An Algorithm for Large-Scale Analog Block Placement with Symmetry Constraints

Prepared by Shing-Weng Fang
Directed by Prof. Hung-Ming Chen

Department of Electronics Engineering

National Chiao Tung University

Hsinchu, Taiwan 300, R.O.C.

2007-01

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recently, analog circuits are developed rapidly in high technology industry. The most important issue for analog signals is the parasitic. We take account of the balance of layout-induced parasitic devices to avoid both degradation of power supply rejection ratio [1] and the balance of layout-induced parasitic since the analog circuits are sensitive to parasitic disturbances, crosstalk, substrate noise, power supply, etc. Placement symmetry can also be used to reduce the sensitivity of thermal gradients in circuits. Failure to balance thermal couplings in differential circuits can introduce unwanted oscillations. Therefore, in order to deal with mismatches, the thermally sensitive devices couples should be placed symmetrically in analog circuits. Indeed, the basic reason for symmetry constraints is the problem of process variations. Considering the basic reason, therefore, the cells of the same symmetry group that has a common axis have better to be placed together and we allow several symmetry groups in our floorplan.

The problem of placing devices with symmetry constraint[1,2,13,14,15] used simulated annealing as an optimization engine based on a packing representation. We classify the representations into two categories. (1) absolute representation, and (2) topological representations[2, 8]. In absolute representation, modules are represented by their absolute coordinates on the chip plane. Illegal overlaps will occur since no restrictions is made to the relative positions between modules. The main

disadvantages of using the absolute representation are the typically high running times and sometimes low-quality placement solutions, and also the need of an increased tuning effort due to the difficulty of predicting an appropriate weight for the overlap penalty. In topological representation, the relative positions between the modules are encoded. To compare with absolute representation, the solution space is much smaller, but complicated computations are needed for checking symmetry feasibility and adjusting the module positions to satisfy the constraints.

There are many ways for blocks placement with symmetry constraints, like TCG-S* [18],O tree [17], and sequence pair [4, 6]. We prefer the the way of sequence pair because of application of the method of symmetry-feasible. Both [4] and [6] offer approaches with sequence pair to dealing with the full cell placement with the symmetry constraints [2, 3]. However, In modern industry, we usually deal with large-scale circuit design. We implement the method of [4] and improve it by combining the symmetry cells first. Second, we know that the cost of placement based on rectangle-packing by the sequence-pair [5] becomes vast when the size is big, especially including symmetry constraints executed by linear programming. So we execute the simulated annealing with the positions of the symmetry and non-symmetry cells in a way of correlation.

1.1 Our Contribution

In this thesis, we implement our algorithm by two parts. First, we combine the cells of the same symmetry group for eliminating whether transitive edges are necessary or not. Second, to avoid the running time is vast due to the linear programming with symmetry constraints, we execute the simulated annealing with the positions of the symmetry and non-symmetry cells in a way of correlation. After that, we confirm our results with symmetry constraints by comparing with the paper, "Ana-

log Placement with Symmetry and Other Placement Constraints", with the same benchmark. As a result,when number of blocks increases, we get a more effective solution in the similar time compared with this paper costs. We will demonstrate the effectiveness of our approach by experiment result.

1.2 Organization of this Thesis

In Chapter 2, we will introduce the symmetry constraints in the cell placement with sequence-pair and even in the physical design, and describe how to handle with it, especially developing trends of analog design. And then describe how the symmetric cells and non-symmetric cells are separated in the floorplan and many kinds of symmetry in the physical design. Finally, we give the problem formulation. In Chapter 3, we discuss our method of allocating positions of symmetry cells and non-symmetry cells into floorplan and how to execute simulated annealing. Our experiment results are presented in Chapter 4. Finally, we give the conclusion of this thesis and future work in Chapter 5.

Chapter 2

Preliminaries

In this chapter, we will introduce the symmetry constraints in the cell placement with sequence-pair and even in the physical design, and describe how to handle with it, especially developing trends of analog design. And then describe how the symmetric cells and non-symmetric cells are separated in the floorplan and many kinds of symmetry in the physical design. Finally, we give the problem formulation.

2.1 Symmetry Constraints

The "symmetry groups" are composed of several blocks that exhibit a form of symmetry and they all share a common symmetry axis. And this symmetry axis can be horizontal or vertical. The " symmetry constraints" are represented by symmetry groups. The symmetry group may include several pairs of symmetry blocks and self symmetry blocks whose center must be placed on the symmetry axis. In a pair of cells placed symmetrically, both of the size and shape of the two blocks are the same, and every one is placed on one side of the axis. The symmetry condition is showed as figure 2.1[4].

About this topic of symmetry constraints, we will introduced two hot application including layout design and cell placement for analog designs bellow.

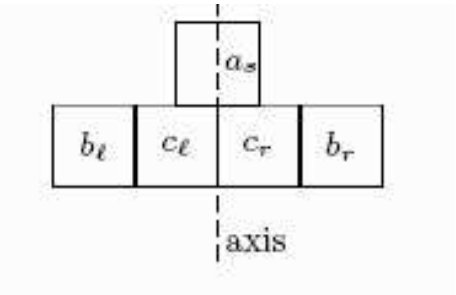

Figure 2.1: one self symmetry block (as) and two pairs of symmetry blocks :(bl , br) , (cl, cr)[4]

2.2 Layout Design with Symmetry Constraints

As cell placement, device matching and layout symmetry are of utmost importance for high performance analog and RF circuits. Symmetric layout of matched transistor alleviates the effects of mismatch in analog/RF circuits. [9]

Recently, the issue of analog layout automation is a hot topic and template-based methods are effective in reuse layout automation for analog circuits such as operational amplifier. But in the generation layout with symmetry constraints of process, the detections for symmetric layouts generated from libraries become very complex as the layouts become large. Moreover, multi-level symmetry constraint generation has built for more effective retargeting large analog layouts [10]. In following, we will introduce the some kinds of devices laid-out symmetrically in trend.

One-dimensional cross-coupled symmetric multi-finger transistor pair(also called interdigitation or interleaveing symmetry): As shown in Figure 2.2, the two transistors, M1 and M2, are arranged symmetrically and the layout has 21 axes of symmetry and 66 unit transistor pairs.

Two-dimensional cross-coupled symmetric multi-finger transistor pair: As shown in Figure 2.3, the two transistors, M1 and M2, are arranged symmetrically and the layout has 12 axes of symmetry and 13 unit transistor pairs.

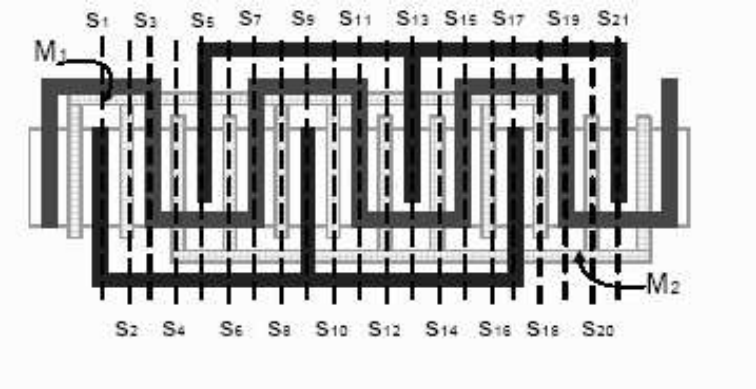

Figure 2.2: One-dimensional cross-coupled symmetric multi-finger transistor pair. The rectangles with dotted patterns represent the poly-silicon layer.[10]

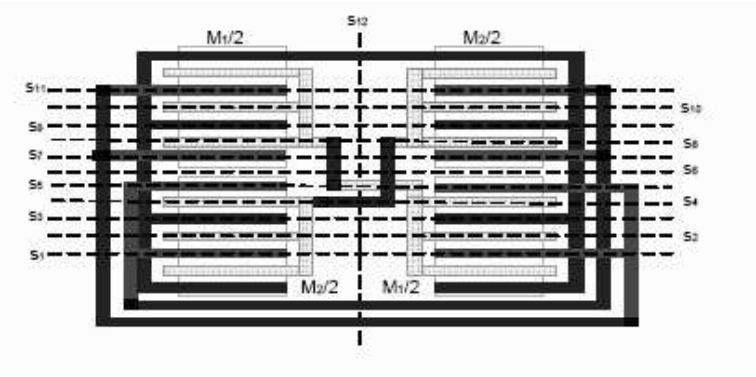

Figure 2.3: Two-dimensional cross-coupled symmetric multi-finger transistor pair. The rectangles with dotted patterns represent the poly-silicon layer.[10]

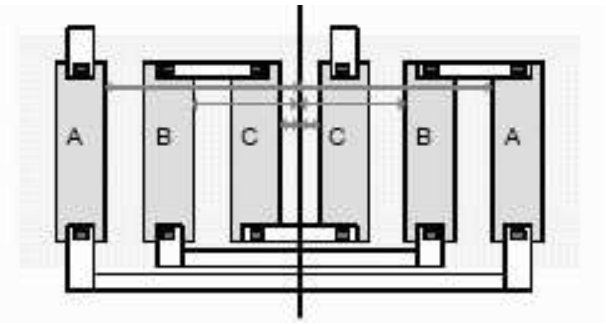

Figure 2.4: One dimensional symmetric layout of on-chip resistors [10]

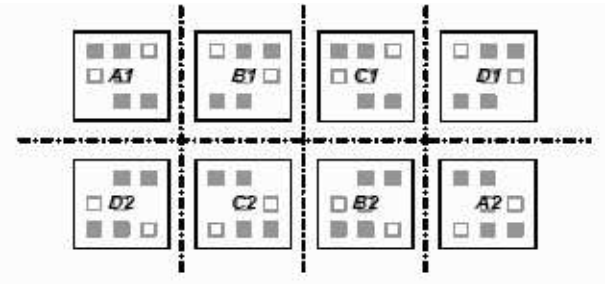

Figure 2.5: Four comparator subcircuits (A, B, C, and D) laid-out in a split-symmetric common-centroid layout. [10]

Matched passive devices: The resistor-chain layout ,as shown in Figure 2.4, including three resistors which are laid-out in an interdigitated fashion with one-dimensional common-centroid symmetry[11]. They would be identical resistances regardless of process and temperature gradients due to match between the resistors.

Matched block: Beside matched devices, blocks are also often identical by design in large analog circuits. For example, as shown in Figure 2.5, it is a 2-bits comparator circuit composed of 4 unit comparators. For high performance, they are laid-out in common-centroid fashion. That is, for all blocks of A1, A2, B1, B2, C1, C2 ,D1, and D2 may be laid-out with one or two dimensionally symmetric layouts or may be translated with respect to each other since all blocks are identical.

The layout for large analog circuits, we must cost large number of symmetry constraints due to the increase in layout size and complexity. This cost is too

expensive. Even considering layout of matched blocks shown in Figure 2.5, the layout in each block may comprise several devices with symmetry constraints like Figure 2.2 2.4, and it is that there are many unwanted symmetry constraints to be computed necessarily, especially in layout of matched blocks.

2.3 Sequence-Pair Placement with Symmetry Constraints

2.3.1 Sequence-Pair Representation Review

A sequence-pair [3] is a pair of module-name sequence like ((4,3,1,6,2,5), (6,3,5,4,1,2)), where 1,2,3,4,5,6 represents a module respectively. We can know the relationship between two modules from the two sequence-pair as follows:

((...*bi*...*b*...),(...*bi*...*bj*...)) means *bi* is on the left of *bj*

((...*bi*...*bj*...),(...*bj*...*bi*...)) means *bi* is below of *bj*

Figure 2.6 shows the grid and the corresponding packing. From the sequence-pair, we can translate it into the grid graph. From the grid, we can know the approximate location of each module and establish a horizontal graph and vertical graph, as showed in Figure 2.7. And then place all modules into the locations according to the horizontal and vertical graph.

The GH(V,E) (V: vertex set, E: edge set) called horizontal constraint graph is constructed as follows:

1) V: source s, sink t, and other vertices label 1,2,3,4,5,6

2) E: (s, x),(x ,t) and (x, x') where x' is right of x in the sequence-pair.

3) Vertex weight: zero for s and t, but the width of blocks for the other.

The vertical constraint graph can be constructed similar. When the number of block is n, the H/V constraint can be obtained in O(n2) time by applying the well-known longest path algorithm[7]. From [3], it is evaluated only in O(n log log n) time.

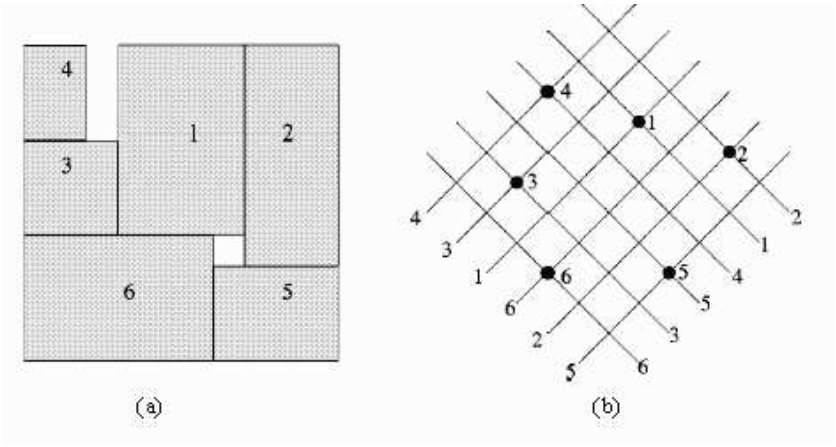

Figure 2.6: (a) Oblique grid for sequence-pair (X,Y) = ((4,3,1,6,2,5) , (6,3,5,4,1,2)). (b) The corresponding packing. [3]

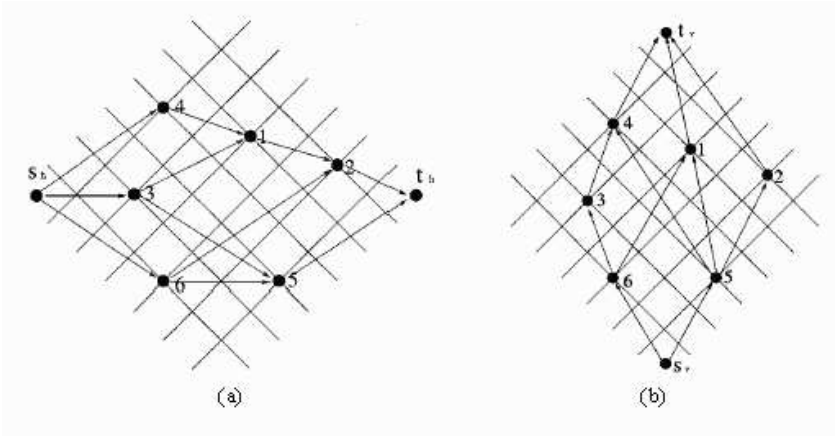

Figure 2.7: (a) In horizontal constraint graph GH(V,E), a path sh412th corresponds to <4 1 2>, a common subsequence of (X,Y) = ((4,3,1,6,2,5),(6,3,5,4,1,2)). (b) In vertical constraint graph Gv(V,E), a path sh634th corresponds to <6 3 4>, a common subsequence of (XR,Y) = ((4,3,1,6,2,5),(6,3,5,4,1,2)).[3]

2.3.2 Cell Placement Methods with Symmetry Constraints

From Balasa's placement method[2,8], if cells a x and y in a symmetry group in sequence-pair S satisfy

$$a^{-1}(\mathrm{x}) < a^{-1}(\mathrm{y}) \Leftrightarrow b^{-1}(\mathrm{sym(y)}) < b^{-1}\ (\mathrm{sym(x)})$$

where (a , b) is the sequence-pair for a symmetric group to a vertical axis,a^{-1}(x) is the position of the cell x in a, and sym(x) is the cell symmetric to cell x, the sequence-pair S is said to be symmetric-feasible[2]. If we choose y = sym(x) (it means x ,y are a symmetric pair), and sym(sym(x)) = x , we rewrite it:

$$a^{-1}(\mathrm{x}) < a^{-1}\ (\mathrm{sym(x)}) \Leftrightarrow b^{-1}\ (\mathrm{x}) < b^{-1}\ (\mathrm{sym(x)})$$

condition (S) shows that any symmetric pair of cells appears in the same order in the sequence a and b. If we choose two cells x ,y belonging to distinct symmetric pairs, we rewrite it:

$$a^{-1}(\mathrm{x}) < a^{-1}\ (\mathrm{y}) \Leftrightarrow b^{-1}\ (\mathrm{sym(y)}) < b^{-1}\ (\mathrm{sym(x)})$$

The symmetric cells appear in reversed order in the sequence a and b.

For example, a sequence-pair representation of placement configuration shown in Figure 2.8 [8] (a, b) = (BCEAFDHG,ABCDEFG). The symmetry group G = (E,F), (B,H)with two pairs of symmetric blocks satisfy the condition, so (a, b) is symmetric-feasible. And we also can find any distinct block x and y in G to satisfy it.

According to this symmetry group condition, the blocks of a horizontal symmetry group will appear in a mirror form in a sequence-pair, we can set that:

a = ...A1...A2...Ax...

b = ...sym(Ax)...sym(A2)...sym(A1)...

Similarly, for a vertical symmetry group, we can set that:

a = ...A1...A2...Ax...

b = ...sym(A1)...sym(A2)...sym(Ax)...

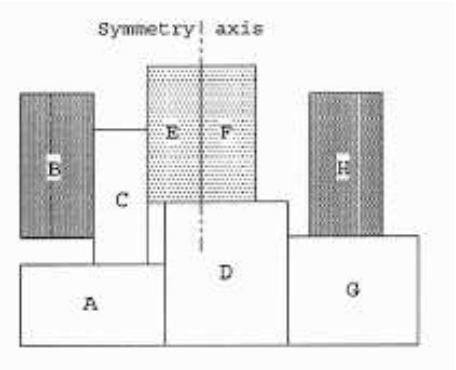

Figure 2.8: Placement with symmetry groups G = (E,F), (B,H)[8]

2.3.3 Previous Works

The authors in [4] studied the problem of cell placement with symmetry constraints for analog IC layout design and proposed methods to obtain a simple constraint graph [16] directly from a sequence pair in O(snloglog$n$+es) time and derive a set of linear constraints expressions from the graph,where the time cost loglog$n$ is utilized the method of obtaining the lower left corner packing [3]. And the time of es is the cost of the proposed algorithm SP-Core, where the SP-Core is to judge whether transitive edge are necessary or not. They also shorten the time required by linear programming, the number of variables and linear expressions are decreased by substituting expressions for dependent variables, and the resultant placement is obtained by solving the linear expressions using linear programming. If the symmetry axis is only vertical, they obtain the placement more quickly by vertical constraint graph based on a sequence-pair.

About symmetry constraints, the authors in [6] studied the problem and instead of handling the constraints by having a penalty term in the cost function to penalize violations, a unified method is proposed that, by adjusting the edge weights in a pair of constraint graphs, can try to satisfy symmetry constraints simultaneously in a candidate floorplan solution.

Given a sequence pair, it will be scanned initially to check if it will be a feasible solution satisfying all the constraints, where the initial condition for sequence pairs are generated only to satisfy the symmetry conditions. After this initial scan, a pair of constraint graph(Hh,Hv) will be built according to the sequence pair to represent the relative positions between the modules. Additional nodes di and constraint edges will be inserted to Hh and Hv to enforce the required symmetry constraints. Then a procedure "symmetry()" will be called to generate the packing and evaluate its cost finally if the "symmetry()" is feasible, where the "symmetry()" is for computations of the variable edge weights with symmetry constraints.

2.3.4 Problem Formulation

The problem we concerned is described as follows. Let M = *m1,m2...,mn* be a set of n rectangular blocks including k symmetry groups composed of pairs of symmetry blocks and self-symmetry blocks. Each block has its height *hi* and width *wi*. The blocks with symmetry constraints are placed symmetrically and contiguously with a common axis in the same symmetry group. The goal is to find an optimal floorplan with minimum cost in (area and wire-length). And all blocks are not allowed overlapped.

Chapter 3

Proposed Methodology

Our method is constructed by two parts. First, considering the blocks with symmetry constraints, they will be constructed with the linear program. Second, all blocks with the symmetric blocks then are constructed by [3] with sequence-pair. The objective is to construct a floorplan F with satisfying symmetry constraints and minimizing a cost function cost(F) = area(F) + wire(F) where is a user given weight, area(F) is total area of F and wire(F) is the total wire length of F measured by the half-perimeter estimation. The important issue is the set of moves in the annealing process. How to set the moves is the key point in order to match our proposed methodology.

3.1 Improved Sequence-Pair Packing for Symmetry Constraints

In sequence-pair packing, we will apply the method with fast evaluation of sequence-pair proposed by [3].

For given a sequence-pair (X, Y), considering placement with x coordinates, both X and Y have n blocks with *1, 2,..., n*. The weight *w(b)* is the width of block b. Block position array P[b], b = 1,2,...,n is used to record the x coordinate of block b. Match array *match*[i], i = 1,2,...,n is record that *match*[i] = j iff X[i] =

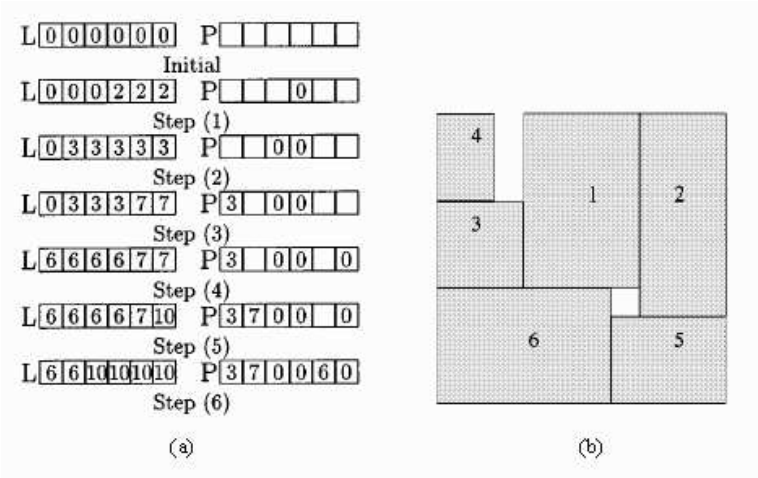

Figure 3.1: (a) The process of packing block i to n. (b) The corresponding packing with sequence-pair (431625, 635412)

Y[j]. And the length array $L[1...n]$ is used to record the length of candidates of the longest common subsequence (LCS). The algorithm is presented in [3]. An example is shown in Figure 3.1.

Now, with applying symmetry blocks that have been constructed by linear program with the algorithm in [3], if a block with symmetry constraints is packed, all blocks of the symmetric group that this block belongs to will be packed sequentially so that all blocks with symmetry constraints are packed together. In the same way, we must update the length array L[1...n]. However, for each symmetric group, we must pick the blocks abuts to source and destination.

3.2 Linear Programming with Symmetry Constraints

3.2.1 Linear Constraint Expressions from Symmetry Constraints and Given Sequence Pair

We construct the linear constraints from H/V constraints of a given sequence pair. For example, the expression of "a is in the left of b" in the horizontal constraints can be derived [12].

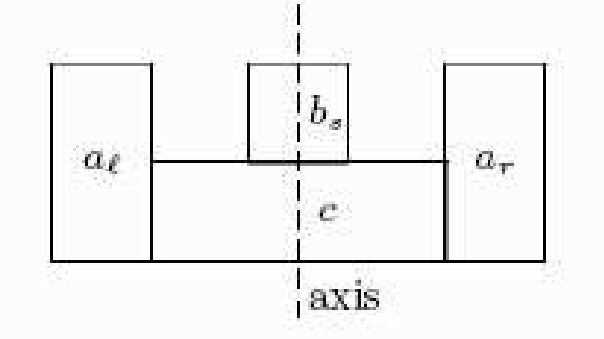

Figure 3.2: blocks placement for seq-pair *(al bs c ar; al c bs ar)*

x(a) + w(a) x(b) is set x(a) - x(b) -w(a)

where x(a) and x(b) are x coordinates of the left edge of block a and b, and w(a) is the width of a.

For symmetry constraints, a block pair *al* and *ar* is symmetry to a vertical symmetry axis. It can be derived

Axis- (x(*al*) + w(*al*)) = x(*ar*) - *Axis* is set

2**Axis* - x(*al*) - x(*ar*) = w(*al*)

where Axis is x coordinate of a vertical symmetric axis for a symmetry group.

For y direction of linear constraints can be derived

y(*al*) = y(*ar*) is set y(*al*) - y(*ar*) = 0

For self symmetric block of as, it can be derived

Axis - (x(*as*) + w(*as*)) = x(*as*) - *Axis* is set

Axis - x(*as*) = w(*as*)/2

3.2.2 Removal of Transitive Edges

It is mentioned in [4], consider a sequence pair(*al bs c ar*; *al c bs ar*) and *w(c)* > *w*(*bs*), the transitive edge (*al, ar*) is required to avoid overlap of *c* and *ar*. The packing is shown in Figure 3.2.

To solve this problem, we construct a simple constraint graph by utilizing the method of obtaining the left corner packing in $O(n \log \log n)$ time from a sequence pair[3]. In our sequence pair, all blocks with symmetry constraints are limited to be arranged in order. That means each block with symmetry constraints is weighted by its width or height, so we don't consider addition constraints because a non-symmetric block isn't added when symmetric blocks are packed.

Take the example mentioned above, the sequence pair will become (*al bs ar; al bs ar*), so we don't consider the overlap of *c* and *ar*.

3.2.3 Set of Parameters with Linear Constraint Expressions

All blocks with symmetry constraints in the sequence-pair are scanned first, and some equations are constructed with placement constraints and symmetry constraints. In placement constraints, block *ni* is set to *xi* in sequence-pair (X, Y) = ((*n1,n2..nk*),(*any order*)) ,the coefficient of *xi* is either 1 or -1 depends on our sequence, and the values of the equation are set to the width of the block where is placed left in horizontal placement. Later, in symmetry constraints, the *xi+1* to *xi+k* are set for symmetry constraints where k is the numbers of symmetry constraints and *xi+k+1* to *xi+k+2* are set for the symmetry axis and the objective function, the coefficient of x is also either 1 or -1 depends on symmetric equations, and the values of the equations is set to 0 for pairs blocks, half of width for self-symmetry blocks. An example of symmetric groups in D70 is shown as Figure 3.3.

3.3 Annealing Process in Placement

We allow several conditions in our annealing process in following. Beside, we propose an efficient way by executing two annealing in a way of correlation. By this way, we can also get a reasonable result in shorter time.

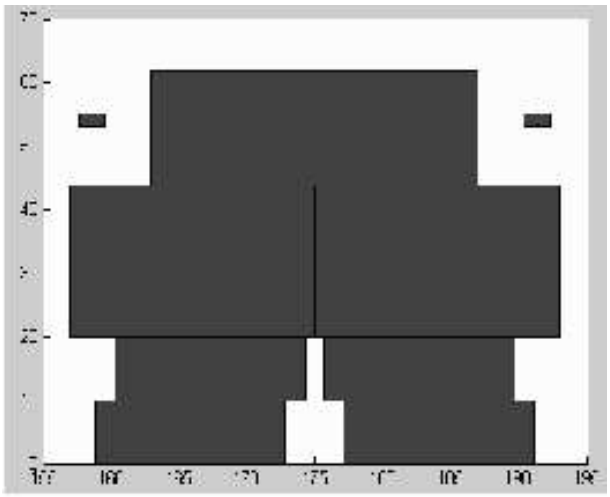

Figure 3.3: The result of a symmetric group in D70 (the symmetry axis is at 175)

3.3.1 Set of Moves

1. Swapping two blocks of the same symmetry group:

Two blocks A and B are picked randomly from a symmetry group, and they do not belong to a symmetry pair. Then we swap A and B in s1 and swap sym(A) and sym(B) in s2. Block A and B can be either belong to a symmetry pair individually or self symmetry.

2. Swapping two cells

A cell may be either an asymmetric block or a symmetry group. Each cell are picked randomly and swap them. For example, if two cells picked randomly are two symmetry groups. After swapping, the sequence pair in s1 is $(a1,a2..A1..A2..ai..Aj)$ originally, it will become $(a1,a2KA2..A1..ai..Aj)$ if we swap $A1$ and $A2$. Where $(a1...ai)$ belong to non-symmetry blocks, $(A1...Aj)$ belong to symmetry groups and Aj would include several symmetry blocks. Notice that the relative ordering between blocks of the same group are not changed. We can also do the similar way in s2.

3. Exchange its width and its height of a block

A block is picked randomly and its width and its weight are changed. If a block which belongs to a symmetry pair is picked, we must also make the corresponding

change to its pair.

4. Rotating a symmetry group

A symmetry group can be symmetry in a horizontal or a vertical axis. But in considerations of application, we allow only all groups that are symmetry in a horizontal or a vertical axis.

3.3.2 Our Approach to Dealing with the Set of Moves

We divide annealing process in two parts. One is to minimize the cost function of every symmetry group and then another is to minimize the cost function of total area including symmetric blocks and non-symmetric blocks. The cost function will be mentioned in following. The two parts are introduced as follows:

Annealing process for symmetric groups: In the process, we apply the set of moves in 3.1.1-1. Each symmetry group has its goal is that minimizing a cost function individually.

Annealing process for all blocks including symmetric blocks and non-symmetric blocks: In this process, we apply the set of moves in 3.1.1-2 4. Here we don't execute the move of 3.1.1-1.

We will execute the part 1 and part 2 in a way of correlation. For example, we may execute a ratio of iterations in part 1 to part 2 to be 1:10, and even more. The cost in part 1 is vast, although the numbers of blocks are smaller than all in our floorplan. In traditional approach, we must consider the move of 3.1.1-1 for all iterations. By the approach we propose, we will evidence to get an efficient solution in a short time and it is more obvious when total blocks in all floorplan are more.

3.3.3 Cost Function

We construct a cost function $cost(F)$ = area(F) + a*wire(F), where F is the foorplan, area(F) is the total area of executed blocks,a is a user given weight, and

wire(F) is the total wire length of F measured by the half-perimeter estimation.

Chapter 4

Experimental Results

We proposed an more effective method of obtaining large-scale block placement satisfying the given symmetry constraints , and compare our result in our approach with [6] and [4]. Here we implemented our algorithm in the C++ programming language in Intel Pentium4 2.8GHz CPU with 2.0GB memory. (1) In comparing to paper [6] with symmetry constraints, the experimental results are implemented on an Intel(R) Xeon(TM) CPU 3.00GHz work station with 2GB memory. (2) In comparing to paper [4] with symmetry constraints, we implement their method by changing our set of moves in traditional way. For a reasonable compare, we valuate between Intel Pentium and Intel(R) Xeon(TM) by an easy program. The ratio of speed of Intel Pentium to Intel(R) Xeon(TM) about 1:1.06. Table 4.1 shows the results of this set of experiment. We compare the dead space with approximate simulation time. The third shows the number of symmetry groups and 5 symmetry groups with 8, 7, 7, 4 and 6 blocks respectively. Notice that the data is assign randomly.

In our approach, we can find that we have a more effective result when the total blocks are increased. When the block numbers increase up to 220, we have a smaller dead space in our approach than that in [6].For the data set of D220,considering the speed of machines, our simulation time will be divided 1.06 time,and the data of time and the time is 21 m. 1.6 s. and the space is 12.08 percent in [4] and the time

Table 4.1: Experimental comparisons between the results of our approach(Pentium4 2.8GHz) and [4](Xeon(TM) 3.00GHz)

Data Set	Block No.	Symmetry Constraints	Our Approach		[4]	
			Time	Dead Space(%)	Time	Dead Space(%)
D50	50	8,7,7,4,6	66s	19.41	74.5s	4.28
D70	70	9,4,9,5,9	2m 10.1s	17.08	2m 12.9s	5.07
D100	100	4,12,4,11,12	4m 38s	16.85	4m 58.2s	8.48
D120	120	5,4,4,7,8	6m 34s	13.03	6m 15.2s	8.49
D170	170	8,7,7,4,6	13m 35.3s	12.69	13m 5.9s	10.14
D220	220	8,7,7,4,6	20m 48.3s	11.84	21m 1.6s	12.08

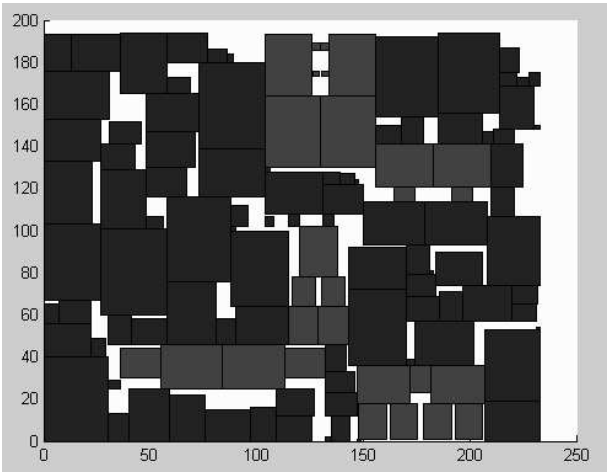

Figure 4.1: D120 with 120 blocks and 5 symmetry groups at Table 4.1 with our approach

is 19m. 57.9s. and the space is 11.84 percent in our approach. However, we can not compile the program of [6] when block numbers is over 250. Figure 4.1 shows the placement of D120 in Table 4.1.

Beside, we add our approach by changing our set of moves in traditional way as shown in Table 4.2. We also run it in the similar simulation time to compare its dead space. We can find it is not effective due to a too short simulation time for this approach. We take about 1 hour to simulate our programming and Figure 4.2 shows the results.

Table 4.2: Experimental comparisons between the results of our approach(Pentium4 2.8GHz) and similar program in [4](Pentium4 2.8GHz)

Data Set	Block No.	Symmetry Constraints	Our Approach		[6]	
			Time	Dead Space(%)	Time	Dead Space(%)
D50	50	8,7,7,4,6	66s	19.41	65s	35.23
D70	70	9,4,9,5,9	2m 10.1s	17.08	2m 8.4s	34.25
D100	100	4,12,4,11,12	4m 38s	16.85	4m 48s	35.61
D120	120	5,4,4,7,8	6m 34s	13.03	6m 39s	32.01
D170	170	8,7,7,4,6	13m 35.3s	12.69	13m 10.3s	25.61
D220	220	8,7,7,4,6	20m 48.3s	11.84	20m 30.3s	24.32

Indeed, we do not run the linear program for symmetry constraints every iteration. And when the size of floorplan becomes larger, the effect of the symmetry groups becomes smaller. We offer an effective way to deal with the large floorplan applied in VLSI designs now.

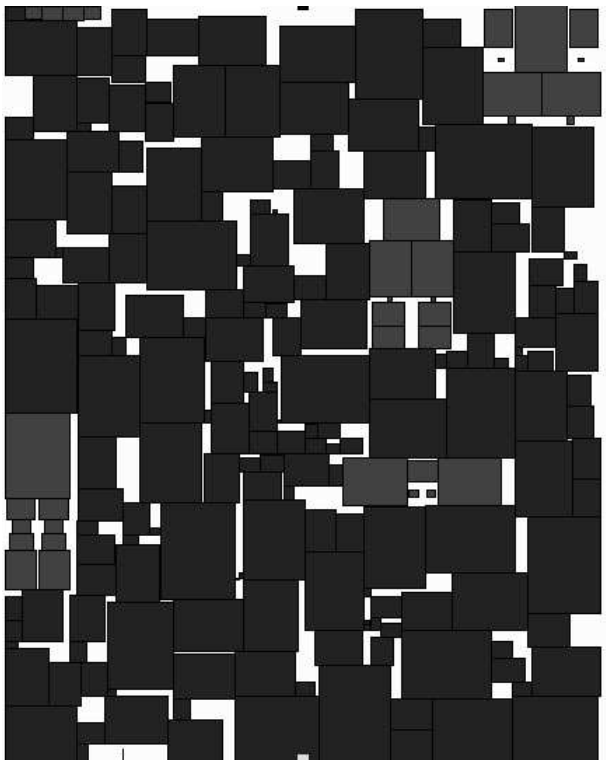

Figure 4.2: D220 with 220 blocks and 5 symmetry groups about 1 hour with dead space of 9.93 percent with our approach

Chapter 5

Conclusion and Future Work

In this thesis, we proposed an more effective method of obtaining large-scale block placement satisfying all the given symmetry constraints. This algorithm has an efficient trend to get a nearly optimum solution and when the sizes of the floorplan become larger, the effect of the symmetry groups becomes smaller. We offer an effective way to deal with the large floorplan applied in VLSI designs now.

Our future problem is experiments on industrial applications of analog circuits. Even it is useful and helps to speed up the automation of analog layout designs with symmetry constraints.

Bibliography

[1] J. Cohn, D. Garrod, R. Rutenbar and L. Carly , "Analog Device-Level Layout Automation," Kluwer Academic Pub., 1994.

[2] F. Balasa and K. Lampaert , "Symmetry within the Sequence-Pair representation in the Context of Placement for Analog Design," IEEE Trans. CAD, vol.19, no.7, pp.721-731, 2000.

[3] K. Krishnamoorthy, S.C. Maruvada and F. Balasa , "Fast Evaluation of Symmetric-Feasible Sequence-Pairs for Analog Topological Placement," 5th IEEE Int. Conf. on ASIC(ASICON), pp.71-74, 2003.

[4] S. Kouda, C. Kodama, and K. Fujiyoshi , "Improved Method of cell Placement with Symmetry Constraints for Analog IC Layout Design," ISPD'06, April 9-12, 2006.

[5] H.Murata, K. Fujiyoshi, and S.Nakatake , "VLSI Module Placement Based in Rectangle-Packing by the Sequence-Pair," IEEE Tran. CAD, vol. 15, no. 12,1996.

[6] Y.C. Tam, Evangeline F.Y. Young, and C. Chu , "Analog Placement with Symmetry and Other Placement Constraints," International Conference on Computer-Aided Design, pp. 5-9, 2006.

[7] H. Murata, K. Fujitoshi, S. Nakatake, and Y. Kajitani , "Rectangle-Packing-Based Module Placement," IEEE International Conference on Computer-Aided. pp. 472-479, 1995.

[8] F. Balasa, S. C. Maruvada, and K. Krishnamoorthy , "On the Exploration of the Solution Space in Analog Placement with Symmetry Constraints," IEEE Tran. CAD, vol. 23, no. 2, 2004.

[9] S. Bhattacharya, N. Janglrajarng, R. Hartono, and C-J. Richard Shi,"Hierarchical Extraction and Verification of Symmetry Constraints for Analog Layout Automation," IEEE Design Automation Conference, pp. 400-405, 2004.

[10] S. Bhattacharya, N. Janglrajarng, and C-J. Richard Shi , "Multi-Level Symmetry Constraint Generation for Retargeting Large Analog Layouts," IEEE Tran. CAD, vol. 25, pp. 945-960, 2004.

[11] A. Hastings , "The Art of Analog Layout," Prentice Hall, 2001.

[12] J.-G Kim and Y.-D Kim , "A linear programming-based algorithm for floorplanning in VLSI design," IEEE Trans. CAD, vol. 22, no. 5, pp. 584-592, 2003.

[13] K. Lampaert, G. Gielen, and W. Sansen , "A Performance-driven Placement Tool for Analog Integrated Circuits," IEEE J. Solid-State Circuits, vol. 30. no. 7, pp. 773-780, 1995.

[14] H. Murata, K. Fujiyoushi, S. Nakatake, and Y. Kajitani , "Rectangle-Packing-Based Module Placement," Proceedings IEEE International Conference on Computer-Aided Design, pages 472-479, 1995.

[15] Evangeline F. Y. Young, Chris C. N. Chu, and M. L. Ho , "Placement Constraints in Floorplan Design," IEEE Transactions on Very Large Scale Integration Systems, vol. 12, no. 7, pp. 735-745, 2004.

[16] R. Okuda, T. Sato, H. Onodera and K. Tamaru , "An algorithm for layout compaction problem with symmetry constraint," IEICE Trans. Fundamentals, vol.J70-A, no. 3, pp. 536-543, 1990 (in Japanese).

[17] Yingxin Pang; Balasa, F.; Lampaert, K.; Chung-Kuan Cheng , "Block placement with symmetry constraints based on the O-tree non-silicing representation," DAC, pages 464 - 467, June 5-9, 2000

[18] Jai-Ming Lin; Guang-Ming Wu; Yao-Wen Chang; Jen-Hui Chuang, "Placement with symmetry constraints for analog layout design using TCG-S," ASPDAC, vol 2, pages 1135 - 1138, 18-21 Jan. 2005