國立交通大學

電子工程學系 電子研究所

碩 士 論 文

考量可偵錯式設計之版圖修正以利於進行聚焦離子束的訊號觀測與電路修正技術

Design-for-Debug Aware Layout Modification for FIB Net Observation and Circuit Editing

研 究 生： 張琮偉

指導教授： 周景揚 教授

中 華 民 國 九 十 九 年 十 一 月

考量可偵錯式設計之版圖修正以利於進行聚焦離子束的訊號觀測與電路修正技術

Design-for-Debug Aware Layout Modification for FIB Net Observation and Circuit Editing

研 究 生： 張琮偉　　Student： Tsung-Wei Chang

指導教授： 周景揚　　Advisor： Jin-Yang Jou

國 立 交 通 大 學

電子工程學系 電子研究所

碩 士 論 文

A Thesis
Submitted to Department of Electronics Engineering and
Institute of Electronics
College of Electrical and Computer Engineering
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in
Electronics Engineering

November 2010
Hsinchu, Taiwan, Republic of China

中華民國 九十九 年 十一 月

考量可偵錯式設計之版圖修正以利於進行聚焦離子束的訊號觀測與電路修正技術

研究生：張琮偉　　　　　　　　　　　　　　　　　　指導教授：周景揚博士

國立交通大學

電子工程學系　電子研究所碩士班

摘要

在現今矽除錯之流程中，晶片的錯誤分析是相當重要的一個環節。錯誤分析結果的優劣，取決於當晶片運行時，所能得到的訊號數量：若所得到的數量越多，則找出錯誤發生位置的成功率越高。在現行錯誤分析的技術中，聚焦離子束的技術被廣泛使用，因其（1）可直接觀察訊號值、以及（2）可直接對晶片進行修改的特性，使得錯誤分析的效率得到大幅度的改善。然而，隨著製程的演進，金屬線的間距與寬度皆變小、金屬層的密度提高，導致我們難以對任意的訊號進行觀察或修改的操作，進而使得錯誤分析的難度提昇。為了解決這個問題，在此篇論文中，我們藉由預先修改繞線，以提高（1）觀察率以及（2）修改率。實驗數據的結果顯示，經由我們的方法修改過的設計，可提高平均約50％的觀察率，修改率亦是原先的兩倍以上。

DESIGN-FOR-DEBUG AWARE LAYOUT MODIFICATION FOR FIB NET OBSERVATION AND CIRCUIT EDITING

Student: Tsung-Wei Chang Advisor: Dr. Jing-Yang Jou

Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University

Abstract

Failure analysis(FA) plays a critical role in today's silicon debugging flow. The efficiency of FA depends on how many net values can be observed while a chip is running. If we acquire more net values, locating the failure will become easier. On the contrary, few values of nets would result in more difficult FA. In modern techniques of FA, *focus ion beam* is widely used since it can directly observe net values and modify chips. These characteristics enhance the efficiency of FA. However, due to the advanced technologies, both the interval between each wire and width of wires become smaller. This phenomenon causes that observing and modifying arbitrary nets become more difficult. Thus, difficulty of FA will raise as well. To conquer this problem, we propose a methodology to modify the routing of a design in advance. In this manner, we can reach the goals: enhancing the (1)observing rate and (2)modifying rate. Our experimental results show that the improvement of observing rate reaches 50% on average and modifying rate doubled compared with the original design.

Acknowledgements

I would like to express my gratitude to all those who gave me the possibilities and assistances to complete this thesis. At first, I deeply appreciate my advisor Professor Jin-Yang Jou. He always makes beneficial suggestions to me and also provides resources of research. I am also appreciate Professor Chia-Tso Chao for his kindly guidance. I would like to thank Meng-Chen Wu and all members of EDA laboratory. Without them, it is impossible for me to accomplish this thesis.

Last but not least, I wish to thank my parents, family and friends for their invaluable support, advice and encouragement throughout my study years.

Tsung-Wei Chang

National Chiao Tung University

Novmeber 2010

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Post-Silicon Validation/Debug

Due to the increasing complexity of modern designs and uncertainty of advanced process technologies, some design errors are difficult to be detected by pure simulation during the design phase and hence are more likely to escape from the current design verification flow, which leads to a low first-silicon success rate for today's modern designs. As a result, *post-silicon debug* (or failure analysis) becomes a critical and necessary step in the current design flow to identify the root causes of the escaped errors based on the failed silicon chips and to further fix them. Therefore, the effectiveness and efficiency of the post-silicon debug will significantly affect the time and cost of achieving design closure [2][7].

Unlike pre-silicon debug whereon the value of internal signals can be obtained easily through simulation, post-silicon debug has no direct access to the internal signals of a failed chip and relies on specialized circuit features or physical probing techniques to observe those internal signals. The specialized circuit features include DFT (design for test) scan-based designs [5] and DFD (design for debug) trace-buffer-based designs [2][3][4][15], which can dump the value of pre-selected flip-flops or internal signals. However, those pre-selected signals may not be near the physical fault locations and the provided visibility is only for a one-cycle snapshot or a limited number of cycles. Therefore, physical probing techniques are still required to observe the value of critical signals for post-silicon debug.

1.2 Physical Probing Techniques

Physical probing techniques include electron beam (E-beam) probe [12], laser voltage probe (LVP) [17], and focused ion beam (FIB) technique [1][10][11]. We will present a brief description of each technique.

1.2.1 E-beam

E-beam probe can observe the signal on the top two metal layers through capacitive coupled voltage contrast, and can further cooperate with FIB mill techniques to probe the signal on the bottom metal layers from backside [13]. However, advanced process technologies can easily contain more than 6 metal layers and hence the observable signals through E-beam probe are limited.

1.2.2 LVP

LVP is a backside probe technique and measures a signal by transforming the amplitude of the reflected laser beam into a voltage. The bandwidth of LVP is about 10 GHz and hence is especially suitable for analyzing delay defect. However, for 65nm technology, its transistor size is already smaller than the resolution of LVP [8]. As a result, in order to use LVP in advance technologies, additional type of cells need to be inserted into the circuit [8], or the cells to be observed need to be replaced with larger cells [9], which results in extra area overhead.

1.2.3 FIB

FIB technique utilizes ion beam to remove the covered inter-layer dielectric (ILD) above the signal to be observed and then deposit metal into the hole to form a probe pad directly connecting the signal. This FIB probing requires no area overhead to the design, is not only limited to the top metal layers, and is relatively cost effective with shorter process time in current industry. In addition, the FIB technique can also be used for circuit editing, such as cutting existing metal and reconnecting it to a desired location (usually pre-placed spare cells). This circuit-editing technique can quickly implement a simple circuit modification to repair the failed chip without going through another tape-out and hence can significantly speed up the whole silicon debug process.

When using FIB probing (or circuit editing), we need to make sure that the metal of other signals above the observation location will not be touched or removed. Otherwise, the probe pad may be connected to multiple signals or the original circuit functionality may be changed. Unfortunately, when the technology node continually shrinks, the circuit layout becomes more dense and more signals may be blocked by the other signals on top of them for FIB probing or circuit editing [14]. We proposed an automatic tool to efficiently identify the locations which can be used to perform the desired FIB circuit editing. However, no current APR (automatic place and route) tool can create a layout which is friendly for applying FIB probing or circuit editing, if needed.

1.3 Organization of the Thesis

In this work, we propose a DFD framework, which modifies the circuit layout to maximize the probability that a signal can be observed by FIB probing as well as the probability that a signal can be repaired through FIB circuit editing. The layout modification is done through a few pre-defined actions, which move a small portion of the existing metal lines to different metal layers with new vias, instead of performing a complete rerouting. Therefore, the proposed DFD framework can be applied in conjunction with any APR tool. Also, the proposed DFD framework can consider the timing critical paths of a circuit and restrict the layout modification to the non-timing-critical paths only, making its impact on the circuit's performance under control. The experimental results based on an 90nm technology demonstrate that the proposed DFD framework can effectively increase the number of signals to be observable or repairable with FIB technique while the overall area and circuit performance remain the same.

The rest of thesis is organized as following. We will introduce the frontside FIB techniques and basic idea about observing signals with routing layers adjusting in Chapter 2. In Chapter 3 and Chapter 4, we will present the usage of FIB techniques and methodologies. Then we will show the experimental results of our work in Chapter 5. Lastly, Chapter 6 will conclude this work.

Chapter 2

Background

Before mentioning our methodologies, we need to introduce the FIB technique including its nature, process, and parameter in this chapter.

2.1 Introduction to FIB

An FIB system operates in a similar manner as a scanning electron microscope (SEM) or a transmission electron microscope (TEM) except that the FIB system utilizes a focused beam of ions (gallium most of the time) instead of a beam of electrons. When operating at a low beam current, a focused ion beam can be used for imaging the sample surface with high resolution. When operating at a high beam current, a focused ion beam can be used for milling the surface. Because the ions are larger, heavier, slower, and positive compared to electrons, the ion beam cannot easily penetrate within individual atoms of the sample and hence can more easily break the chemical bonds of the substrate atoms, which makes FIB suitable for surface milling [6].

Figure 2.1 illustrates an example of using FIB technique to observe a target signal inside a chip. In Figure 2.1(a), the surface milling is performed by applying a focused ion beam of Ga+ to hit the surface of inter-layer dielectric (ILD), break the bonds of a certain amount of surface material, sputter out ions (mostly positive ions), and gradually form a hole right above the target signal. Meanwhile, an electron beam is applied to the surface to neutralize the sputtered positive ions, and sometimes certain gas (such as XeF2) is also applied to assist the etching (mainly for preventing the re-deposition of the sputtered surface material). Next, in Figure 2.1(b), FIB is used to deposit metal (Pt in this case) onto the dug hole. When the ion beam hits the gas of metal, the metal will chemisorb on the surface through FIB-assisted chemical vapor deposition [6]. The deposition of metal then forms a probe pad for the target signal.

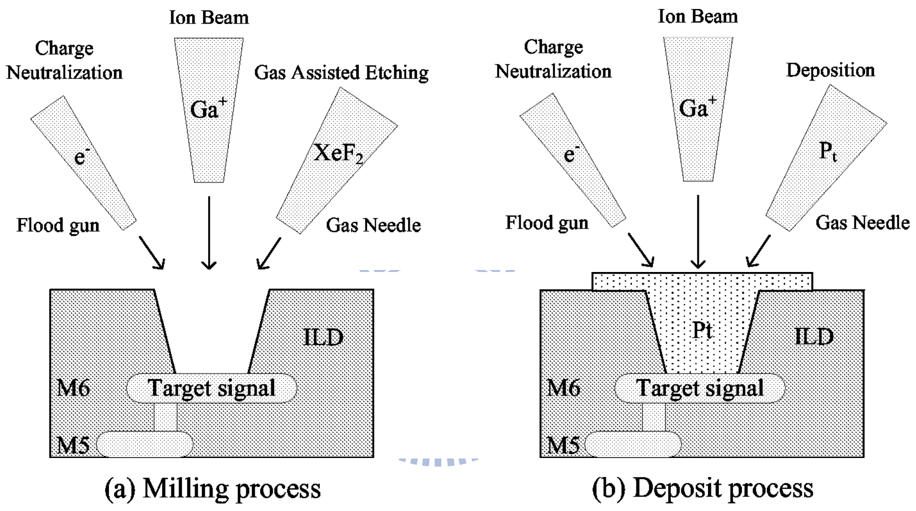


Figure 2.1: An example of FIB probing using (a) FIB surface mill and (b) FIB deposition.

2.2 Parameter of FIB

In order to successfully finish an FIB probe or FIB circuit editing, the area of the bottom of the FIB-dub hole (usually square), defined as *baseline window*, needs to be large enough. The larger the baseline window, the higher the probability of a successful FIB action. Since the focused ion beam or its reflection from the surface may also hit the edge of the dug hole during the surface milling, the edge of the dug hole is not directly orthogonal to the baseline window but a few angels outward from bottom to top, as illustrated in Figure 2.2. As a result, for each higher metal layer, the width of its transverse section with the dug hole is an offset larger. In other words, if we plan to dig a hole to a lower metal layer, the area saved for the top of the hole needs to be larger even though the size of the baseline window is the same for all metal layers.

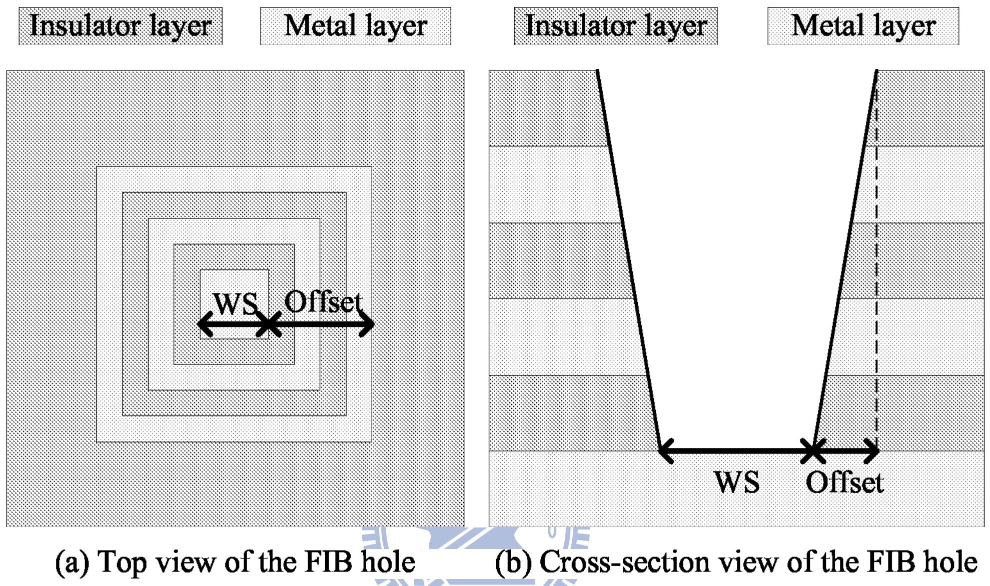


Figure 2.2: The different views of a FIB hole.

For general FIB technologies used in current failure-analysis companies, the minimal sufficient width of the baseline window, denoted as WS, is around 800nm to 1000nm. The slope of the hole's edge is around 1-to-10 (x-direction versus y-direction). For the UMC 90nm technology used in our experiment, such a 1-to-10 slope will create a 5nm offset for the transverse width of the hole on each higher metal layer.

Chapter 3

FIB Observable Nets

In this chapter, we will explain our methodology to maximize the observable nets for FIB techniques. First, we will define *Net Observation Rate* and describe the methodologies to modify the design netlist. Then, we will explain how to make a net observable by the combination of operations. Besides, we will propose a ranking algorithm to decide the order of modification. Lastly, the overall flow will be presented to illustrate our framework.

3.1 Introduction

To confirm an error candidate during the debug process, failure analysis engineers often need to directly observe the signal of some specific nets in the design netlist. Since defects may occur anywhere in the netlist after manufacturing, it is always desirable to have as many post-silicon observable nets as possible.

With the FIB technique, we know that the value of a certain net can be detected. However, with growing of complexity of designs, it is impossible to probe all the nets in the netlist. Usually, more than half nets in the netlist cannot be observed directly. Therefore, it is essential to apply some operations with layout to increase observable nets.

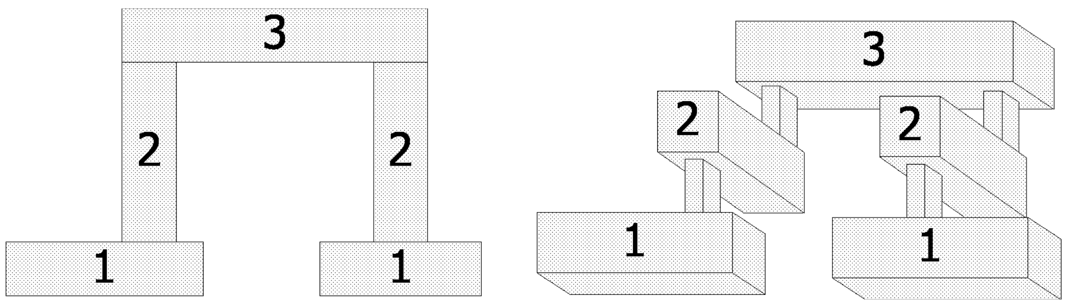
Section 3.2 will introduce the definition of *Net Observation Rate* and formulate our problem. Section 3.3 will introduce the operations with a layout and an overall flow of our framework. Lastly, Section 3.4 will give a short conclusion to this chapter.

3.2 Preliminary

This section presents the definition of *Net Observation Rate* precisely, and then formulate the problem we attempt to solve.

3.2.1 Net Observation Rate

A net in a design netlist may correspond to several metal segments across different metal layers. As shown in Figure 3.1, the net is distributed between layer 1-3 and each segment connects each other with vias. A net n is defined as *FIB observable* if all of the following conditions hold: (1) we can dig an FIB hole inside which no metal of other nets locates originally, (2) the width of the baseline window exceeds the minimal sufficient width WS, and (3) the baseline window of the hole overlaps at least WS long of any metal segment of the net n. *Net Observation Rate(NOR)* can then be defined as the portion of *FIB observable nets* in a design netlist as the following equation.



(a) Top view of a net   (b) 3D view of a net

Figure 3.1: Illustration of a net in different layers.

$$NOR = \frac{\text{number of } \textit{FIB observable} \text{ nets}}{\text{number of total nets}} \times 100\%$$

Net Observatoin Rate indicates how many nets can be applied to FIB techniques. In other words, this rate shows how much additional information we will receive in failure analysis.

3.2.2 Problem Formulation

According to the FIB techniques and the discussion from previous section, if more nets are reserved at the higher layers, we could have more observable nets and higher successful FIB results of a design. Thus, the problem of observable nets maximization can be defined as following:

INPUT

- The original layout file
- The width of baseline window for FIB holes
- The offset for FIB holes

OUTPUT

- The modified layout file

OBJECTIVE

- Maximize NOR with frontside FIB probing

CONSTRAINT

- Minimize the modification to the original layout

3.3 Methodology

In this section, we will first introduce the proposed operations to change a routing. Then, we will demonstrate how to combine these operations to make a net observable. Third, we will determine the order of modification of nets. Lastly, we will present the overall flow of our proposed methodologies.

3.3.1 Basic Operations

To change the layout, we propose two operations, *moving-up* and *moving-down*, and describe each of them as following. To minimize the impact of delay, the operations we proposed will only change the routing layer of nets.

3.3.1.1 Moving-up

The first operation is *Moving-up* which moves the target segment up to higher routing layers. To minimize the impact of the original routing, we only move partial segment up. For higher successful rate of FIB, the length of partial segment we move should reach the width of *baseline window* as well as possible. First, we check whether there is sufficient space for adding both vias and segment at higher layers. If above conditions hold, then we partition the target segment, move the one whose upper space is free to higher layer, and maintain the connection of these segments by adding vias. We use Figure 3.2 for illustration.

In Figure 3.2(a), segments a, c, and d can be directly observed. Since the space among segments a, d, and c is enough for via insertion and the space above segment b in $M6$ is free, we divide segment b into three segments b_1, b_2, and b_3. Then, segment b_2 is moved to $M6$ and new vias are inserted to maintain the connection. After modification, we can observe segments a, b, and c as shown in Figure 3.2(b). In Figure 3.2(b), segment d needs to find another location for observing since it is

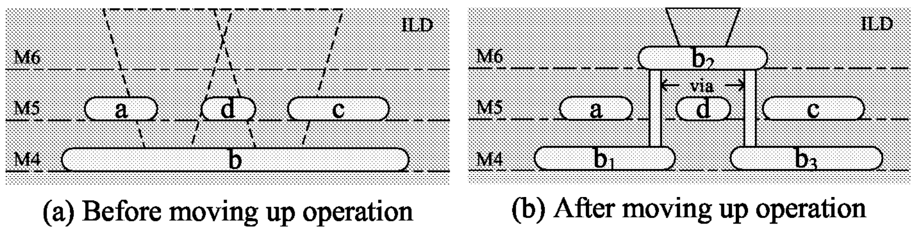


Figure 3.2: (a)Segment b cannot be observed due to insufficient space for FIB milling. (b)Segment b is partitioned into three parts, b_1,b_2 and b_3, and then b_2 is moved to $M6$. Segment b_2 then becomes observable.

blocked by segment b_2.

3.3.1.2 Moving-down

Moving-down is our second operation. Similar to *Moving-up*, we first check whether there is sufficient space for adding both vias and segment(s) at the lower layers. If above conditions hold, then we move the whole segment to lower layer, and maintain the connection by adding vias.We use Figure 3.3 for illustration.

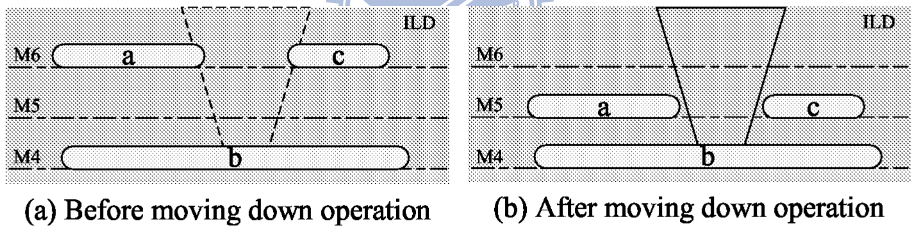

ILD
M6
a
c
M5
M4
b
(a) Before moving down operation
ILD
M6
M5
a
c
M4
b
(b) After moving down operation

Figure 3.3: (a)Segment b cannot be observed due to segments a and c. (b)Segments a and c have been moved down to release the space. As a result, segment b then becomes observable.

In Figure 3.3(a), the necessary space for observing segment b is blocked by segments a and c. To make segment b observable, we first check the space below segments a and c,and find out that space below segments a and c at $M5$ is free. Then, segments a and c are moved to $M5$ to release the space for segment b. After successfully moving these segments down, segment b is finally observable as shown in Figure 3.3(b).

3.3.2 Combinations

In Section 3.3.1.1 and Section 3.3.1.2, we have already introduced our operations in detail. In this section, we will explain how we use them to modify a layout.

While we attempt to modify an unobservable segment, *Moving-up* is usually applied first since it gives little impact to design netlist than *Moving-down*. If applying *Moving-up* to the target segment only can not make it observable, then we will apply *Moving-down* to the segment(s) which block(s) our target segment. However, *Moving-down* often comes with sacrifices, that is, the moved down segment often becomes unobservable after applying this operation. Therefore, unless the observable segment of the net possesses more than one observable segment or another unobservable segments can be observed while applying modifications, we will not apply *Moving-down* to the observable segment. If it still does not work, we will apply both *Moving-up* and *Moving-down*. If an unobservable segment can be observed according to the above methodology, we call this segment a *potential segment*.

3.3.3 Ranking

Every modification is bound to affect the whole design. That is, different orders of modification will lead to different final NOR. Therefore, we must find a modification order to maximize NOR. For a start, we should give each net a score for ranking.

To model the cost of making nets observable, we use number of movements to describe the impact of original design. For example, in Figure 3.2(b), we use one operation to move part of segment b to $M6$, so the cost of making segment b observable is 1. In Figure 3.3(b), the cost of segment b is 2, because we move down

```
Algorithm: Ranking for unobservable nets
Input: The set of unobservable nets, the layout file,
       the offset for FIB holes, and the width of baseline window
Output: The ranking list of unobservable nets
1.  for each unobservable net $n_i$
2.    check all segments of net $n_i$
3.      if none of segments is potential segment
4.        then set infinity as a score of $n_i$
5.      else
6.        for each potential segment $s_j$
7.          evaluate and record the cost of modification of $s_j$
8.        set minimum cost as a first-priority score of $n_i$
9.        set number of potential segment as a second-priority score of $n_i$
10. sort unobservable nets according to the first-priority score
11.   if the first-priority scores are equal
12.     then sort according to the second-priority score
13. return ranking list of unobservable nets
```

Figure 3.4: The flow of ranking unobservable nets.

two segments to release the space for segment *b*. Then, we use this information to help us determine the order of list.

We show the flow of ranking unobservable nets at Figure 3.4. First, we check whether there is any *potential segment* among a net n_i. If n_i does not have any *potential segment*, we set an infinity score to it. In other words, n_i is impossible to be observable under our methodology (Lines 3-4). If there is at least one *potential segment*, we evaluate the cost of every *potential segment* (Lines 6-7). After analyzing every *potential segment*, we set the minimum cost among *potential segments* as a first-priority score and number of *potential segments* as a second-priority score(Lines 8-9). After applying this analysis to all unobservable nets, we then determine the order of list. This list will be sorted according to first-priority scores from minimum to maximum (Line 10). If there are two or more nets keep the same score, second-priority score will be considered to determine the order of these nets. Order of second-priority score is identical to first-priority score, which is from minimum to maximum (Line 11-12). Finally, we obtain the ranked list of unobservable nets (Line 15).

3.3.4 Main Flow

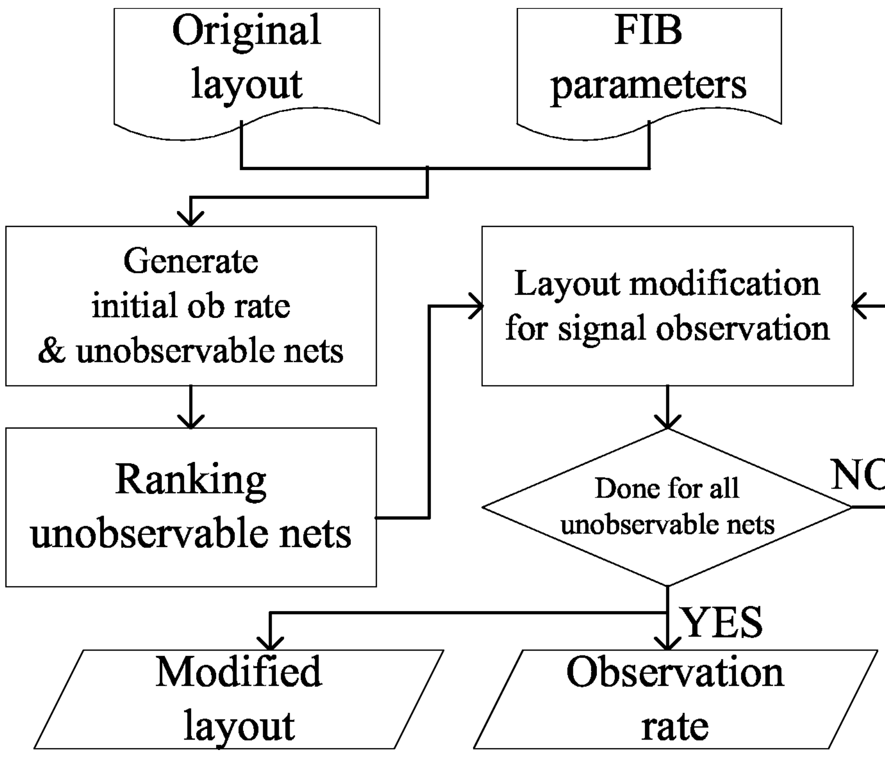

Original layout
FIB parameters
Generate initial ob rate & unobservable nets
Ranking unobservable nets
Layout modification for signal observation
Done for all unobservable nets
NO
YES
Modified layout
Observation rate

Figure 3.5: The flow of observable signals maximization

In this section, we explain the flow of observable nets maximization with Figure 3.5. There are two parts of the inputs: the one is the original layout, and the other is the set of parameters of FIB techniques. By these parameters, we can set the aspect ratio and the width of *baseline windows* according to different FIB techniques.

First, with the aspect ratio and the width of *baseline windows*, we analyze the original layout to obtain the initial *Net Observation Rate* and the set of unobservable nets. After determining the order of list of unobservable nets, we apply the modifications in sequence to all unobservable nets in the list. Finally, we acquire the modified layout and *Net Observation Rate*. To confirm the correctness of modified layout, we will also feed it back to commercial tools to check if it is still a legal layout.

3.4 Summary

In this chapter, we introduce *Net Observation Rate* and formulate the problem, maximizing FIB observable nets via layout modification. By a combination of the two operations, *Moving-up* and *Moving-down*, we can make some unobserved nets observable. We also present a strategy to determine the order of modification. Lastly, we propose our overall flow for maximization of FIB observable nets.

Chapter 4

Circuit-Editing

In Chapter 3, we have already introduced one of the usages of FIB techniques, *FIB Observation*, and come up with a framework to maximize *Net Observable Rate*. In this chapter, we will introduce another usage of FIB techniques, *Circuit Editing*. Similar to *Net Observable Rate*, we also define *Circuit Editing Rate* and describe the methodologies to improve it. The ranking algorithm and overall flow will be presented as well.

4.1 Introduction

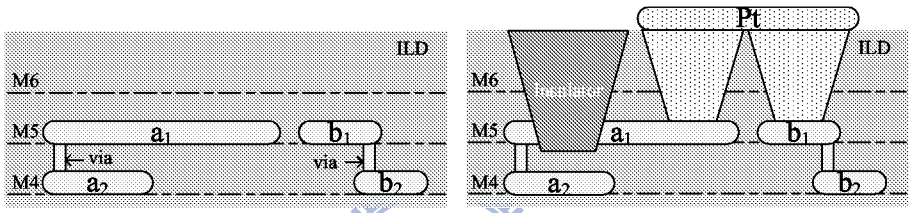
Circuit editing is also an important technique to reduce the time of failure analysis, besides net observation mentioned in Chapter 3, . Different from net observation which only observes the value of certain nets, circuit editing attempts to repair the design through cutting existing metal or reconnecting a segment to a desired location (usually a pre-placed spare cell). Circuit editing technique can quickly and simply modify a circuit to repair the failed chip without going through another tape-out. Hence, it can significantly speed up the whole silicon debug process. Therefore, it is also desirable to have as many post-silicon circuit-editable nets as possible.

We will introduce the definition of *Circuit Editing Rate* and formulate our problem in Section 4.2, and our operations with a layout and an overall flow of our framework in Section 4.3. Lastly, we will give a short conclusion to this chapter in Section 4.4.

4.2 Preliminary

This section will present the definition of *Circuit Editing Rate* precisely, and then formulate the problem we attempt to solve.

4.2.1 Circuit Editing Rate

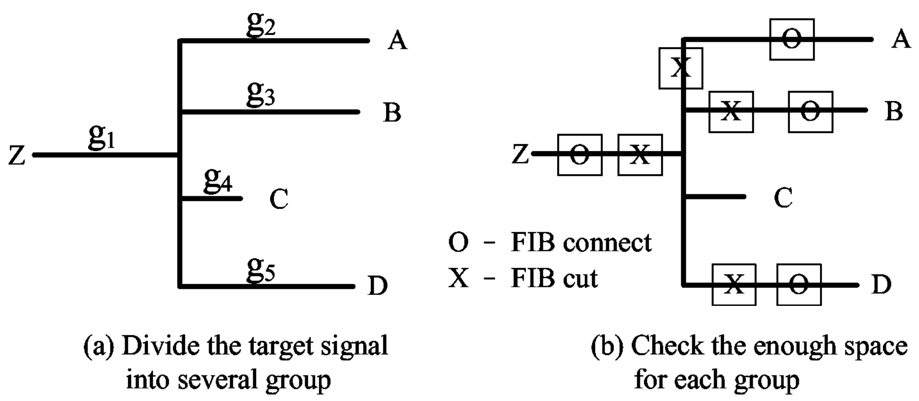


(a) Before circuit editing (b) After cutting and re-connection

Figure 4.1: Cut and reconnection of circuit editing.

Unlike net observation, circuit editing needs to cut and reconnect the desired segments. That is, for a successful circuit editing, we reserve two feasible locations for cutting and reconnecting as shown in Figure 4.1, while net observation only needs one for observing.

Every segment of a net will apply circuit editing individually if necessary. Thus, we partition a net to several *groups* according to the effects: acquiring the same result after modifying. To simplify partitioning, we only group the segments which lie between pins and branches. That is, the number of groups of a net equals to the number of pins this net connects if the fanout of this net is more than one. In Figure 4.2(a), pin Z drives four pins, A, B, C, and D. Therefore, there are five groups, g_1 to g_5.



Z
g1
g2
g3
g4
g5
A
B
C
D
(a) Divide the target signal into several group
Z
O
X
A
B
C
D
O - FIB connect
X - FIB cut
(b) Check the enough space for each group

Figure 4.2: Check the condition of signal for circuit editing. g_4 cannot be edit due to no enough space.

Thus, in circuit editing, we take a group as a unit instead of a net. Our concern will be how many groups is editable. That is, we would like to know the number of groups which possess at least two feasible locations. In Figure 4.2(b), groups g_1, g_2, g_3 and g_5 are all editable due to sufficient feasible locations. In contrary, group g_4 is not editable because there is not enough suitable location to edit.

Circuit Editing Rate(CER) can then be defined as the portion of editable groups in a design.

$$CER = \frac{\text{the number of editable groups}}{\text{the number of total groups}} \times 100\%$$

4.2.2 Problem Formulation

According to the FIB techniques and the discussion from previous section, the problem of circuit-editable groups maximization can be defined as following:

INPUT

- The original layout file

- The width of baseline window for FIB holes
- The offset for FIB holes

OUTPUT

- The modified layout file

OBJECTIVE

- Maximize the CER with frontside FIB probing

CONSTRAINT

- Minimize the modification to the original layout

We treat a *group* as a subnet and attempt to make it possess at least two probing locations. Therefore, the problem of maximizing CER will be similar to the one of maximizing NOR. Differences between these two problems are as follow: (1)we divide the nets to several subnets which are called *groups*, and (2)CER needs two probing locations in a group instead of one location in a net.

4.3 Methodology

After formulating the problem, we will propose our methodology to enhance CER in this section. Due to the similarity between net observation and circuit editing, the operations are also the same in *Moving-up* Section 3.3.1.1 and *Moving-down* Section 3.3.1.2. Therefore, in this section, we only focus on their differences: ranking and flow.

```
Algorithm: Ranking for uneditable groups
Input: The set of uneditable groups, the layout file,
       the offset for FIB holes, and the width of baseline window
Output: The ranking list of uneditable groups
1.  for each uneditable group $g_i$
2.    check all segments of group $g_i$
3.      if sum of probing segments and potential segments
          is less than two
4.        then set infinity as a score of $g_i$
5.      else
6.        for each potential segment $s_j$
7.          evaluate and record the cost of modification of $s_j$
8.        if original number of probing segment is 1
9.          then set minimum cost as a first-priority score of $g_i$
10.       else
11.         sum the smallest two cost and set as a first-priority score of $g_i$
12.       set number of potential segments as a second-priority score of $g_i$
13. sort uneditable groups according to the first-priority score
14.   if the first-priority scores are equal
15.     then sort according to the second-priority score
16. return ranking list of uneditable groups
```

Figure 4.3: The flow of ranking uneditable groups.

4.3.1 Ranking

With the same reason as maximizing NOR, we will also propose a ranking algorithm to determine the order of modification as shown in Table 4.3.

First, we check whether the sum of probing segments and *potential segments* among a group n_i is equal or larger than two. If the sum is less than two, we set an infinity score to it. In other words, g_i is impossible to be edited(cut and reconnect) at the same time under our methodology (Lines 3-4). If the sum is equal or more than two, we evaluate the cost of every *potential segment* (Lines 6-7). After analyzing every *potential segment*, we set a first-priority score to g_i according to the number of probing segments. If the number of probing segments is one, we set the minimum cost among *potential segments* as a first-priority score. If it is zero, we sum the smallest two cost among *potential segments* and set the sum as a first-priority score (Line 8-11). We set the number of *potential segments* as a second-priority score(Lines 12). After applying this analysis to all unobservable nets, we then determine the order of list. This list will be sorted according to first-priority scores from minimum to maximum (Line 13). If there are two or more nets keep the same score, second-priority score will be considered to determine the order of these groups. Order of second-priority score is identical to first-priority score, which is from minimum to maximum (Line 14-15). Finally, we obtain the ranked list of uneditable groups (Line 16).

4.3.2 Main Flow

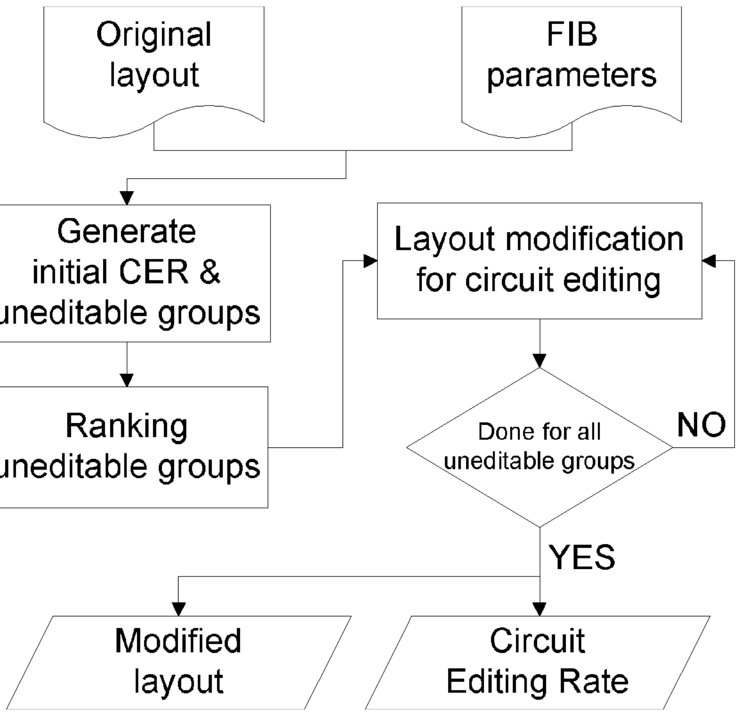

Original layout
FIB parameters
Generate initial CER & uneditable groups
Layout modification for circuit editing
Ranking uneditable groups
Done for all uneditable groups
NO
YES
Modified layout
Circuit Editing Rate

Figure 4.4: The flow of observable signals maximization

The flow of maximizing CER is mostly similar to the one of maximizing NOR as shown in Table 4.4. The difference is that we take a *group* as our unit instead of a net. There are two parts of the inputs: the one is the original layout, and the other is the set of parameters of FIB techniques. By these parameters, we can set the aspect ratio and the width of *baseline windows* according to different FIB techniques.

First, with the aspect ratio and the width of *baseline windows*, we analyze the original layout to obtain the initial *Circuit Editing Rate* and the set of uneditable groups. After determining the order of list of uneditable groups, we apply the modifications in sequence to all uneditable groups in the list. Finally, we return the modified layout and *Circuit Editing Rate*. To confirm the correctness of modified layout, we will also feed it back to commercial tools to check if it is still a legal layout.

4.4 Summary

In this chapter, we introduce *Circuit Editing Rate* and formulate the problem, maximizing FIB editable groups via layout modification. By a combination of the two operations, *Moving-up* and *Moving-down*, we can make some groups editable. We also present a strategy to determine the order of modification. Lastly, we propose our overall flow for maximization of FIB editable groups.

Chapter 5

Experimental Results

We implement our layout modification algorithm in C++ and obtain the experimental results at our workstation. The benchmarks we used are the large circuits from ISCAS'89 and ITC'99. The library is set for UMC 90nm and we constraint metal 6 to be the top layer for routing signals. We first synthesis these benchmarks with Synopsys Design Compiler, and the process of placement and routing are completed with Cadence SoC encounter. The SoC encounter outputs the original layout file in DEF format and timing information in SPEF format. Our flow is applied to the original layout file and outputs the modified layout file. Finally, we feedback the modified layout file to encounter for verifying the correctness.

5.1 Maximization for FIB Observable Nets

Layer	Original layout			Modified layout		
	Signal	Ob.	NOR(%)	Signal	Ob.	NOR(%)
M6	1992	1628	9.94	6210	5954	36.37
M5	2346	1640	10.43	4843	4813	29.40
M4	3083	1241	7.58	2480	2278	13.91
M3	7200	940	5.74	1766	997	6.09
M2	1578	126	0.76	917	126	0.76
M1	168	7	0.00	151	7	0.00
Total	16367	5582	34.10	16367	14175	86.61

Table 5.1: The experimental results for each layer of b17.

First, we set the width of baseline window $(WS)$to be 800nm and the aspect ratio of FIB holes to 10:1 and these parameters could be adjusted for different FIB milling procedures. As shown in Table 5.1, the first experiment describes the number of observable signals at each routing layer of original and modified layouts of b17. The first column is the routing layer. The second and forth columns are the number of signal first appearing in the layout. The third and sixth columns are the number of observable signals for each layer. For example, M6 has 1628 observable signals among 1992 signals, which are first appearing in the layout. Because we modify the layout, the numbers of signal are different at each layer between the original and modified layout. The forth and last column are NOR at each routing layer of original and modified layout. From table 5.1, the signals at lower routing layers (M1-M3) are hard to be observed. Our proposed flow targets on moving signals to upper layers for the frontside FIB techniques, and it greatly increases the observable signals and NOR.

Circuit	Signal	NOR (%)			
		Initial	Upper	Random	Ranking
s38417	8761	55.71	97.87	95.16	95.43
s38584	5132	62.37	97.37	94.91	95.05
s35932	4967	68.67	99.62	96.88	96.98
b17	16367	34.11	90.65	77.21	86.61
b20	6061	38.69	91.62	87.19	87.81
b21	6153	34.44	93.45	88.39	89.27
b22	9196	37.73	92.12	84.05	88.39

Table 5.2: NOR for each benchmark.

To evaluate the ranking methodology, we perform the proposed operations to make signals be observable in random selection. We compare NOR of random selection with our ranking method. Table 5.2 shows the experimental results for the benchmarks. The first column is the name of circuit. The second column is number of total signals. The third column is NOR of initial layout. The forth column is the upper bound of NOR for each layout, because it is impossible to observe all signals. The fifth column is NOR of random selection and we take the average of 20 experimental results. The last column is NOR with our ranking method. By comparing the results of random selection and our ranking method, NORs of ranking are all better than random selection. From the experiments, we observe that the observation rate can greatly be rised to90% in average. For s35932, its observation rate is 96.13%, which means almost all signals can be observed with our proposed flow.

Layer	WS=800nm			WS=1000nm		
	Signal	Ob.	NOR(%)	Signal	Ob.	NOR(%)
M6	6210	5954	36.37	7106	6601	40.33
M5	4843	4813	29.40	4266	4307	26.32
M4	2480	2278	13.91	1997	1769	10.81
M3	1766	997	6.09	1821	406	2.48
M2	917	126	0.76	1017	82	0.50
M1	151	7	0.00	160	8	0.00
Total	16367	14175	86.61	16367	13173	80.49

Table 5.3: The experimental results for each layer of b17 with different WS.

Circuit	Signal	NOR (%)			
		Initial	Upper	Random	Ranking
s38417	8761	49.37	96.51	92.37	93.06
s38584	5132	43.37	96.51	92.97	93.45
s35932	4967	44.67	99.37	95.92	96.13
b17	16367	19.45	85.65	68.13	80.49
b20	6061	23.59	87.25	81.57	83.07
b21	6153	20.35	89.55	83.39	85.02
b22	9196	23.24	88.03	77.58	83.63

Table 5.4: NOR of each benchmark with WS=1000nm.

We also set different window size to demonstrate the flexibility of our proposed flow. Table 5.3 and Table 5.4 demonstrate the experimental results with WS of 1000nm. Table 5.3 shows the observable signals at each layer under different WS. Even with different WS, our flow greatly increases the observable signals at upper layers. Compared with Table 5.2, the initial NORs are greatly dropping because we have to reserve more space for large FIB observation window. After applying our proposed flow, the upper bound of NOR for each benchmark is almost the same with small WS (800nm). NORs with our ranking method are also better than random selection. According to the experiments, with our flow, NOR of each benchmark is very close to the upper bound. Even with different window size, our proposed flow is still very efficient to improve NOR of each benchmark.

5.2 Maximization for Circuit Editing

Circuit	Group	Original layout		Modified layout	
		CER(%)	NOR(%)	CER(%)	NOR(%)
s38417	18454	11.72	55.71	24.53	95.16
s38584	14779	6.73	62.39	14.45	95.30
s35932	14691	7.97	68.67	15.41	97.50
b17	50174	2.51	31.34	8.85	77.31
b20	18426	3.80	38.69	11.33	87.56
b21	18716	2.77	34.44	10.71	88.82
b22	27840	2.97	35.25	9.65	84.68

Table 5.5: The CER and corresponding NOR for each benchmark.

Circuit	Group	WS=800nm		WS=1000nm	
		CER(%)	NOR(%)	CER(%)	NOR(%)
s38417	18454	24.53	95.16	22.67	92.10
s38584	14779	14.45	95.30	13.79	93.36
s35932	14691	15.41	97.50	14.27	96.42
b17	50174	8.85	77.31	7.80	67.88
b20	18426	11.33	87.56	10.26	81.80
b21	18716	10.71	88.82	9.94	83.55
b22	27840	9.65	84.68	8.75	77.68

Table 5.6: The CER and corresponding NOR with different WSs.

To present the improvement of increasing CER with our proposed algorithm, we demonstrate the experimental results in Table 5.5. The first column is the name of circuits. The second column is the number of group as we mentioned before. The third and fifth columns show the CER of original and modified layout with our algorithm. The forth and last columns are NORs of corresponding modified layout. Table 5.6 show the experimental results with different WS. From the experimental results, the CERs of modified layouts are almost two to three times compared to the initial layouts and NORs of modified layouts are almost equivalent as layout with only observable signals maximization. Thus, with our proposed algorithm, we can increase NORs and CERs at the same time.

5.3 Timing impact with layout modification

Applying modifications to design may affect the timing, and cause the design fail due to violations of setup time and hold time. Therefore, we should take timing impact into consideration.

The proposed flow affects the original design. Due to different coupling capacitance of layout modification, the timing of each path may differ from the original design. If we modify nets which belong to critical paths, the performance of the design may be dropped and become unpredictable.

We take the information of critical paths as the input for the proposed flow as shown in Figure 5.1. Once we have the information of timing-critical paths, we lock the nets which belong to critical paths and do not modify them while performing the proposed flow for maximizing NOR or CER.

For the experiments of reducing impact to critical paths, we extract the 50 critical paths of original layout, lock the signals on these paths, and apply the

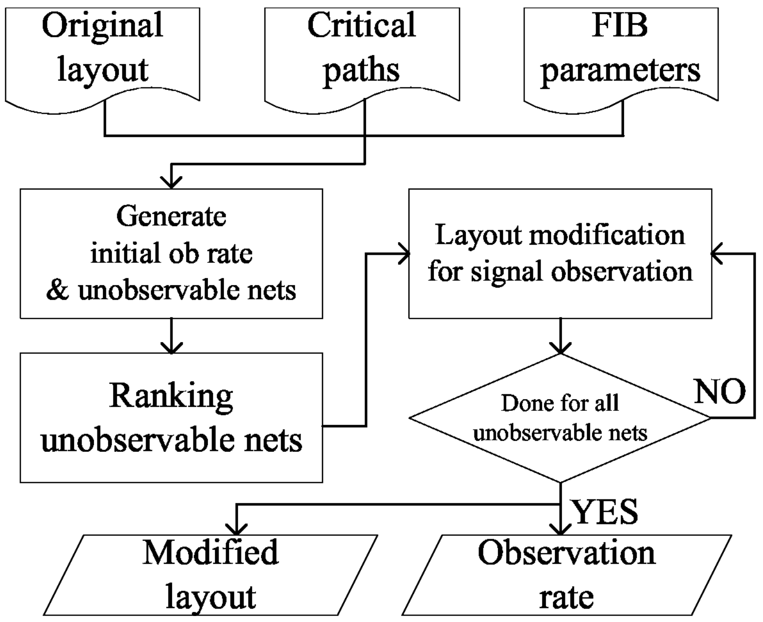

Original layout
Critical paths
FIB parameters
Generate initial ob rate & unobservable nets
Layout modification for signal observation
Ranking unobservable nets
Done for all unobservable nets
NO
YES
Modified layout
Observation rate

Figure 5.1: The flow of observable signals maximization with critical paths

proposed flow for increasing NOR and CER. Then, we evaluate the timing difference of these paths between layouts with and without information about critical paths. As shown in Table 5.7 and Table 5.8, the first column is the name of circuits. The second column is the timing of most critical paths in the original layout. The third and fifth columns are the timing of most critical path in the two kinds of layout. The time unit is ns. The forth and sixth columns are NORs (Table 5.7) and

Circuit	Ori.	Original flow		Flow with critical paths		
	Cri.	Cri.	NOR	Cri.(ns)	NOR (%)	ave. (%)
s38417	7.11	7.11	95.43	7.11	93.80	0.02
s38584	6.89	6.89	95.05	6.89	94.35	0.00
s35932	9.60	9.60	96.98	9.60	96.30	0.01
b17	6.60	6.59	86.61	6.59	86.61	0.15
b20	6.86	6.86	87.81	6.86	84.92	0.05
b21	5.90	5.89	89.27	5.89	86.23	0.04
b22	9.71	9.71	88.39	9.71	87.61	0.14

Table 5.7: The timing differences and NORs among original layout, modified layout without and with critical paths.

Circuit	Ori.	Original flow		Flow with critical paths		
	Cri.	Cri.	CER	Cri.(ns)	CER (%)	ave. (%)
s38417	7.11	7.11	24.53	7.11	23.91	0.02
s38584	6.89	6.89	14.45	6.89	14.23	0.01
s35932	9.60	9.60	15.41	9.60	15.22	0.02
b17	6.60	6.59	8.85	6.59	8.71	0.11
b20	6.86	6.86	11.33	6.86	10.85	0.06
b21	5.90	5.89	10.71	5.89	10.19	0.05
b22	9.71	9.71	9.65	9.71	9.42	0.40

Table 5.8: The timing difference and CERs among original layout, modified layout without and with critical paths.

CERS (Table 5.8) for each benchmarks. The last column is the average difference in percentage of the 50 paths.

Chapter 6

Conclusions

In this work, we propose the methodology to increase *Net Observation Rate* and *Circuit Editing Rate* after the layout of design is obtained. With minimal timing impact, the operations we proposed only change the routing layers of original design. The experimental results show that observable nets are more than 90% after applying our method. For more feasible locations for circuit editing, the experimental results also show much improvement with our method. In the future, we will try to develop further strategies for the unobservable nets.

Bibliography

[1] M. T. Abramo and L. L. Hahn, "The Application of Advanced Techniques for Complex Focused-Ion-Beam Device Modification", *Microelectronics Reliability*, Vol. 36, Issues 11-12, pp. 1775-1778, 1996.

[2] M. Abramovici, P. Bradley, K. Dwarakanath, P. Levin, G. Memmi, and D. Miller, "A Reconfigurable Design-for-Debug Infrastructure for SoCs", *Design Automation Conference*, pp. 7-12, 2006.

[3] E. Anis and N. Nicolico, "On Using Lossless Compression of Debug Data in Embedded Logic Analyis", *International Test Conference*, pp. 1-10, 2007.

[4] E. Anis and N. Nicolico, "Low Cost Debug Architecture using Lossy Compression for Silicon Debug", *Design Automation, and Test in Europe*, pp. 1-6, 2007.

[5] M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing*, Kluwer, Boston, 2000.

[6] FEI Company, "Focused Ion Beam Technology, Capabilites and Applications", http://www.fei.com.

[7] R. Goering, "Post-Silicon Debugging Worth a Second Look", *EETimes*, Feb. 05, 2007.

[8] J. Nonaka, "Design for Failure Analysis by using LVP Measurement Elements", *Semi Technology Symposium*, pp45-48, 2003.

[9] J. Nonaka, T. Ishiyama, and K. Shigeta, "Design for Failure Analysis inserting Replacement-type Observation Points for LVP", *International Testing Conference*, pp. 1-10, 2009.

[10] C. G. Talbot, M. Park, N. Richardson, P. Alto, and D. Masnaghetti, "IC Modification with Foucused Ion Beam System", *U.S. patent 5,140,164*, 1992.

[11] D. C. Shaver and B. W. Ward, "Integrated Circuit Diagnosis Using Foucused Ion Beams", *Journal of Vacuum Science & Technology B; Microelectronics and Nanometer Structures*, Vol. 4, Issuse 1, pp. 185-188, 1986.

[12] C. Shawn, C. C. Tsao, and T. R. Lundquist, "Measuring back-side voltage of an integrated circuit", *U.S. Patent 6,872,581 B2*, 2005.

[13] R. Schlangen, R. Leihkauf, U. Kerst, C. Boit, and B. Kruger, "Functional IC Analysis Through Chip Backside With Nano Scale Resolution - E-Beam Probing in FIB Trenches to STI Level", *International Symposium on the Physical and Failure Analysis of Integrated Circuits*, pp. 35-38, 2007.

[14] Y.-R. Wu, S.-Y. Kao, and S.-A. Hwang, "Minimizing ECO routing for FIB", *VLSI Design Automation and Test*, pp. 351-354, 2010.

[15] J.-S. Yang and Nur A. Touba, "Expanding Trace Buffer Observation Window for In-System Silicon Debug through Selective Capture", *VLSI Tset Symposium*, pp. 345-351, 2008.

[16] J.-S. Yang and Nur A. Touba, "Automated Selection of Signals to observe for Efficient Silicon Debug", *VLSI Tset Symposium*, pp. 79-84, 2009.

[17] W.-M. Yee, M. Paniicia, T. Eiles, and V. Rao, "Laser Voltage Probe (LVP): a Novel Optical Probing Technology for Flip-Chip Package Microprocessors", *Internation Symposium on the Physical and Failure Analysis of Integrated Circuits*, pp. 15-20, 1999.