Project Report - Group 13

Synthesis and Optimization of digital systems

Francesco Ricci	Samuele Yves Cerini	Flavia Caforio
252959	256813	257750

1 Script structure

The script is composed by three main parts:

1. **Dynamic power optimization - design downsize**: most of the design cells are downsized: this improves the dynamic power consumption (and leakage in some degree) but increases slack.
2. **Leakage optimization**: it performs the swap from LVT to HVT of non critical cells (ranking based on slack) until the leakage saving requirement is met.
3. **Slack optimization**: the size of selected cells is increased, to minimize their delay (and therefore improve the overall slack).

2 Script procedures

The procedures used in the script are:

- **savings_init**: computes and returns the initial leakage power consumption.
- **savings_check{savings leak_start_power}**: computes the current leakage power consumption, and performs the difference between the initial one (computed by savings_init). It returns 1 when the savings are achieved, otherwise 0.
- **LVT2HVT{cell_list}**: given a cell list, it swaps each cell from LVT to HVT.
- **HVT2LVT{cell_list}**: given a cell list, it swaps each cell from HVT to LVT.
- **leakage_opt2{savings N leak_start_power}**: it finds all the remaining LVT cells in the design and ranks them in terms of slack. Then, performs the swap of N cells at a time (using LVT2HVT procedure).
- **LVT_remaing{}**: computes how many LVT cells are still present in the current design. It returns the difference between the total number of cells and the remaining LVT cells. The reason behind this choice is that is not advantageous to swap all LVT cells (in terms of delay, and power consumption) and we also observed that some LVT cells have not a corresponding HVT one.
- **upsize{cell_list percentage} - downsize{cell_list percentage}**: these procedures swap a given cell (or cell list) with a different cell (whose size depends on percentage passed as input).
- **getworstcells{numberofpaths}**: it extracts the cells composing the current critical paths and deletes eventual double occurrences. Then it ranks them following a certain cost function. Used in slack optimization.
- **slack_finder{}**: it saves a timing report and parse it to find the current slack of the design.