Synthesis and Optimization of Digital Systems – 2022/23

Low Power Contest Report

GROUP 14
Bidone Matilde - s317729
Marchei Alessandro - s314807
Terzano Tommaso - s317650

1. Introduction

The goal of this project was to design and test a TCL command for PrimeTime capable of executing a post-synthesis power optimization. The script uses the static dual Vth optimization tecnique, which exploits two twin technology libraries, LVT and HVT.

The first one describes a set of cells with a low threshold voltage, ensuring a low propagation delay but at the same time causing an high leakage current and thus an high power consumption.

On the contrary, HVT cells are designed to have a high threshold voltage, producing a high propagation delay but at the same time enabling a low power consumption.

Since the cost function defined by the contest's rules includes as a parameter also the CPU time, it was another important factor to take into account while developing the algorithm.

With this in mind, we fine-tuned our script paying attention on one hand to maximize the power savings, on the other hand to execute the script in a reasonable time.

2. Algorithm

Our strategy consisted in various steps, in order to maximize the chances to achieve a successful optimization. The main idea is to start from a circuit synthetized using all LVT cells, and then to swap to HVT only those cells having the highest slack with some iterations. The steps followed by our algorithm are:

1) Initialize every cell with the LVT library in order to get the fastest possible solution.
2) Build a list of cells ordered by decreasing slack. This step is done only one time at the beginning, in order to save CPU time.
3) Swap the 2% highest slack cells to HVT library. We saved this number (2% of the total number of cells) in a variable and then the algorithm works on that value and not on the percentage anymore. In other words, at the next iteration, if not violations are recorded, the number of cells to swap will remain the same and will not be the 2% of the remained LVT cells. We decided to use as initial value a percentage value and not a fixed one because in this way it is more adaptable on different kinds of circuits. We did a lot of tests to find the better percentage, and this value seems to be a good choice.
4) Check if the contest's constraints are met.
5) If one violation is recorded, swap back the HVT cells of the last transformation to LVT, divide the number of swapped cells by 2, and rerun the transformation. Then return to 4).
6) If all the constraints are met, return to 3).

In order to save CPU time, the algorithm also uses a counter for the number of iterations done. We decided that in any case after 3 iterations the main function returns.

For saving CPU time, we also decided that the main function returns if the normalized power (i.e., leakage_final/leakage_initial) becomes lower than 0.8.