元　智　大　學

資訊工程研究所

碩士論文

Post-Layout Multiple Vias Insertion

在佈局後多倍鑽孔的植入

研 究 生：王建富

指導教授：林榮彬 博士

中華民國九十三年七月

在佈局後多倍鑽孔的植入

Post-Layout Multiple Vias Insertion

研　究　生：王建富　　Student: Chien-Fu Wang

指導教授：林榮彬　　Advisor: Rung-Bin Lin

元 智 大 學

資 訊 工 程 研 究 所

碩　士　論　文

A Thesis

Submitted to Institute of Computer Engineering and Science

Yuan Ze University

in Partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Computer Engineering and Science

July 2004

Chungli, Taiwan, Republic of China.

中華民國 九十三 年 七月

Post-Layout Multiple Vias Insertion

Student: Chien-Fu Wang   Advisor: Dr. Rung-Bin Lin

Institute of Computer Engineering and Science
Yuan Ze University

Abstract

As deep submicron technology moves to nanometer. More and more logic gates can be put in the same chip. More wires and hence more vias will be on the same chip. Vias play an important role in making connections between different levels of metal wiring. Once any one of these vias has a defect in the chip during manufacturing, it will increase the probability of malfunction and product cost. In order to improve the reliability for the wiring connections between different layers, multiple vias need to be inserted. The objective of via duplication problem is to duplicate as many vias on as possible. In this thesis, we develop a post-routing MVIT(Multiple Vias Insertion Tool) for solving via duplication problem. MVIT is based on a bipartite graph model for modeling the via duplication problem and those surrounding via candidates. The Ford-Fulkerson algorithm is used for achieving maximizing matching for vias and those corresponding via candidates. Otherwise, MVIT will keep away from obstacles and other nets that will cause short-circuit when it inserts the via candidate. It also

provides the flexible option to set-up the priority of on-pin, on-track and on-critical path. In our experiments, ISCAS benchmark circuits are used to evaluate our MVIT and Cadence Silicon Ensemble's double-cut via function. Although, our vias insertion rate of experimental results are lower than Silicon Ensemble's. But for an existing layout is still a good choice to insert double or multiple vias with no overhead.

Contents

書名頁……………………………………………………………...…………………ii

摘要………………………………………………………………………..……...iii

Abstract…………………………………………………………………………iv

Contents…………………….……………………….………………..…….…..…...v

List of Figures……………………….……………..………………………….…..vii

List of Tables……………………………………………..……………………..…viii

Symbol Definitions……………………………………………………………….ix

Chapter 1 Introduction………………………...…………………………....1

1.1 Background and Motivation………..…………………..……………………1

1.2 Scope of the Work……………………...……………………..………………3

1.3 Thesis Organization……………………..…………………………..………..3

Chapter 2 Related Work……………………………..……………...….4

2.1 During Layout Double Vias Insertion…..……………………………………4

2.2 Post-Layout Multiple Vias Insertion ………………………….…………..6

Chapter 3 Post-Routing Via Duplication Approaches ..……….…….8

3.1 Heuristic…………………………………………..... …..…………….8

3.1.1 Bipartite Graph Modeling…………………………………………...9

3.1.2 Ford-Fulkerson Algorithm……………………………………………12

3.1.3 Breadth First Search for Partitioning…………………………………13

3.1.4 Prioritizing the Via Candidates………………………………………..14

3.2 Maximization of Multiple Vias Insertion..…………………………………17

Chapter 4. Experimental Results……………………………………..19

4.1 Design Environment and Flow …….…………………………….……………...19

4.2 Experiments of Heuristic……….…..……………………………..………..…21

4.2.1 Weighted Double Vias Insertion…...……………….…………………….22

4.2.2 Weighted Multiple Vias Insertion ….……..……………………...….26

Chapter 5 Conclusions………………………………………………...27

References……………………..……………………………….…28

List of Figures

Figure 1.1 Basic via duplication ………….…………………………….....…...2

Figure 2.1.1 Double-cut vias definition in LEF……………………………………….5

Figure 2.1.2 Different types of double-cut vias……………………………………….5

Figure 2.2 Four “Via Candidates” of on track and off track for single via……………7

Figure 3.1 Bipartite matching graph ……….…………………………………..…..9

Figure 3.2(a) A part of layout……………………………………………………..10

Figure 3.2(b) Via candidates (Simplification)..……….……………………………..10

Figure 3.2(c) Corresponding bipartite graph.………………………………………..11

Figure 3.3 The corresponding flow network.………………………………………..13

Figure 3.4 Sharing via candidate…………………………………………………..15

Figure 3.5(a) An example of critical path..…………………………………………..16

Figure 3.5(b) A part of critical layout…....…………………………………………..16

Figure 3.5(c) Insert the double via in critical path..…………………...…………..16

Figure 3.6(a) An example of on-pin via candidate…………………………………..17

Figure 3.6(b) Insert the double via in on-pin via candidate..………………………..17

Figure 4.1 Design Environment and Flow of MVIT…………………………………20

List of Tables

Table 4.2.1 Double Vias Insertion for Row Utilization 80% (No M1_M2 Via).……..23

Table 4.2.2 SE Double Vias Insertion for Row Utilization 80% (No M1_M2 Via)…23

Table 4.2.3 Double Vias Insertion for Row Utilization 95% (No M1_M2 Via)……..24

Table 4.2.4 SE Double Vias Insertion for Row Utilization 95% (No M1_M2 Via…..24

Table 4.2.5 Double Vias Insertion for Row Utilization 80%…………………..25

Table 4.2.6 Double Vias Insertion for Row Utilization 95%…………………..25

Table 4.2.7 Multiple Vias Insertion for Row Utilization 80%…………...……26

Table 4.2.8 Multiple Vias Insertion for Symbol Definitions

V_i : the *i*th via in the layout

VC_i^W : a western via candidate of the via *i*

VC_i^E : a eastern via candidate of the via *i*

VC_i^S : a southern via candidate of the via *i*

VC_i^N : a northern via candidate of the via *i*

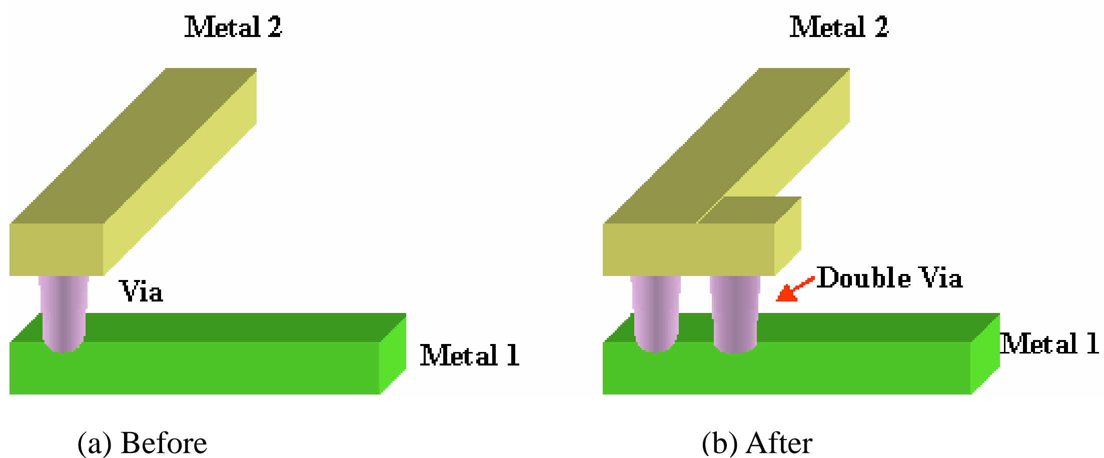
Chapter 1. Introduction

1.1 Background and Motivation

As deep submicron technology moves from 0.25, 0.18 micron… even to nanometer, more and more logic gates can be put in the same chip. More wires and hence more vias will be on the same chip. Vias play an important role in making connections between different levels of metal wiring. Every via in an integrated circuit layout has probable manufacturing defects affecting its electrical characteristic and function. Once any one of these vias has a defect in the chip during manufacturing, it will increase the probability of malfunction and product cost potentially. In order to strengthen the wiring connections between different layers, multiple vias can be inserted. The objective of via duplication problem is to duplicate vias on as many wiring connections as possible. Figure 1.1 shows a picture of via duplication.

There are two approaches to via duplication. This first one such as Cadence Silicon Ensemble supports double-cut vias duplication during placement. Users must specify four-way vias per each via defined in LEF (Library Exchange Format) file in advance. 100% double-cut vias duplication can be guaranteed, but floorplan and utilization adjustment are usually performed to complete the routing.

The second one duplicates vias on existing layouts post routing where permitted by the design rules. In [Har01], [All02], [Leu03] they simply inserted double or multiple vias for each via and set a simple priority (i.e. on-track double via will be preferred than off-track double via). They didn't run standard benchmarks for experiments and show vias insertion relationship between row utilization and circuit size. They also didn't guarantee to get an optimal solution for the basic via generation technique. To improve this, we developed a heuristic to solve this problem. The heuristic built a bipartite graph for the underlying problem and solved vias maximum matching using Ford-Fulkerson algorithm. In view of the above problem, it is worth revisiting the via duplication problem. Therefore, in this thesis we propose to design a Multiple Vias Insertion Tool (MVIT).



(a) Before (b) After

Figure 1.1 Basic via duplication

1.2 Scope of the Work

We developed a post-layout MVIT for maximizing multiple via insertion. We first formulated this problem as a bipartite graph model. The graph representation be brought up for speedily processing a large number of via candidates. Any form of pre-existing obstacles such as power bus, cell obstacle, cell pin, etc. will be taken into account. The Verify Connectivity of Cadence Silicon Ensemble and the DRC of Cadence ICFB can be used to verify the experimental results obtained respectively from our heuristic.

1.3 Thesis Organization

In this chapter, we have elaborated on our motivation for developing the MVIT. Chapter 2 will carry out the survey of some related work. Chapter 3 will detail our algorithm for adding extra vias on a post-routing design. The experimental results are presented in Chapter 4. The last chapter draws some conclusions and outlines the future work.

Chapter 2. Related Work

Via duplications can be made or planned at the IC design stages, floorplan, placement and routing. The related work can be classified as during-layout double vias insertion and post-layout multiple vias insertion.

2.1 During-Layout Double Vias Insertion

Cadence Silicon Ensemble (SE) version 5.1 or later support double via insertion [And98]. Before running this feature, a user must define four-way (north, south, east and west) via candidates for each default via in LEF files as shown in Figure 2.1.1.

After defining the types of double vias, SE router can flexibly choose the most suitable double vias defined in advance for routing. For each default via, SE router will generate a proper double via. If SE router can't complete the routing because of double vias setting in LEF that causes more congested routing area and more violation, it is necessary to adjust the related setting during floorplan to enlarge the routing area. There are several types of double vias showed in Figure 2.1.2 and possibly be implemented by SE router.

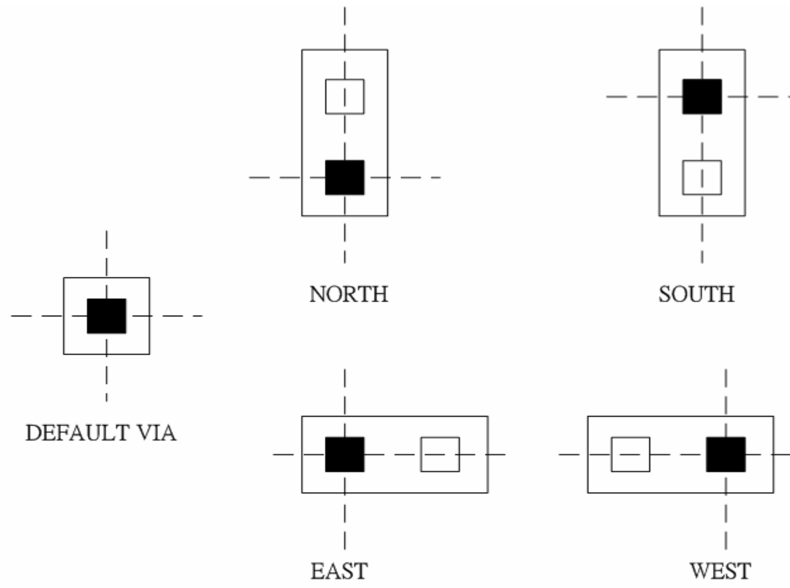


Figure 2.1.1 Double-cut vias definition in LEF

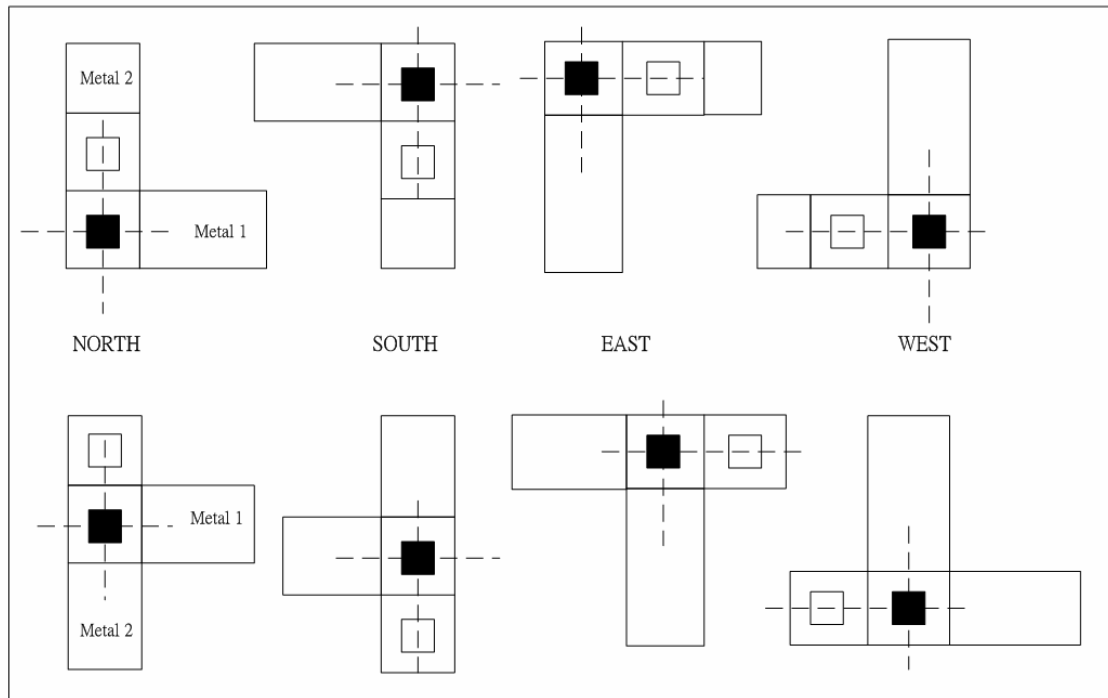


Figure 2.1.2 Different types of double-cut vias

2.2 Post-Layout Multiple Vias Insertion

This approach inserts multiple vias into an existing layout after routing. In order to add a second or more via for each default via, it is necessary to choose the best position of via candidates which didn't violate design rules.

In [Har01], the author discussed the relationship of via critical area and diameter of defects. The margin for tolerating via failure is also calculated. In [Leu03], the author not only focuses on single via failure, but also performs the following layout considerations to increase multiple vias insertion rate:

(a) Proper planning during global and detailed routing can significantly improve the success rate by avoiding local congestion or at-capacity routing on adjacent layers.

(b) Detailed router can detour wires to allow the addition of redundant vias in otherwise infeasible regions.

All related papers, [Har01], [All02], [Leu03] use the via generation technique shown in Figure 2.2. In Figure 2.2, the authors assumed that there are only two layers for this case (metal 1 and metal 2). One default via connects these two layers. Normally, this default via has four via candidates (VCs) (south, north, east and west). We use four symbols, VC^S, VC^N, VC^E, VC^W, to represent the four VCs, respectively. VC^N, VC^E are "on track" via candidates. VC^S, VC^W are "off track"

via candidates. The three papers mentioned above also have the same setting for VC (i.e. on track via candidate will be preferred than off track via candidate) and the reasons as follows:

1. On track via candidates use less resource: for example, double via, VC^N only consists of VC itself and one more metal 1 segment which connects between VC^N and default via. It will not impact so much on existing layout.
2. On track via candidates save more space: no extra overhang, so the users can put much more via candidates in congested layout.

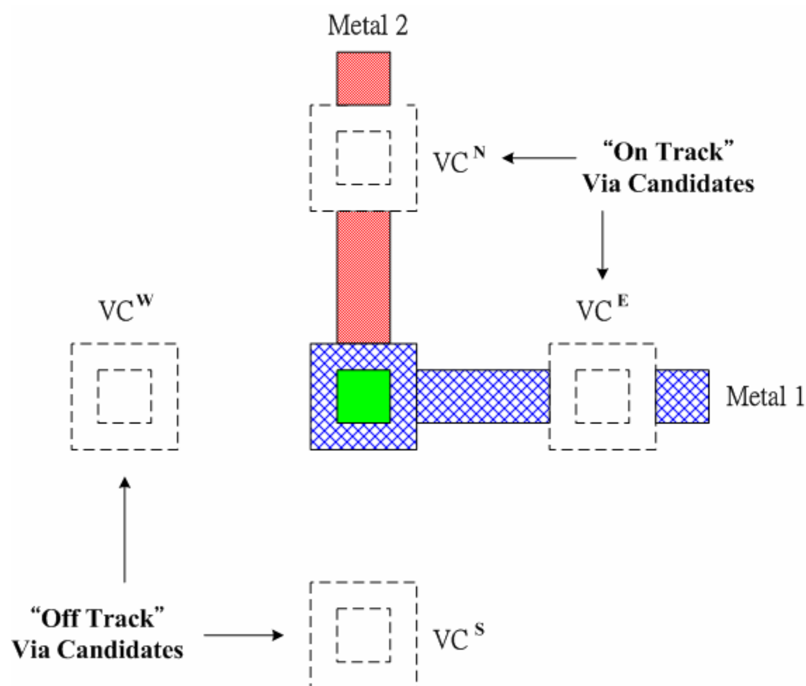


Figure 2.2 Four "Via Candidates" of on track and off track for single via

Chapter 3. Post-Routing Via Duplication Approach

This chapter describes the proposed via duplication approach- Multiple Vias Insertion Tool (MVIT)- in great detail. First, we will construct a graph model to represent via duplication problem. Next, we will propose a heuristic to solve this problem. The heuristic needs to take into account obstacles, stacked via, double-via insertion priority and layout partitioning for speeding up the running time.

3.1 Heuristic

It is very important to choose the best via candidate for each default via. Because an eligible position may be shared by many via candidates, the problem of selecting a best position is not trivia. In this thesis, we proposed a heuristic to solve this problem systematically.

The pseudo code is showed as follows for describing our heuristic:

```
1   MVIT() {
2           Construct the bipartite graph G.,
3           Finding the via clusters by BFS
4                   mark the cluster number for via cluster
5           Prioritizing via candidates
6           Do (Convert each via cluster to corresponding flow network f) {
7                 Assign highest 2 via candidates for via of each via cluster into flow
8                 add vertex, source s and sink t to V
9                 assign unit capacity to each edge
10                Ford-Fulkerson Algorithm {
11                      initialize flow f to 0
12                      while there exists an augmenting path p{
13                            do augment flow f along p
14                      }
15                }
16          } until (no vertex in graph G)
17  }
```

3.1.1 Bipartite Graph Modeling

In the proposed graph model $G = (V, E)$, we first denote the vias as V_1 to V_m and the distinct double-via candidates as d_1 to d_n. Let V_i be a via vertex, d_i be a via candidate vertex. Therefore, for $V_i \in V$ and $d_j \in V$ if a double-via candidate d_j can be assigned to a via, V_i, then we create an edge $(V_i, d_j) \in E$ as such a bipartite graph can be established. In this group, a via can have more than one candidate vias. A via candidate can be shared by more than one via. The double-via optimization problem can be formulated as a bipartite graph matching problem which can be solved by a polynomial algorithm with time complexity:

$$O\left(\min\left(|V-D|, |D|\right) |E|\right) \qquad (3.1)$$

where

D is the set of via candidates and

$|D|$ is the number of double-via candidates

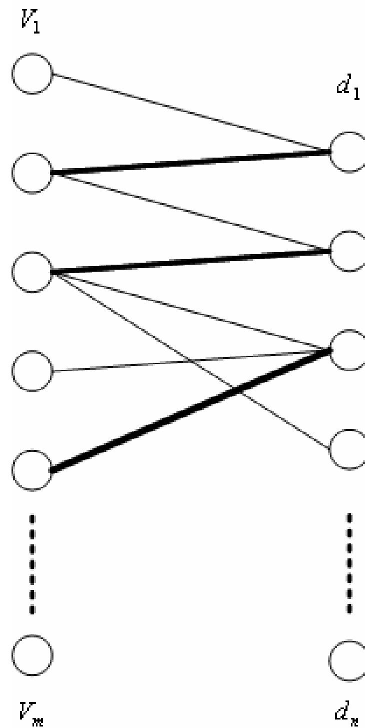


Figure 3.1 Bipartite matching graph

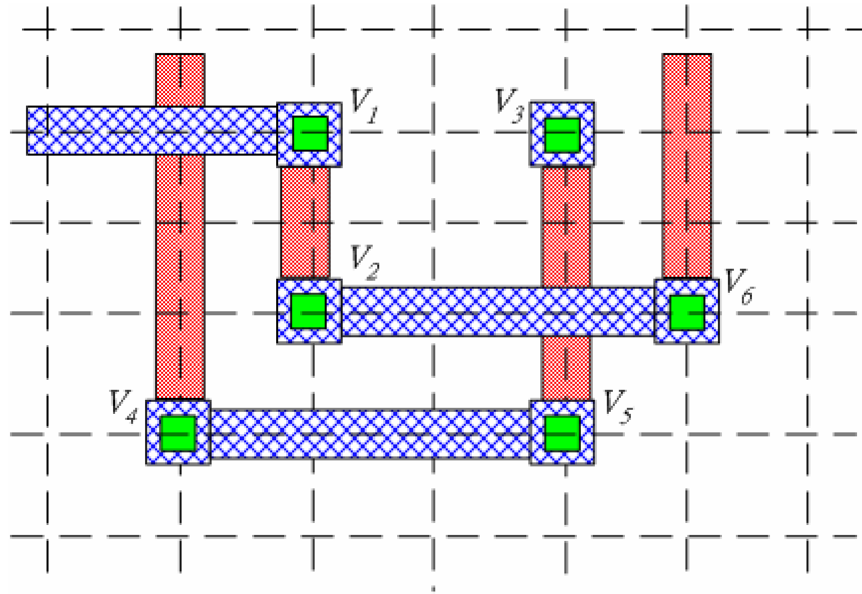


Figure 3.2(a) A part of layout

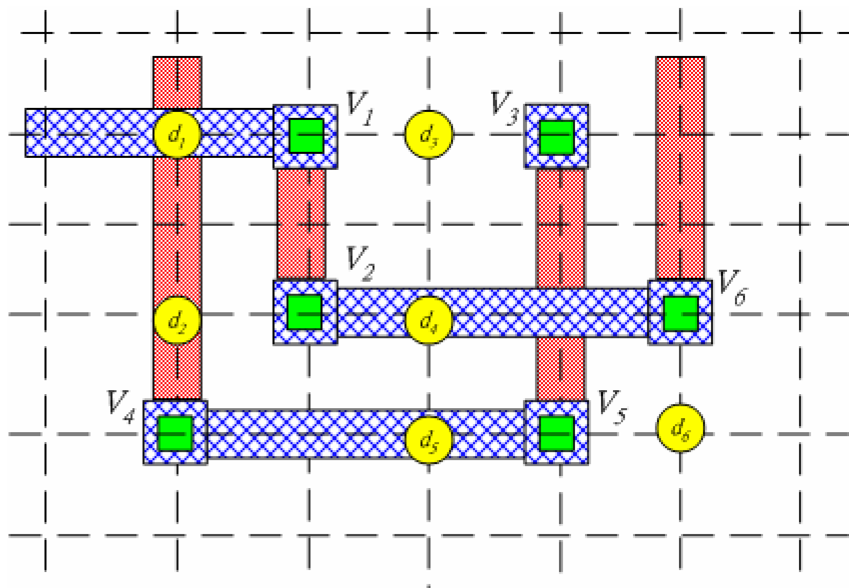


Figure 3.2(b) Via candidates(simplification)

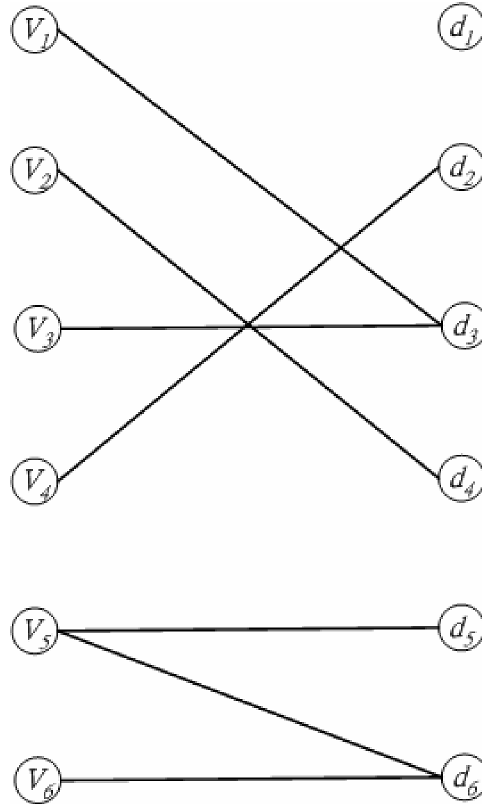


Figure 3.2(c) Corresponding bipartite graph

There is a part of layout in Figure 3.2(a) for describing the translation from physical layout to Bipartite graph. There are only two nets and six vias(V_1~V_6) in the layout. All vias are assumed to be M1_M2 vias. Vertical wire segments are Metal 2 and Horizontal wire segments are Metal1.

In order to show the example concisely, we simplify the four-way(multiple) via candidates and leave some via candidate(d_1~d_6) for description in Figure 3.2(b). Besides, all double via candidates are just chosen arbitrarily with no preference for a via candidate.

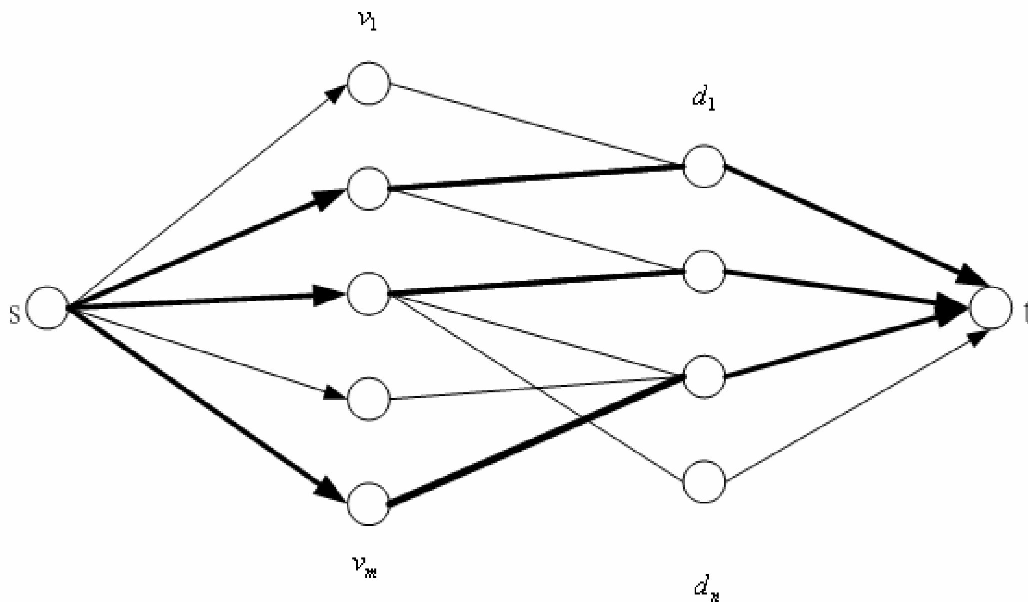
For V_1 , it has d_1 and d_2 via candidates. We can't choose d_1 due to short to another wire. We can choose off-track VC, d_3 and create an edge (V_1, d_3). For V_2, it has d_2 and d_4 via candidates. We can't choose d_2 due to short to another wire(conflict to other wires). We can choose on-track VC, d_4 and create an edge (V_2, d_4). For V_3, it only has d_3 off-track VC. It is shared (conflicting) by V_1. For this situation, we still create an edge (V_3, d_3) and we can find a maximum matching using Ford-Fulkerson algorithm describes in next section.

For V_4, it only has *d2* on-track VC. We choose it and create an edge(V_4, d_2). For V_5, it has d_5 on-track VC and d_6 off-track VC. We create two edges (V_5, d_5) and (V_5, d_6). For V_6, it only has d_6 off-track VC. We create an edge (V_6, d_6). We show the completed Bipartite Graph in Figure 3.2(c)

3.1.2 Ford-Fulkerson Algorithm

We use the Ford-Fulkerson algorithm to find a maximum matching in an undirected bipartite graph in polynomial time. First, we need to construct a flow network in which flows correspond to matching as shown in Figure 3.2. We add source *s* and sink *t* to *V*, and we let $V' = V \cup \{s, t\}$. Second, we assign unit capacity to each edge. Finally, we run the Ford-Fulkerson algorithm.

The Ford-Fulkerson algorithm is an iterative approach. We start with flow $f(v, d) = 0$ for all $v, d \in V$, giving an initial flow of value 0. At each iteration, we increase the flow value by finding an "augmenting path". We repeat this process until no augmenting path can be found. This process yields a maximum flow.



3.1.3 BFS(Breadth First Search) for Partitioning

In order to solve the huge bipartite graph matching problem efficiently, we can partition all vias in graph into several via clusters according to the relationship of

sharing position of via candidates.

We use BFS(Breadth-First Search) on the original bipartite graph to find the clusters. Given a graph $G = (V, E)$ and a source vertex, breadth-first search systematically explores the edges of G to discover every vertex that is reachable from the source vertex. The time complexity of BFS is $O(V + E)$.

3.1.4 Prioritizing the Via Candidates

We assign weights to each edge before using Ford-Fulkerson algorithm. The large a weight is, the higher the matching priority is.

We set up the priority from high to low as follows:

1. Shared Via Candidate: A via candidate is shared by two or more vias. An example is showed in Figure 3.4. In this example, eastern via candidate of V_1 and western via candidate of V_2 are on the same position. Therefore, they share the same via candidate.
2. On-Critical Path Via Candidate: A via candidate overlays the critical path. An critical path example is showed in Figure 3.5(a). Then, we focus on the part of the layout for the critical path shown in Figure 3.5(b). We can give the (via, via candidate) pair a larger weight. After inserting VC, the final result is shown in Figure 3.5(c).
3. On-Pin Via Candidate: A via candidate overlays the pin of a targeted. An on-pin via candidate example is as shown in Figure 3.6(a). The on-pin via candidates have the higher priority to be inserted as shown in Figure 3.6(b).
4. On-Track Via Candidate: A via candidate on the wire segment that belongs to the same net with via.

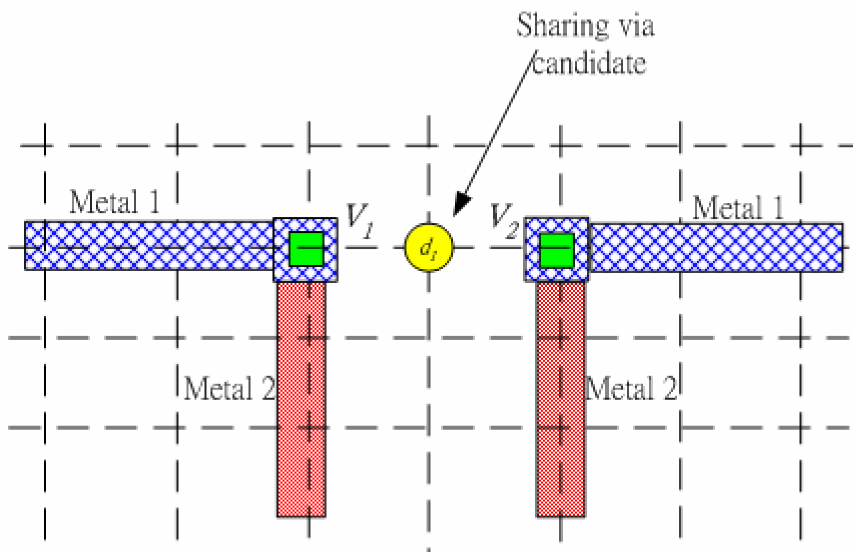


Figure 3.4 Sharing via candidate

xample of critical path

Critical Path

Critical Path

V_1

Metal 2

V_2

Metal 1

Critical Path

V_1

Metal 2

V_2

Metal 1

Figure 3.5(c) Insert the double via in critical path

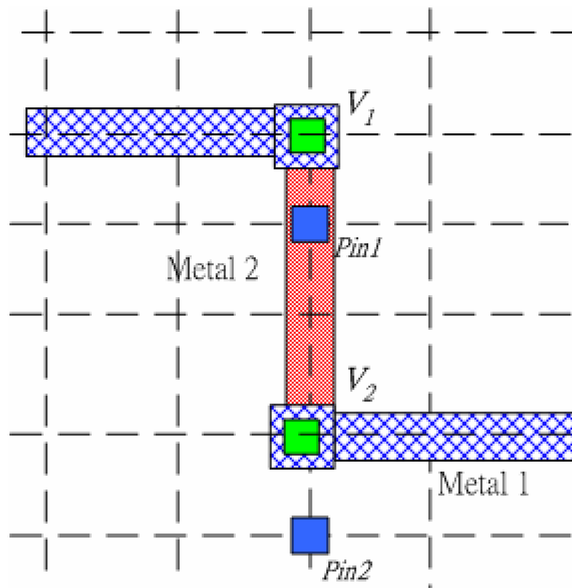


Figure 3.6(a) An example of on-pin via candidate

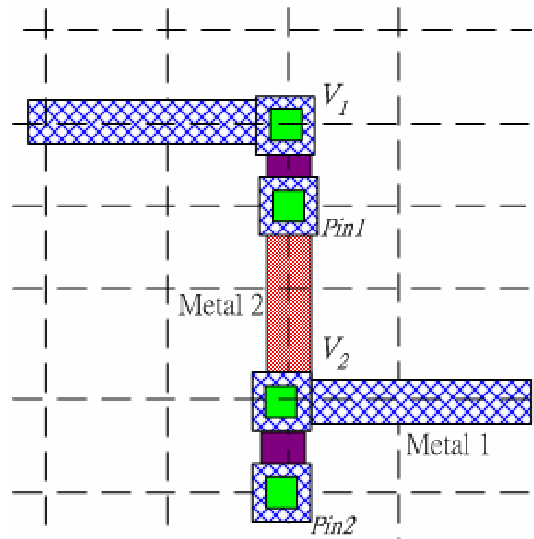


Figure 3.6(b) Insert the double via at an on-pin via candidate

3.2 Maximization of Multiple Vias Insertion

In this section, we would like to maximize as many vias insertion as possible. The bipartite graph model is used several times to maximize via insertion. We detailed the processes of Multiple Vias Insertion Tool as follows:

1. Constructed the bipartite graph model

For each given post-routing DEF file, we record all wires information that includes all vias. Then, we read the LEF file and record all cell macro which includes all obstacles and pin information for identifying via candidates.

2. Prioritizing via candidates

As described in Section 3.1.4, we assign weights to each edge for the matching priority.

3. Finding via clusters by BFS

This step, we partition the huge bipartite graph into several via clusters. A cluster is formed by a set of vias and via candidates that form maximum connected graph. First, we start from a via as source vertex and use breadth-first-search to explore the edges systematically that are reachable from the source vertex. If the BFS complete the search, all clusters are found.

4. Repetitively for multiple via insertion

For each via cluster, it has a corresponding sub-bipartite graph. For double via insertion, we run the Ford-Fulkerson algorithm once for each via c luster. Hence, at most one via candidate can be selected. For multiple via insertion, we run the Ford-Fulkerson algorithm several times for each via cluster.

Chapter 4. Experimental Results

Our approaches for solving “Double and Multiple Vias Insertion” have been implemented in the C language. In order to evaluate our approach, we compare the experimental results obtained by our MVIT to those obtained by the “double-cut via” function of Cadence Silicon Ensemble.

ISCAS benchmark circuits are used in our experiments. The benchmark circuits are placed and routed by Cadence Silicon Ensemble Version 5.4. After this is done, we get the post-routing Cadence DEF (Design Exchange Format) [Cad03a][Cad03c] file for import to our MVIT.

4.1 Design Environment and Flow

In Figure 4.1 shows the design environment and the flow of MVIT. The input to MVIT consists of a post-routing DEF file and LEF file. Note that the DEF file in the flowchart , including circuit netlist with placement and routing information, and the LEF file in the flowchart, containing the descriptions of a process technology and standard cell library, could help us construct a routing grid model.

After MVIT gets the desired input files, DEF and LEF files, it will construct a data structure based on the information given by the DEF and LEF. The most important thing is to perform multiple vias insertion for an existing layout and get a modified DEF file.

First, we construct the bipartite graph model based on a given DEF file. Second, we perform the violation check for each via candidate. Third, we partition the

huge graph model into several sub-graph using BFS. Fourth, we assign weights for each via candidate according the rule defined in Section 3.1.4. Fifth, we implemented the Ford-Fulkerson's algorithm for processing the underlying bipartite graph model. When each time the Ford-Fulkerson is run, it will maximize the matching in each sub-graph. In other words, it will select as many via candidates as possible to be included in the layout. If we prefer multiple vias insertion, we need to run this stages several times, otherwise, only one time can be done for double vias.

Finally, we need to modify the DEF for those newly-added vias. The target DEF file will be generated and verified by enabling SE Connectivity. If no error is found in SE verification report, the design processes is completed

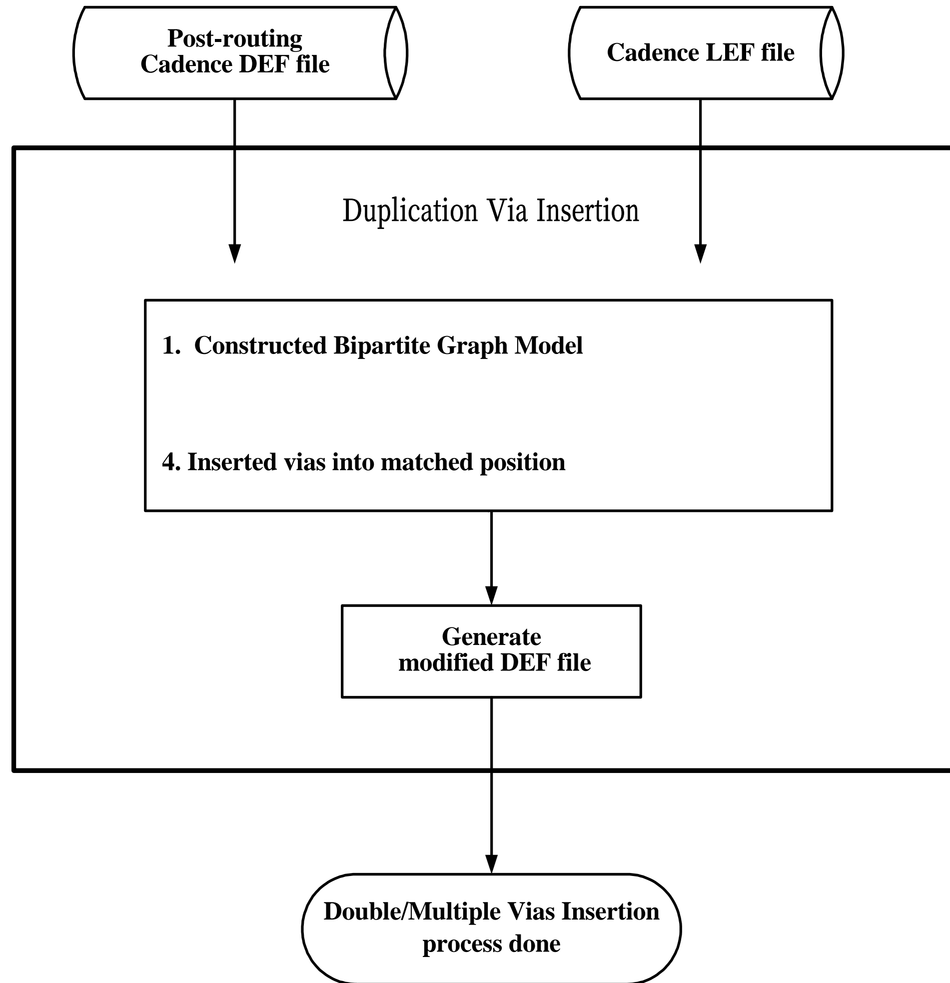


Figure 4.1 Design Environment and Flow of Double/Multiple Via Insertion Tool

4.2 Experiments of Heuristic

We have three sets of experiments obtained by the proposed heuristic. The first set of experiments is shown in Table 4.2.1~ Table 4.2.4. The second set of experiments is shown in Table 4.2.5~ Table 4.2.6. The third set of experiments is shown in Table 4.2.7~ Table 4.2.8. A brief description is given as follows:

1. Table 4.2.1~Table 4.2.2 show the results for the circuits with row utilization 80%. The results are obtained by our Double-Via Insertion Tool and Silicon

Ensemble Double-Cut Via Function. (No M1_M2 Via be inserted)

2. Table 4.2.3~Table 4.2.4 show the results for the circuits with row utilization 95%. The results are obtained by our Double-Via Insertion Tool and Silicon Ensemble Double-Cut Via Function. (No M1_M2 Via be inserted)
3. Table 4.2.5~Table 4.2.6 show the results for the circuits with row utilization 80% and 95%. The results are obtained by our Double-Via Insertion Tool.
4. Table 4.2.7~Table 4.2.8 show the results for the circuits with row utilization 80% and 95%. The results are obtained by our Multiple-Via Insertion Tool.

4.2.1 Weighted Double Vias Insertion

The Weighted Double Vias Insertion Tool has been implemented and tested on the ISCAS benchmark circuits. The experimental results are shown in Table 4.2.1. This experiments didn’t insert M1_M2 via candidates for the purpose of comparing the results with those obtained by Silicon Ensemble. The results obtained by SE are given in Table 4.2.2.

In Table 4.2.1, the second column is the number of vias in post-routing layout, the third column is the number of double vias we inserted, the fourth column is the number of on-track double vias, fifth column is the number of vias which have no via candidate, the sixth column is the total wire length for regular wires, the final column is the percentage of the vias (other than M1_M2) that do not have via candidates.

We observe the third column, the number of vias be inserted, in Table 4.2.1 and Table 4.2.2. Obviously, SE’s duplicated via count is greater than ours. The last column in Table 4.2.1, the value is about 0%~ 13.1%, but in Table 4.2.2, the value is 0.0%. Because SE Ensemble inserting double vias when it routes the nets. SE router can dynamically choose the appropriate double via which is defined in LEF. Therefore, it can achieve 100% double via insertion for M2_M3 and M3_M4. Moreover, we can see the sixth column, regular wire length. Most of large circuit are use longer regular wire length in the layout obtained by SE.

DVIT Program, Double Vias Insertion, No Via1 (Core Row Utilization: 80%)						
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	Regular Wire Length(Microns)	M2_M3 M3_M4 without VC (%)
s27	52	12	10	40	604	0.0%
s298	365	119	93	246	4415	1.4%
s510	773	307	219	466	10364	5.0%
s832	958	341	240	617	12215	3.9%
s1423	1966	968	733	998	21021	0.8%
s5378	4655	2277	1646	2378	63268	2.3%
s13207	10667	4322	3170	6345	159973	13.1%
s15850	11890	5677	4273	6213	169634	4.1%
s35932	33559	14929	6766	18630	447319	
s38417	39360	17169	10951	22191	569481	

Table 4.2.1 Double Vias Insertion for Row Utilization 80% (no Via1)

Cadence Silicon Ensemble V5.4 , Double Vias Insertion (Core Row Utilization: 80%)						
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	Regular Wire Length(Microns)	M2_M3 M3_M4 without VC (%)
s27	53	16	10	37	518	0.0%
s298	362	129	104	233	4412	0.0%
s510	794	349	263	445	10745	0.0%
s832	958	375	289	583	12594	0.0%
s1423	1939	1046	788	893	20707	0.0%
s5378	4986	2917	2203	2069	64769	0.0%

Table 4.2.2 SE Double Vias Insertion for Row Utilization 80% (no Via1)

Table 4.2.3 and Table 4.2.4 are with row utilization 95%. Although we change

the row utilization from 80% to 95%, but the insertion rate is increased.

DVIT Program, Double Vias Insertion, No Via1 (Core Row Utilization: 95%)						
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	Regular Wire Length(Microns)	M2_M3 M3_M4 without VC (%)
s27	54	13	10	41	547	0.0%
s298	395	153	105	242	4598	2.5%
s510	895	348	251	547	11470	12.5%
s832	1058	365	261	693	13546	11.2%
s1423	2041	1124	830	917	22475	1.7%
s5378	5124	2815	1993	2309	65026	4.4%
s13207	11123	4287	2996	6836	151657	18.0%
s15850	12697	6648	4906	6049	166677	3.6%
s35932	35813	22249	12321	13564	477078	
s38417	41853	20242	12975	21611	559437	

Table 4.2.3 Double Vias Insertion for Row Utilization 95% (no Via1)

Cadence Silicon Ensemble V5.4 , Double Vias Insertion (Core Row Utilization: 95%						
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	Regular Wire Length(Microns)	M2_M3 M3_M4 without VC (%)
s27	50	5	5	45	511	0.0%
s298	408	172	136	236	4719	0.0%
s510	849	404	312	445	11517	0.0%
s832	1010	469	344	541	12291	0.0%
s1423	2151	1266	965	885	23590	0.0%
s5378	5457	3590	2738	1876	67714	0.0%

Table 4.2.4 SE Double Vias Insertion for Row Utilization 95% (no Via1)

Table 4.2.4 and Table 4.2.5 are obtained by our DVIT program. And we add the last column, number of on-pin vias, to these 2 tables. It shows how many vias are

on-pin.

DVIT Program, Double Vias Insertion (Core Row Utilization: 80%)					
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	# of On-Pin Vias
s27	52	43	24	9	10
s298	365	287	179	78	66
s510	773	607	390	166	110
s832	958	759	482	199	165
s1423	1966	1631	1065	335	322
s5378	4655	3854	2477	801	808
s13207	10667	7677	4908	2990	1602
s15850	11890	9350	6085	2540	1789
s35932	33559	20155	9039	13404	2514
s38417	39360	28417	17519	10886	4981

Table 4.2.5 Double Vias Insertion for Row Utilization 80%

DVIT Program, Double Vias Insertion (Core Row Utilization: 95%)					
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	# of On-Pin Vias
s27	54	45	23	9	10
s298	395	296	183	99	64
s510	895	609	405	286	121
s832	1058	704	467	354	144
s1423	2041	1703	1076	338	351
s5378	5124	4127	2582	997	863
s13207	11123	7323	4311	3800	1967
s15850	12697	20273	9917	2490	2762
s35932	35813	27895	14445	7918	4225
s38417	41853	30135	17954	11718	5917

Table 4.2.6 Double Vias Insertion for Row Utilization 95%

4.2.2 Weighted Multiple Vias Insertion

Table 4.2.7 and Table 4.2.8 are obtained by our MVIT program. The third column,

Number of vias, increase dramatically due to the four-way via candidates be inserted. The last column, number of on-pin vias are also increased comparing with Figure 4.2.5.

MVIT Program, Multiple Vias Insertion (Core Row Utilization: 80%)					
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	# of On-Pin Vias
s27	52	99	41	9	13
s298	365	581	281	78	74
s510	773	1083	568	166	138
s832	958	1304	692	199	192
s1423	1966	3554	1710	334	428
s5378	4655	7833	3856	800	1026
s13207	10667	15995	7679	2986	2042
s15850	11890	19672	9588	2536	2403
s35932	33559	59032	15857	13399	3423
s38417	39360	65188	27678	10875	6671

Table 4.2.7 Multiple Vias Insertion for Row Utilization 80%

MVIT Program, Multiple Vias Insertion (Core Row Utilization: 95%)					
Circuits	Original # of Vias	# of Vias are inserted	# of On-Track Vias	# of Vias without VC	# of On-Pin Vias
s27	54	95	41	9	12
s298	395	589	293	99	81
s510	895	940	535	286	137
s832	1058	1127	629	354	162
s1423	2041	3523	1672	335	503
s5378	5124	7579	3830	997	1067
s13207	11123	14410	6749	3797	2432
s15850	12697	20273	9917	2490	2762
s35932	35813	75101	24654	7906	5551
s38417	41853	65555	28273	11703	7643

Table 4.2.8 Multiple Vias Insertion for Row Utilization 95%

Chapter 5. Conclusions

5.1 Contribution

A heuristic is proposed to solve the Multiple Vias Insertion problem. The approach is based on a bipartite graph model described in Section 3.1. It can be solved by Ford-Fulkerson algorithm. Practical issues such as pins in macro cells and obstacles are taken into consideration. Although our insertion rate of post-routing MVIT lower that SE's during routing double via function due to SE uses longer regular wire length, more vias and larger chip area. But for an existing layout, our MVIT provides a different kind of flexible option for inserting duplicated vias and uses less regular wire length for inserting duplicated vias. Moreover, MVIT can insert duplicated M1_M2 via into cell row and doesn't cause any violation.

5.2 Future Work

1. Because the stacked vias consist of multiple vias that may conflict with other nets coincidently and such conflict edges will destroy the bipartite graph structure. We will try to find a better heuristic to solve this problem.
2. We'll try to move tracks in order to increase the insertion of double/multiple vias in a congested layout.

References

[All02] Gerard A. Allan, "Yield/Reliability Enhancement using Automated Minor Layout Modifications," IEEE/SEMI Advanced Semiconductor Manufacturing Conference, pp. 252-261, 2002.

[And98] Andy, "*Minimum-Area Rule and Double-Cut Contact*," Application Note of Cadence Design Systems, Inc., 1998.

[Cad03a] Cadence Design Systems, Inc., "*LEF/DEF Language Reference*," Product Version 5.5, 2003.

[Cad03b] Cadence Design Systems, Inc., "*LEF C/C++ Programming Interface*," Product Version 5.5, 2003.

[Cad03c] Cadence Design Systems, Inc., "*DEF C/C++ Programming Interface*," Product Version 5.5, 2003.

[Cha99] Chin-Chih Chang and Jason (JingSheng) Cong," An Efficient Approach to Multilayer Layer Assignment with an Application to Via Minimization," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 18, No. 5, 1999.

[Che03] Shu-Yu Chen, "Constrained Via Minimization for Multilayer Interconnects with Stack Vias," Yuan Ze University, 2003.

[Chu02] Solon Chuang, "*Experience Sharing:Physical Implementation in SiS*," SiS Corporation Ltd., 2002.

[Cor01] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, "*Introduction to Algorithms*" McGraw-Hill Book Company, 2001

[Goe03] Richard Goering, "TSMC manager offers 90 nm design tips," EE Times, URL: *http://www.eedesign.com/story/OEG20030417S0029*, 2003

[Har01] Neil Harrison, Albuquerque, "A Simple Via Duplication Tool for Yield Enhancement," Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, pp. 39-47, 2001.

[Kah03] Andrew B. Kahng, "*On Design-Manufacturing Integration*," University of California, San Diego, 2003

[Leu03] Hardy Kwok-Shing Leung, "Advanced Routing in Changing Technology Landscape," Proceedings of the International Symposium on Physical Design, pp.118-121, 2003.

[Wil00] Doug Dreibelbis and Reg Wilcox, "Hidden Benefit #3: Design for Profit," IBM Microelectronics Fourth Quarter, pp.10-13, 2000

[Wil02] Ron Wilson, "Failures Plague 130-nanometer IC Processes," EE Times, URL: *http://www.eetimes.com/story/OEG20020826S0022*, 2002