ASIC/FPGA Design and Verification

Part 4: CMD Line (Cont'd)

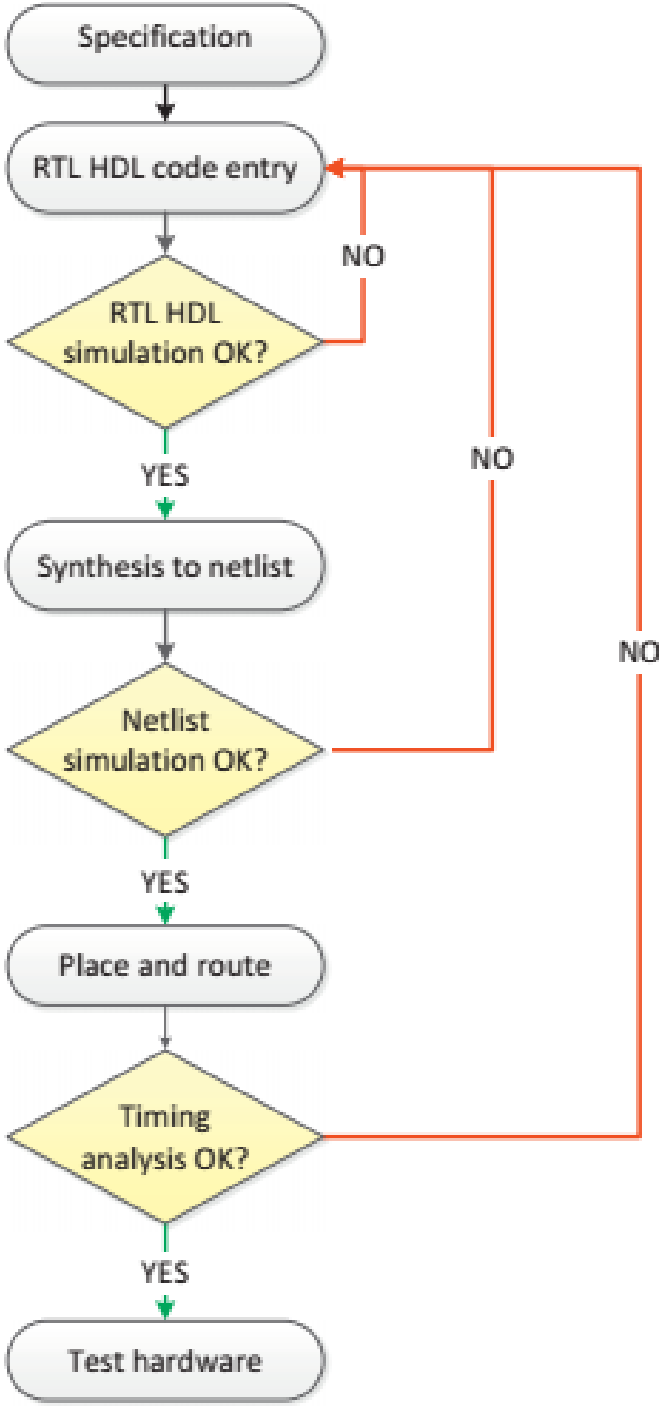
Tuan Nguyen-viet

Nov, 18-20, 2024

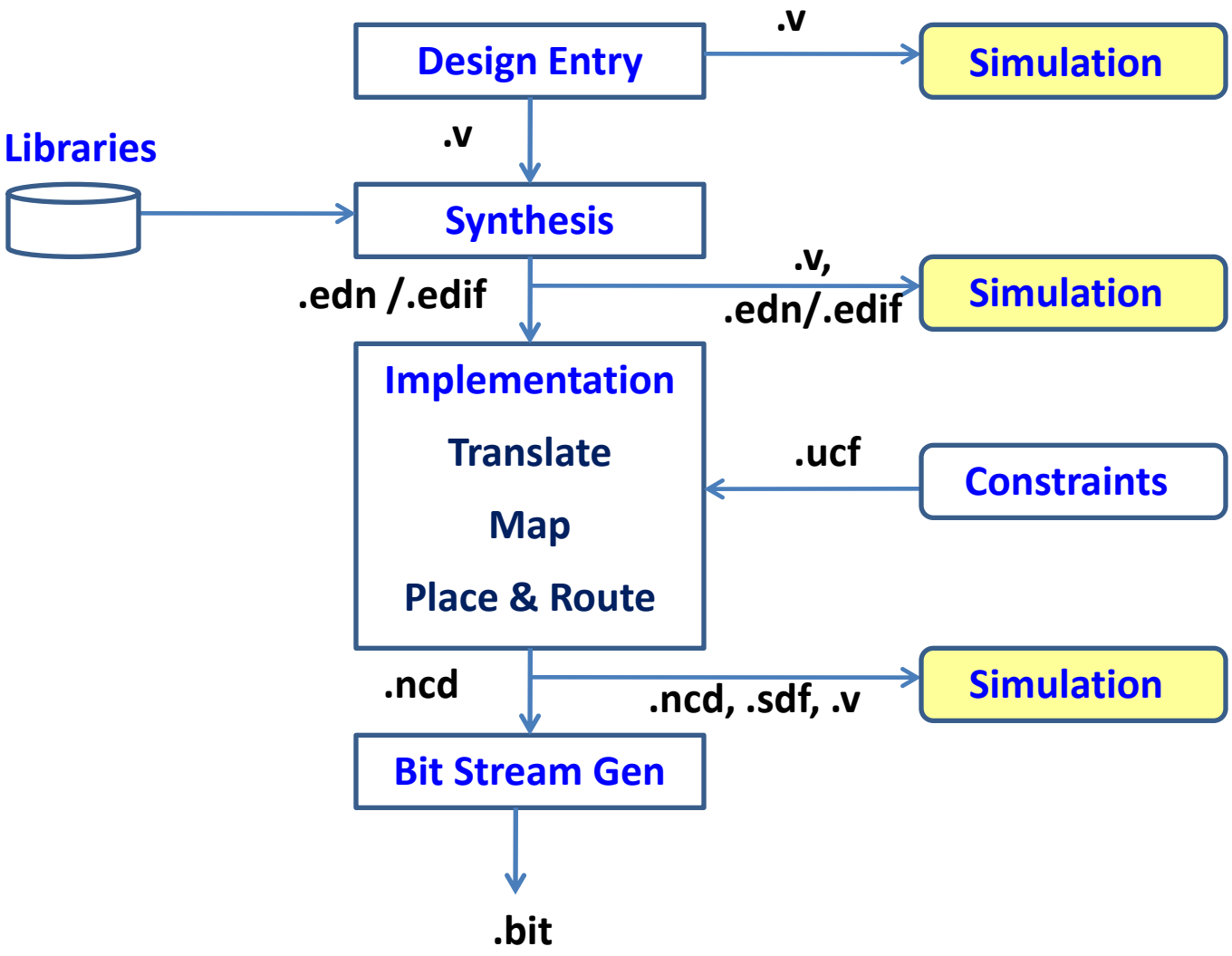
Functional Simulation

- As soon as a project is created in the Vivado Design Suite,
  - we can run behavioral simulation (see **_P3**).

- We can run functional and timing simulations on the design after successfully running synthesis and/or implementaton.


Specification
RTL HDL code entry
RTL HDL simulation OK?
NO
YES
Synthesis to netlist
Netlist simulation OK?
NO
YES
Place and route
Timing analysis OK?
NO
YES
Test hardware

RTL Design Verification (see Slide _P3)



Vivado Design Suite User Guide

Logic Simulation

UG900 (v2022.1) April 21, 2022

Vivado Design Suite User Guide

Using Tcl Scripting

UG894 (v2022.1) June 8, 2022

Netlist Design Verification

```
C:\Users\Family>d:
```

```
D:\fpga_projects\adder\src>vivado -mode tcl
```

```
Vivado%
```

```
Vivado% write_verilog -mode funcsim s_adder_netlist.v
D:/fpga_projects/adder/src/s_adder_netlist.v
Vivado% _
```

```
`timescale 1 ps / 1 ps

(* NotValidForBitStream *)
module s_adder
   (a,
    b,
    sum);
  input [31:0]a;
  input [31:0]b;
  output [31:0]sum;

  wire [31:0]a;
  wire [31:0]a_IBUF;
  wire [31:0]b;
  wire [31:0]b_IBUF;
  wire [31:0]sum;
  wire [31:0]sum_OBUF;
  wire \sum_OBUF[11]_inst_i_1_n_0 ;
  wire \sum_OBUF[11]_inst_i_1_n_1 ;
  wire \sum_OBUF[11]_inst_i_1_n_2 ;
  wire \sum_OBUF[11]_inst_i_1_n_3 ;
```

```
wire \sum_OBUF[7]_inst_i_1_n_2 ;
wire \sum_OBUF[7]_inst_i_1_n_3 ;
wire \sum_OBUF[7]_inst_i_2_n_0 ;
wire \sum_OBUF[7]_inst_i_3_n_0 ;
wire \sum_OBUF[7]_inst_i_4_n_0 ;
wire \sum_OBUF[7]_inst_i_5_n_0 ;
wire [3:3]\NLW_sum_OBUF[31]_inst_i_1_CO_UNCONNECTED ;

IBUF \a_IBUF[0]_inst
     (.I(a[0]),
      .O(a_IBUF[0]));
IBUF \a_IBUF[10]_inst
     (.I(a[10]),
      .O(a_IBUF[10]));
IBUF \a_IBUF[11]_inst
     (.I(a[11]),
      .O(a_IBUF[11]));
IBUF \a_IBUF[12]_inst
     (.I(a[12]),
      .O(a_IBUF[12]));
IBUF \a_IBUF[13]_inst
     (.I(a[13]),
      .O(a_IBUF[13]));
```

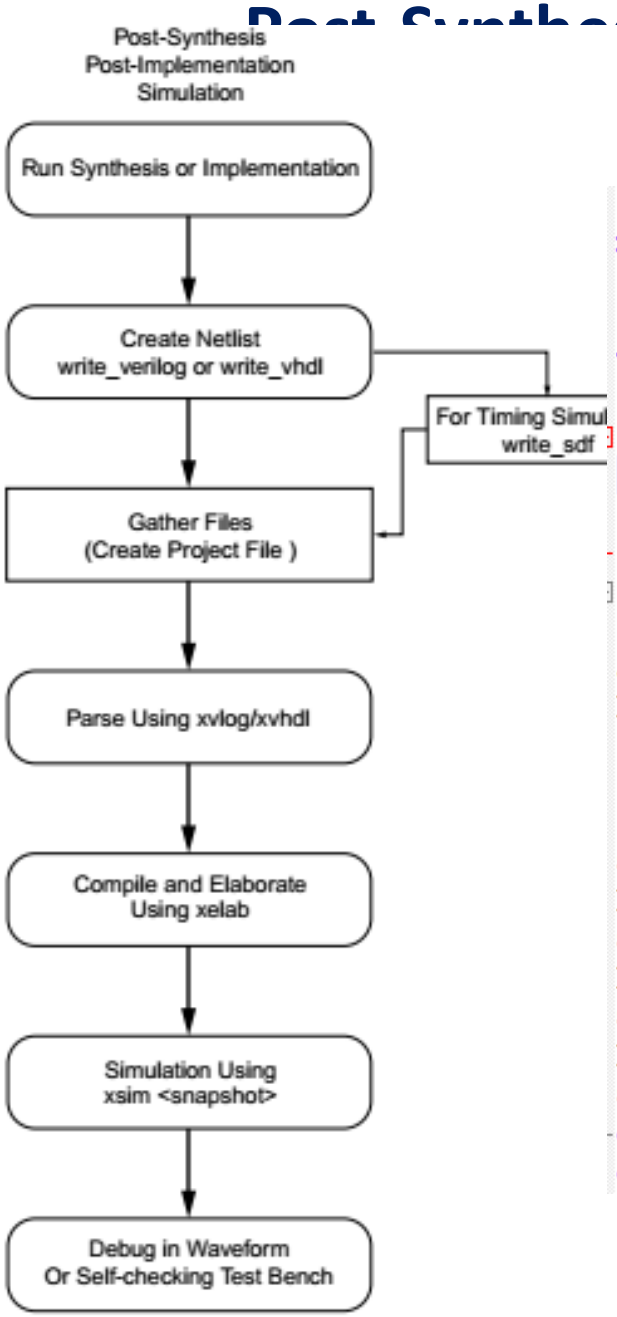
```
LUT2 #(
  .INIT(4'h6))
  \sum_OBUF[7]_inst_i_3
     (.I0(a_IBUF[6]),
      .I1(b_IBUF[6]),
      .O(\sum_OBUF[7]_inst_i_3_n_0 ));
LUT2 #(
  .INIT(4'h6))
  \sum_OBUF[7]_inst_i_4
     (.I0(a_IBUF[5]),
      .I1(b_IBUF[5]),
      .O(\sum_OBUF[7]_inst_i_4_n_0 ));
LUT2 #(
  .INIT(4'h6))
  \sum_OBUF[7]_inst_i_5
     (.I0(a_IBUF[4]),
      .I1(b_IBUF[4]),
      .O(\sum_OBUF[7]_inst_i_5_n_0 ));
OBUF \sum_OBUF[8]_inst
     (.I(sum_OBUF[8]),
      .O(sum[8]));
OBUF \sum_OBUF[9]_inst
     (.I(sum_OBUF[9]),
      .O(sum[9]));
endmodule
```

PC › Local Disk (D:) › fpga_projects › adder › src

Name

- .Xil
- adder_tb.v
- s_adder.v
- s_adder_netlist.v
- vivado.jou
- vivado.log

Post-Synthesis / Post-Implementation Simulation

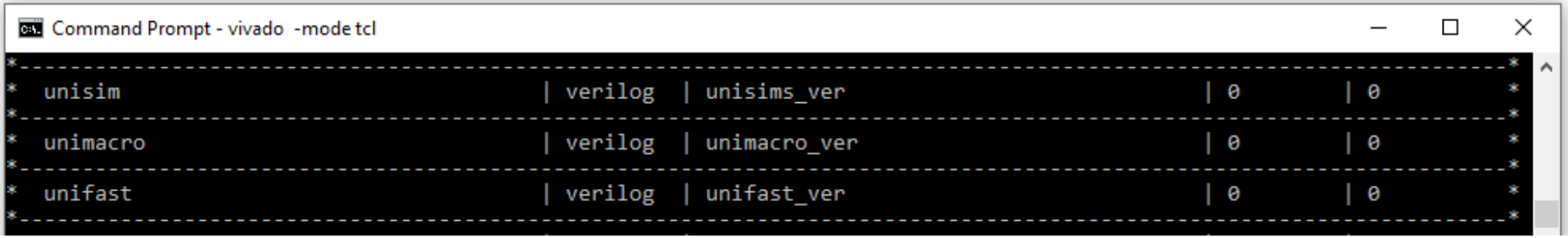


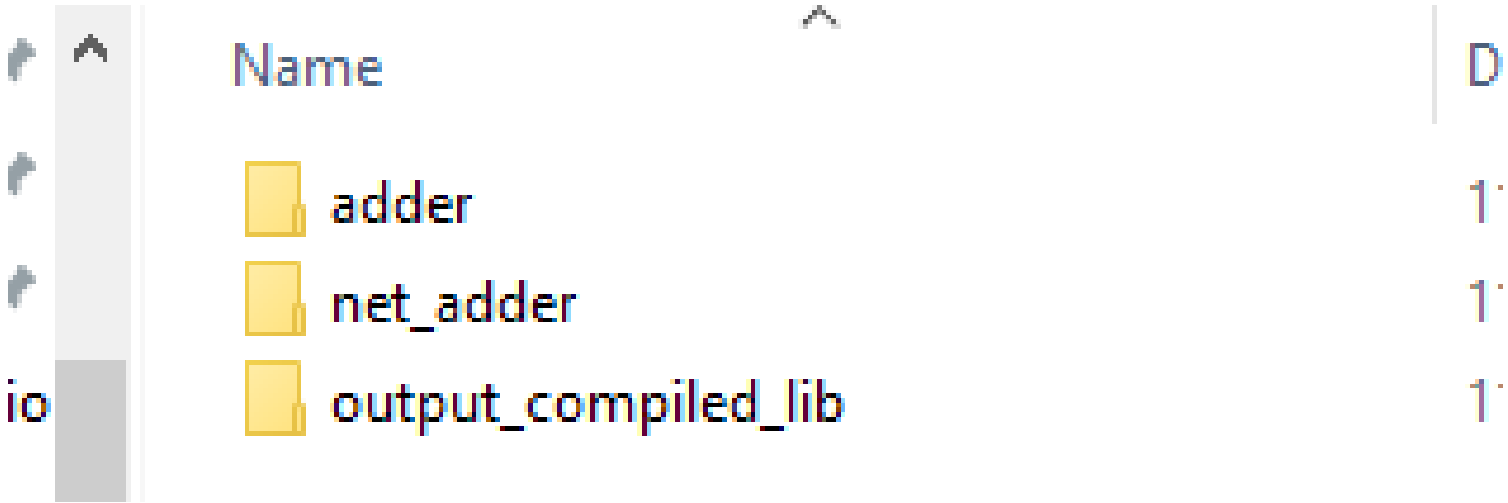

```
`timescale 1ns / 1ps
module adder_tb;
reg [31:0] a;
reg [31:0] b;
wire [31:0] sum;
// Instantiate the simple_adder module
s_adder_netlist dut ( .a(a),
//s_adder dut ( .a(a),
                    .b(b),
                    .sum(sum));
initial begin
// Initialize inputs
a = 32'b00;
b = 32'b00;
// Monitor changes to inputs and output
$monitor("Time: %0t | a: %b, b: %b, sum: %b", $time, a, b, sum);
// Stimulus for the test
#10 a = 32'b01;
b = 32'b01;
#10 a = 32'b10;
b = 32'b01;
#10 a = 32'b11;
b = 32'b11;
#10 $finish; // End simulation
end
endmodule
```

compile_simlib -language verilog -dir {D:\ fpga_uvm \output_compiled_lib} -simulator questa -library unisim -family artix7

compile_simlib -language all -dir {D:\ fpga_uvm \output_compiled_lib} -simulator questa -library unisim -family artix7

```
Vivado% compile_simlib -language verilog -dir {D:\fpga_projects\output_compiled_lib} -simulator questa -library unisim -
family artix7
WARNING: [Vivado 12-5377] Language specific library compilation for IPs is not supported. By default, the libraries will
 be compiled for all languages.
INFO: [Vivado 12-4753] Extracting data from the IP repository...(this may take a while, please wait)...
```




Name
D
adder
net_adder
output_compiled_lib
io

Verilog UNISIM Library

Verilog UNISIM Library

The Verilog UNISIM library is located at *<Vivado_Install_Dir>/data/verilog/src/unisims*.

In Verilog, the individual library modules are specified in separate HDL files.

This allows the -y library specification switch to search the specified directory for all components and automatically expand the library.

The Verilog UNISIM library does not have to be specified in the HDL file prior to using the module.

Verilog is case-sensitive, so ensure that UNISIM primitive instantiations adhere to an uppercase naming convention, for example, BUFG.

If you use precompiled libraries, use the correct simulator command-line switch to point to the precompiled libraries.

The following is an example for the Vivado simulator:

```
-L unisims_ver
```

REF: https://adaptivesupport.amd.com/s/article/64052?language=en_US

Verilog UNISIM Library

- The Verilog UNISIM library is located at <*Vivado_Install_Dir*>/*data/verilog/src/unisims*.
- In Verilog, the individual library modules are specified in separate HDL files.

  This allows the -y library specification switch to search the specified directory for all components and automatically expand the library.
-
- The Verilog UNISIM library does not have to be specified in the HDL file prior to using the module.

  Verilog is case-sensitive, so ensure that UNISIM primitive instantiations adhere to an uppercase naming convention, for example, BUFG.
- If you use precompiled libraries, use the correct simulator command-line switch to point to the precompiled libraries.

  The following is an example for the Vivado simulator:

  -L unisims_ver

REF: https://adaptivesupport.amd.com/s/article/64052?language=en_US

Table 9: **Environment Variable Setting for Third-Party Simulators**

Simulator	Linux	Windows

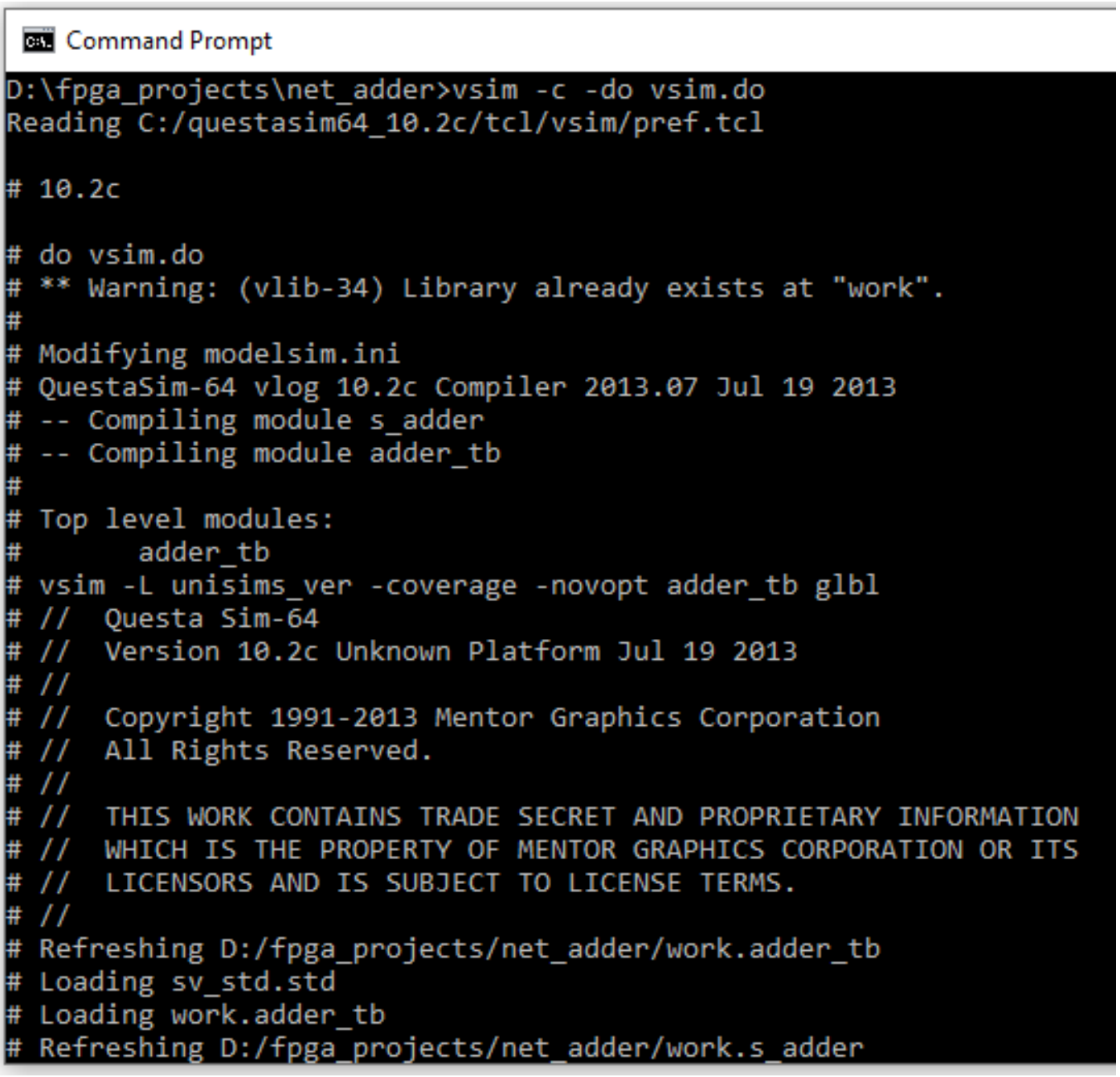
Modelsim	`setenv MODEL_TECH <tool installation path>`
	`setenv LM_LICENSE_FILE <license file>`
	`setenv PATH ${MODEL_TECH}/bin:$PATH`
		`set MODEL_TECH=<tool installation path>`
		`set LM_LICENSE_FILE=<license file>`
		`set Path=%MODEL_TECH%\win32;%Path%`

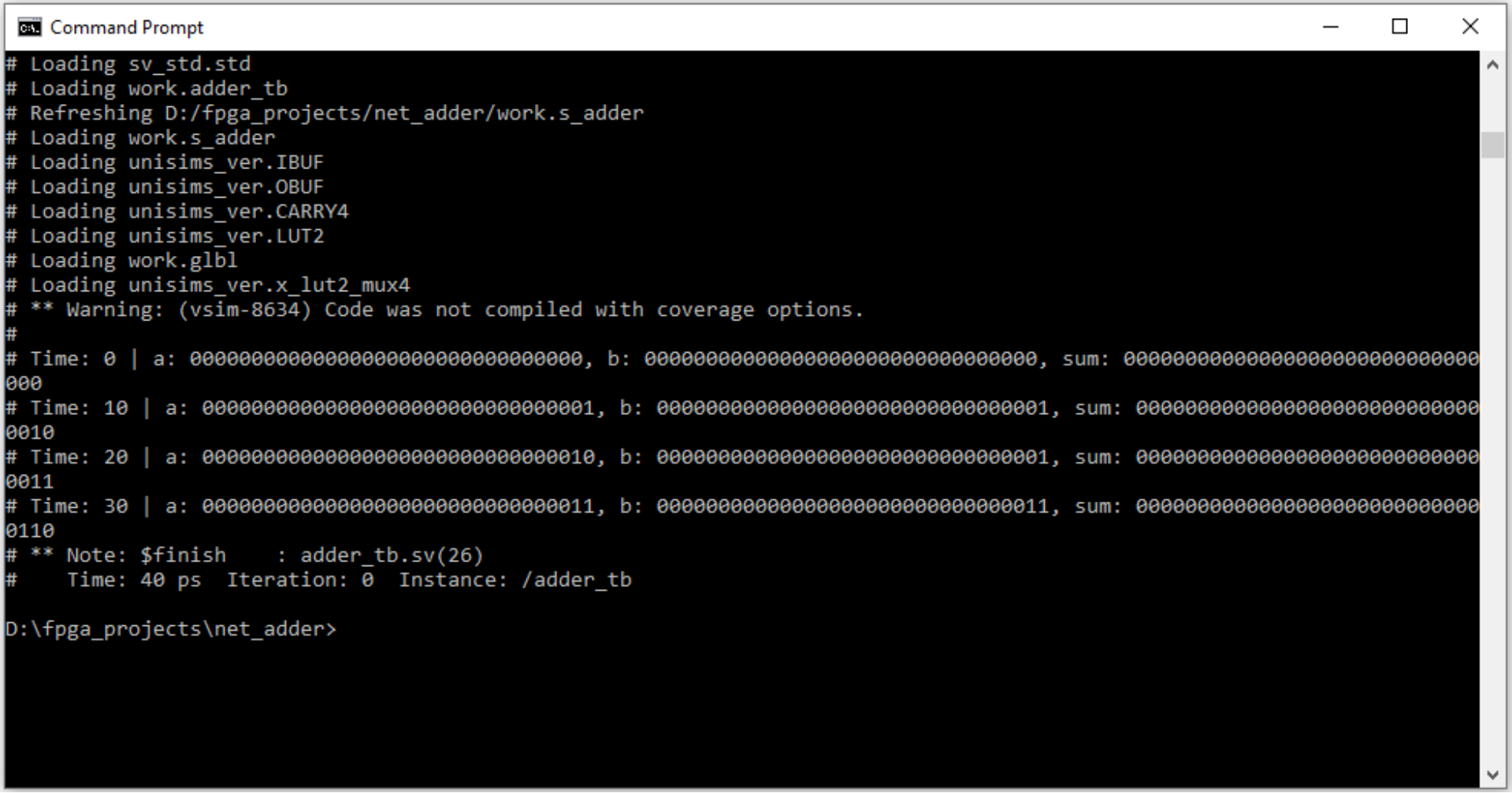
Questa	`setenv MODEL_TECH <tool installation path>`
	`setenv LM_LICENSE_FILE <license file>`
	`setenv PATH ${MODEL_TECH}/bin:$PATH`
		`set MODEL_TECH=<tool installation path>`
		`set LM_LICENSE_FILE=<license file>`
		`set Path=%MODEL_TECH%\win32;%Path%`

REF: UG900

```
[Library]
;others = $MODEL_TECH/../modelsim.ini
;
; VITAL concerns:
;
; The library ieee contains (among oth
```

```
vlib work
vmap work work

# Compilation: ----------------------------------------
vlog adder_tb.sv

# Simulation: ----------------------------------------
vsim -novopt -L unisims_ver -coverage adder_tb glbl

add wave -r sim:/adder_tb/*
run -all

```


Command Prompt
D:\fpga_projects\net_adder>vsim -c -do vsim.do
Reading C:/questasim64_10.2c/tcl/vsim/pref.tcl
10.2c
do vsim.do
** Warning: (vlib-34) Library already exists at "work".
#
Modifying modelsim.ini
QuestaSim-64 vlog 10.2c Compiler 2013.07 Jul 19 2013
-- Compiling module s_adder
-- Compiling module adder_tb
#
Top level modules:
adder_tb
vsim -L unisims_ver -coverage -novopt adder_tb glbl
// Questa Sim-64
// Version 10.2c Unknown Platform Jul 19 2013
//
// Copyright 1991-2013 Mentor Graphics Corporation
// All Rights Reserved.
//
// THIS WORK CONTAINS TRADE SECRET AND PROPRIETARY INFORMATION
// WHICH IS THE PROPERTY OF MENTOR GRAPHICS CORPORATION OR ITS
// LICENSORS AND IS SUBJECT TO LICENSE TERMS.
//
Refreshing D:/fpga_projects/net_adder/work.adder_tb
Loading sv_std.std
Loading work.adder_tb
Refreshing D:/fpga_projects/net_adder/work.s_adder


Command Prompt
Loading sv_std.std
Loading work.adder_tb
Refreshing D:/fpga_projects/net_adder/work.s_adder
Loading work.s_adder
Loading unisims_ver.IBUF
Loading unisims_ver.OBUF
Loading unisims_ver.CARRY4
Loading unisims_ver.LUT2
Loading work.glbl
Loading unisims_ver.x_lut2_mux4
** Warning: (vsim-8634) Code was not compiled with coverage options.
#
Time: 0 | a: 00000000000000000000000000000000, b: 00000000000000000000000000000000, sum: 00000000000000000000000000000
000
Time: 10 | a: 00000000000000000000000000000001, b: 00000000000000000000000000000001, sum: 00000000000000000000000000000
0010
Time: 20 | a: 00000000000000000000000000000010, b: 00000000000000000000000000000001, sum: 00000000000000000000000000000
0011
Time: 30 | a: 00000000000000000000000000000011, b: 00000000000000000000000000000011, sum: 00000000000000000000000000000
0110
** Note: $finish   : adder_tb.sv(26)
Time: 40 ps Iteration: 0 Instance: /adder_tb
D:\fpga_projects\net_adder>

Thank You