YIELD ANALYSIS AND IMPROVING THE VIA INSERTION RATE WITH MULTI-VIA MECHANISMS

Y. Jhansirani
M. Tech
jansirani_y@yahoo.co.in

Abstract: In chip designing the yield analysis is very important task. Yield is defined as number of working chips over total manufacturing chips. In modern technology yield and redundancy is decreases due to failure of via. In order to avoid this failure of via we can use multi-via mechanism. These mechanisms we can do in post routing stage. By doing this we can expect congestion issues. To overcome this problem we can combine double via & rectangular via patterns that overcomes the limitation of double via insertion. finally the results shows that the proposed project will gives the result of improving yield and increasing insertion rate by using multi-via mechanism.

Keywords: yield, congestion, via, redundancy, multi-via.

I INTRODUCTION

As we know all about the technology of VLSI (very large scale integration). Integration generally defined as various components can be placed on a single chip. That requires chip area we can call it as technology. Now the present technology is on 18nm in that we are using 7 metal layers. Always top metal layer is used for power supply and below layer is for clock. 2nd, 3rd layers to place macros. Generally contact is connection to source, drain or poly while Vias is used to make connection between 2 metal layers. Vias are generally made of tungsten while contact is made using aluminum as shown in fig 1.

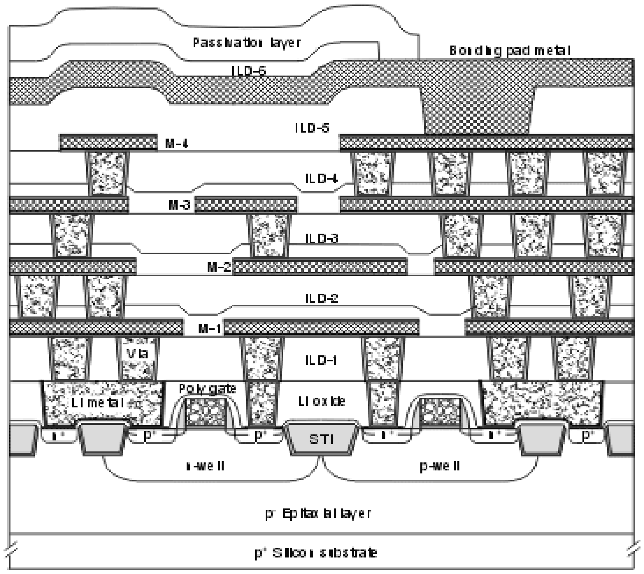


Fig 1 metal layers and vias and contacts

In this study, a novel via pattern, rectangle-via, is the first introduced in redundant via insertion research for full chip design. Figures 1(a-c) show the dimension of single via, double-via and rectangle-via, respectively. Note that as process scales into 65*nm* or beyond, the rectangle-via can be used for redundant via insertion, whose characterization depends on offerings of foundries. Fig. 1(d) illustrated that the rectangle-via can be treated as two seamless single-vias in physical layout geometries, and the area of rectangle-via is indeed two times the size of single-via. The rectangle-via has the same function with traditional double-via for yield improvement and has lower resistance than that of single via. Moreover, the rectangle-via uses less metal coverage area than traditional double-via does. In other words, the occupied routing area of rectangle-via is used more effectively than that of double-via.

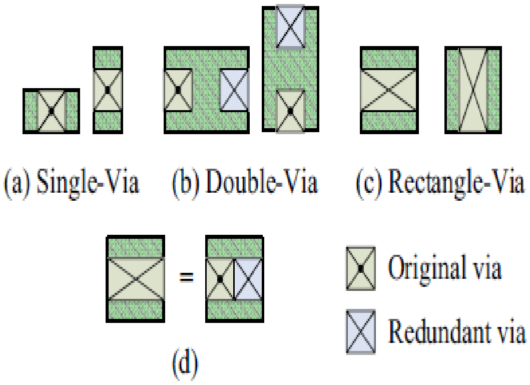


Fig 1.1 comparison of routing patterns

The voids in via is a serious issue in manufacturing. To avoid that problem we need backup vias that also known as redundant vias. In our project using multi-via mechanism. In this we use more than one via for single connection. Because if even one connection is defective it can be access with other connection that is double-via concept.

II PRINCIPLES OF SI ANALYSIS

A digital system can be examined at three levels of abstraction: logic, circuit theory, and electromagnetic (EM) fields. The logic level, which is the highest level of those three, is where SI problems can be easily identified. EM fields, located at the lowest level of abstraction, comprise the foundation that the other levels are built up on[3]. Most of the SI problems are EM problems in nature, such as the cases of reflection, crosstalk and ground bounce. Therefore, understanding the physical behavior of SI problems from EM perspective will be very helpful. For instance, in the following multi-layer packaging structure shown in Figure a switching current in via a will generate EM waves propagating away from that via in the radial direction between metal planes. The fields developed between metal planes will cause voltage variations between planes (voltage is the integration of the E-field). When the waves reach other vias, they will induce currents in those vias. And the induced currents in those vias will in turn generate EM waves propagating between the planes. When the waves reach the edges of the package, part of them will radiate into the air and part of them will get reflected back. When the waves bounce back and forth inside the packaging structure and superimpose to each other, resonance will occur. Wave propagation, reflection, coupling and resonance are the typical EM phenomena happening inside a packaging structure during signal transients. Even though EM full wave analysis is much more accurate than the circuit analysis in the modeling of packaging structures, currently, common approaches of interconnect modeling are based on circuit theory, and SI analysis is carried out with circuit simulators. This is because field analysis usually requires much more complicated algorithms and much larger computing resources than circuit analysis, and circuit analysis provides good SI solutions at low frequency as an electrostatic approximation Typical circuit simulators, such as different flavors of SPICE, employ nodal analysis and solve voltages and currents in lumped circuit elements like resistors, capacitors and inductors. In SI analysis, an interconnect sometimes will be modeled as a lumped circuit element. For instance, a piece of trace on the printed circuit board can be simply modeled as a resistor for its finite conductivity. With this lumped circuit model, the voltages along both ends of the trace are assumed to change instantaneously and the travel time for the signal to propagate between the two ends is neglected. However, if the signal propagation time along the trace has to be considered, a distributed circuit model, such as a cascaded RL- C network, will be adopted to model the trace. To determine whether the distributed circuit model is necessary, the rule of thumb is – if the signal rise time is comparable to the round-trip propagation time, you need to consider using the distributed circuit model.

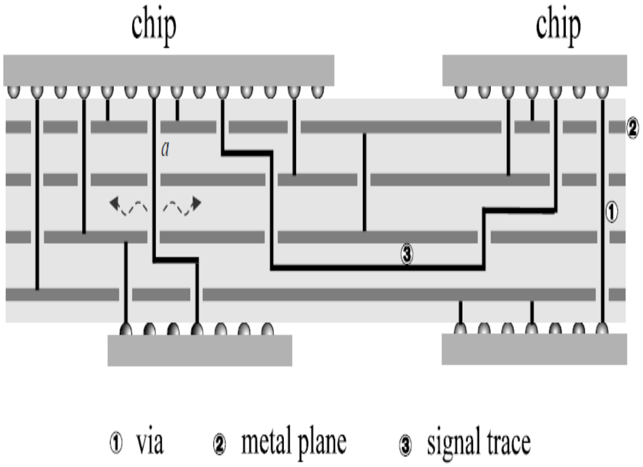


Fig 2 multi layer packaging structure

III. PROPOSED METHODS

As discussed in many previous works, via redundancy problem can be modeled by maximum bipartite matching problem as shown in Fig. 4(a) and (b). Given a routing layout, we first construct an undirected bipartite graph *Gb* = (*V, E*) which contains

$V = \{u \cup v : u \in L\ set, v \in R\ set\}$ and

$E = \{(u, v) : u \in L\ set, v \in R\ set\}$,

Where all vertices in *V* are classified into *L set* and *R set*. Here, *L set* is the set of original single-vias and *R set* is the set of redundant via candidates. For $u \in L\ set$ and $v \in R\ set$, the edge $(u, v) \in E$ declares the relationship between single-vias and its redundant via candidates. Therefore, maximum bipartite matching problem is declared as the maximum matching number of via redundancy.

A Weighted Bipartite Graph construction

Unlike the normal method to solve the maximum bipartite matching problem, we give the different weight for each edge in the bipartite graph. As shown in Fig. 3(a), we classify the redundant via candidates (R1-R6) into on-track candidates and off-track candidates. On-track candidates declare the redundant via candidates which are on the same metal with the original via (R4 and R5). The rest of redundant via candidates are off-track candidates (R1, R2, R3 and R6), which are not located on the metals. Since the on-track candidates can be used by its own original vias and the off-track candidates can be used by all near single-vias, we consider the on-track candidate as higher priority than off-track candidates. The weight of edges

$w(u, v)$ is defined as follows: $w(u, v) = 2$, if *v* is on-track candidate

1, if *v* is off-track candidate.

B Flow Network Construction

Based on the constructed weighted bipartite graph, we construct the flow network. Here, the vertex *s* and vertex *t* are added in the graph at first. The capacity of each edge is then defined as follows:

$c(s, u) = \infty$,

$c(v, t) = \infty$ and

$c(u, v)$ =2, if *v* is on-track candidate

1, if *v* is off-track candidate

Where all vertices $u \in L\ set$ of *Gb* and all vertices $v \in R\ set$ of *Gb*. In order to perform the maximum matching of bipartite graph and priority consideration of via candidates at the same time, we give the different capacity for each edge. We assume that the capacities from *s* to *u* and from *v* to *t* are ∞. Then, we assume that the capacity from *u* to *v* are two if *v* are on-track candidates and the capacity of off-track candidates are one. After the capacity of each edge is assigned, each flow of path *p* will only be determined by the capacity $c(u, v)$. Next, we sort all vertices *u* by their degrees $d(u)$ to declare the amount of each vertex's via candidates. Note that the vertices with fewer candidates maybe starve if they are not assigned with higher priority, so we should sort these vertices with increasing order of their degrees. As a result, the solution is also the maximum matching of bipartite graph. Figure3 shows an example of the flow network construction and the operation of modified Ford-Fulkerson algorithm. As shown in Fig. 3(a) and (b), we model the relationship between original vias and their redundant via candidates into the flow network. Besides, we assign a different capacity for each edge which is shown in Fig. 3(b). Since R4 and R5 are the on-track candidates, the edge capacities of $c(V\ 2,R4)$ and $c(V\ 3,R5)$ are two, and the residual edges' capacities between *V* s and *R*s are one. For example as shown in Fig. 3(c), a path *p* which starts from vertex *s* and goes to vertex *t* via V2. Here, there are three candidates (R2, R3 and R4) for V2. Since we pick up the highest capacity for each DFS level in our proposed method, the capacity $c(V\ 2,R4) = 2$ will be directly selected in this way. Also, $c(V\ 3,R5) = 2$ will be selected in the same way. Last, Fig. 3(d) shows the result of redundant via insertion in this example. Since there are on-track candidates for V2 and V3, the redundant vias of V2 and V3 are on-track, and V1 gets a redundant via which is off-track.

C. Redundant Via Insertion (RVI)

To implement redundant via insertion in our method, we use *RV I* to change thevia type with design rule consideration. First, we declare each single-via on post-routing layout with data structure *V s*, which is:

```
structure Vs {
status; // sv, rv or dv
x; y; // coordinate
};
```

Where the *status* could be *sv* (single-via), *rv* (rectangle-via) or *dv* (double-via). The *x* and *y* are the coordinates for presenting the location of the *V s*. Then we declare the feasible redundant via candidates with data structure *Rs*, which is:

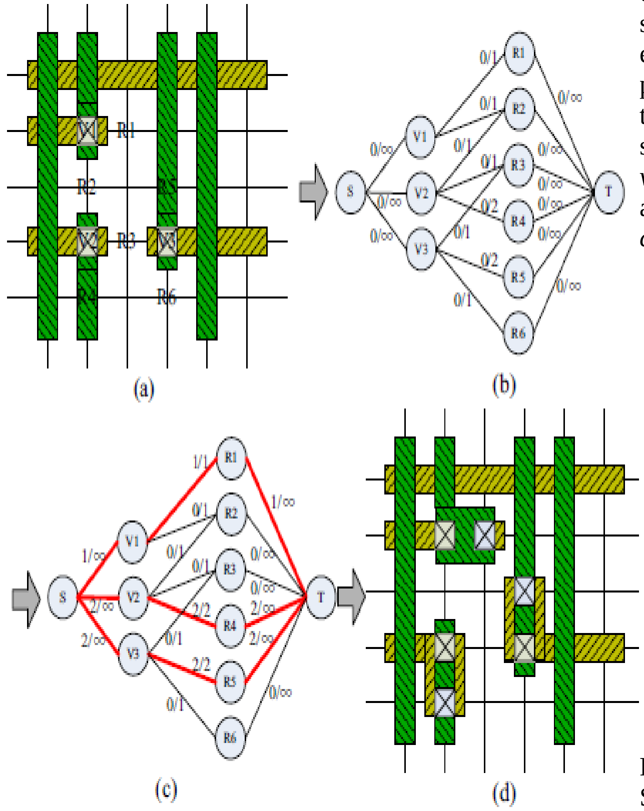


Fig 3 modified redundancy problem

```
structure Rs {
x; y; // coordinate
vector; // u, d, l or r
alive; // d or a
verify count; // default 0
on track; // y or n
};
```

where *x* and *y* is the location of the *Rs*, and *vector* is the vector point to this *Rs* from its *V s*, its value could be *u* (up), *d* (down), *l* (left) or *r* (right). The *alive* means that the *Rs* is really feasible or not, its value could be *d* (dead) or *a* (alive). The *verify count* is the times of checking design rule and its default value is 0. Last, *on track* means the *Rs* is on the same net with its *V s* or not, and its value could be *y* (yes) or *n* (no). After declaring the data structure of each original single-via and redundant via candidate, we represent each single-via by *V i*, where $0 \leq i \leq n$, and *n* is the total number of single vias. We also represent each redundant via candidate by *Rj*, where $0 \leq j \leq m$, and *m* is the total number of redundant via candidates. Then we model the relationship between *V i* and its *Rj* into Bipartite Graph which is described previously. As shown in Fig. 6, we check the bounding box for each *Rj*. The area of bounding box is determined by the area of double-via plus the minimum spacing *α*. The minimum spacing is one of the design rules which determine the minimum space between metals in the same metal layer. For example, the minimum spacing is 0.1*um* in 65*nm* process. For each *Rj*, we first check the bounding box of two sides which are perpendicular to the *vector*. If one side does not violate with design rule, then we perform *verify count* += 1 and check another side. If another side also does not violate with design rule, we perform *verify count* += 1 again

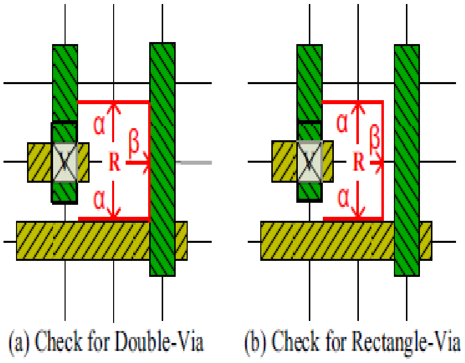
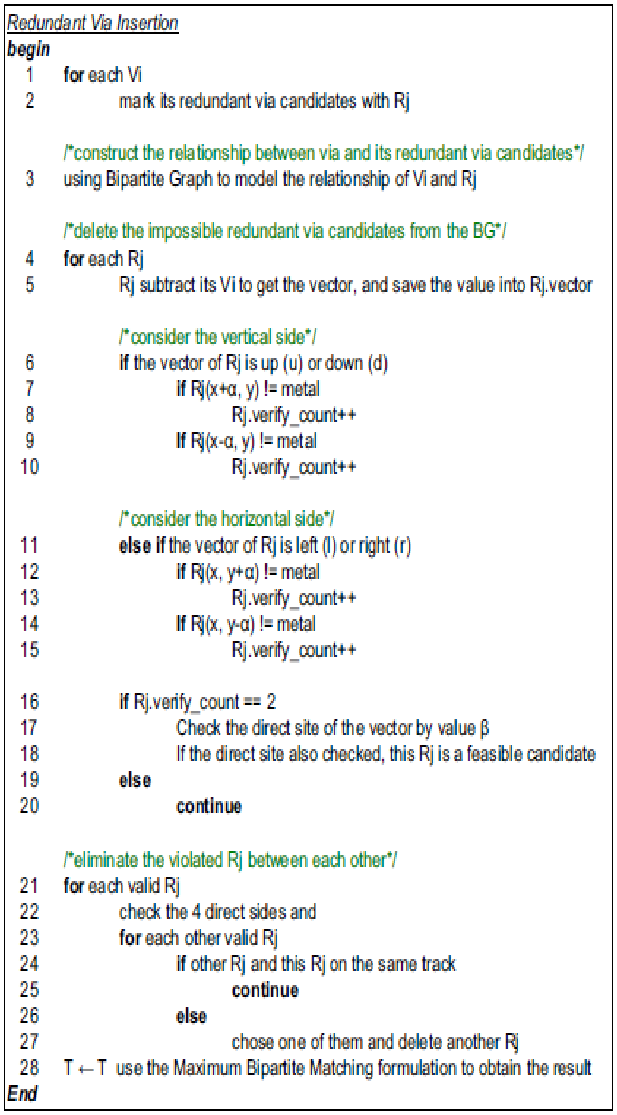


Fig 3.1 design rule check for redundant via

So far, if the *verify count* == 2, it means that both sides of vector are not violated with design rule. After the *verify count* checked, we next check the front side of the *vector* by the value *β* as shown in Fig. 6(a) and (b). Here, the value *β* is determined by the redundant via pattern; if the redundant via pattern is double-via, the value *β* would be equal to *α*. On the other hand, if the redundant via pattern is rectangle-via, the value *β* would be equal to 1/2 *α* since the rectangle-via is smaller than double-via. Once these three sides are checked, we can delete the infeasible *Rj* from the bipartite graph and simplify the bipartite graph. Last, for each residual *Rj*, we check them if they violate with other *Rj* or not. If they violate with each other, then we will delete one of them until no violation and simplify the bipartite graph again. After these steps, the rest of *Rj* are the feasible redundant via candidates and we can obtain the result by solving maximum bipartite matching problem which is declared in Section III-A. In the end, the *RV I* algorithm is summarized in Fig. 7.

C. Enhancement of the Insertion Rate for Redundant Via

Since the double-via is the mainstream of redundant via, we perform our *RV I* by double-via first. As the same result with others' proposed methods, there are some single-vias still left on IC layout after inserting the double-vias in interconnects. As shown in Fig. 3.1(a) and (b), after we perform *RV I* by double-via to insert the double-vias in this high density interconnects, many vias (the via with circles) are still single via. To enhance the insertion rate of redundant via based on the result shown in Fig. 9(b), we try to insert the rectangle via into this IC layout. In general, the commercial placed and-routed tools (such as Synopsys IC Compiler) treat the rectangle-via as a larger single-via and replace the single via directly. However, it may change the IC layout and affect timing or other redundant vias' insertion. In comparison with the commercial placed-and-routed tools, we treat the rectangle via as a kind of redundant via. It can easily be performed by *RV I* algorithm, thus refining the insertion rate without changing the IC layout. fig 3.2 shows the flowchart of the proposed scheme associated with double-via and rectangle-via insertion. In the first round, we perform *RV I* to insert the double-vias. After performing each round of *RV I*, we should check the output information which contains whether the rectangle-via or not.



```
Redundant Via Insertion
begin
    for each Vi
        mark its redundant via candidates with Rj

    /*construct the relationship between via and its redundant via candidates*/
    using Bipartite Graph to model the relationship of Vi and Rj

    /*delete the impossible redundant via candidates from the BG*/
    for each Rj
        Rj subtract its Vi to get the vector, and save the value into Rj.vector

        /*consider the vertical side*/
        If the vector of Rj is up (u) or down (d)
            If Rj($x+\alpha$, y) != metal
                Rj.verify_count++
            If Rj($x-\alpha$, y) != metal
                Rj.verify_count++

        /*consider the horizontal side*/
        else If the vector of Rj is left (l) or right (r)
            If Rj(x, $y+\alpha$) != metal
                Rj.verify_count++
            If Rj(x, $y-\alpha$) != metal
                Rj.verify_count++

        If Rj.verify_count == 2
            Check the direct site of the vector by value $\beta$
            If the direct site also checked, this Rj is a feasible candidate
        else
            continue

    /*eliminate the violated Rj between each other*/
    for each valid Rj
        check the 4 direct sides and
        for each other valid Rj
            If other Rj and this Rj on the same track
                continue
            else
                chose one of them and delete another Rj
    T ← T  use the Maximum Bipartite Matching formulation to obtain the result
End
```

Fig 3.2 RVI algorithm

IV CONCLUSION

This paper proposed that yield and redundancy is increases by using multi-via mechanism in post routing stage. Experiment result significantly improved 5.9% redundant via insertion rates of total vias and reach up to 11.3% improvement of redundant via insertion rates in via1s. A nanometer-scale redundant via insertion scheme is presented, especially, the rectangle-via would be predicted to replace double-via since manufacturing yield of nanometer-scale enhanced and settled.

REFERENCES

[1] H.-Y. Chen, M.-F. Chiang, Y.-W. Chang, L. Chen, and B. Han, "Full-chip routing considering double-via insertion," *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, no. 5, pp. 844 –857, may 2008.
[10] K.-Y. Lee, T.-C. Wang, and K.-Y. Chao, "Post-routing redundant via insertion and line end extension with via density consideration," in
Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference, 5-9 2006, pp. 633 –640.
[11] J.-T. Yan, B.-Y. Chiang, and Z.-W. Chen, "Yield-driven redundant via insertion based on probabilistic via-connection analysis," in *Electronics, Circuits and Systems, 2006. ICECS '06. 13th IEEE International Conference*, 10-13 2006, pp. 874 –877.
[12] H. Yao, Y. Cai, Q. Zhou, and X. Hong, "Multilevel routing with redundant via insertion," *Circuits and Systems II: Express Briefs, IEEE Transactions*, vol. 53, no. 10, pp. 1148 –1152, oct. 2006.
[13] *IC Compiler*, www.synopsys.com.