**ASIC Synthesis**

Synthesis of the RTL-level designs into a gate-level Verilog netlist is performed by means of Synopsys Design Compiler (SDC) and a commercial 45nm CMOS technology process library (PCS/my\_design/synthesis/). A set of Tcl scripts supporting the synthesis flow of the design, in particular **flow\_defines.tcl**, **flow\_synth\_v2.tcl** and **flow\_write\_spice.tcl**, which can be found as a part of either **64b66b/** or **8b10b/** implementations. These also provide capabilities of automated synthesis process to be performed with any other technology library.

**flow\_defines.tcl** file provides linking to the definitions, logical descriptions and timing information of various logical gates (standard cells) of this particular library. In general case, these should be substituted with the corresponding cell library location information and environmental variables.

The translation of RTL-level design into a gate-level semantic is performed via invocation of commands specified in **flow\_synth\_v2.tcl** understandable to Synopsys Design Compiler. Each of synthesizable constructs is optimized against a specific design rules (defined partially in **flow\_synth\_v2.tcl**), optimization constrains (fanout, area and timing requirements for all design paths are specified in **flow\_defines.tcl** and **Sources/module\_name.sdc** Tcl scripts) and environmental descriptions (variations in voltage, temperature and modelling wire loads – all provided by **flow\_defines.tcl** and CMOS library itself).

The final result produced by Design Compiler is optimized on the gate-level and mapped to the library cell descriptions of the technology process in use (saved as Verilog HDL gate-level netlist).

Once the synthesis process has been completed **flow\_synth\_v2.tcl** script generates various report files providing: timing and area information, cells used in the design, clocks, ports and buses information, constraints, and other information. The obtained logs together with the synthesized designs are kept in **out.synth/** folder.

Timing closure analysis of synthesized designs is a major consideration that allows the circuit function properly. Synopsys DC performs static timing analysis checking the design paths for timing violations under worst conditions. There are common ways to rectify the setup and hold time violations inside the circuit, which are not described in this brief. The obtained log files (**out.synth/**) as well as Design Vision graphical interface provide a convenient way of design debugging and examination.

Once all design objectives are met, the circuits can be presented in two main forms: Verilog HDL netlist and HSPICE (generated using **flow\_write\_spice.tcl** script) gate-level netlist. The next step considers checking of these against the original RTL designs in simulation environment.

**Getting started:**

1. Place your RTL-design (design.v) into **Sources/** folder and create the corresponding \*.sdc file (design.sdc) specifying the synthesis timing requirements for the design;
2. Invoke the $$ **./run\_synthesis.sh design** command via specifying the design name that needs to be synthesized. To run the batch mode synthesis, execute $$./GO\_Synthesis.sh command; make sure that the files inside the script are listed correctly; by default, $$./GO\_Synthesis.sh command inside PCS/codec/synthesis/ with run a batch synthesis for all designs used with this PCS implementation;
3. Make sure the timing objectives are met after the synthesis process.

**Post-synthesis verification**

In order to verify the correctness of produced hardware during the synthesis process and its correspondence to the initial HDL descriptions, the verification environment has been created (PCS/my\_design/simulation/). Testbench files, that are stored in Sources/behavior/ folder can instantiate the modules provided and test them with custom input patterns (Sources/input\_pattern). The tests comparing the behaviour of synthesised modules against results obtained from corresponding RTL-code simulations are performed by means of Mentor Graphics ModelSim via invocation of **run\_simulation.sh**. The script interacts with **sim\_modelsim.fdo** or **sim\_modelsim\_behav.fdo** files depending whether a gate-level or RTL level verification tests need to be produced. Any other simulation environments can be used as an alternative.

During the simulation process ModelSim environment can produce **Value Charge Dump (VCD)** files to log switching activity of the entire module under analysis or any functional block composing the circuit. To generate VCD files incorporate **$dumpfile()** and **$dumpvars()** statements inside the testbench modules using corresponding module instantiation names. The basic use of this functionality can be found in **Sources/behavior/testbench.v** testbench module.

Switching activity file is further used for power estimations of Verilog-represented gate-level circuits performed in collaboration with Synopsys PrimeTime PX suite.

HSPICE gate-level netlists can be also used for design verification and power analysis. Please refer to PMA part of the toolkit for representative examples.

**Power estimations**

Power analysis is initiated by **design\_power.scr** script file and performed by means of Synopsys PrimeTime PX suite. The tool requires synthesized gate-level netlist, switching activity file, recorded by the corresponding simulation tool and technology process library specifications as input parameters.

The total power consumed by a design is classified into static and dynamic power groups, and is evaluated on the basis of the declared number of combinatorial and sequential elements composing the circuit. While the static or leakage power is considered to be constant over the entire simulation period (it is only dependent on voltage, temperature and state of transistors), the dynamic power tends to depend on the frequency of the logic state transitions. Further, the dynamic power analysis is performed on the cell basis.

The internal power model of the CMOS cells is defined by technology process library, while the dynamic switching power is estimated as a result of state transitions on cells outputs. Latter power group classes are evaluated over the entire simulation period and can be represented as energy-time evolution waveforms.

**Getting started:**

1. Update the **sim\_modelsim.fdo** and **sim\_modelsim\_behav.fdo** files with the corresponding locations of all the design files used in the implementation; specify the location testbench file and linking to the technology process library use. Modify simulation and waveform analysis options for Mentors ModelSim.
2. Update the **design\_power.scr** power script with the corresponding design and technology process library information.
3. Update the testbench file with corresponding dump variables /output files to record the switching activity of the circuit. Specify the \*.vcd file name in design\_power.scr.
4. Update run\_simulation.sh script file to link the Synopsys and Mentors tools to the design. Run $$ ./run\_simulation.sh to start simulation and power analysis.