國立交通大學

電子工程學系電子研究所碩士班

碩士論文

微調電路佈局以適合聚焦離子束技術

Repair Friendly Layout Generation for FIB Technology

研究生：陳昱安

指導教授：趙家佐博士

中華民國一〇二年九月

微調電路佈局以適合聚焦離子束技術

Repair Friendly Layout Generation for FIB Technology

研究生：陳昱安 Student：Yu-AnChen

指導教授：趙家佐 Advisor：Mango Chia-Tso Chao

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 ChiaoTung University
In partial Fulfillment of the Requirements
for the Degree of
Masterof Science
in
Electronics Engineering
August 2013
Hsinchu, Taiwan, Republic of China

中華民國一〇二年九月

微調電路佈局以適合聚焦離子束技術

學生：陳昱安　　指導教授：趙家佐

國立交通大學

電子工程學系　電子研究所

摘　要

這篇論文提出將針對以完成繞線的電路，以盡量在不影響原本電路設計的情況下，對繞線進行簡單操作，以提高聚焦離子束完成電路修補的比例，並確保最後最大延遲時間不會增加。 首先探討聚焦離子束在電路中的所受到的限制，定義出各個在調整成能執行聚焦離子束的電路修補時所造成的代價，根據這個代價去設計架構–

ReFL(Repair Friendly Layout Generation for FIB Technology)，以提升電路修補的效能，最後去比較不同作法所提升的比例，並且分析結果，討論未來可做的方式。

Repair Friendly Layout Generation for FIB Technology

Student: Yu-An Chen Advisor: Dr. Chia-Tso Chao

Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University

Abstract

This thesis proposes a methodology for the layout, which is completed routing and placement. There are some simple operations of adjustment for the metals without effecting the circuit seriously. These operations could make the rate of FIB repair rise, and keep the critical path delay. First, there are introduction of limitation of FIB. Then, we define the cost which spend for adjustment of FIB repairable cell. According the cost, we design a framework for rising the rate of FIB repairable cell called "ReFL"(Repair Friendly Layout Generation for FIB Technology). Finally, it compares other methods and ours by experimental result, and discuss the future work.

誌　　謝

首先感謝家中支持我讀碩士班。以及指導老師 趙家佐教授在各方面的提點。在校期間與實驗室同學間的討論也使我獲益良多，陳擴安學長教導我很多關於這研究主題需要的背景，盧敬和學弟提供關於 OpenAccess 的資料，這份研究可以寫出來甚至有再往前推進的可能，都離不開大家適時的幫助。

其他在校期間與我有過互動的同學們也是我能走到這一步的助力：同研究室的林政偉、穆思邦、楊皓宇、張玟翔、徐浩文、王易民、黃欽遠、黃召穎，以及蔡佳達、陳昭宏、廖偉翔，還有許多和我一起修課的同學們。

2013 年 9 月

目錄

表目錄

圖目錄

第 一 章　介 紹

第一章 第1節 錯誤診斷

在先進製程晶片的開發時，往往存在著一些缺陷，為了縮短在重新下線的時間和製作光罩的成本，因此，有許多錯誤測試和診斷(Testing and Diagnosis)技術廣泛運用在晶片的錯誤分析中。隨著製程技術的進步與電路設計越加複雜，要診斷出電路中錯誤的原因也更加困難，除了製程進步導致原先錯誤診斷儀器的解析度(Resolution)不敷使用外，另外常常在設計晶片初期並沒有考慮到在錯誤發生，以致這些錯誤本身不但難以模擬、分析，甚至重現錯誤都相當困難 [3]。就算許多電路都已經採用全掃描鏈設計，電路中大部分訊號還是無法觀察到的[2]。

為了減少在製程後段對錯誤測試和診斷的困難度，發展出很多方法，其中最為廣泛的為 DFT (design for testing) scan-based designs [3]和 DFD (design for debug) trace-buffer-based designs [4][5][6][7]。 在晶片設計階段時，會放入一些特定功能的原件、電路，能幫助晶片在測試階段時可以在輸出端量測一部分的訊號。若要觀察或修正電路中錯誤節點的訊號，會先使用 scan-based 的測試方式，縮小發生錯誤的區域，再使用實體探測技術(Physical probing technique)，量取各節點的波形訊號，判斷哪一個節點或者是哪一段線路出問題。

第一章 第2節 實體探測技術

常見的實體探測技術有電子束探測(E-Beam, Electron Beam Probing)[8]、雷射電壓探測(LVP, Laser Voltage Probing)[9] 、聚焦離子束探測(Force Ion Beam Probing，簡稱 FIB Probing) [10][11][12]。E-Beam 是利用發射電子束在 metal 上，並觀測撞擊出的二次電子而得到電位高低，因為反射的電子被容易被目標以外的 metal 影響，因此通常只能針對較上層的 metal 進行訊號量測。 LVP 則是將雷射由晶片下方打入閘極的空乏區，藉由反射回來雷射的強弱判斷高低電位，雷射電壓探測的解析度並不足以在 0.18um 以下的製程中準確的觀測訊號。

FIB Probing 能蝕刻氧化隔離層(inter-layer dielectric , ILD)，然後蝕刻到要量測的 metal 上，並填入金屬，在晶片上產生一個可量測的接觸點。 FIB 是上述中最直接的量測技術，可觀察晶片任何指定位置，且不需太多時間，但是限制的目標位置旁邊以及上方再一定得範圍中不能有其他的 metal。

FIB 的技術除了能觀測訊號外，還能對進行電路修補 (circuit repair)，FIB 能夠將原本的電路蝕刻、連接，這是其他探測技術所做不到的，利用 Spare Cells 來進行錯誤的驗證，以大幅降低驗證的時間與光罩的成本。

第一章 第3節 MFOB

MFOB (Framework for Maximizing FIB Observable Rate) [14] 完整提到如何去判斷一個節點能否做 FIB probing，並提出數種操作繞線的方式使得原本無法做 FIB Probing 的節點能被改善，而這些能被改善的節點將受到評估對電路的影響，量化這些影響後，給予各個節點一個 cost，依 cost 的排序開始調整，調整過程中不斷去更新 cost 和排序，最後能明顯增加節點做 FIB probing 的比例。

第一章 第4節 我們的做法

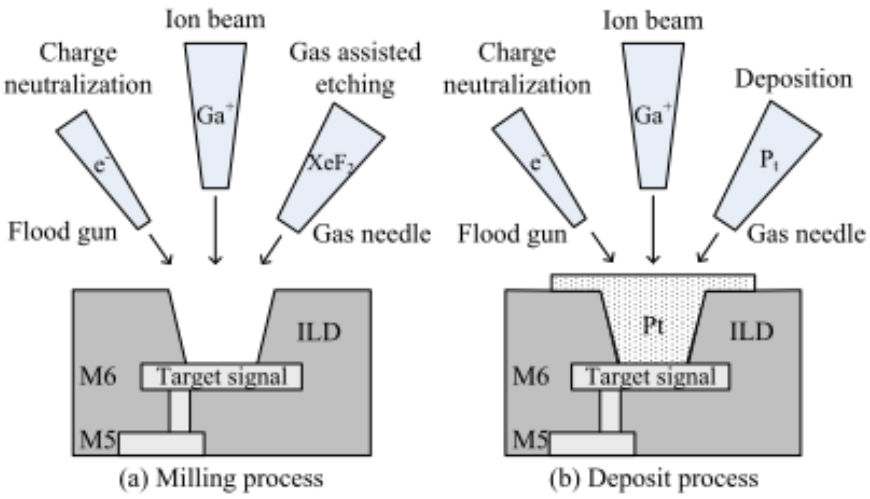
在這份研究中，我們延伸之前的研究，對 SOC Encounter 合成的 Def 檔進行處理，設計 framework 將原先電路中的 cell 能做 FIB repair 的比例改善，和原先 FIB probing 調整相比，需要定義出 cell 的能否做 FIB repair 的條件，和改善各個 cell 間繞線的 cost，並考慮 repair 後 cell 的延遲時間 (delay time)，估計 FIB repair 增加的負載，最後決定出調整 cell 的順序，使 FIB repair 的成功比例最大化。

在接下來的章節，會先簡介 FIB 的原理、限制，和 MFOB 中我們沿用的一些技巧。 接著會描述我們所設計的 framework、cell cost 的定義、調整 spare cell 時(spare cell)的額外繞線，然後是實驗的內容以及結果，最後針對實驗結果的分析，以及相關題目未來可以改進的地方。

第 二 章　背 景

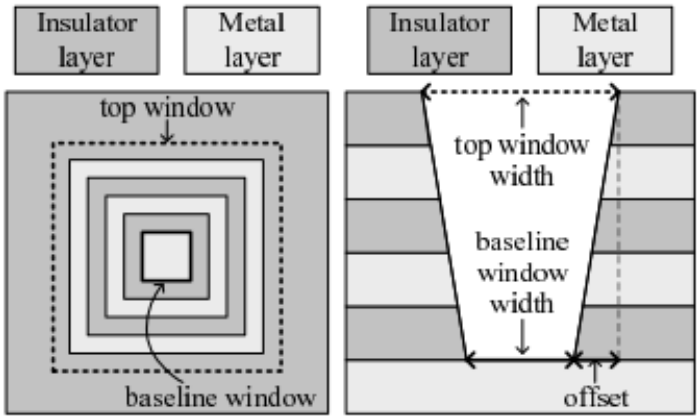
第二章 第1節 FIB 原理

Focused Ion Beam (FIB) 是一種發射帶電粒子的技術，能對晶片執行兩種動作:蝕刻(etching)和蒸鍍(deposit)。進行蝕刻時(圖表 1(a))，會在電場中對 Ga+離子加速打向的晶片的氧化隔離層，當離子束打入後，會使得氧化隔離層上的原子被離子化，或打斷試樣基材的化學鍵，最後在目標 metal 上蝕刻出一個洞，也能加入特定的氣體(如 XeF_2)以加速蝕刻的反應。蒸鍍(圖表 1(b))，通常是為了避免外來物質入侵或將訊號引至隔離層外，因此在蝕刻的洞中補上絕緣層或金屬層，要填上絕緣層時多半是使用 Thermoplastic Elastomers 加上 Siloxane 及 O_2 以鍍上 SiO_2，要填入鎢會使用 $W(CO)_6$，要填入鉑則會使用 $C_9H_{16}Pt$。FIB 最直接的功能是用探針去觀察經過蝕刻和蒸鍍處理的 metal，藉此得到電路中某一節點的訊號，以幫助電路進行錯誤診斷。



圖表 1 FIB 操作示意圖 (a)蝕刻 (b)蒸鍍

圖 2(a)為 FIB 蝕刻的上方示意圖，其中淺色的金屬層，深色的為隔離層，其中 top window 為蝕刻過後最上層所挖掉的面積，而 baseline window 為目標 metal 所需要的留下面積，從 baseline window 到 top window 會根據所挖的深度由小到大，而 baseline window 為一個定值(1000nm X 1000nm)，因此，當目標 metal 所在的 layer 越底層，最上方的 top window 的面積也會越大。圖 2(b)為 FIB 蝕刻後的剖面圖。本身蝕刻後的洞會有一個規律的斜率，大約是 1/10，能預測對某一 metal 做 FIB 時，所須要留下沒有 metal 的空間。以圖 2 目標是在從上而下的第三層金屬層，在 UMC 90nm 的製程下，每一層的高度為 250nm，baseline window width 的為 1000nm，則 top window 為 1250nm X 1250nm。

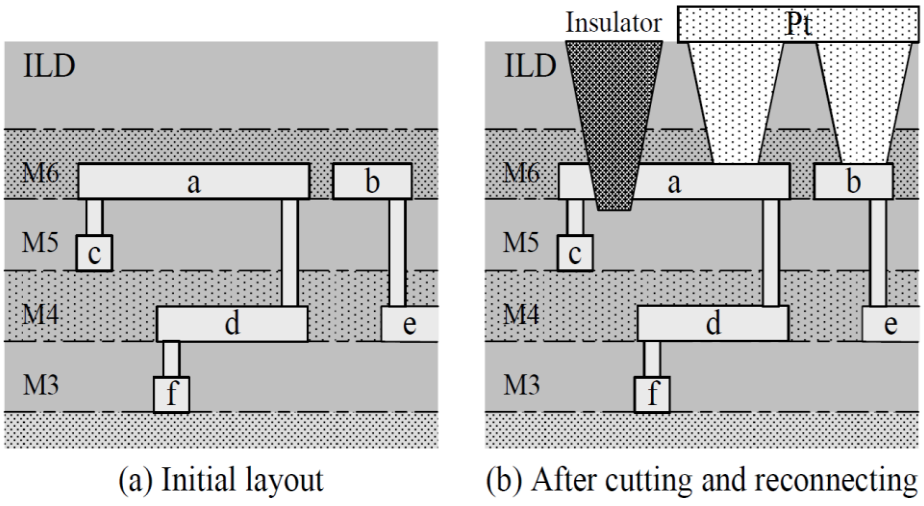


圖表 2 (a)蝕刻後的上方示意圖(b)蝕刻後的剖面圖。

第二章 第2節 FIB 電路修補

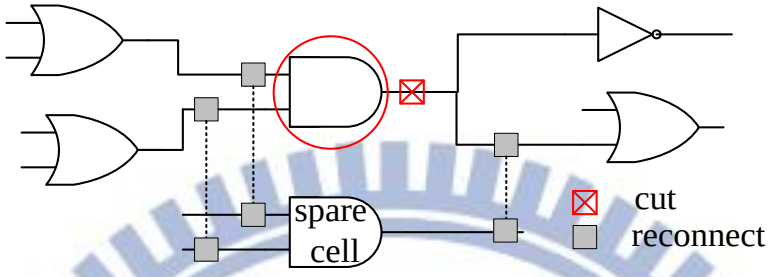
FIB 在半導體業中最常用在電路修補(circuit repair)或叫電路編輯(circuit edit)，特別是將某些有問題的接線切斷、接上某幾個該要導通的訊號，以免去等待修改 layout 後重新經過 foundry 製作所需要的時間，並可以馬上送回驗証部門進行功能上的驗証。FIB 所使用的 Ga^+在射進試樣後可以植入試樣中，把試樣表層非晶態化（amorphous），利用這種性質便可在微米尺度對試樣研磨或修改。此外還有在 FIB 過程中加入氣態金屬將金屬原子鍍至試樣表面，來連接 spare cell 至原先的電路中以修改電路。

圖表 3 為電路修補的示意圖。目標是將 b 上的訊號引到 a 的節點上，先蝕刻 a metal 並將絕緣物質填入以截斷來至 c 的訊號，接著將 a 及 b 做 FIB 蝕刻並蒸鍍 Pt，以圖 3 這個範例來說至少需要三個能做 FIB 的位置。



圖表 3 電路修補範例(a)原始電路(b)修補後電路。

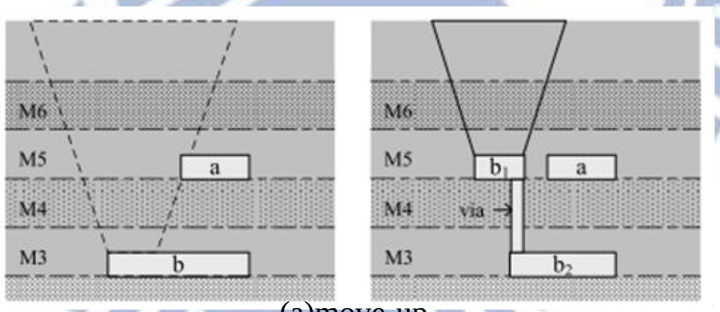
實作上，會將可能損壞的原件，利用 FIB 電路修補的方式，將其替換成空白原件(圖表四)，以幫助電路的錯誤診斷。範例中，被圈選的 cell 是要被替換目標，首先會決定出要替換的 spare cell，通常會選擇最近並相同的空白原件，接著去截斷(cut)目標 cell 的輸出，且截斷的位置必須在 stem 上，以確保訊號不會影響到之後修補的電路，連結(reconnect)的位置則沒有這樣的限制。



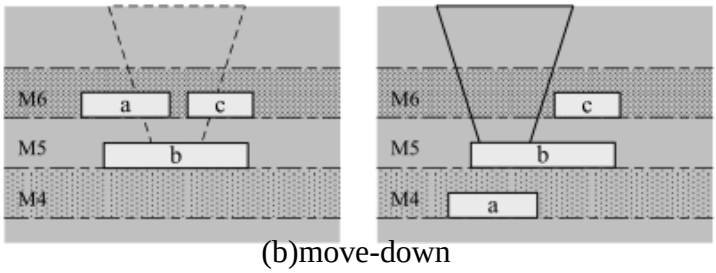
圖表 4 FIB 電路修補範例。

第二章 第3節 調整電路的方法

接下來會介紹在 MFOB 中使用的調整電路的方法，這些方法只會對部分 metal 進行上下移動，並不影響總 metal 的長度，以對，主要分成三個方式，(1)move-up、(2)move-down、(3)swapping。



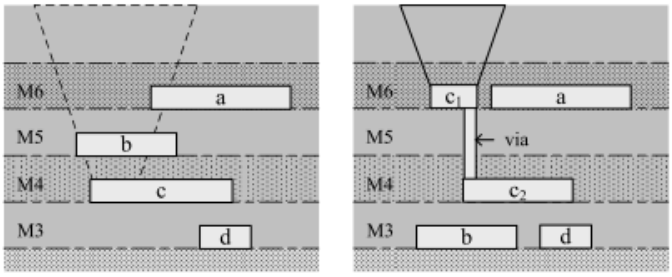
(a)move-up



(b)move-down

圖表 5 電路調整範例(a)move-up(b)move-down。

圖表 5(a)中 b 為需要做 FIB 的 metal，但蝕刻的範圍會有 a，因此將 b metal 的一部分往上移，使 b_1 能做 FIB。圖表 5(b)中完全被 a、c 遮住，若將 a 往下移則 b 的一部分即可行 FIB。

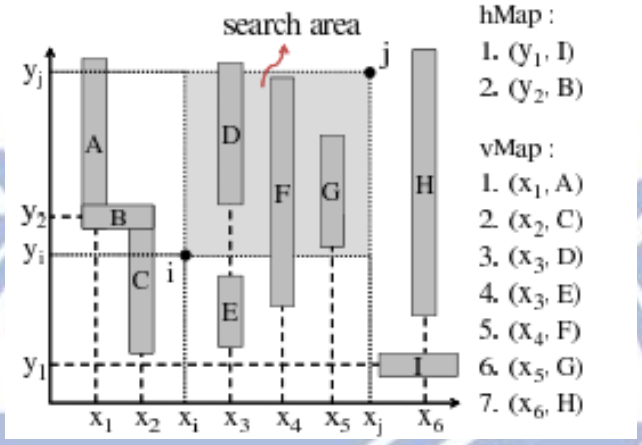


圖表 6 電路調整範例 swapping

圖表 6 swapping 調整的範例，swapping 會同時執行 move-up 和 move-down，c 為要觀察的目標，單純使用前兩種方都無法使 c 能被觀察，因此先將下方有空間的 b 先做 move-down，在對 c 使用 move-up，使得 c_1 最後得以被觀察到。這些方法在執行後，被調整成可觀察的 metal 將會標記上 fixed，被標記 fixed 的 metal 之後將不能再被移動和遮擋，因此，當調整了某一段 metal 能可觀察，必定會造成另一段的 metal 不能被觀察到，所以會優先使用影響的 metal 數會較少的方法－move-up 和 move-down，最後才考慮用 swapping 去調整電路。

第二章 第4節 資料結構

MFOB 中常常會需要從晶片上找出特定區域中的 metal 進行分析，以判斷如何去調整電路，或能否被觀察到，建立能快速找出這些 metal 的資料結構是很重要的。因此 MFOB 中使用的存取 metal 的資料結構為 STL 的"map"，建立 *hMap* 和 *vMap* 分別存取橫方向(horizontal)和縱方向(vertical)的 metal。其中 *hMap* 的 key 為 y 座標，*vMap* 的 key 為 x 座標。



圖表 7 尋找區域中 metal 範例。

圖表 7 為一尋找特定區域中 metal 範例。圖中淺灰色區塊為我們所指定的區域，其中 *hMap* 、*vMap* 是分開處理的，以 *vMap* 為例，搜尋區域在 x 軸的範圍是 x_i 到 x_j，在這範圍內根據 *vMap* 的 key 可以找出 D、E、F、G，之後再針對這些 metal 的 y 比對，則會發現 E 不在區域內，*hMap* 將 key 換成 y 也是同理。

第 三 章　問 題 構 成 及 實 作

第三章 第1節 問題構成

FIB 比較常用在 0.18um 以上的設計中，主要是因為 FIB 挖洞時所需預留的空間，在較小的製程中難以找到，在前人的論文中(MFOB)成功的使 FIB probing 在 90nm 和 65nm 變得可行，我們在這篇論文中會以改良的方法去提升 FIB repair 得比例，關於 repair 的定義在第二章第二節介紹了，要解決的問題如下:

1. spare cell 難以觀察:

   spare cell 的 pin 都落在 layout 的最底層，上面難以預留空間給 FIB，並且在做 FIB repair 時，需針對所有 pin 做 FIB，因 90nm 的 cell 往往各 pin 的距離沒有超過 baseline window width，以至於 FIB 重疊而失敗。

2. FIB 的時序分析:

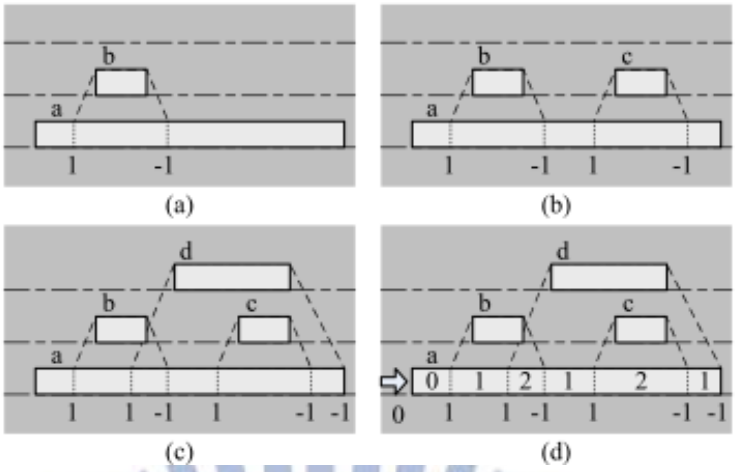
   FIB repair 執行時，會因為 FIB 的額外接線所產生的負載，導致延遲時間過長，若要能成功 repair 則必須考慮增加的延遲時間。

3. 調整的順序:

   spare cell 調整時會導致周遭的 spare cell 難以調整，且每個一般的 cell 在調整時，也會導致一般的 cell 和 spare cell 無法做 FIB，因此每個 cell 之間都存在著一種競爭關係，「如何使 FIB repairable 的比例做大的提升」，這是這篇最大的問題。

第三章 第2節 定義調整 cell 的代價

在 MFOB 中會給予每個 metal 一個 cost，而 cost 得定義為調整 metal 為可觀察的，至少會影響到多少其他的 metal。在這篇論文中也會使用類似的方法去定義每個正常 cell，給各個 cell 一個 cost，定義為調整這些 cell 至可修復的(repairable)至少需要影響多少 metal。



圖表 8 計算單一 metal cost 的範例。

首先，針對每個 metal 的 cost 做計算有一個演算法，圖表 8 範例為 layout 的剖面圖，a 為我們想要計算 cost 的 metal，根據先前的資料結構，我們找出 b、c、d 三個會影響到 a 做 FIB 的 metal，圖表 8(a)為開始，先將 b 放入計算，依照 FIB 蝕刻時斜率 1/10，從 b 畫出斜線到 a 上，分別記錄+1 和-1 的位置，而兩點之間的範圍，就是 a 做 FIB 時，被 b 所遮蓋到的地方，同理，c、d 也是。接著再從左至右將這些點的直疊加，並將質填入對應的區間，且調整區間至少滿足 baseline window width，這就是每個區間的 cost，最後 metal 的 cost 則是區間中最小的 cost。

$$Rank_{Net}(N) = \min_S rank(m_N) \ , m_N \in M_N \qquad (1)$$

算式(1)為計算節點 N 的 cost，中 M_N 為節點 N 所包含的 metal，這算式代表每個節點的 cost 由所擁有的 metal 最小的 cost。

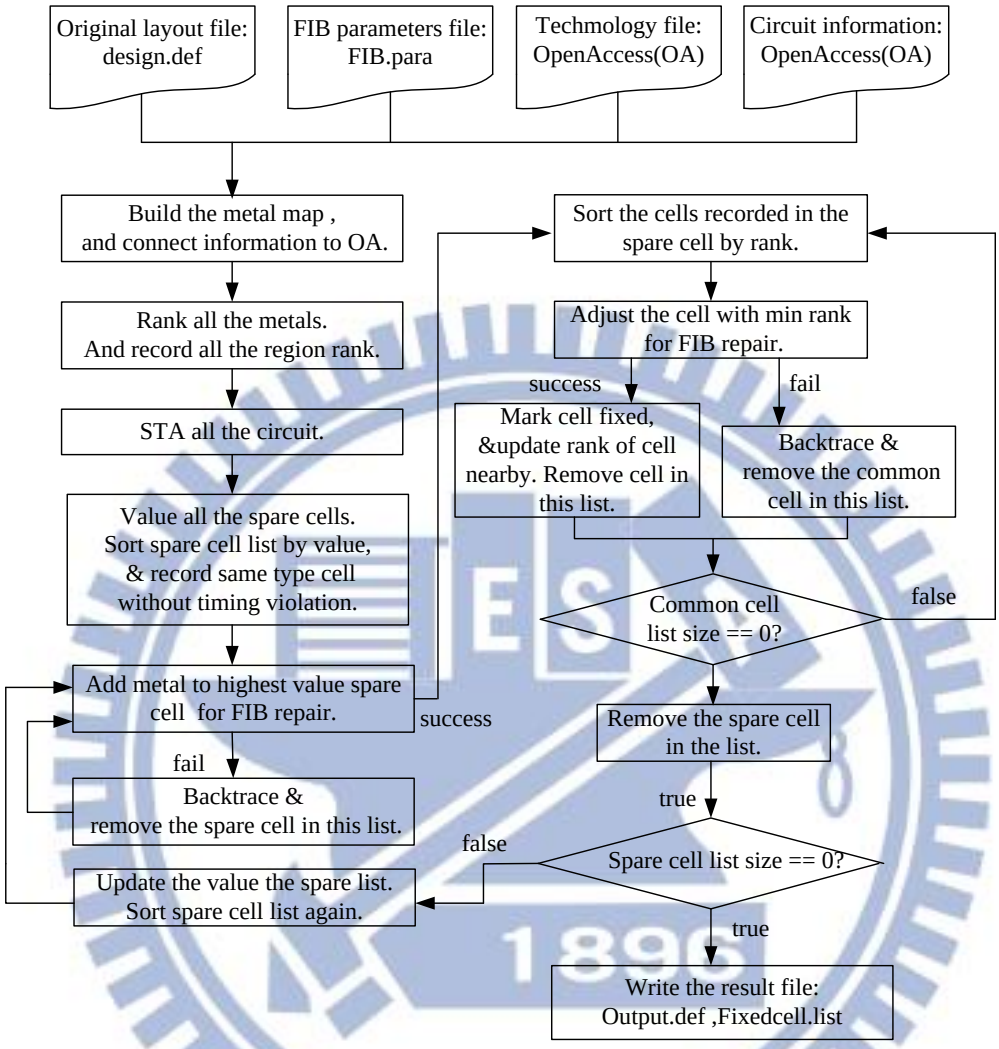
$$Rank_{stem}(C) = \min_S \ rank(m_{N_o}) \ , m_{N_o} \in M_{N_o,stem} \ , N_o = \text{output net of C} \qquad (2)$$

算式(2)中 C 為目標 cell，N_o 為 C 在輸出端的節點，$M_{N_o,stem}$ 為 cell C 在節點 N_o 的為 stem 的 metal，算式(2)代表的是如果將 cell C 調整成可修補的，調整電路在 stem 上的 metal 最小的 cost。

$$Rank_{cell}(C) = \sum_{k=1}^{Pin.size} Rank_{Net}(N_{pin}[k]) + Rank_{stem}(C) \qquad (3)$$

算式(3)綜合算式(1)及算式(2)，可以算出 cell C 在調整電路後成為可修復的，過程中至少會影響多少 metal 無法被觀察。N_{pin} 為 cell C 在輸出和輸入端的節點。這裡所計算的 cost 都是比較樂觀的數值，定義為「至少」影響多少 metal 無法被觀察，除了 cost = 0 可以保證能被 FIB 修補，其餘的 cell 和節點再調整皆會受到所在位置的 metal 密度和標記 fixed 的 metal 影響，導致最後移動的 metal 超出預期，甚至無法調正成可修補的狀態。

第三章 第3節 framework



圖表 9 電路調整流程圖。

第三章 3-1節 檔案讀取

首先會讀進 file 檔，除了原本在 MFOB 中會讀取的 def 檔和 FIB.para，一樣讀進我們得資料結構，但是為了加速程式的開發和其他程式的相容性，這邊使用了 OpenAccess(OA)的資料結構來存取 tech file 和 circuit information 的檔案，OA 為 Cadence 所發表，屬於一種公開的資料結構，並定義 VLSI 相關參數及含式命名，以下會介紹我們讀取的 file:

- design.def: 為描述實體 layout 的 file 包含 metal 及 cell 位置的資訊。
- FIB.para: 描述 FIB 的參數，含 baseline window width、蝕刻後的斜率。

接下來的檔案是由 OpenAccess 所存取的

- tech.lef: 描述實體 celllayout 的檔案，包含 pin 角的座標、via...等。
- netlist.v: 描述 gate-level 電路。
- tech.lib: 描述各種 celltiming 的資料。
- circuit.spef: 紀錄各節點間的負載
- circuit.sdf: 紀錄各 cell 的延遲時間。

第三章 3-2節 前置處理及時序分析

當檔案讀取完成後，會先建立 metal map，接著我們會計算全部的 metal 的 cost，並且將各個區間的 cost 存下，之後會對所有的 cell 進行 Static timing analysis，將 critical path delay 設成 cycle time，之後將 cell 各自的 time slack 存下。

有了 slack 的資料後，會開始決定 spare cell 的 expectation，expectation 為各自 spare cell 所能修復的 cell 以及其中的 cost 所決定，在下一個章節，將會有更詳細的 expectation 的說明。每個 spare cell 可以找到相對應 cell-cell type 相同、修復後 timing 不會違反限制，將這些 cell 放入這些 spare cell 對應的 list 中。

在 FIB 後 timing 的分析必須考慮到 layout 中實體的狀態增加的負載，關於 FIB repair 後阻值計算，使用的是以下公式(4)

$$C = \text{area} \times \text{CPERSQDIST} + \text{length} \times \text{EDGECAP} \quad (4)$$

CPERSQDIST 代表的是上下平行面積之間的產生的電容計算參數，而 EDGECAP 則為相同 layer 中 metal 側面所產生的電容計算參數，這兩個參數值來至相對應的 lef 檔，並且在不同的 layer 值也會不同。在這裡先評估 FIB 蒸鍍 Pt 的洞所產生電容，baseline window 為 1000*1000 為公式中的 area，每層的寬度受蝕刻的斜率決定，再將各層產生的電容疊加起來。而在晶片外的拉線，寬度為 1000nm，這裡外部的拉線只考慮直角拉線的最短距離，也就是兩個相連點 x 和 y 的差值相加。

整體公式如下(5)(6)(7):

$$C_{hole} = \text{area} \times \text{CPERSQDIST}_{layer} + \sum_{k=layer}^{Top\_layer} \text{length}_k \times \text{EDGECAP}_k \quad (5)$$

$$C_{line} = (1000\text{nm} \times \text{distance}) \times \text{CPERSQDIST}_{top\_layer+1} \quad (6)$$

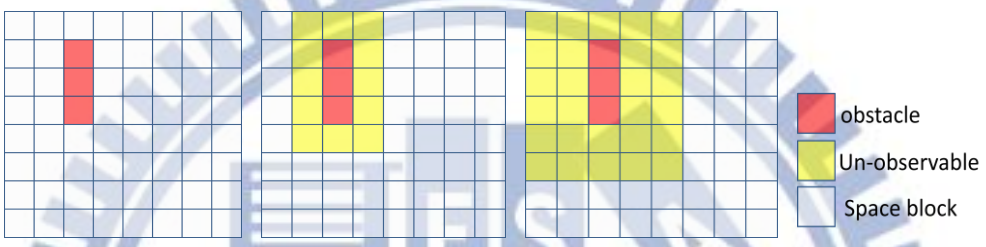
$$C_{total} = C_{hole1} + C_{line} + C_{hole2} \quad (7)$$

將算出來的負載代入 STA 計算，時序分析的 table 已存在 OpenAccess 的資料結構中，就能判斷有無 timing violation 的產生。

第三章 3-3節 新增 spare cell 的 metal

在主要的迴圈中，我們會先選擇最高 expectation 的 spare cell 來進行新增 metal 的動作，因為 90nm 的製程中，spare cell 被其他的 metal 遮住上方且面積較小無法直接做修補的動作，因此在這裡會先新增一些 metal 以幫助做 FIB 觀測，在這裡會使用到一些 detail route 的方式。

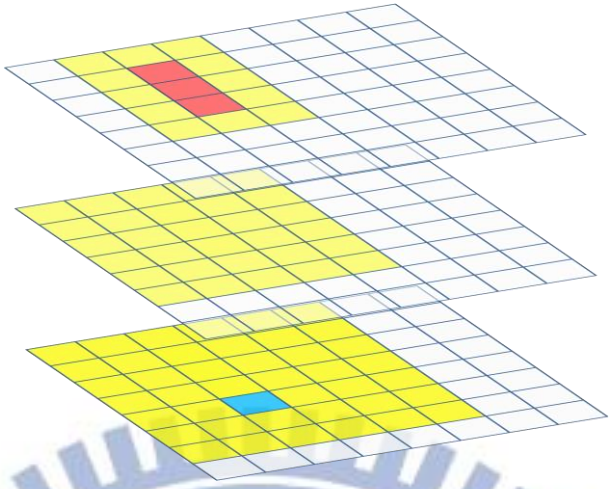
在這裡使用的是 rip-up-and-reroute (RNR)，在使用 RNR 的繞線前必須先建立一個三維矩陣的資料結構，並將其中的障礙物(obstacle)和終點位置決定，並且障礙物還會產生導致不能被觀察的區域，圖表 10 為一個三維矩陣的範例。



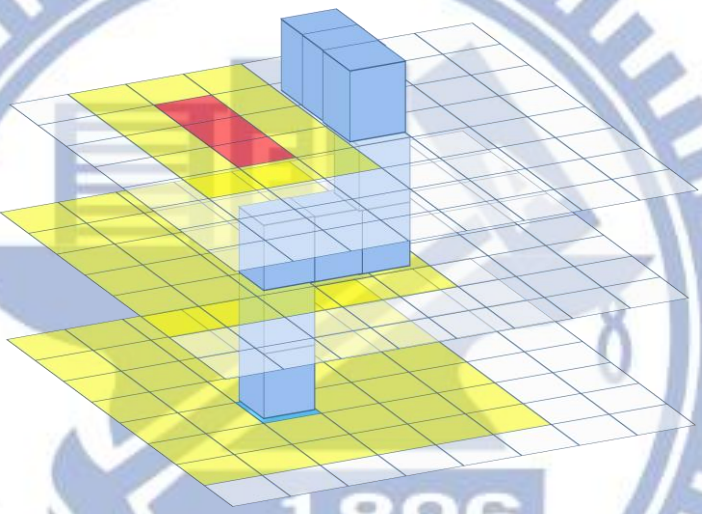
(a)layer4 (b)layer4 (c)layer3

圖表 10 障礙物在矩陣上範例(a)layer4 原始的矩陣、(b)修飾後 layer4、(c)修飾後的 layer3

在圖表 10 中(a)描述最原始在 layer4 上有一段 metal，在這裡標記為障礙物(紅色)，而(b)裡的黃色區塊為因障礙物而無法做 FIB 區域，而(c)為較下層的 layer，黃色的區間在越下層會增加越多。



(a)



(b)

圖表 11 改良後 RNR 範例(a)最原始的電路(b)新增 metal 後的電路。

在圖表 11 為一個專門對尋找 FIB 觀察點的 RNR，圖(a)中藍色的格子為起始的點，也就 spare cell 需拉出 metal 的 pin 位置，如同原本的 RNR，從起始點開始往外找，直到找到空白的位置時，則開始計數，當在同一層連續走到空白方塊滿足 baseline window width 時(在範例中是三個)，範例中為了清楚解釋演算法，格子之間尺寸與實作不同，最後則回傳 true 和這些新增的 metal，並準備執行接下來的一般 cell 的調整。若在到達步數限制和 RNR 沒有能選擇的格子，則回傳 false，並將這 spare cell 移出調整的 list。

spare cell 得對上面需做 FIB repair 的 pin 都執行一樣的操作，當調整完一個 spare cell 後，會使得附近的 metal 非常壅擠，且 spare cell 常常會放在比鄰的位置，導致周遭的 spare cell 將難以做修復。

第三章 3-4節 spare cell 的 expectation 和 cost

接續上一段最後所提的，調整一個 spare cell 將會使附近的 spare cell 無法被調整，因此需要一個方法去決定那些 spare cell 先被調整，才能使得有最多的 cell 能被 repair，這裡提出了一個判斷的依據「expectation」和「cost」，當 expectation 越高，我們則會優先處理，expectation 在這裡代表的 spare cell 約能修復多少個 cell。

$$E[spr] = \sum_{i=0}^{C_R.size} P(Cost(C_R[i])) \qquad (8)$$

Circuit	Probability			
	Cost 1	Cost 2	Cost 3	Cost 4
s35932	0.612	0.571	0.606	0.612
s38417	0.473	0.439	0.542	0.473
s38584	0.156	0.225	0.224	0.157
avg	0.597	0.484	0.202	0.119

公式[8]可以看做是將每個 cell 在不同 cost 下可以 repair 機率總和，C_R 為 spare cell C 可修復的一般 cell，當 cost 為 0 時則保證可以 repair，若 cost 為 1 或 2 時大約有一半的機率能被調整成功，這數值只是大約的機率，為的是分辨出能大量修復其他 cell 和 cell type 使用率特別高的 cell 能優先處理，當 expectation 差距不足 1 時，則繼續比較 cost，cost 為定義為 spare cell 平均每調整一個 cell 至 repairable 至少會影響到多少 metal，用之前計算的 cost 公式(3)帶入。

$$AvgCost(spr) = \frac{\sum_{k=0}^{C_R.size} Cost_{cell}(C_R[k])}{C_R.size} \qquad (9)$$

這裡C_R這各集合在計算中將會把 cost 大於 5 的移除，因為在某些難以修復的 cell 會存在極大的 cost 值，移除後可避免極端大的 cost 值影響 cost 的計算，算出能較精準比較出值得修理的 cell。另外在 expectation 和 cost 的計算中，皆會排除以 fixed 過的 cell，以確保找出能修最多的 spare cell。

第三章 3-5節 主程式迴圈

在主程式中每個 spare cell 調整，彼此之間皆存在著競爭關係，排序後由最第一的 spare cell 開始做調整，如果無法調整至能做 repair 則它從序列中移出，繼續對序列中排序最高的 spare cell 調整。當調整成功時，則會進入裡面的迴圈，對目前這個 spare cell 所對應的一般 cell 去排序，從 cost 最低的開始去調整，原

因與之前 spare cell 類似，因為當調整一個 cell 會導致其他的 cell 無法調整，優先出處理對其他 metal 影響較少 cell 先調整，使得修復比率能最大化。

在對一般 cell 調整時，先找輸出端的 stem 去調整，在 stem 上有一段 metal 能做 FIB 後，先記錄起來，再繼續找到另一段能做 FIB 位置，如此，在輸出端上能做截斷(cut)和連接(reconnect)的點都有了，接者，對輸入端的節點做調整，在 MFOB 中對調整節點的選擇較為複雜，原因是 MFOB 中考慮的為整個電路的節點選擇；在這裡的選擇，通常只有幾個節點能選擇先後順序，所以指考慮 cost 最低的先做調整，只要有一個節點調整失敗時，這個 cell 就算是無法調整，此時回傳 fail，並還原已做調整的電路，將這個 cell 移出序列，繼續執行下一個。當所要求的節點都能做 FIB 時，則調整成功回傳 true，將這個 cell 設為 fixed 移出序列，重複之前的步驟對下個一般 cell。當序列的 size 為零時，則結束迴圈。

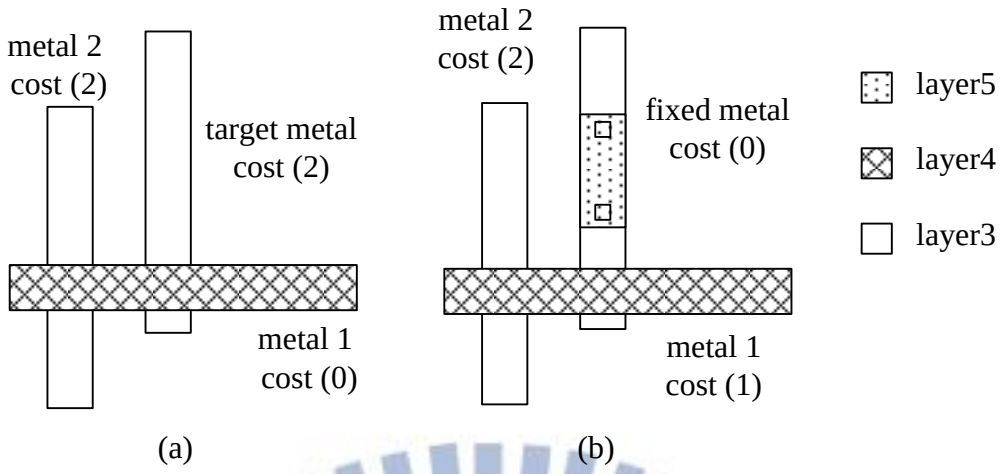
當 spare cell 調整完對應的一般 cell 時，需要去對電路中的各種資訊去更新，但是如果對整個電路重新分析，cost 全部的 metal 以及 cell，這樣執行程式的時間將會過長，因此需要去記錄在各種情況下，可能導致的各種 cost 改變或 spare cell 的 expectation 和 cost 的改變，這些將會在之後的章節繼續討論。

整個迴圈結束時，spare cell 的序列將會是空的，也就是說就算某部分的 cell 無法修復都其他的 cell，仍將會完成它的調整，以增加在真正實作上做 FIB repair 的可能性，之後將會把修改後的，以 def 檔的形式吐出，和一個紀錄可做 repair 的 cell 序列。

第三章 第4節 區域性的數值更新

在之前的章節提到，每當我們調整完一部分的電路，或是將某個 cell 或 metal 設為 fixed 時，都會影響一些沒做調整的電路，這些影響會越來越嚴重，如果程式中都沒去處理，最後，程式在排序時所用的 cost、expectation、cost 都將會沒有意義。因此，程式中會需要在電路調整的後，去更新上述所提到的那些數據，以確保每次調整的 cell、節點都是符合我們所預期的。

由於不可能針對全電路去重新計算，我們必須在調整電路時，細心地將可以能影響到的 metal 記錄下來，這裡先針對 metal 調整時所作的方式說明。



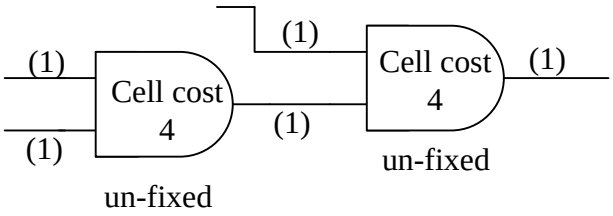
圖表 12 調整電路後更新 cost 範例(a)原先電路(b)調整後電路。

圖表 12 為一個更新 metal 的 cost 範例。在之前提到調整 metal，再調整 metal 前會以目標 metal 為中心會定出一個搜尋範圍範圍內，而在這範圍內的 metal 才有機會影響到 target metal 的執行 FIB，相對的，當目標 metal 調整時，會受到影響的也只有這範圍內的 metal。在圖表 12 中找到的是 metal 1 和 metal 2，當找出這些 metal 後，其中會影響目標的 metal，為 layer 數相同或大於目標 metal 的 layer；則相對的調整後的 metal(fixed)，會影響到的其他 metal 則是那些 layer 比它小的 metal，因此能推斷出會變動的 cost 值的 metal 的條件為

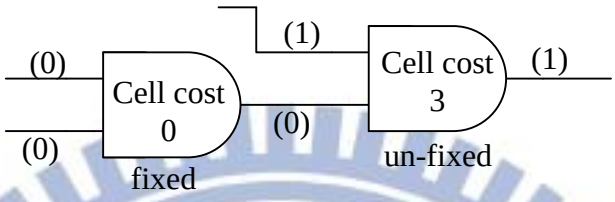
1. metal 位於搜尋區間內。
2. metal layer > original target metal
3. metal layer < fixed metal layer

在圖表 12 中，最後找到的為 metal1，而對 metal 1 的 cost 進行更新，這裡將會記錄下這些找到的 metal，並對這些 metal、所屬的節點、節點所屬的 cell，將數值重新計算。

除此之外，還有節點所連接的 cell 也都需要更新 cost，如下圖範例(圖表 13)：



(a)



(b)

圖表 13 調整後 cell 更新 cost 範例(a)調整前(調整後)

圖表 13(a)為調整前的 cost 數值(公式(3))，當 cell 調整完後各個 metal 後，前後連的 cell 也須更新自己的 cost 如圖 13(b)所示。

第 四 章　實 驗 數 據

在這一份研究中所用到的測試電路分別是 iscas85 的 s35932、s38417、s38584 以及 ITC99 的 b17。這裡我們平均的去放電路中用到的 cell type，用的是 SOC Encounter 提供的指令 *createSpareModule* 和 *placeSpareModule*，執行這些指令插入 spare cell 是在 placement 之後，表格 1 為一些電路的基本資訊。

表格 1 電路資訊

Circuit	critical path(ns)	Density(%)	total cells	spare cells
s35932	2.075	85.52%	8233	1335
s38417	1.304	88.53%	8141	511
s38584	1.596	87.07%	9572	504
b17	5.050	88.15%	15623	676

大部分的電路沒插入 spare cell 前的原本 density 都在 80%以上，spare cell 放放後，大約增加 3%~5%，layer 的總數為 6 層。然後表格二為不同策略下最後 repairable 的比率，

表格 2 修改後電路 repairable 比率

Circuit	FIB repair rate(%)				
	Spr only	Random	AvgCost	Expect	ReFL
s35932	1.7%	13.0%	16.9%	19.3%	22.0%
s38417	3.1%	9.5%	10.5%	11.5%	16.7%
s38584	1.2%	6.7%	12.6%	12.8%	15.3%
b17	1.1%	4.2%	5.4%	5.7%	6.1%
avg	1.8%	8.5%	10.6%	12.3%	15.0%

90nm 的製程以下，在電路中完全沒調整的情況下，FIB repair 能成功的機率近乎沒有，原因在於 cell 本身就太小，導致就算很多 spare 上面沒有 metal，各自 pin 的上所挖 FIB 的範圍會銑重疊到，而不符合限制；或是 pin 的長度不滿足 FIB 最短的限制:1000nm。表格二中的 initial 的方式為 spare cell 經過調整，一般 cell 則無，最後得到的比例；random 則是隨機排序 spare cell 的序列，然後依序調整 spare cell，及其對應的一般 cell；Ours，則是 ReFL 的做法；這裡可以看出隨便排序的 spare cell 所做出來，和我們的 framework 做出來的比較，進步的比例大

約可以到兩倍。

表格二中的"non STA"則是在原本的 framework 中，不考慮 FIB repair 後電路所增加的延遲而得到的結果，這樣的條件下執行 ReFL 的結果，會出現只要有一個 cell type 的 spare cell 能調整成功，其他的一般 cell 就能跟著調整，因 FIB repair 會增加很重的負載，在不考慮 timing 的問題下，會大大提高 repairable 的比例。最後"mix"為在 ReFL 的正常 flow 做完後，將剩下 timing 不過的 cell 要做完調整的比例，也就是說 mix 裡符合 timing 的比例為 ReFL 那一欄所示，而剩下的比例 cell 在做 FIB repair 的時候會影響到 critical path delay 的增加，如果錯誤診斷的模式下有較大的 timing slack 則可對者些 cell 做 FIB repair。

第 五 章　結 論

在本研究中，針對電路的 def 檔中的 metal 進行微調，這些電路可沿用原先 routing 和 placement 所用的演算法，微調後的電路能保持的原本地 critical path delay，並使 FIB repair 的比例增加。

FIB repair 在 90nm 以下的電路中難以使用，需要針對在最底層的 spare cell 再增長 metal 的動作，使用改良後 RNR 的 detail routing 的演算法。另外我們使用在 MFOB 中調整電路的方法，這些方法可以有效的增加 FIB observe 的比例，且 metal 的長度不會增加，讓電路的改動能在最小，並且估計 FIB repair 後會增加多少負載，在調整的過程中，去檢查 FIB repair 後的延遲。

實作中，我們證明了 ReFL 的確能有效地改善 FIB repair 的比例，其中 initial 的方法，證明了我們有需要對一般的 cell 做調整。我們設計了一套計算的方式排列找出當前最適合的 spare cell 進行調整，比較 spare cell 排序上使用 random 的方法，其餘兩者的條件相同，最後證明了能提升約兩倍的修復比例。最後我們也列出了不考慮 timing 的情況可修復的比例，並且也提拱了完成 ReFL 後，再將 timing 可能不過的 cell 也做調整的比例，因為前者所需修復的 spare cell 較少，所以留下比較多的空間來修復一般的 cell，但比例也相當接近。

在 repair 上的結果上，timing 導致失敗的比例相當嚴重，主要是因為在電路中央的 spare cell 能修復的比例很低，如果能再繞線前先在這些 spare cell 上可供觀察的空間，就可應該能大幅提升可修復的比例，但可能會影響到原本電路的效能。如果不管 timing，在修復的比例上跟 MFOB 所做的節點 FIB 觀察的最大化相比，感覺相當接近可修復的極限的，如果要能使這些值在更大化，就要從繞線的時候開始考慮，但這樣勢必會傷害到原本的繞線演算法，或是能考慮到從晶片背後做的 FIB，又會有其他完全不同的限制，需要去重新設計其他的演算法。

第 六 章　參 考 書 目

[1] Kuo-An Chen, " Design-for-Debug Layout Adjustment for FIB Probing and Circuit Editing" Test Conference (ITC), IEEE International, 2011

[2] Kai-hui Chang, Igor L. Markov, Valeria Bertacco, "Automating Post-Silicon Debugging and Repair." International Confernce on Computer-Aided Design (pp. 91-98). 2007

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

[4] 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

[5] E. Anis and N. Nicolico, ”On Using Lossless Compression of Debug Data in Embedded Logic Analysis”, International Test Conference, pp. 1-10, 2007.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] D. C. Shaver and B. W. Ward, ”Integrated Circuit Diagnosis Using Focused Ion Beams”, Journal of Vacuum Science & Technology B; Microelectronics and Nanometer Structures, Vol. 4, Issue 1, pp. 185-188, 1986.