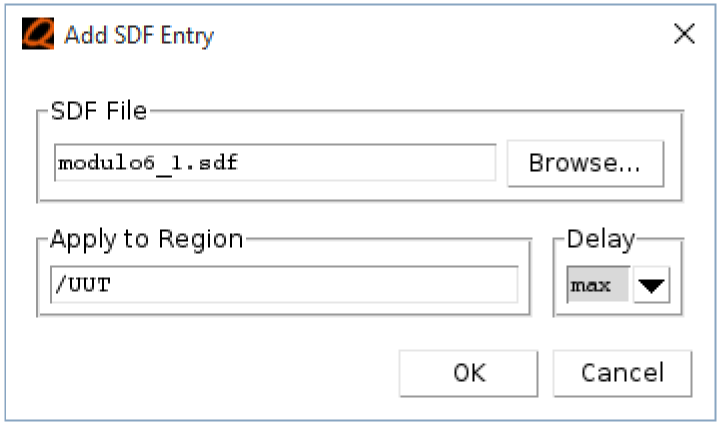
-sdfnoerror Reduce SDF errors to warnings to enable simulation with missing hold times, etc.

-L CMOS8HP Library of gate-level Verilog models.

-t ps Timing resolution consistent with SDF.

Simulating with SDF & testbench

vsim -sdfmax /UUT=modulo6_1.sdf -sdfnoerror -L CMOS8HP -t ps modulo6_bench


Add SDF Entry

SDF File: modulo6_1.sdf

Apply to Region: /UUT

Delay: max

<u>Compile options</u>

-sdfmax /UUT=modulo6_1.sdf

Apply delays only to design (UUT) within the testbech. Can also use –sdfmin or –sdftyp.

-sdfnoerror Reduce SDF errors to warnings to enable simulation with missing hold times, etc.

-L CMOS8HP Library of gate-level Verilog models.

-t ps Timing resolution consistent with SDF.

Testbench: *modulo6_bench.v*

```
timescale 1 ns / 10 ps
module modulo6_bench ();                               //no top-level inputs
   reg clk, clearbar, load_countbar;                   //counter control inputs
   reg [2:0] Din;                                       //counter parallel inputs
   reg [2:0] Qint;                                     //expected counter outputs
   wire [2:0] Qout;                                    //outputs driven by counter
   integer i;

  // instantiate the modulo6 counter
  modulo6 UUT (.CLEARbar(clearbar), .L_Cbar(load_countbar), .CLK(clk), .I(Din), .Q(Qout));

 // produce 20 ns period clock
 initial clk <= 0;                  //initial state 0
 always  #10 clk <= ~clk;           //toggle every 10ns

  // produce other inputs
  initial begin
       Din = 3'b101;                //inputs for load test
       clearbar  = 1'b1;            //clearbar inactive initially
       load_countbar  = 1'b1;       //select load function initially
       #5  clearbar <= 1'b0;        //activate clearbar at time 5
       #10 clearbar <= 1'b1;        //deactivate clearbar at time 15
       if (Qout !== 3'b000) $display("ERROR Qout %b did not reset to 0\n",Qout);
       #20;                         //wait for load of 101
       if (Qout !== Din) $display("ERROR Qout %b did not load %b \n",Qout,Din);
       Din = 3'b010;                //load changing all 3 ffs
       #20;                         //wait for load of 010
```

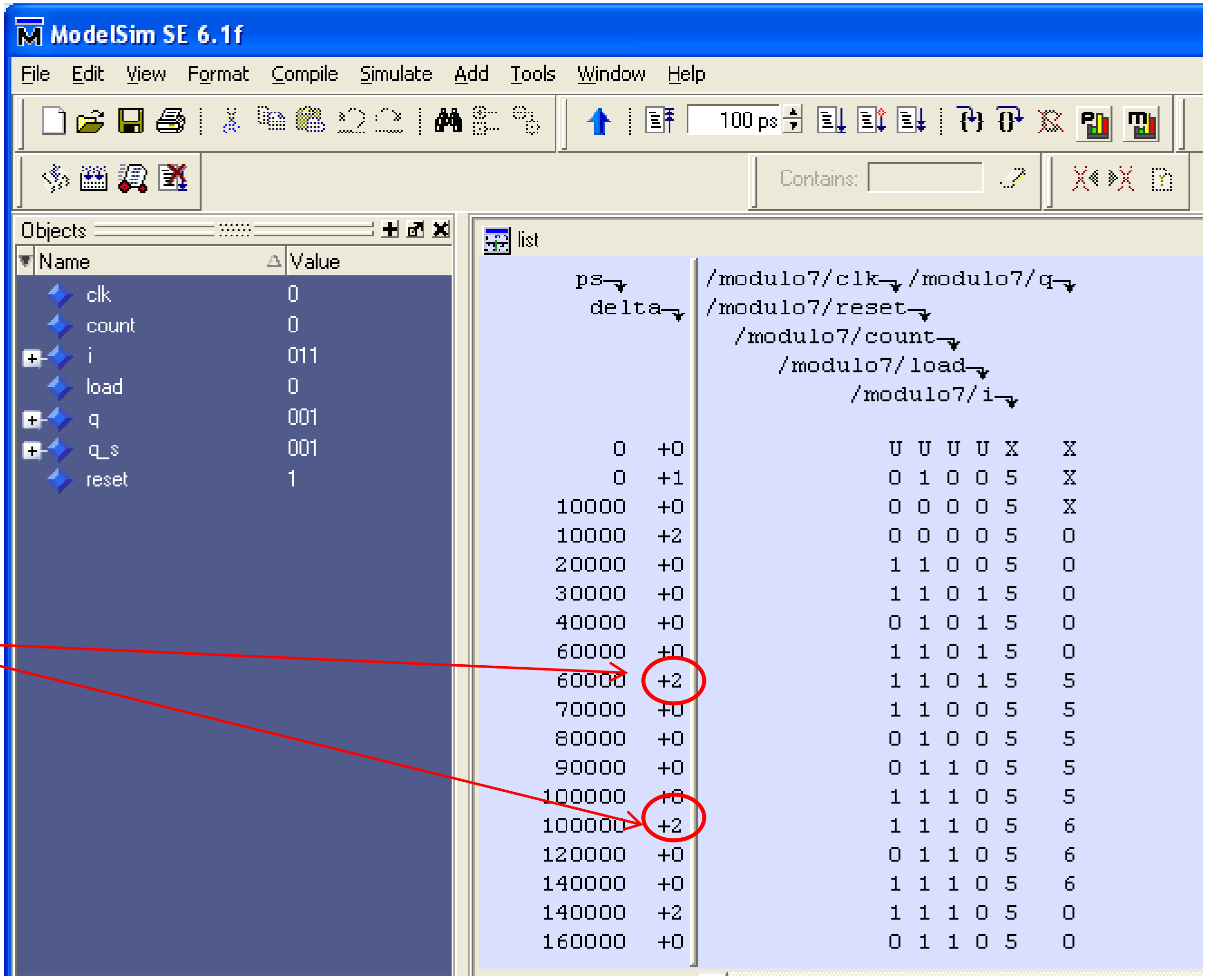
Continued on next slide

Testbench: *modulo6_bench.v*

(Continued)

```
      if (Qout !== Din) $display("ERROR Qout %b did not load %b \n",Qout,Din);
      Din = 3'b101;                        //load changing all 3 ffs
      #20;                                 //wait for load of 101
      if (Qout !== Din) $display("ERROR Qout %b didnot load %b \n",Qout,Din);
      #5  clearbar <= 1'b0;                //activate clearbar at time 5 to reset bits 2 and 0
      #15 clearbar <= 1'b1;                //deactivate clearbar at time 15
      if (Qout !== 3'b000) $display("ERROR Qout %b did not reset to 0\n",Qout);
      Din = 3'b010;                        //load 010
      #20;                                 //wait for load of 010
      if (Qout !== Din) $display("ERROR Qout %b didnot load %b \n",Qout,Din);
      #5  clearbar <= 1'b0;                //activate clearbar at time 5 to reset bit 1
      #15 clearbar <= 1'b1;                //deactivate clearbar at time 15
      if (Qout !== 3'b000) $display("ERROR Qout %b did not reset to 0\n",Qout);
      Qint <= Qout;                        //start count from current state
      #20 load_countbar <= 1'b0;           //disable load & enable count
      for (i=0; i<12; i=i+1) begin         //do two complete count cycles
          Qint = (Qint + 1) % 6;           //expected modulo-6 count
          #20;                             //wait until after count
          if (Qint !== Qout) $display("ERROR Qout %b count not %b \n",Qout,Qint);
      end
   end
endmodule
```

Simulation without SDF


ModelSim SE 6.1f
File Edit View Format Compile Simulate Add Tools Window Help
100 ps
Contains:
Objects
list

Name	Value
clk	0
count	0
i	011
load	0
q	001
q_s	001
reset	1

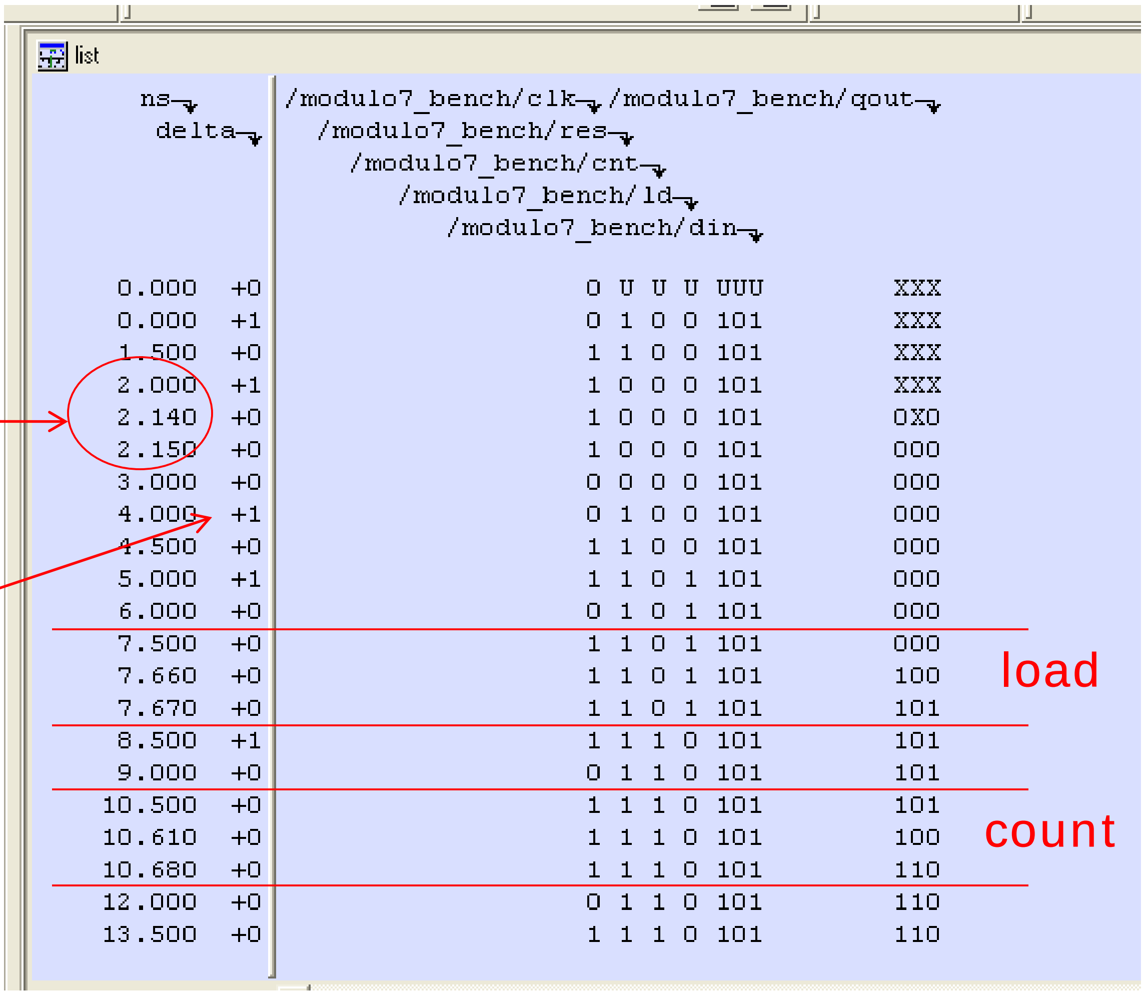
```
       ps    /modulo7/clk  /modulo7/q
    delta    /modulo7/reset
               /modulo7/count
                   /modulo7/load
                        /modulo7/i

        0  +0            U  U  U  U  X    X
        0  +1            0  1  0  0  5    X
    10000  +0            0  0  0  0  5    X
    10000  +2            0  0  0  0  5    0
    20000  +0            1  1  0  0  5    0
    30000  +0            1  1  0  1  5    0
    40000  +0            0  1  0  1  5    0
    60000  +0            1  1  0  1  5    0
    60000  +2            1  1  0  1  5    5
    70000  +0            1  1  0  0  5    5
    80000  +0            0  1  0  0  5    5
    90000  +0            0  1  1  0  5    5
   100000  +0            1  1  1  0  5    5
   100000  +2            1  1  1  0  5    6
   120000  +0            0  1  1  0  5    6
   140000  +0            1  1  1  0  5    6
   140000  +2            1  1  1  0  5    0
   160000  +0            0  1  1  0  5    0
```

Note “delta” delays for behavioral model.

Simulation using SDF file

Note actual delays within test circuit.

Delta delays at testbench level.



```
list
          ns     /modulo7_bench/clk /modulo7_bench/qout
           delta   /modulo7_bench/res
                     /modulo7_bench/cnt
                        /modulo7_bench/ld
                           /modulo7_bench/din

     0.000  +0              0  U  U  U  UUU     XXX
     0.000  +1              0  1  0  0  101     XXX
     1.500  +0              1  1  0  0  101     XXX
     2.000  +1              1  0  0  0  101     XXX
     2.140  +0              1  0  0  0  101     0X0
     2.150  +0              1  0  0  0  101     000
     3.000  +0              0  0  0  0  101     000
     4.000  +1              0  1  0  0  101     000
     4.500  +0              1  1  0  0  101     000
     5.000  +1              1  1  0  1  101     000
     6.000  +0              0  1  0  1  101     000
     7.500  +0              1  1  0  1  101     000
     7.660  +0              1  1  0  1  101     100
     7.670  +0              1  1  0  1  101     101
     8.500  +1              1  1  1  0  101     101
     9.000  +0              0  1  1  0  101     101
    10.500  +0              1  1  1  0  101     101
    10.610  +0              1  1  1  0  101     100
    10.680  +0              1  1  1  0  101     110
    12.000  +0              0  1  1  0  101     110
    13.500  +0              1  1  1  0  101     110
```