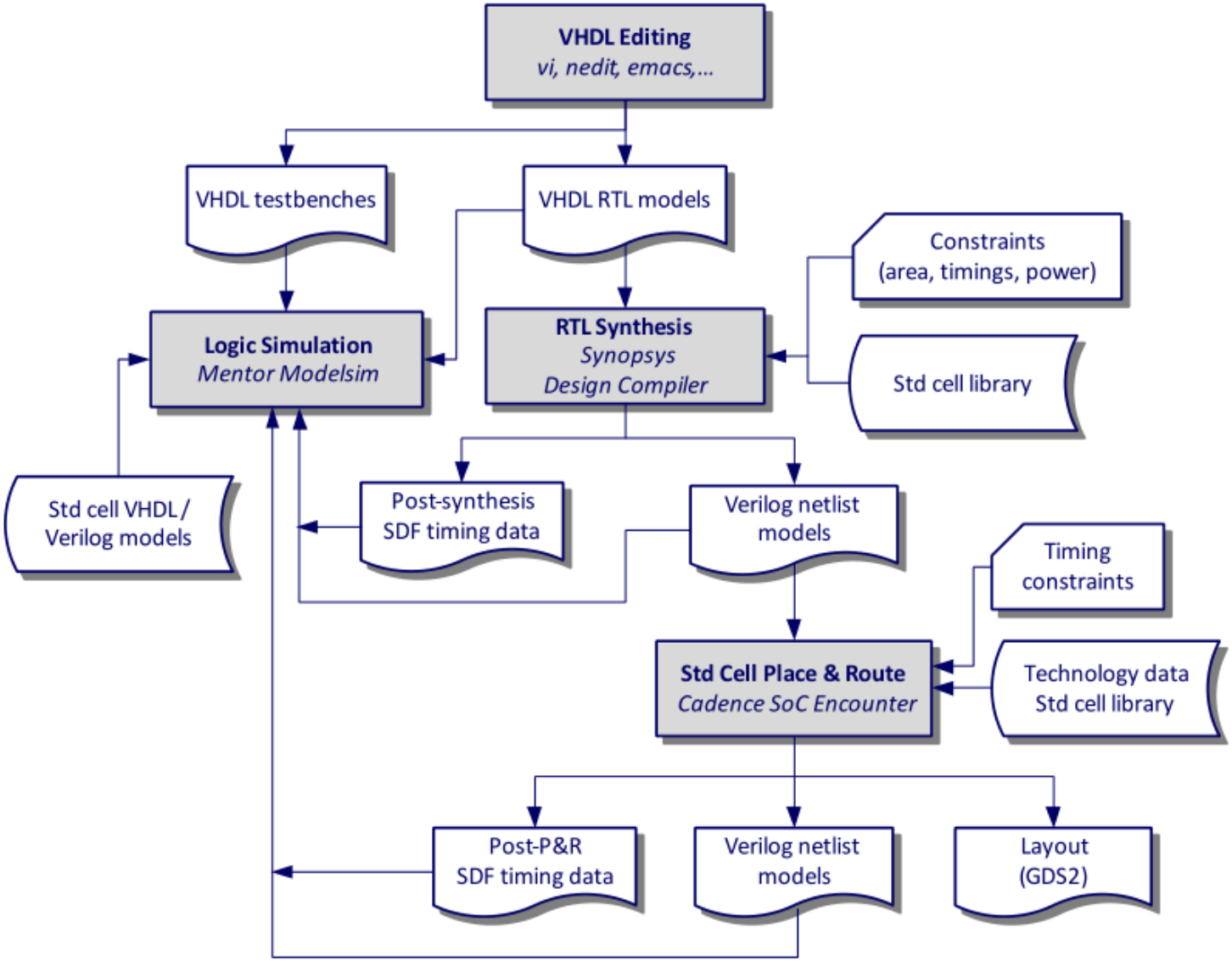
1st Students' Meeting
Something about the Digital Flow
I/II

Serena Panati

Monday, January 20, 2014

DF Flowchart



HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993

- **Verilog** - Std. IEEE 1364-2005

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993
  - *VHSIC Hardware Description Language*
- **Verilog** - Std. IEEE 1364-2005

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993
  - *VHSIC Hardware Description Language*
  - developed at the U.S Department of Defense (1980)
- **Verilog** - Std. IEEE 1364-2005

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993
  - *VHSIC Hardware Description Language*
  - developed at the U.S Department of Defense (1980)
- **Verilog** - Std. IEEE 1364-2005
  - created by Prabhu Goel and Phil Moorby in 1984

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993
  - *VHSIC Hardware Description Language*
  - developed at the U.S Department of Defense (1980)
- **Verilog** - Std. IEEE 1364-2005
  - created by Prabhu Goel and Phil Moorby in 1984
  - first named *Automated Integrated Design Systems* (later renamed to *Gateway Design Automation* in 1985)

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

HDL Model

Hardware Description Languages:

- **VHDL** - Std. IEEE 1076-1993
  - *VHSIC Hardware Description Language*
  - developed at the U.S Department of Defense (1980)
- **Verilog** - Std. IEEE 1364-2005
  - created by Prabhu Goel and Phil Moorby in 1984
  - first named *Automated Integrated Design Systems* (later renamed to *Gateway Design Automation* in 1985)
  - *Gateway Design Automation* was purchased by **Cadence Design Systems** in 1990.

Text editors: gedit, nedit, emacs, vi...
Examples: encoder 8b/10b + non-synthesizable VHDL code $\star$

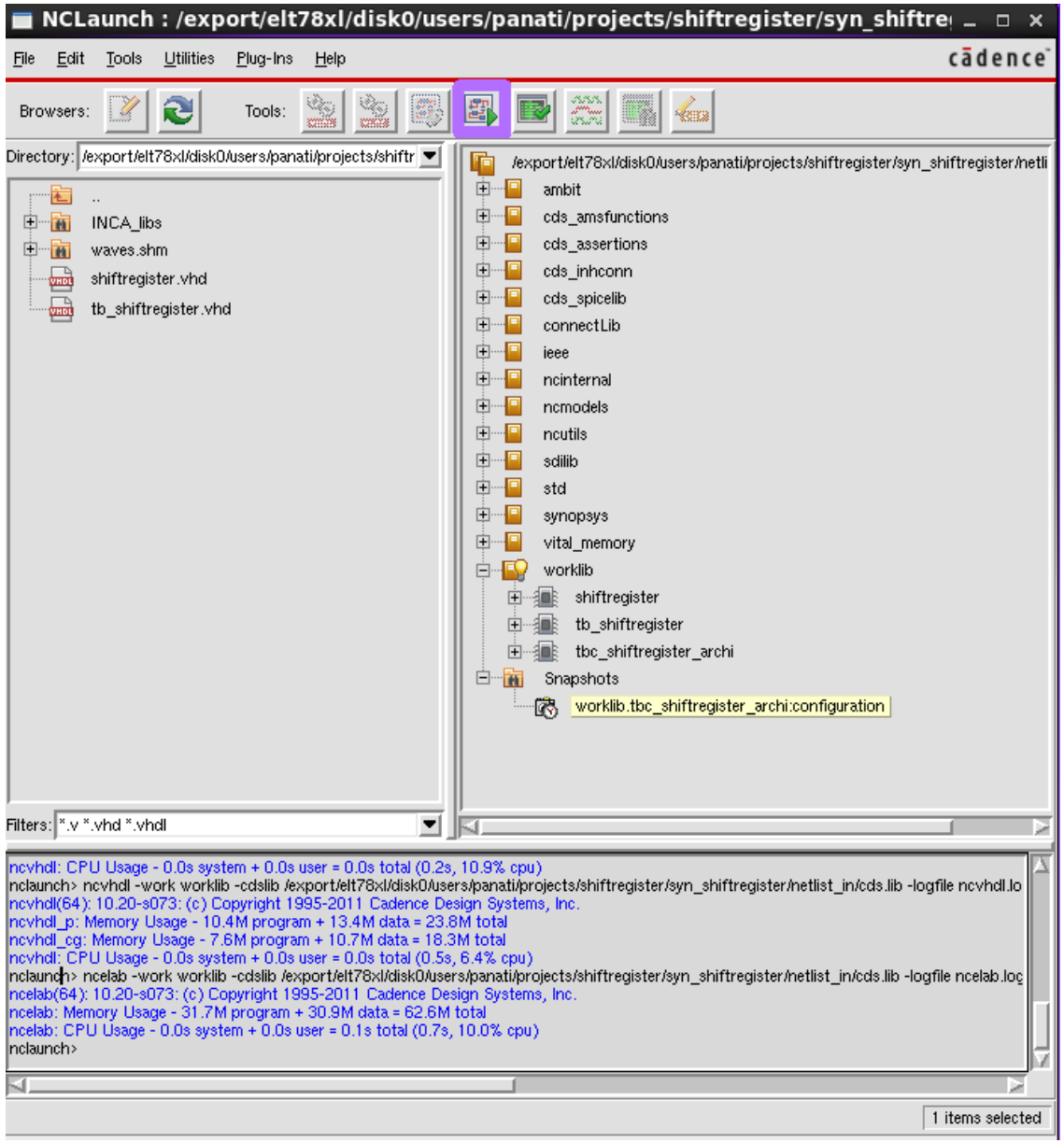
Pre-synthesis logic simulation

→ Cadence IC 6.1.5 (also IC 6.1.4 is ok)

Tools:

- **NCLaunch** (compiler + elaborator)
- **SimVision** (simulator)

Pre-synthesis logic simulation



cds.lib and hdl.var - Examples

- **cds.lib** $\star$
- **hdl.var** $\star$

cds.lib and hdl.var - Examples

- **cds.lib** $\star$
  - ASCII text file
- **hdl.var** $\star$

cds.lib and hdl.var - Examples

- **cds.lib** $\star$
  - ASCII text file
  - defines which libraries are accessible and where they are located
- **hdl.var** $\star$

cds.lib and hdl.var - Examples

- **cds.lib** $\star$
  - ASCII text file
  - defines which libraries are accessible and where they are located
  - contains statements that map logical library names to their physical directory paths
- **hdl.var** $\star$

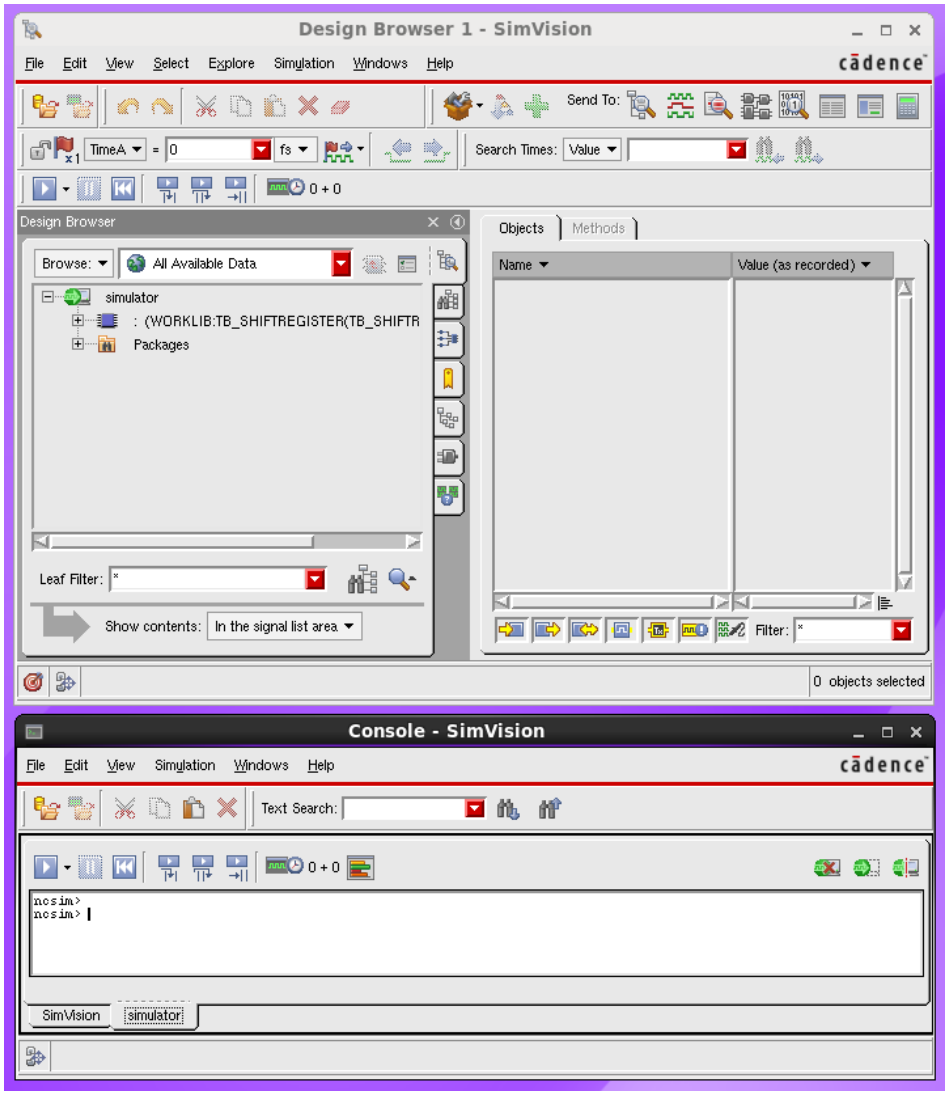
cds.lib and hdl.var - Examples

- **cds.lib** $\star$
  - ASCII text file
  - defines which libraries are accessible and where they are located
  - contains statements that map logical library names to their physical directory paths
- **hdl.var** $\star$
  - optional ASCII text file

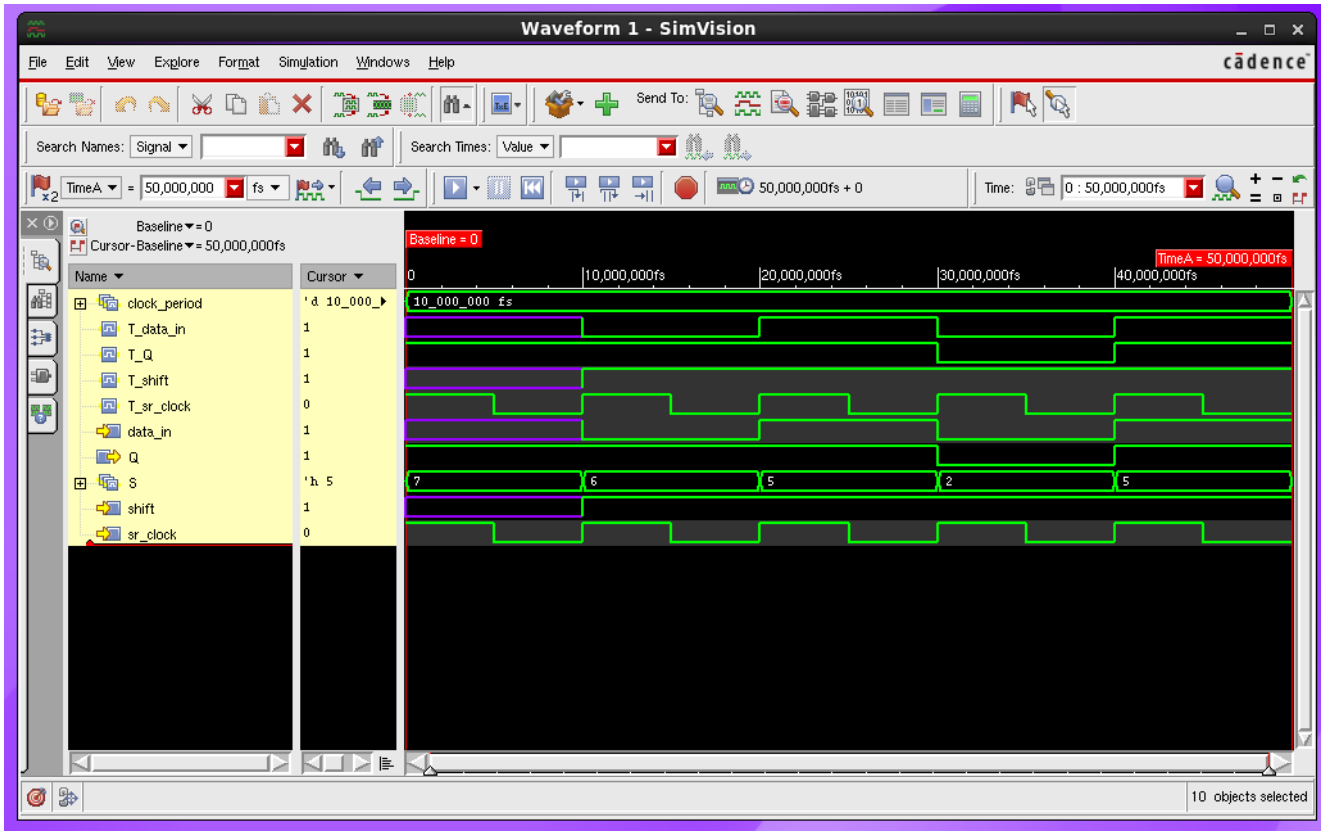
cds.lib and hdl.var - Examples

- **cds.lib** $\star$
  - ASCII text file
  - defines which libraries are accessible and where they are located
  - contains statements that map logical library names to their physical directory paths
- **hdl.var** $\star$
  - optional ASCII text file
  - contains definitions for system variables. One of these variables is the WORK variable that specifies the work library in which the compiler stores compiled objects and other derived data.

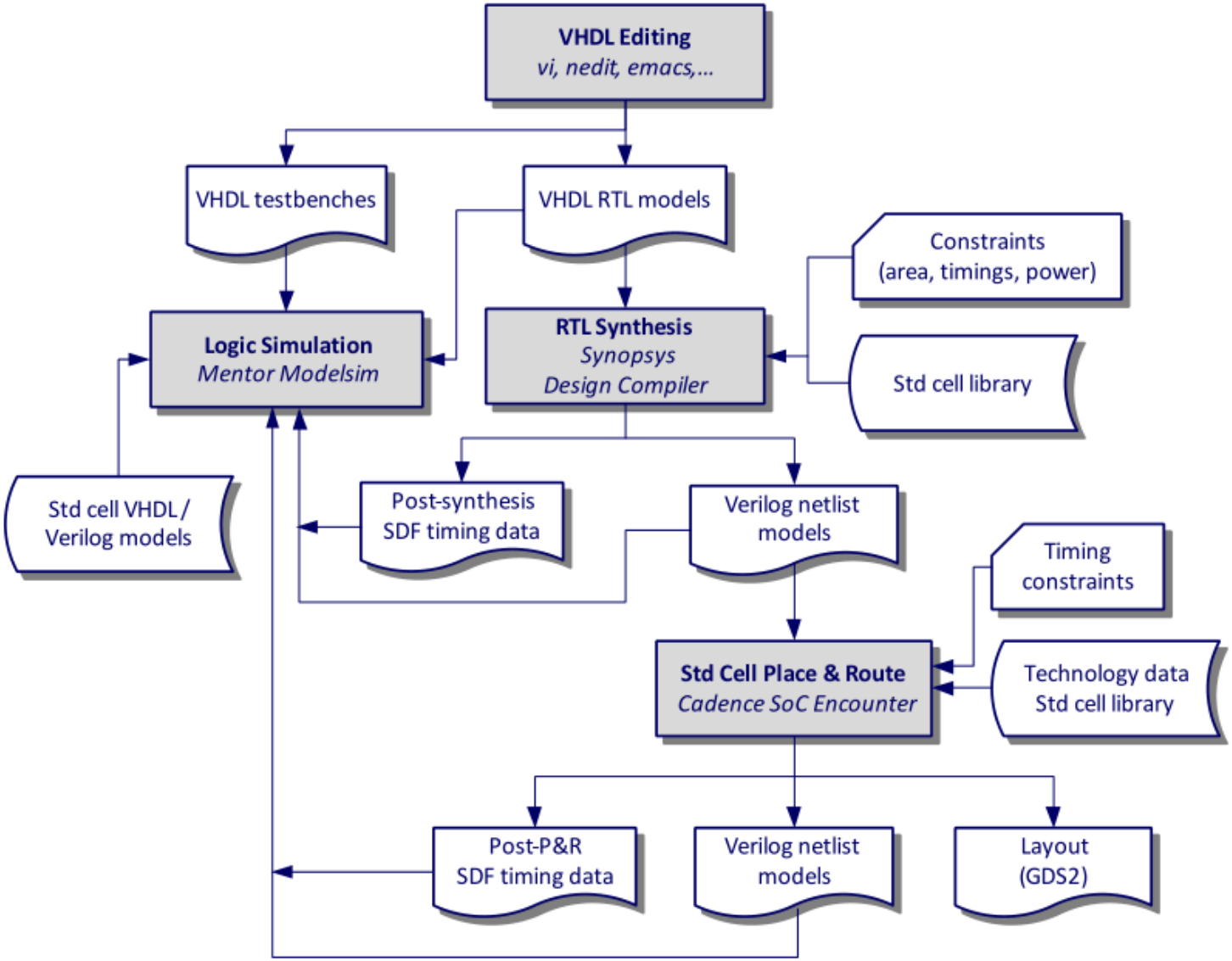
Pre-synthesis logic simulation



Pre-synthesis logic simulation



DF Flowchart



Register Transfer Level Synthesis

$\rightarrow$ Cadence EDI 9.12

Tools:

- **Encounter RTL Compiler**

What we need to define:

- timing $\rightarrow$ .sdc scripts $\star$

Register Transfer Level Synthesis

$\rightarrow$ Cadence EDI 9.12

Tools:

- **Encounter RTL Compiler**

What we need to define:

- timing $\rightarrow$ .sdc scripts $\star$
- constraints (area, power...) $\rightarrow$ .tcl scripts $\star$

Register Transfer Level Synthesis

→ Cadence EDI 9.12

Tools:

- **Encounter RTL Compiler**

What we need to define:

- timing → .sdc scripts ⋆
- constraints (area, power...) → .tcl scripts ⋆
- process corners libraries (.lib, .lef, .CapTbl) ⋆

Register Transfer Level Synthesis

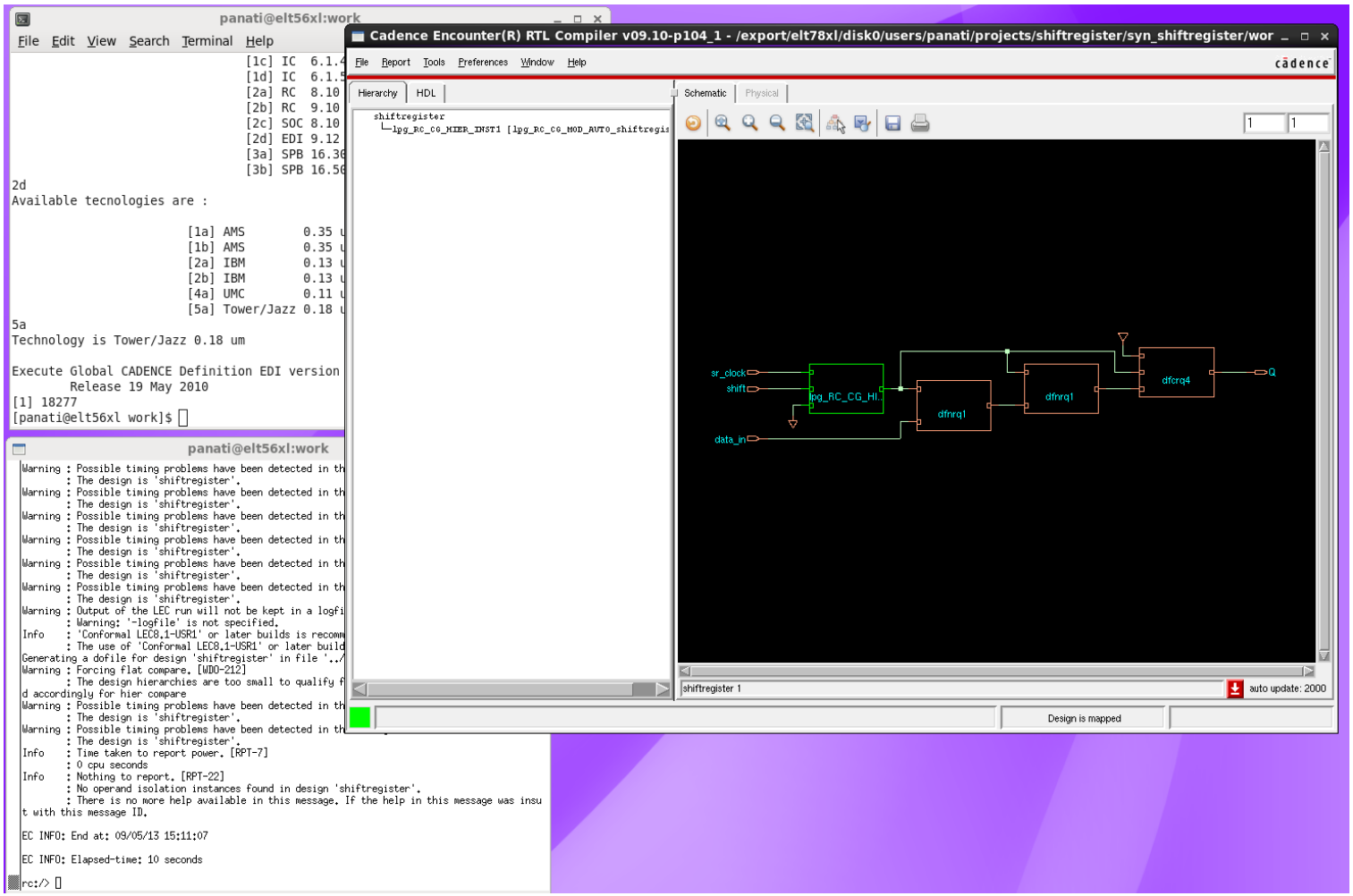
LEF: Library Exchange Format (LEF) is a specification for representing the physical layout of an integrated circuit in an ASCII format. It includes design rules and abstract information about the cells. LEF only has the basic information required at that level to serve the purpose of the concerned CAD tool. It helps in saving valuable resources by providing only an abstract view and thus consuming less memory head. LEF is used in conjunction with Design Exchange Format (DEF) to represent the complete physical layout of an integrated circuit while it is being designed. LEF was developed by Cadence Design Systems.[via Wikipedia]

PROCESS CORNERS: In semiconductor manufacturing, a process corner is an example of a design-of-experiments (DoE) technique that refers to a variation of fabrication parameters used in applying an integrated circuit design to a semiconductor wafer. Process corners represent the extremes of these parameter variations within which a circuit that has been etched onto the wafer must function correctly. A circuit running on devices fabricated at these process corners may run slower or faster than specified and at lower or higher temperatures and voltages, but if the circuit does not function at all at any of these process extremes the design is considered to have inadequate design margin. [via Wikipedia]

CAPTABLE: Captable file contains resistance and capacitance values which are used to model the interconnect parasitics of a design. Captable files usually contains resistance value and capacitance values. The capacitance values for a range of width and spacing of each metal used in the corresponding technology are defined. This information is used while extracting the nets.

GATES LIST EXAMPLE check it! $\star$

Register Transfer Level Synthesis



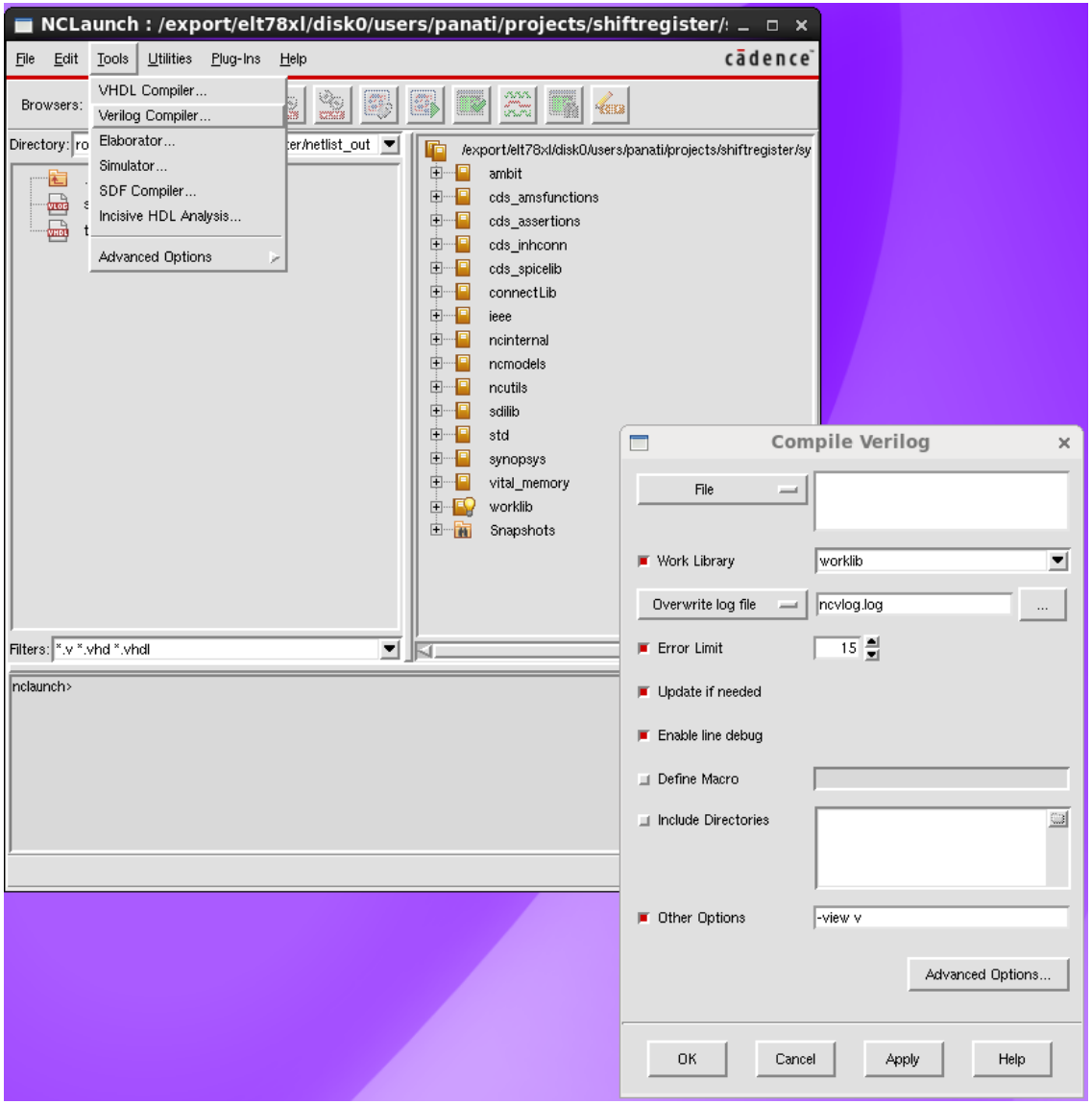
Post-synthesis logic simulation

Tools:

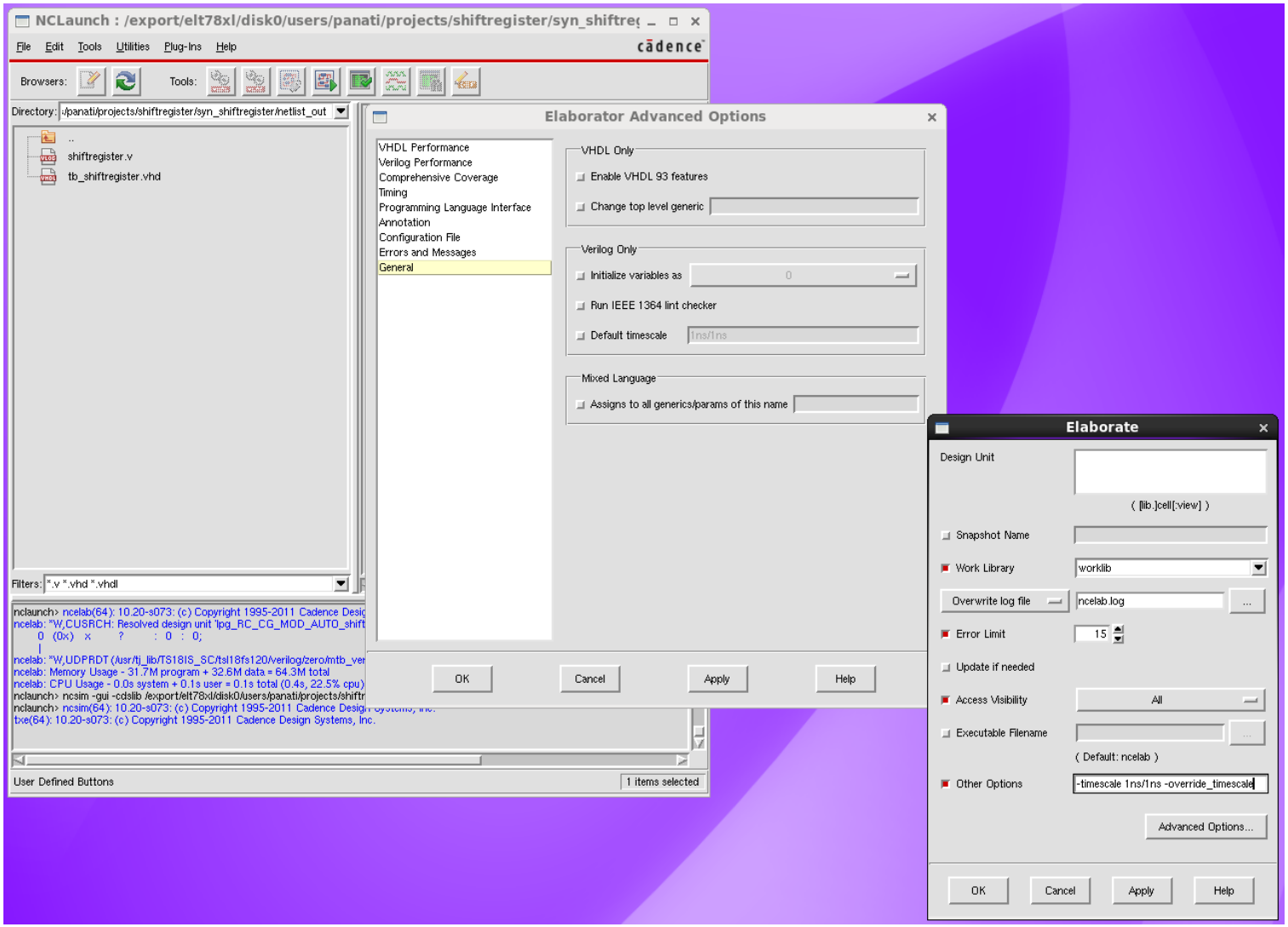
- **NCLaunch** (compiler + elaborator)
- **SimVision** (simulator)

→ Synthesis output: **Verilog!** Some trouble with the VHDL test bench...

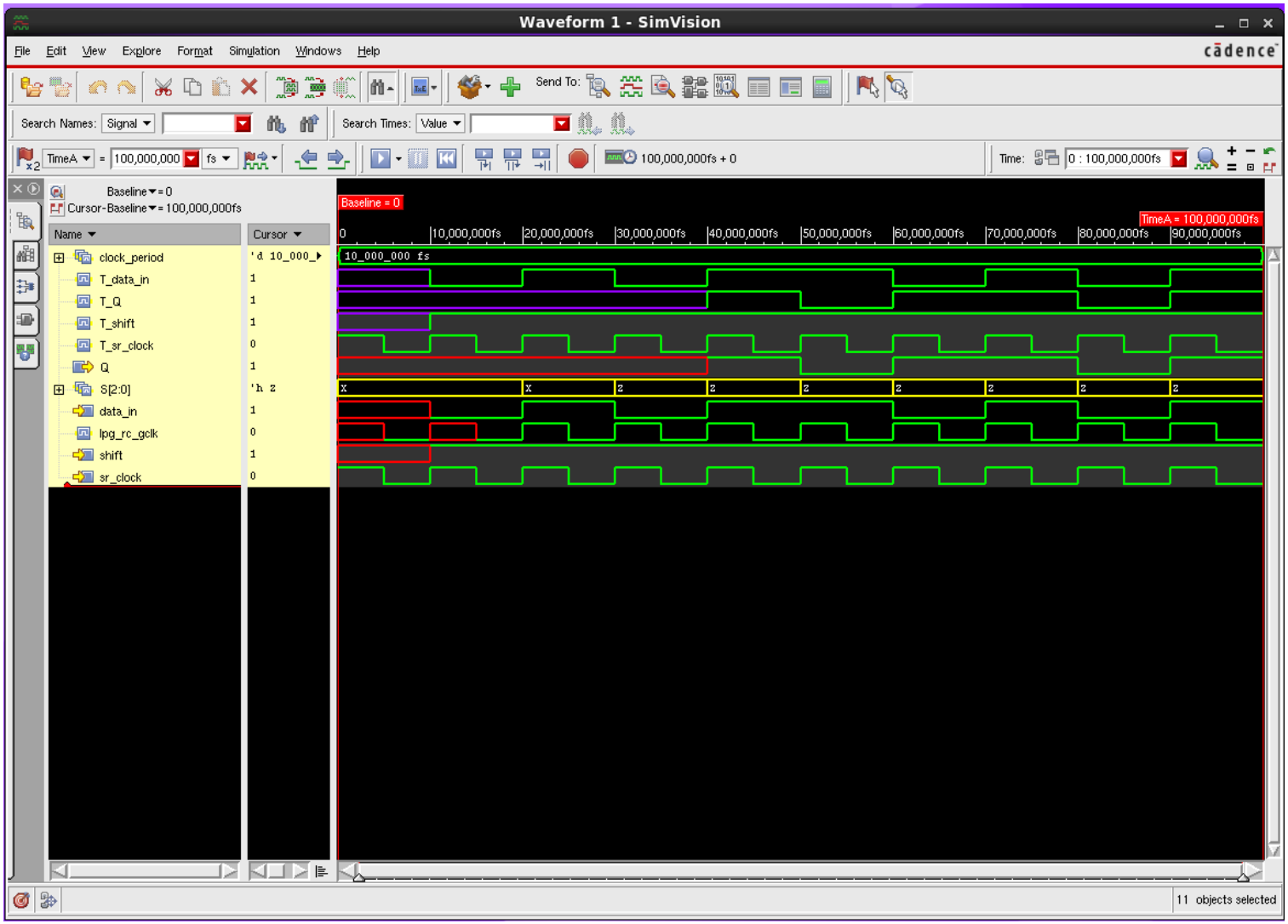
Post-synthesis logic simulation



Post-synthesis logic simulation



Post-synthesis logic simulation



Links and resources

- Top Down Digital Flow Tutorial:
  http://lsm.epfl.ch/files/content/sites/lsm/files/shared/Resources%20documents/TopdownDF.pdf
- Cadence Tools Overview (not only digital...):
  http://www.phys.hawaii.edu/∼varner/PHYS476_Spr10/Lectures/CadenceTools.pdf
- Cadence NCLaunch Tutorial:
  http://www.ee.virginia.edu/∼mrs8n/soc/nclaunchtut.pdf

Next meetings

- Automatic place and route (PnR)

Next meetings

- Automatic place and route (PnR)
- Simulation post-PnR

Next meetings

- Automatic place and route (PnR)
- Simulation post-PnR
- GDS II $\rightarrow$ Virtuoso

Thank you all, guys...