Politecnico di Torino

Report

Matteo Battilana s281389
Salvatore Gabriele La Greca s281589
Giovanni Pollo s290136

1 Explanation

The idea of the code is to switch cells depending on two priorities in order to reduce leakage power, dynamic power and used area of a combinational circuit. The algorithm is divided into two main different steps, run in order:

1. Swap from LVTs to HVTs: changed accordingly to a certain priority
2. Swap performing a gate resizing: changed from a bigger cell to a smaller one based on a priority

At each change in both the two steps, the slack constraint is checked. If the new slack is lower of the allowed one, the last applied change is reverted.

1.1 Swap from LVTs to HVTs

The first thing that is executed is the computation of the priority based on the leakage power of each cell. Higher leakage power means higher priority.

The selection of this type of priority is based on previous tests in order to decide the best one. Previous experiments were using priority based on the ratio between the difference of leakage power and difference of delays, between HVT and LVT. We noticed that using a much greedy priority we obtained better results since the computation is much faster.

Then, after the priority for each cell is defined, the swap can start. In order to avoid useless swaps, only LVT cells are replaced with the corresponding HVT. The swap is done using the PT command `size_cell` along with `update_timing -full`. The latter command is used in order to avoid mistakes in computation of the new slacks, since it forces a full timing update.

1.2 Swap performing a gate resizing

After the dual-Vth cell assignment, done in the previous step, the first thing that is executed is the computation of the priority based on area times the dynamic power of each cell. Higher value means higher priority.

$$P_{areadyn} = Area \cdot P_{dyn}$$

Then, after the priority for each cell is defined, the swap can start. This is performed step by steping starting from the original cell size and going lower and lower until new possible swaps are available. The swap is done using the PT command `size_cell` along with `update_timing -full`. The latter command is used in order to avoid mistakes in computation of the new slacks, since it forces a full timing update.

2 Benchmarks

Figure 1: Results using c432 library

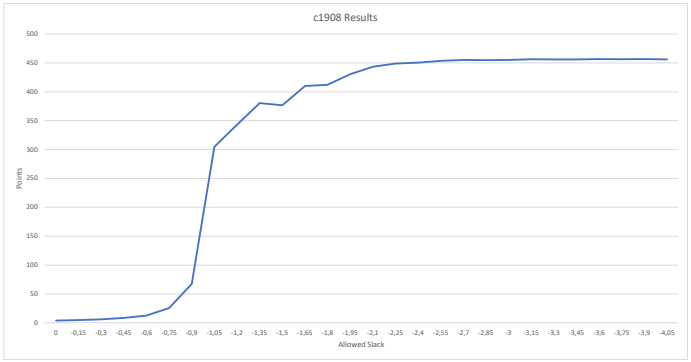

Figure 2: Results using c1908 library

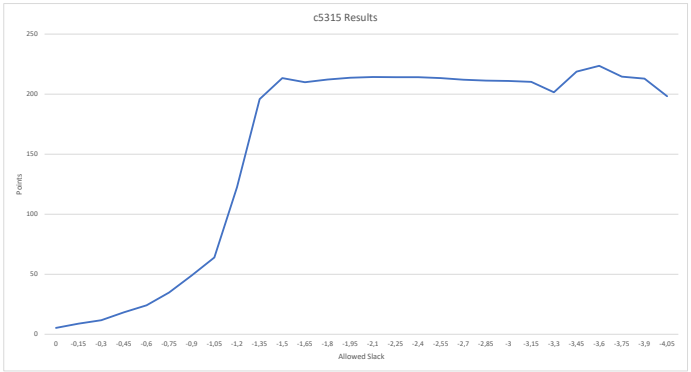

Figure 3: Results using c5315 library