ASIC Design of the OpenSPARC Floating Point Unit

In the *ASIC Design I* course, basic design flow steps of IC design were covered starting with a given RTL description (such as a VHDL, Verilog file) until timing verification. The steps of the IC physical design flow, including logic synthesis, floorplanning, placement, clock tree synthesis and routing, were introduced. In this project, you are asked to complete the ASIC design of actual processors using the Synopsys design flow. The RTL description for the following processor is provided for the ASIC design project:

- `OpenSPARC™ T1`.

In March 2006, the complete design of Sun Microsystems' UltraSPARC™ T1 microprocessor was released—under OpenSPARC™ T1 name and in 2008, its successor, OpenSPARC™ T2, was also released in open-source form. These are the first 64-bit microprocessors ever open-sourced. They are also the first CMT (chip multithreaded) microprocessors ever open-sourced.

The core is provided with extensive documentation from the Synopsys community. Teams of two-to-four back-end designers will work together towards a common goal. Team of one is acceptable. Teams are expected to use the ASIC design from Synopsys tools, where support is available. Documentation is available on BBLearn and in previous lab manuals.

Deliverables on the project for 50% of the total credit are:

- Project report (not to exceed 10 pages), which is similar to the lab reports and should include the following information **in tabular format**:
  - DC compiler:
    - Number of cells in the design,
    - Dynamic/leakage power consumption at gate level.
  - IC Compiler:
    - Floorplan and setup stage details,
    - Core utilization,
    - Clocking and powering schemes (Critical paths, clock generation, clock distribution, buffering, VDD/GND rings and straps),
    - SDC file,
    - Report of timing violations after CTS, performed fixes and final timing report (briefed),
    - Power estimates (model, worst case, average case),
    - Routing layers information,
    - DRC, wirelength and congestion results after routing.
  - Primetime:
    - Critical paths information with graphical highlights on schematic and verbal description,
    - Delay insertion and global clock skew.
    - ECO changes for timing closure.
- Presentations of progress, final status and improvements after final status.

- Electronic versions of the scripts to perform your back-end ASIC design work. Comments within the scripts printed onto the screen are especially welcome to make is easier to follow your flow. For instance, run script design.scr, which prints "Working on Synthesis" onto the screen, pauses for user input upon completion, outputs "Working on Placement on X by Y mm area" and continues, etc.

Deliverables for the 50% of the total credit will be assigned through *coolness* points. The coolness will be judged by the instructors might include a wide range of accomplishments including (but not limited to):

- Lowest power, smallest area, highest speed implementations (compared with each other and against other team's designs as well),
- Various ECO updates for low power or improved slack with extensive documentation. For instance, perform *useful skew*, report changes to skew as well as to the circuit,
- Performing an ASIC implementation to have multiple cores on the same chip. Check *opencores.org* for ideas.
- Using a tool not covered in the labs, such as `Star-RCXT` for parasitic extraction (for more accurate simulation) or `Power Compiler` for low power implementation.
- Performing a transistor level simulation on your design using `Nanosim/HSPICE` to report power consumption on chip or clock tree only.
- Writing a script to report the location of clock tree buffers and sinks, min and max data path delays between them and the propagated clock delays (for clock skew scheduling),
- Comparing Cadence `SOC Encounter` (placement and routing) results with Synopsys `IC Compiler` on identical circuits.
- Performing DRC/LVS checks, using padframes to prepare a design ready-to-ship for manufacturing (tape-out).

The "coolness" component of the project can be completed within the context of the course coverage. However, the definition is left open-ended for those "student-designers" willing to experiment with the tools or work on other cores from OpenCores such as a DSP core. The students are encouraged to seek input from peers, instructors, graduate students and other resources (*e.g.* IEEE publications) for ideas. Note that not all entries above have the same "coolness" nor will the same entry have similar coolness for groups of varying size or academic standing (e.g. undergraduate v.s. graduate).

Synopsys 90nm library is recommended for the ASIC design.

Project lecture files are on WebCT under folder "Project". The project lab files are on the class server in the following path for `OpenSPARC`:

- `/home/DREXEL/yt74/ECEC473/OpenSPARC`