Ideas for Project 3

Your group has to pick one of the following ideas and communicate it to me by email before April 14th.

1) Design optimization (3 Groups)

This is an extension to the 2nd project (STA). Apply techniques described in the lecture to automatically optimize the design (mainly: gate resizing, buffering, and cloning). The optimizations target, mainly, timing then area. Your input is the design gate level netlist, the library and the results of the STA. The output is the optimized gate level netlist.

2) DEF Viewer (2 Groups)

Develop a web based (HTML and JavaScript) DEF viewer that displays the layout of the design before and after routing. The viewer inputs are the design DEF file and the library LEF file. The viewer has to be interactive so that:

- The user can get information about a cell by clicking on it
- The viewer can highlight cells from certain type (e.g., fill cells, flip-flops, buffers)
- The viewer can highlight the clock tree (as requested by the user)
- The viewer can highlight the power distribution network (as requested by the user)
- Generate power distribution rings (as requested by the user)
- Display/hide the wires (display all wires, display M1 wires, display M2 wires, etc.,)

3) Interactive Timing paths viewer/optimizer (manual) (2 Groups)

Display timing paths to the user with the slacks and the delays annotated on the gates. Also, the wires must be labeled by their delays. Allow the user to size up and size down the cells (from the available library cells) as well as inserting buffers. The tool must automatically update the slacks and the delays on the graph as the user modifying the path. Also, the tool has to generate a netlist to reflect the changes (ECOs) done by the user.

4) Enhance CloudV FSM Editor/Compiler (2 Groups)

Enhance the FSM editor (e.g., multi-lines support for the state, add tools bar, online error checking, etc.,). Enhance the FSM to Verilog compiler (e.g., better errors checking, better Verilog RTL structure, etc.,)

5) Introducing Automatic Clock Tree Generation to QFlow (2 Groups)

Automatically create a balanced clock tree:

- Place the design to get the locations of the flip-flops (clock tree end points).
- A network of buffers will be created to distribute the clock targeting zero clock skew (Verilog netlist).
- The synthesized clock tree will be placed as well as the flip-flops on an empty floor plan. The flip flops placements should be identical to those of step 1
- The synthesized clock tree will be merged to the original gate level net list.
- Finally, the rest of the cells will be placed on the design obtained from step 3.

6) Propose your own (open)

Have to discuss your group's idea with me to get my approval before April 14th.