UCLA EE201A Project: Automated Inter-Chip Pin Assignment

Tiefang Li - 205035362
Zaurbek Tsorojev - 805029443

I. Introduction

THE goal of this project is to develop a pin assignment tool in C++ using the OpenAccess API. We start with an initial default pin placement and improve it to minimize routing wirelength. This report describes our work.

II. Our algorithm

A. How it works

The algorithm we implemented works in the following way. First, for every macro in the design, check if its master cell has already been assigned. If it has, simply rotate the current macro to the angle achieving the smallest HPWL. If the master hasn't been assigned yet, the pins of the macro will be assigned for the first time. Here, for every pin in the macro, we will find its moving direction, i.e. the direction where to move the pin in order to reduce the wirelength. The way we do this is by taking the average position of all the pins connected to this pin. An example is illustrated in figure 1.

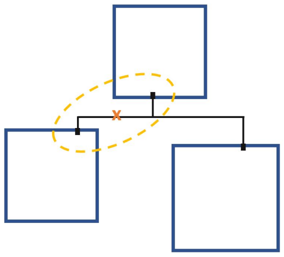

Fig. 1: Simulation results for the network.

For the pin on the far right on figure 1, the average position of pins it is connected to (represented by an orange "x") is on the top-left. However, as the pin is already on the top edge of the macro, its moving direction is then only left. The advantage of proceeding this way, is that we never calculate HPWL. We just find the direction once and then place our pins on the best legal position depending on this direction.

The next step is finding a valid pin position. For every pin, we calculate the max moving step according to the perturbation and the pin's moving direction. Then we find the new pin location based on this max step. After that, we check if the new pin's location is valid, i.e. at least at a distance of minimum pitch from any other pin of the macro. If the location is not valid, we go back step by step until the pin is in a valid location. Once it is valid, we physically move the pin to this position and we update the invalid region.

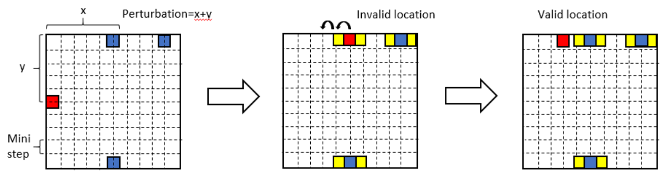

Fig. 2: Example of finding valid pin location

For example, in figure 2, we move the red pin to the top-right. The perturbation is between 10 and 11 min-steps, so the max move is 10 steps. If we move the pin by these 10 steps, the new location is in the invalid area (yellow) of the other pin (blue). So we move our red pin several steps back until we find a valid position.

Once all the pins of the current macro have been assigned, we find all the macros in the design that have the same master and give them the same pin assignment. Note that if macros have different orientations, you have to rotate the pin assignment to match it.

After doing this, we iterate the whole algorithm again, until there are unasigned macros left in the design.

B. Pseudocode

```
for every macro in the design do
    if master already assigned then
        Rotate macro to chose angle with min HPWL;
    else
        for every pin in the macro do
            Find pin direction;
            Calculate the max move;
            Find the initial new pin location;
            while pin location is invalid do
                Move the pin location one step back;
            end
            Update the invalid area array;
            Physically move the pin to this valid position;
        end
        for every other macro in the design do
            if macro has same master then
                Give it same pin assignment, that matches
                  macro orientations
            end
        end
    end
end
```

III. Results

Our improvements of mean wirelength (Wmn), maximum net wirelength (Wmax) and run time for every benchmark under three input rules are listed in following table.

Metric Score				
Benchmark	Input rules	Wmn	Wmax	Running time(s)
aes_top	min	0	0.060	1.563
	rand	0	0.063	1.71
	max	0	0.076	1.676
sbox_x2	min	0	0	0.348
	rand	0	0	0.373
	max	0.022	0	0.386
sbox_x2_vert	min	0.121	0	0.375
	rand	0.122	0	0.376
	max	0.134	0	0.39
sbox_x4	min	0	0	0.53
	rand	0	0	0.532
	max	0	0	0.525
des3_perf_opt	min	0.019	0.106	0.573
	rand	0.021	0.110	0.51
	max	-	-	-
mips_alu	min	0.136	0.009	1.646
	rand	0.113	0.033	1.677
	max	-	-	-
rockettile_x2	min	0.051	0.065	0.693
	rand	0.050	0.075	0.709
	max	0.052	0.074	0.716
Average	min	0.065	0.048	0.818
	rand	0.061	0.056	0.839
	max	0.058	0.043	0.739

Overall, we got around 6% improvement for wirelength mean and 5% improvement for maximum net wirelength. The average run time is less than 1s, which is pretty fast. We were ranked $n°9$ in the class with a score of 122.38/210.

Benchmarks sbox_2 and sbox_2_vert have the shortest run time ($\sim 0.37s$), while benchmarks aes_top and mips_alu have the longest ($\sim 1.65s$).

Some benchmarks were not improved because we decided to remove the macro rotation from our code as well as the diagonal moving directions. The reason for this is that, although our code worked for every benchmark, the rotation was slightly shifting macros for aes_top, so our design could not enter in Innovus. To avoid loosing all our points for this benchmark, we removed the macro rotation, which lowers all our wirelength metrics. We also removed the pin's diagonal moving directions, because a few pins were not respecting the minimum pitch.

IV. Methods we tried

Our research process was to first review some literature about the topic [1] [2] [3]. Based on that, the very first method we tried to implement was the concentric circles method [4]. We review several papers for that. However, we thought that this was not the most efficient method, because we are essentially simply rotating all the pins together. To achieve better improvements, we thought about creating 4 groups of pins (one group per macro edge), that we could move independently which gives more flexibility. We abandoned this idea, because it was still not flexible enough. Move the whole group together in one direction is inefficient as some pins will produce better results when they are moved in opposite directions.
Later, we briefly considered a topological pin assignment [4] [5], but quickly moved to design our own algorithm as we believed it would achieve better performances.

V. Improvements

The following can be done to improve our current implementation:

1) Correct the macro rotation for it to work in all situations.
2) Correct the diagonal direction moving to respect the minimum pitch and reduce the wirelength.
3) Take the macro that gives the best result as a reference for the hierarchy, instead of choosing a random one.
4) Set an order to move the pins, based on their wirelength and congestion improvement.
5) When a pin is connected to a net with multiple endpoints far from each other, copy this pin so that one will go closer to one group of outer pins and the copied pin will go closer to the other group. This will reduce the wirelength, but at the expense of a higher perturbation.

References

[1] T. Meister, J. Lienig, *"Novel Pin Assignment Algorithms for Components with Very High Pin Counts"*, Proceedings Design, Automation and Test in Europe, 2008, pp. 837-842.
[2] M. Pedram, M. M.-Sadowska and E. S. Kuh, *"Floorplanning with Pin Assignment"*, IEEE Conference: Computer-Aided Design, 1990, pp. 98-101.
[3] C. Kyung, *"Three-step pin assignment algorithm for building block layout"*, IEEE Electronics Letters, October 1992, Vol. 28, pp. 1882-1884.
[4] A. B. Kahng, J. Lienig, I. L. Markov, J. Hu, *"VLSI Physical Design: From Graph Partitioning to Timing Closure"*, Springer, 2011, pp. 83-85
[5] H. Nelson Brady, *"An Approach to Topological Pin Assignment "*, IEEE Transactions On Computer-Aided Design, Vol. CAD-3, no.3, July 1984, pp. 250-255.