I493370

(19)中華民國智慧財產局

(12)發明說明書公告本 **(11)證書號數：TW I493370 B**

(45)公告日： 中華民國 104 (2015) 年 07 月 21 日

(21)申請案號：102107593 (22)申請日： 中華民國 102 (2013) 年 03 月 05 日

(51)Int. Cl.： *G06F17/50 (2006.01)* *G06F11/07 (2006.01)*

(71)申請人：國立交通大學 (中華民國) NATIONAL CHIAO TUNG UNIVERSITY (TW)
新竹市大學路 1001 號

(72)發明人：江蕙如 JIANG, IRIS HIU RU (TW) ；楊喻名 YANG, YU MING (TW) ；何松庭 HO, SUNG TING (TW)

(74)代理人：蘇建太；林志鴻

(56)參考文獻：

TW 201250862A TW 201310913A

US 2009/0007041A1

Hau-Yu Chang; Iris Hui-Ru Jiang;Yao-Wen Chang，"Timing ECO Optimization via Bezier Curve Smoothing and Fixability Identification"， IEEE， 2011。

審查人員：林文琦

申請專利範圍項數：9 項 圖式數：15 共 55 頁

(54)名稱

工程變更之保持時間修復方法

ECO HOLD TIME FIXING METHOD

(57)摘要

近代電路設計中，往往需要考量可能面臨到的動態變異，因此，電路設計會考慮最糟的情況，並透過預留的時序保護帶來確保電路能夠正確的運作，但也因此降低的電路的效能。為了消除時序保護帶，習知技術使用可復原式電路。然而，可復原式電路將會面臨到更嚴峻的短路徑問題。本發明為了使得可復原式電路的錯誤偵測與修正能夠發揮作用，因此提供一工程變更之保持時間修復方法，以解決短路徑填補問題。異於習知技術的是本發明使用全域觀點來決定填補數值與位置。此外，本發明進一步提出粗質延遲填補與細質延遲填補方法，根據所決定的填補數值來加以實現。

Modern IC designs are exposed to a wide range of dynamic variations. Traditionally, a conservative timing guardband is required to guarantee correct operations under the worst-case variation, thus leading to performance degradation. To remove the guardband, resilient circuits are proposed. However, the short path padding problem in resilient circuits is severer than conventional integrated circuit design. Therefore, the invention provides an engineering change order (ECO) hold time fixing method for the short path padding problem to enable the timing error detection and correction mechanism of resilient circuits. Unlike prior art, the invention determines the padding values and locations with a global view. Moreover, the invention proposes coarse-grained and fine-grained padding allocation methods to further achieve the derived padding values at physical implementation. Experimental results show that method of the invention is promising to validate timing error resilient circuits.

其為流程圖，故無符號。

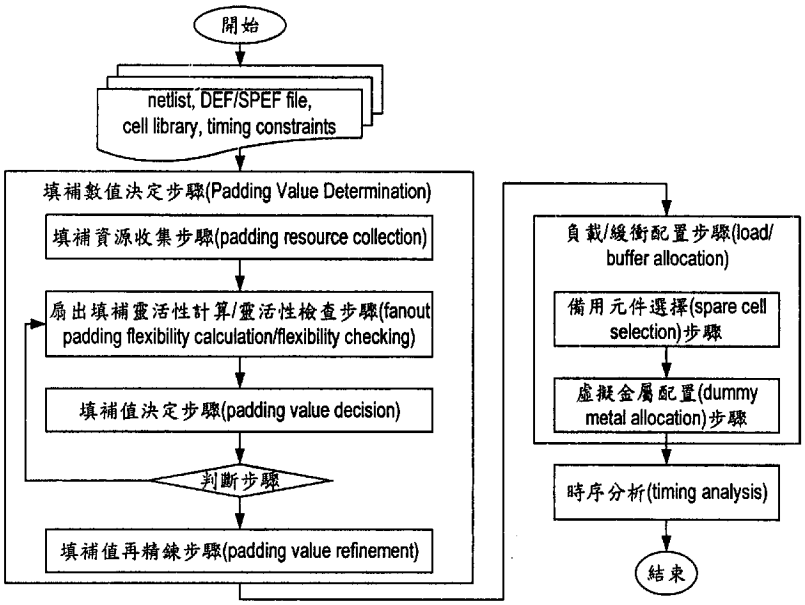

圖 4

I493370

公告本

發明摘要

※ 申請案號：102107593

※ 申請日：102. 3. 05　　※IPC 分類：G06F 17/50 (2006.01)
11/07 (2006.01)

【發明名稱】(中文/英文)

工程變更之保持時間修復方法

ECO hold time fixing method

【中文】

近代電路設計中，往往需要考量可能面臨到的動態變異，因此，電路設計會考慮最糟的情況，並透過預留的時序保護帶來確保電路能夠正確的運作，但也因此降低的電路的效能。為了消除時序保護帶，習知技術使用可復原式電路。然而，可復原式電路將會面臨到更嚴峻的短路徑問題。本發明為了使得可復原式電路的錯誤偵測與修正能夠發揮作用，因此提供一工程變更之保持時間修復方法，以解決短路徑填補問題。異於習知技術的是本發明使用全域觀點來決定填補數值與位置。此外，本發明進一步提出粗質延遲填補與細質延遲填補方法，根據所決定的填補數值來加以實現。

【英文】

Modern IC designs are exposed to a wide range of dynamic variations. Traditionally, a conservative timing guardband is required to guarantee correct operations under the worst-case variation, thus leading to performance degradation. To remove the guardband, resilient circuits are proposed. However, the short path padding problem in resilient circuits is severer than conventional integrated circuit design. Therefore, the invention provides an engineering change order (ECO) hold time fixing method for the short path padding problem to enable the timing error detection and correction mechanism of resilient circuits. Unlike prior art, the invention determines the padding values and locations with a global view. Moreover, the invention proposes coarse-grained and fine-grained padding allocation methods to further achieve the derived padding values at physical implementation. Experimental results show that method of the invention is promising to validate timing error resilient circuits.

【代表圖】

【本案指定代表圖】：圖（ 4 ）。

【本代表圖之符號簡單說明】：

其為流程圖，故無符號。

【本案若有化學式時，請揭示最能顯示發明特徵的化學式】：

「無」

發明專利說明書

【發明名稱】(中文/英文)

工程變更之保持時間修復方法

ECO hold time fixing method

【技術領域】

【0001】 本發明係關於積體電路的技術領域，尤指一種工程變更之保持時間修復方法。

【先前技術】

【0002】 隨著積體電路製程技術的不斷演進，以往對電路影響不大的效應，如今在先進製程的設計階段都必須要加以考慮，才能確保電路的正常運作。例如種種因為動態變異而造成的影響，像是工作電壓的變化、製程的變異、溫度的波動、軟錯誤(soft error)以及電晶體的退化，而造成時序上的錯誤(timing error)。在傳統的電路設計當中，往往使用最壞情況設計(worst-case design)，設計者會預留時序保護帶(timing guardband)，來保證即使遇到最糟糕的條件下，電路都能在正確的時間內完成運算。然而，由於預留時序保護帶(timing guardband)的關係，也因此降低了電路效能。

【0003】 為了消除時序保護帶(timing guardband)對電路效能的影響，許多擁有錯誤偵測和修正的可復原式電路

(resilient circuits)被提了出來。例如 D. Ernst et al.在 MICRO, pp. 7–18, 2003 所發表的論文「Razor: a low-power pipeline based on circuit-level timing speculation」中，提出 Razor 正反器(Razor flip-flop)來用以偵測錯誤的電路。圖 1 係習知 Razor 正反器之使用示意圖，其藉由額外的儲存元件陰影拴鎖器(shadow latch)110 來取樣組合邏輯 120 的輸出，此儲存元件 110 由經過延遲的時脈 clk_dly 所控制，再將主正反器 130 與陰影拴鎖器(shadow latch)110 的輸出經由一互斥反或閘(XNOR)140 做比對。如果比對結果發現有錯誤，就會發出偵測到錯誤的訊號，並把造成錯誤結果的指令重新計算來做錯誤修正。

【0004】 然而對可復原式電路(resilient circuits)而言，保持時間(hold time)是個非常重要議題。例如以圖 1 的電路為例子，如果下個週期的信號經由短路徑(short path)而被延遲的時脈所偵測到的話，會造成錯誤的判斷。為了要避免錯誤判斷的發生，短路徑(short path)的延遲至少要超過錯誤偵測窗框(error detection window)w，例如一般時脈 clk 和延遲時脈 clk_dly 的相位差。一錯誤偵測窗框 w 會對保持時間裕度(hold time margin)產生額外的要求，這個問題也存在於其它論文所提出的的可復原式電路(resilient circuits)中。由於對保持時間裕度(hold time margin)的額外要求，使得可復原式電路(resilient circuits)所面臨的短路徑填補(short path padding)問題更加的複雜。

【0005】 對於延遲填補(delay padding)的問題，習知技

術提出了對短路徑(short path)塞入緩衝器(buffer)來增加其延遲的方法。

【0006】 習知技術中，延遲填補(delay padding)會與時脈偏移排程(clock skew scheduling)的方法一起在邏輯再合成(logic resynthesis)的階段設法降低時脈週期，其目標是決定每條路徑(path)的填補延遲(padding delay)而非該在哪位置插入延遲(insert delay)。

【0007】 相較之下，另一種習知技術會決定該在哪個位置插入延遲(insert delay)。 N. V. Shenoy et al.在 ICCAD, pp. 156-161, 1993 所發表的論文「Minimum padding to satisfy short path constraints」提出了藉由線性規劃的解決方法。由於線性規劃的計算時間比較長，所以不適合用於大型設計。針對線性規劃方法不適合用於大型設計，一種先前技術提出了貪婪啟發式演算法。其中一種貪婪的方式是針對最大建立鬆弛(setup slack)的邏輯閘來做填補，目的是為了不要傷害到最長路徑延遲(longest path delay)。另一種貪婪的方式是針對最多保持違規路徑(hold violating paths)經過的邏輯閘來做填補，目的是為了減少總填補延遲(total padding delay)。

【0008】 圖 2 係一習知將時序錯誤可復原式電路(timing error resilient circuits)整合至一設計之流程圖。由於考慮時序保護頻帶(timing guardband)，基於保守時序週期(conservative clock period)的邏輯合成和時序分析後，目標時序週期(target clock period)和所述錯誤偵測區間 w 則

會被確定。S_{th}及H_{th}代表目標時序週期對保守時序週期的比率。其最長路徑延遲超過目標時鐘週期之時序可疑的正反器(flip-flop)會被可復原式電路所取代。正反器(flip-flop)被可復原式電路取代前，設計會被再合成，此時，時序可疑的正反器(flip-flop)會被分配有一個額外的保持時間裕度(hold time margin)以覆蓋該錯誤偵測窗框(error detection window)w。在正反器(flip-flop)被可復原式電路取代後，執行佈局及繞線(placement & routing)。由於顯著的保持時間裕度(hold time margin)及保持違規(hold violations)可能仍然存在於一個佈局及繞線後的設計中，因此，最後執行短路徑填補。

【0009】 根據先前技術所提出的啟發式方法，無法妥善的填補所有短路徑(short path)。圖 3A 至圖 3D 係習知技術填補短路徑之示意圖。圖 3A 電路之邏輯閘 g_1，g_2，g_3發生保持違規(hold violations)。如果反覆地針對最大建立鬆弛(setup slack)的邏輯閘來做填補或是針對最多保持違規路徑(hold violating paths)經過的邏輯閘來做填補，會得到圖 3B 和圖 3C 的結果。由圖 3B 和圖 3C 可以發現邏輯閘 g_2的保持違規(hold violations)無法解決。但是如果以圖 3D 的填補方式，是可以解決所有的保持違規(hold violations)。因此可以發現，如果在填補的時候只有採取區域觀點(local view)，可能會有無法修補的結果。然而，即使找到了一組不錯的填補方法(padding solution)，因為緩衝器能提供的延遲是固定的，仍然可能會在實體具現(physical

implementation)時失敗。例如，希望對邏輯閘 g_2 增加 0.2 單位的延遲，而緩衝器只能提供 0.15 單位的延遲和 0.25 單位的延遲，仍然會導致填補失敗。因此習知之保持時間修復之技術實仍有改善的空間。

【發明內容】

【0010】 本發明之目的主要係在提供一工程變更之保持時間修復方法，以全域觀點找出填補數值，並在粗質延遲填補(coarse-grained delay padding)中使用備用元件(spare cell)，而細質延遲填補(fine-grained delay padding)中使用虛擬金屬(dummy metal)，而可成功地修復一已佈局及繞線設計的保持時間。

【0011】 為達前述之目的，本發明提出一種工程變更之保持時間修復方法，其係對一已佈局及繞線設計藉由塞入最少的電容(capacitance)來填補完該已佈局及繞線設計的短路徑(short path)，該方法包含一填補數值決定步驟(Padding Value Determination)及一負載/緩衝配置步驟(load/buffer allocation)。該填補數值決定步驟接收該已佈局及繞線設計，依據一元件庫(cell library)、一時序限制(timing constraints)及一時序分析報告(timing analysis report)，以決定並輸出該已佈局及繞線設計之每一個閘所需填補(padding)的數量與位置。該負載/緩衝配置步驟依據一備用元件(spare cells)資訊、一虛擬金屬資訊(dummy metal information)、及該已佈局及繞線設計之每一個閘所需填補

(padding)的數量與位置，以填補完該已佈局及繞線設計的短路徑(short path)。

【圖式簡單說明】

【0012】

圖 1 係習知 Razor 正反器之使用示意圖。

圖 2 係一習知將時序錯誤可復原式電路整合至一設計之流程圖。

圖 3A 至圖 3D 係習知技術填補短路徑之示意圖。

圖 4 係本發明一種工程變更之保持時間修復方法的流程圖。

圖 5A 及圖 5B 係本發明增加短路徑(short path)的延遲之示意圖。

圖 6A 及圖 6B 係本發明對一已佈局及繞線設計執行填補數值之示意圖。

圖 7 係本發明一有界盒子(bounding box)中的對應備用元件之示意圖。

圖 8 係本發明以動態規劃解次集合加總解時的表格之示意圖。

圖 9 係本發明競爭相同的備用元件之示意圖。

圖 10 係本發明填補閘/走線選擇最佳的次集合加總解之示意圖。

圖 11 係本發明衝突填補閘/走線之虛擬金屬的示意圖。

圖 12 係本發明所使用最大流量網絡方法之示意圖。

圖 13 係本發明基準統計的結果之示意圖。

圖 14A及圖 14B係本發明的方法與習知技術的填補數值比較之示意圖。

圖 15 係本發明的方法與習知技術的負的建立鬆弛及負的保持鬆弛(negative setup slack/negative hold slack)比較之示意圖。

【實施方式】

【0013】 圖 4係本發明一種工程變更之保持時間修復方法的流程圖，該工程變更之保持時間修復方法係對一已佈局及繞線設計藉由塞入最少的電容(capacitance)來填補完該已佈局及繞線設計的短路徑(short path)。

【0014】 本發明所使用的元件時序模型(cell timing model)是根據 Synopsys 公司的 liberty library。閘延遲(gate delay)以查表(lookup table)的方式儲存，以輸入迴轉(input slew)和輸出電容(output capacitance)為索引去查表即可得到閘延遲(gate delay)。每條網絡(net)的走線延遲(wire delay)會加在對應的驅動閘(driving gate)上。閘的輸出電容(output capacitance)包含走線負載(wire loading)、扇出閘(fanout gate)的輸入電容(input capacitance)和輸出接腳電容(output pin capacitance)。G. D. Hachtel and F. Somenzi 在其 2006 出版的「Logic Synthesis and Verification Algorithms」一書中，提出負載決定現象(loading dominance phenomenon)，亦即，改變輸出負載(output loading)對閘延遲(gate delay)

的影響遠大於改變輸入延遲(input slew)。因此，本發明藉由增加閘的輸出電容(output capacitance)來增加延遲，亦即在本發明中，所增加的延遲將被轉換為電容(capacitance)。此外，閘的輸出電容(output capacitance)不能大於元件庫(cell library)中所定義的最大負載電容(maximum load capacitance)。

【0015】 短路徑填補(short path padding)對傳統的電路設計是很重要的，並且在可復原式電路(resilient circuits)的問題當中更具有挑戰性。為了使可復原式電路(resilient circuits)的錯誤偵測和修正能夠有效運作，本發明致力於解決短路徑填補(short path padding)的問題，短路徑填補(short path padding)的問題可視為：給定一個已佈局及繞線設計(placed and routed design)、元件庫(cell library)、備用元件(spare cells)資訊、虛擬金屬資訊(dummy metal information)、時序限制(timing constraints)、及時序分析報告(timing analysis report)後，本發明藉由塞入最少的電容(capacitance)，來填補完所有的短路徑(short path)並且滿足時序限制(timing constraints)。

【0016】 由於在設計流程的早期階段，時序(timing)與實體資訊(physical information)仍不夠明確，因此本發明在後佈局階段(post-layout stage)執行短路徑填補(short path padding)。圖 5A 及圖 5B 係本發明增加短路徑(short path)的延遲之示意圖。如圖 5A 所示，本發明藉由插入緩衝器，來增加短路徑(short path)的延遲。如圖 5B 所示，本發明藉

由加掛額外的負載電容(load capacitance)，來增加短路徑(short path)的延遲。因此，元件(cell)與金屬走線都能夠用來增加短路徑(short path)的延遲。通常電路設計會在佈置(placement)階段預先布置備用元件(spare cell, redundant cell)。虛擬金屬(dummy metal)亦可增加延遲，因此在本發明中，虛擬金屬(dummy metal)可以視為電容(capacitance)的額外來源並且大小可以被調整。因此，在後佈局階段(post-layout stage)可以藉由對備用元件(spare cell)與虛擬金屬(dummy metal)的重新繞線來完成填補。根據 Synopsys 公司的 Liberty library 規範，對短路徑(short path)所增加的延遲所轉換為增加的負載電容(load capacitance)或插置緩衝器(inset buffer)的數量，係可以經由查表所得到。

【0017】 如圖 4 所示，本發明一種工程變更之保持時間修復方法包含一填補數值決定步驟(Padding Value Determination)及一負載/緩衝配置步驟(load/buffer allocation)。

【0018】 該填補數值決定步驟(Padding Value Determination)接收該已佈局及繞線設計，依據一元件庫(cell library)、時序限制(timing constraints)及一時序分析報告(timing analysis report)，以決定並輸出該已佈局及繞線設計之每一個閘所需填補(padding)的數量與位置。該填補數值決定步驟(Padding Value Determination)係以全域觀點(global view)來產生所需填補(padding)的數量與位置。

【0019】 該負載/緩衝配置步驟依據一備用元件(spare

cells)資訊、一虛擬金屬資訊(dummy metal information)、及該已佈局及繞線設計之每一個閘所需填補(padding)的填補數值與位置，以填補完該已佈局及繞線設計的短路徑(short path)。該負載/緩衝配置步驟藉由增加額外的負載電容(load capacitance)與插置緩衝器(inset buffer)來實現。為了達成該填補數值決定步驟所產生的填補數值(padding value)，該負載/緩衝配置步驟分成粗質延遲填補(coarse-grained delay padding)與細質延遲填補(fine-grained delay padding)兩階段來處理。在粗質延遲填補(coarse-grained delay padding)中，使用備用元件(spare cell)，而細質延遲填補(fine-grained delay padding)使用虛擬金屬(dummy metal)。

【0020】 一般來說，愈多的填補延遲(padding delay)代表著需要更多的資源用於填補上。因此，本發明首先縮小所有的填補延遲(padding delay)，然後將每個閘/走線(gate/wire)的填補延遲(padding delay)轉換成負載/緩衝器(load/buffers)。然而，此會面臨到兩個問題，第一個問題是要選擇哪個邏輯閘來填補延遲(padding delay)，第二個問題是不能傷害到該已佈局及繞線設計中的閘之建立時間(setup time)。

【0021】 第一個問題的困難在於，如果對靠近主要輸入(primary input)的閘來做填補的話，可以很容易的就符合時序限制(timing constraints)，但全部的填補數值(padding value)卻會增加。如果對較多短路徑(short paths)所經過的gate閘來做填補的話，可以降低全部所使用的填補數值

(padding value)，但是卻可能違反時序限制(timing constraints)。在圖 2A 中，如果對閘 g_2 與閘 g_3 來填補 0.3 單位的延遲，這樣一來雖然滿足時序限制(timing constraints)，但總共需要 0.6 的填補數值(padding value)。如果對閘 g_1 來填補 0.3 單位的延遲，發現經由閘 g_2 到 o_1 的這條短路徑(short path)會無解。因此，為了要處理這個問題，決定填補數值(padding value)與位置的同時必須要有全域觀點(global view)。

【0022】 該填補數值決定步驟(Padding Value Determination)包含一填補資源收集步驟(Padding resource collection)、一扇出填補靈活性計算/靈活性檢查步驟(fanout padding flexibility calculation/flexibility checking)、一填補值決定步驟(padding value decision)、一判斷步驟、及一填補值再精鍊步驟(padding value refinement)。

【0023】 由於本發明的工程變更之保持時間修復方法適用於在後佈局階段(post-layout stage)，故首先需對可填補資源進行蒐集。可用於對每個閘進行填補的資源包括該已佈局及繞線設計之一個閘其扇出網絡(fanout net)的一有界盒子(bounding box)中的對應備用元件(spare cells)資訊及對應虛擬金屬資訊(dummy metal information)，故在該填補資源收集步驟(padding resource collection)中，依據該備用元件(spare cells)資訊及該虛擬金屬資訊(dummy metal information)，以收集該已佈局及繞線設計之一個閘其扇出網絡(fanout net)的一有界盒子(bounding box)中的對應備用

元件(spare cells)資訊及對應虛擬金屬資訊(dummy metal information)。

【0024】 邏輯閘 g_i 的最大填補電容 $C_{max}(i)$ 係為元件庫(cell library)中所定義最大輸出電容的最小值及及其可用之填補資源。因此一正反器之輸入或一主要輸出的最大填補電容 $C_{max}(i)$ 為 0。一個位於邏輯閘 g_i 與邏輯閘 g_j 之間的走線之最大填補電容 $C_{max}(i,j)$ 亦可如最大填補電容 $C_{max}(i)$ 定義，不予贅述。最大填補電容 $C_{max}(i)$ 及 $C_{max}(i,j)$ 代表上限，但仍保留靈活性(flexibilities)來設置填補數值。

【0025】 該扇出填補靈活性計算/靈活性檢查步驟(fanout padding flexibility calculation/flexibility checking)計算該已佈局及繞線設計之一個閘的扇出填補靈活性(fanout padding flexibility) $P_F(i)$。為了要以全域觀點(global view)來決定填補數值與位置，首先對所有保持違規邏輯閘(hold violating gate)的扇出錐(fanout cone)計算扇出填補靈活性(fanout padding flexibility) $P_F(i)$。

【0026】 在電路設計的過程當中，通常會表示成有向圖(directed graph) $K = (G,E)$。該有向圖 $K = (G,E)$ 係表示有向圖 K 係由閘 G 及邊 E 所組成。該有向圖 K 的每個節點 $g_i \in G$ 表示一個邏輯閘，並在其上以 $D(i)$ 表示閘延遲(gate delay)。每一條邊 $e(i,j) \in E$ 表示連著閘的導線。節點 g_i 的輸出端之建立到達時間(setup arrival time) $A(i)$ 係以下列公式表示：

$$A(i) = \max_j\{A(j)|e(j,i) \in E\} + D(i)。$$

節點 g_i 的輸出端之建立需要時間(setup required time)R(i)係以下列公式表示：

$$R(i) = \min_k\{R(i,k)|R(i,k) = R(k) - D(k), e(i,k) \in E\}。$$

節點 g_i 的輸出端的保持到達時間(hold arrival time)a(i)係以下列公式表示：

$$a(i) = \min_j\{a(j)|e(j,i) \in E\} + D(i)。$$

節點 g_i 的輸出端之保持需要時間(hold required time)r(i)係以下列公式表示：

$$r(i) = \max_k\{r(i,k)|r(i,k) = r(k) - D(k), e(i,k) \in E\}。$$

【0027】 邊 e(i,j) ∈ E 係由節點 g_i 指向節點 g_j，紀錄該邊 e(i,j)的建立邊緣鬆弛(setup edge slack)S(i,j)係以下列公式表示：

$$S(i,j) = R(i,j) - A(i)。$$

節點 $g_i \in G$ 所記錄的建立節點鬆弛(setup node slack)S(i)係以下列公式表示：

$$S(i) = \min_j\{S(i,j)|e(i,j) \in E\} = R(i) - A(i)。$$

邊 e(i,j) ∈ E 紀錄由節點 g_i 指向節點 g_j 的保持邊緣鬆弛(hold edge slack)H(i,j)係以下列公式表示：

$$H(i,j) = a(i) - r(i,j)。$$

節點 $g_i \in G$ 所記錄的保持節點鬆弛(hold node slack)H(i)係以下列公式表示：

$$H(i) = \min_j\{H(i,j)|e(i,j) \in E\} = a(i) - r(i)。$$

以圖 3A 的例子，可計算出 H(2,1) = 0.1 - 0.4 = -0.3，$H(2,o_1)$ = 0.1 - 0.3 = -0.2，H(2) = -0.3。

【0028】 每個節點 g_i 的安全填補數值(safe padding value)$P_{saf}(i)$可用下列公式表示：

$$P_{saf}(i) = \min\{S(i), |\min\{0, H(i)\}|, P_{\max}(i)\}。$$

其中，$P_{max}(i)$為節點或邏輯閘 g_i 的最大填補延遲(maximum padding delay)。該最大填補延遲(maximum padding delay) $P_{max}(i)$係由最大填補電容 $C_{max}(i)$轉換而來，因此一正反器之輸入或一主要輸出的最大填補延遲(maximum padding delay) $P_{max}(i)$為 0。

【0029】 根據以上的定義，對短路徑(short path)所經過的節點 g_i 增加 t 延遲且 $t \leq P_{saf}(i)$時，滿足此條件的情況下是不會違反時序限制(timing constraints)。

【0030】 節點 $g_i \in G$ 所記錄的扇出填補靈活性(fanout padding flexibility)$P_F(i)$係以下列公式表示：

$$P_F(i) = \begin{cases} 0, & g_i \in PO \text{ or } H(i) \geq 0; \\ \min\{0, \min_{e(i,j)\in E}\{H'(i,j)\}\} - H(i), & otherwise. \end{cases}$$

當中，$H'(i,j) = H(i,j) + P_F(j) + P_{saf}(j)$。

$$S'(i) = \min_j\{S(i,j) - (S'(j) - S(j)) - P_{saf}(j)|e(i,j) \in E\}。$$

H'(i)和 S'(i)表示在扇出都填補最大可允許的值情況下的更新後之鬆弛。$P_{saf}(i)$也會隨著動態地做更新。

【0031】 本發明對每一個閘 g_i 定義扇出填補靈活性

(fanout padding flexibility)$P_F(i)$，來表示閘 g_i 的扇出錐(fanout cone)之中所允許的最大填補數值。對於保持滿足的閘(hold satisfying gate)或是主要輸出(primary output)，其扇出填補靈活性 $P_F(i)$的值會是 0。當扇出都填補最大可允許的數值情況下，對於保持違規邏輯閘(hold violating gate)g_i，其扇出填補靈活性 $P_F(i)$的值會是閘 g_i 目前的保持節點鬆弛(hold node slack)H(i)和扇出邊緣(fanout edge)最小的更新後的保持邊緣鬆弛(hold edge slack)之差異。

【0032】 根據上述數學公式，扇出填補靈活性 PF(i)會從主要輸出(primary output)朝著主要輸入(primary input_計算回去。例如以圖 3A 的例子，根據數學公式可以得到以下數值：

$P_F(o1) = 0.0$, PF(FF2) = 0.0；

$P_F(1) = \min\{0, (-0.3+0.0+0.0)\}-(-0.3) = 0.0$；

H'(1) = -0.3, S'(1) = 0.4；

$P_F(2) = \min\{0, (-0.2+0.0+0.0), (-0.3+0.0+0.3)\}-(-0.3) = 0.1$；

$H'(2) = \min\{(-0.3+0.0+0.3), (-0.2+0.0+0.0)\} = -0.2$；

$S'(2) = \min\{(0.4-0.0-0.3), (0.3-0.0-0.0)\} = 0.1$；

$P_F(3) = \min\{(-0.3+0.0+0.3), 0.0\}-(-0.3) = 0.3$；

$H'(3) = \min\{(-0.3+0.0+0.3)\} = 0.0$；

$S'(3) = \min\{(1.0-0.0-0.3)\} = 0.7$。

【0033】 根據以上的數學公式，本發明可由以下的式子來檢查扇出填補靈活性(fanout padding flexibility)$P_F(i)$。如果$|\min\{0, H(i)\}| > P_F(i), g_i \in PI$,PI 為主要輸入(primary

input)，表示保持違規(hold violations)無法由對邏輯閘做填補來解決。如果$S(i,j) < |\min\{0, H(i,j)\}|, e(i,j) \in E$表示保持違規(hold violations)無法由對走線做填補來解決。

【0034】 該填補值決定步驟(padding value decision)，依據該一個閘的扇出填補靈活性(fanout padding flexibility)$P_F(i)$以計算該一個閘的一填補值 P(i)。在前述考慮全域觀點的情況下所計算得到扇出填補靈活性(fanout padding flexibility)$P_F(i)$，接下來便可由此來決定填補數值。針對每個保持違規邏輯閘(hold violating gate)g_i以拓樸次序(topological order)來計算出其填補數值。扇出填補靈活性(fanout padding flexibility)$P_F(i)$表示邏輯閘 g_i的扇出錐(fanout cone)之中所允許的最大填補數值。因此，對保持違規邏輯閘(hold violating gate)g_i所做的填補只需要考慮扣除扇出錐(fanout cone)的影響之後所剩下未修完的負的保持鬆弛(negative hold slack)。例如，安全填補數值(safe padding value)和扇出填補靈活性(fanout padding flexibility)$P_F(i)$的差異。因此，由前述的數學公式及符號，該一個閘 g_i的該填補值係以下列公式表示：

$$P(i) = \max\{P_{saf}(i) - P_F(i), 0\}。$$

【0035】 $P_{saf}(i)$代表在邏輯閘 g_i的扇入閘(fanin gates)經過填補之後的安全填補數值(safe padding value)。當決定了每個邏輯閘 g_i的填補數值之後，增加的延遲會對其扇出閘(fanout gate)的到達時間(arrival time)造成影響。因此，需對填補的邏輯閘 g_i的扇出邊緣鬆弛(fanout edge slack)更

新。扇出邊緣鬆弛(fanout edge slack)更新可以用下列公式表示：

$$S(i,j) = R(i,j) - A(i) - P(i),$$

$$H(i,j) = P(i) + a(i) - r(i,j)。$$

【0036】 對填補的邏輯閘 g_i 的扇出邊緣鬆弛((fanout edge slack)做更新之後，扇出閘(fanout gates)的建立節點鬆弛(setup node slack)及保持節點鬆弛(hold node slack)也會跟著重新計算。

【0037】 圖 6A 及圖 6B 係本發明對一已佈局及繞線設計執行填補數值之示意圖。以圖 6A 做為決定填補數值的例子，假設 $P_{max}(2) = 0.5$, $P_{max}(3) = 0.4$, $P_{max}(1) = 0.4$。根據扇出填補靈活性(fanout padding flexibility)$P_F(i)$，可以得到以下數值：

$P_{saf}(2) = \min\{0.3, |-0.3|, 0.5\} = 0.3$, $P(2) = 0.3-0.1 = 0.2$；

$S(2,1) = 1.1-0.7-0.2 = 0.2$, $H(2,1) = 0.2+0.1-0.4 = -0.1$；

$P_{saf}(3) = \min\{0.3, |-0.3|, 0.4\} = 0.3$, $P(3) = 0.3-0.3 = 0.0$；

$S(3,1) = 1.1-0.1-0.0 = 1.0$, $H(3,1) = 0.0+0.1-0.4 = -0.3$；

$S(1) = \min\{1.0, 0.2\} = 0.2$, $H(1) = \min\{-0.3, -0.1\} = -0.3$；

$P_{saf}(1) = \min\{0.2, |-0.3|, 0.4\} = 0.2$, $P(1) = 0.2-0.0 = 0.2$。

【0038】 但是決定完填補數值之後，從圖 6A 可以發現，經由邏輯閘 g_3 到邏輯閘 g_1 的短路徑(short path)仍然有 -0.1 的保持違規(hold violations)，這是由於高估扇出填補

靈活性(fanout padding flexibility)$P_F(i)$的緣故。經由重新執行該扇出填補靈活性計算/靈活性檢查步驟(fanout padding flexibility calculation/flexibility checking)與該填補值決定步驟(padding value decision)，將得到的填補數值做累加，可以得到圖 6B 的結果，所有短路徑(short path)都得以解決。因此，該扇出填補靈活性計算/靈活性檢查步驟(fanout padding flexibility calculation/flexibility checking)與該填補值決定步驟(padding value decision)兩步驟會做重複的計算直到保持違規(hold violations)都得以解決或是無法再進一步改善為止。因此在該判斷步驟中，其判斷該已佈局及繞線設計之每一個閘的短路徑(short path)之保持違反(hold violation)是否均已解決或是違反(violation)已無法再刪減(eliminated)，若否，則重回扇出填補靈活性計算/靈活性檢查步驟(fanout padding flexibility calculation/flexibility checking)。

【0039】 本發明希望能夠更進一步的減少在邏輯閘上的全部填補數值，因此如果仍然有未解決的保持違規(hold violations)，本發明會使用填補走線(padding wires)對邏輯閘增加其延遲。

【0040】 在本發明的填補值決定步驟(padding value decision)中，填補位置(padding location)會盡量選在比較靠近主要輸出(primary output, PO)的地方，但是對於分叉的短路徑(forked short paths)，此會使得所使用的填補數值因此而增加。如圖 6A 所示，根據前述的計算所得到的結果總共

需要 0.5 的填補延遲(padding delay)，但由於邏輯閘 g_4 是分叉路徑(forked paths)，故以圖 6B 的填補延遲(padding delay)方式，可以進一步減少總共所使用的填補數值為 0.4。

【0041】 本發明希望能夠藉由對填補數值再行刪減，使填補數值最後決定的位置會是有較多短路徑(short paths)所經過的邏輯閘。因此，當該判斷步驟判定該已佈局及繞線設計之每一個閘的短路徑(short path)之保持違反(hold violation)均已解決或是違反(violation)已無法再刪減(eliminated)時，該填補值再精鍊步驟(padding value refinement)以反向拓樸次序(reverse topological order)計算該已佈局及繞線設計之一個閘的再精鍊填補值(refined padding value)，以再刪減該已佈局及繞線設計之一個閘的填補值。

【0042】 本發明透過以下的定義及數學公式，來達成降低填補數值的目的。一反向填補值(reverse padding value)$P_{rev}(i)$係以下列公式表示：

$$P_{rev}(i)=\begin{cases} P(i), & if\ g_i\ has\ only\ one\ hold\ violating\ fanin; \\ 0, & otherwise. \end{cases}$$

反向填補值(reverse padding value)$P_{rev}(i)$表示每個填補的邏輯閘 g_i 有多少的填補能夠朝著扇入(fanin)方向傳遞。為了避免動到結合路徑(joined path)的填補數值，因此本發明只針對只有一個扇入保持違規(fanin hold violation)的邏輯閘來做處理。如果邏輯閘 g_i 有保持邊緣鬆弛(hold edge slack)小於填補數值時，表示邏輯閘 g_i 有保持違規(hold violation)

的情況發生，如此一來便可以對填補之後所更新的結果找出原本的保持違規(hold violation)。

【0043】 一新增安全填補數值(added safe padding value)$P_{add}(i)$係以下列公式表示：

$$P_{add}(i) = \min\{S(i), P_{\max}(i) - P(i)\}。$$

新增安全填補數值(added safe padding value)$P_{add}(i)$是考慮保持限制(setup constraint)與最大填補延遲(maximum padding delay) $P_{max}(i)$，計算後得到每個邏輯閘還能夠增加多少填補。該再精鍊填補值(refined padding value)$P_{ref}(i)$係以下列公式表示：

$$P_{ref}(i) = P(i) + \min\{P_{add}(i), \min_j\{P_{rev}(j) | H(i,j) < P(i)\}。$$

【0044】 根據以上的定義，以反向拓樸次序(reverse topological order)可以算出每個邏輯閘的再精鍊填補值(refined padding value)$P_{ref}(i)$，新的結果能夠更進一步的減少全部的填補數值。

【0045】 如果電路的建立鬆弛(setup slack)很緊或是最大輸出電容(maximum output capacitance)的限制，導致對邏輯閘做填補後，仍然有無法解決的保持違規(hold violation)的話，本發明會在再精鍊(refinement)之後，透過填補於走線上(padding on wires)來解決剩下的保持違規(hold violation)。亦即，該填補值再精鍊步驟(padding value refinement)更包含一走線填補值(wire padding value)步驟。

【0046】 該走線填補值(wire padding value)步驟係計算

一走線 e(i,j)的一走線填補值(wire padding value)，以再刪減該已佈局及繞線設計之一個閘/走線的填補值，該走線填補值(wire padding value)P(i,j)係以下列公式表示：

$$P(i,j) = \min\{S(i,j), |\min\{0, H(i,j)\}|\}。$$

【0047】 該走線填補值(wire padding value)P(i,j)會以拓樸次序(topological order)來做決定。根據時序庫(timing library)，最後決定每個邏輯閘的填補遲延(padding delay)會轉換成填補負載(padding load)或填補緩衝器(padding buffer)。

【0048】 該負載/緩衝配置步驟包含一找尋備用元件候選者(finding spare cell candidates)步驟、一備用元件選擇(spare cell selection)步驟、一虛擬金屬配置(dummy metal allocation)步驟。

【0049】 為了達成該填補數值決定步驟所產生的填補數值(padding value)，該負載/緩衝配置步驟分成粗質延遲填補(coarse-grained delay padding)與細質延遲填補(fine-grained delay padding)兩階段來處理。

【0050】 從元件庫(cell library)可以得到每個元件(cell)所能提供的電容，經由轉換得到每個元件(cell)所能提供的延遲。但由於是元件(cell)離散的關係，使得備用元件(spare cell)可能無法滿足閘/走線所要的填補負載(padding load)或填補緩衝器(padding buffer)。因此，在粗質延遲填補(coarse-grained delay padding)中，使用備用元件(spare cell)，而細質延遲填補(fine-grained delay padding)使用虛擬

金屬(dummy metal)。

【0051】 該找尋備用元件候選者(finding spare cell candidates)步驟，對該已佈局及繞線設計之需要進行填補的閘/走線(gate/wire)產生備用元件候選者(spare cell candidates)，其係將每個需要進行填補的閘的扇出網絡(fanout net)的一有界盒子(bounding box)中的對應備用元件(spare cells)規劃為需要進行填補的閘能夠使用的資源。圖7係本發明一有界盒子(bounding box)中的對應備用元件之示意圖。如圖7所示，能提供填補給待填補閘 g_2 的備用元件(spare cell)為 s_1，s_2 與 s_3，能提供填補給待填補閘 g_2 與 g_3 之間走線的備用元件(spare cell)為 s_2。接下來要決定如何使用這些備用元件(spare cell)來做閘/走線的填補。備用元件(spare cells)與重新走線(rewiring)所提供的延遲，應該盡可能地接近每個閘所要的填補延遲(padding delay)，但是不能超過填補延遲(padding delay)，避免發生時序違規(timing violation)。

【0052】 本發明對一個閘/走線找到合適的備用元件候選者(spare cell candidate)來做填補的問題，係以次集合加總解(subset sum solution)來處理。次集合加總解(subset sum solution)可以透過動態規劃(dynamic programming)來得到最佳解。做動態規劃(dynamic programming)時的步階大小(step size)會取決於是否能夠兼顧效率與準確度。例如圖7中，待填補閘 g_2 的填補延遲(padding delay)是 0.25，能夠使用的備用元件(spare cells)為 s_1，s_2 與 s_3。圖8係本發明以

動態規劃(dynamic programming)解次集合加總解(subset sum solution)時的表格之示意圖，此時決定的步階大小(step size)是 0.05。可以發現當填補延遲(padding delay)是 0.25 時，s_2 與 s_3 的組合會是最佳解，因此將 s_2 與 s_3 紀錄為 g_2 的備用元件候選者(spare cell candidates)。同樣的以此也可以決定出填補走線(padding wire)的備用元件候選者(spare cell candidates)。

【0053】 然而，對於不同的填補閘/走線(padding gate/wire)的次集合加總解(subset sum solution)，可能會競爭相同的備用元件(spare cell)的情形。圖 9 係本發明競爭相同的備用元件之示意圖。如圖 9 所示，g_1 與 g_2 都想以 s_3 來做填補。為了要處理資源競爭的問題，本發明對所有填補閘/走線(padding gate/wire)紀錄使用者定義允許範圍之內的多組次集合加總解(subset sum solution)。如圖 8 所示，當填補數值(padding value)是 0.25 時，如果定義範圍為 0.05，則 $\{s_1\}$ 與 $\{s_2, s_3\}$ 均會被紀錄為備用元件候選者(spare cell candidates)。接下來，本發明要對每個填補閘/走線(padding gate/wire)，來決定用哪些備用元件(spare cell)來做填補。

【0054】 該備用元件選擇(spare cell selection)步驟其係將每個需要進行填補的閘/走線選擇最佳的次集合加總解(subset sum solution)，填補完該已佈局及繞線設計的短路徑(short path)。對填補閘/走線(padding gate/wire)決定用哪些備用元件(spare cell)來做填補的時候，不同的閘/走線

(gate/wire)會有資源競爭的問題。因此，本發明對填補閘/走線(padding gate/wire)記錄了多組備用元件候選者(spare cell candidates)。在該備用元件選擇(spare cell selection)步驟中，本發明將決定填補閘/走線(padding gate/wire)所要使用的備用元件(spare cell)。備用元件選擇(spare cell selection)問題可視為設定包裝問題(set packing problem)，是 NP-hard 的問題。本發明為了有效率的處理，先讓每個填補閘/走線(padding gate/wire)選擇最佳的次集合加總解(subset sum solution)。如果發生衝突的情況，將有衝突情況的填補閘/走線(padding gate/wire)集合起來，並且以其次集合加總解(subset sum solution)的數量來做從小到大的排序。最後再以此順序，讓每個填補閘/走線(padding gate/wire)選擇最佳的次集合加總解(subset sum solution)。圖 10 係本發明填補閘/走線選擇最佳的次集合加總解之示意圖。如圖 10 所示，閘 g_1 對應至備用元件 $\{s_4\}$、閘對應至備用元件 $\{s_2, s_3\}$、閘 g_3 對應至備用元件 $\{s_5, s_6\}$。

【0055】 在之前的粗質延遲填補(coarse-grained delay padding)，本發明使每個填補閘(padding gate)選擇備用元件(spare cells)。如果選擇後的備用元件(spare cells)仍然無法滿足要求的填補延遲(padding delay)，則剩下不足的延遲(delay)經過計算轉為電容(capacitance)，並且在細質延遲填補(fine-grained delay padding)階段利用虛擬金屬插置(dummy metal insertion)來達成。因此，該虛擬金屬配置(dummy metal allocation)步驟利用虛擬金屬插置(dummy

metal insertion)，填補完該已佈局及繞線設計的短路徑(short path)。

【0056】 填補閘時所能使用的虛擬金屬資源(dummy metal resource)係取決於其扇出網絡(fanout net)的有界盒子(bounding box)內未被使用的繞線資源(routing resource)。如果不同的閘/走線(gate/wire)的有界盒子(bounding box)有重疊的話，會發生互相爭奪相同虛擬金屬資源的情形。圖 11 係本發明衝突填補閘/走線(padding gate/wire)之虛擬金屬(dummy metal)的示意圖。將獨立有界盒子(independent bounding box)內的虛擬金屬稱作獨立虛擬金屬(independent virtual metal)，反之稱為相依虛擬金屬(dependent virtual metal)。本發明首先使用獨立虛擬金屬來做填補，如果還是有未解決閘/走線(gate/wire)的話，再以最大流量網絡(maximum flow network)方法，分配相依虛擬金屬。

【0057】 圖 12 係本發明所使用最大流量網絡(maximum flow network)方法之示意圖。最大流量網絡(maximum flow network)方法包含源節點(source node) s 及匯集節點(sink node) t。每個填補閘/走線對應的節點 g_i 和節點 d_i 表示相依虛擬金屬。連接節點 s 與節點 g_i 的邊緣能力(edge capacity)為節點 g_i 的剩餘待填補電容(remaining padding capacitance)，連接節點 d_i 與節點 t 的邊緣能力(edge capacity)為節點 d_i 所能提供的電容(capacitance)。如果節點 g_i 的有界盒子(bounding box)有覆蓋到節點 d_i 的範圍，則連接節點 g_i 與節點 d_i 的邊緣能力(edge capacity)為無限大。如圖 11 所

示，分配完獨立虛擬金屬(independent virtual metal)之後，閘(或又稱為節點)g_1與閘g_2和走線w_3仍然有未解決填補電容(unfixed padding capacitance)，其分別為 0.15，0.2 和 0.1。虛擬金屬的重疊區域(overlapping region)可以提供的電容分別為 0.25 與 0.2。對應的流量網絡(flow network)與最大流量(maximum flow)如圖 12 所示。根據計算完的流量，可以以此來決定用哪些虛擬金屬來修補剩下的填補電容。如果仍然有未解決填補電容(unfixed padding capacitance)，代表可能因為資源有限而無法加以實現，此時會回到填補數值決定步驟重新決定填補數值，讓資源比較少的閘不要給超過該閘所能負荷的填補數值。

【0058】 圖 13 係本發明基準統計(benchmark statistics)的結果之示意圖，其中，Circuit 指的是電路名稱，#Gate 指的是組合邏輯閘數，#FF 指的是正反器的數量，#SFF 指的是有時序疑慮的正反器的數量，Conservative clock period (ns) 指的是考慮時序保護帶(timing guardband)的時序週期，THS 指的是來自有時序疑慮的正反器之全部為負數的保持鬆弛(total negative hold slack)。每個電路是由 Synopsys 及 Cadence 公司的商業程式工具根據 55-nm 製程來做合成、佈局、與繞線。同時也使用這些工具來驗證電路時序。本發明中所使用 S_{th}=75%與 H_{th}=25%的設定，其為為現今電路設計常用的設定。

【0059】 圖 14A 及圖 14B 係本發明的方法與習知技術的填補數值比較之示意圖。圖 14A 及圖 14B 中的 Padding

delay 指的是全部所決定的填補數值(padding value)總和。LP 指的是 N. V. Shenoy et al.在 ICCAD, pp. 156–161, 1993 所發表的論文「Minimum padding to satisfy short path constraints」提出線性規劃(linear programming)方法以 IBM 公司的 ILOG CPLEX Optimizer 實現。Greedy 1 指的是 Y. Sun et al.在美國專利 US Patent 7,278,126 中提出每次選擇最大建立鬆弛(setup slack)的閘來做填補的方式。Greedy 2 指的是每次選擇最多保持違規路徑(hold violations path)經過的閘來做填補。由圖 14A 及圖 14B 所示可知，因為本發明有考慮全域觀點，所以本發明的方法能夠解決每一個電路的保持違規(hold violations)。線性規劃(LP)方法在處理大問題時，會因為時間的問題而無法處理。Greedy 1 與 Greedy 2 有很大機率會無法處理所有的保持違規(hold violations)並且因為需要反覆的計算，所以需要花費較多時間。

【0060】 圖 15 係本發明的方法與習知技術的負的建立鬆弛及負的保持鬆弛(negative setup slack/negative hold slack)比較之示意圖。比較了本發明的負載/緩衝配置與 N. V. Shenoy et al.方法的結果。如同前述所提到，即使在 N. V. Shenoy et al.得到了最佳的結果，但是依此結果映射(mapping)過去仍然可能有保持違規(hold violations)未能解決。以粗質延遲填補(coarse-grained delay padding)與細質延遲填補(fine-grained delay padding)兩階段來處理，本發明可以成功實現該填補數值決定步驟(Padding Value Determination)所決定的填補數值。備用元件(spare cell)與

虛擬金屬(dummy metal)的組合提供了本發明彈性地來做電容配置(capacitance allocation)，由於離散緩衝器延遲的問題(discrete buffer delay problem)，虛擬金屬(dummy metal)扮演著不可或缺的腳色。

【0061】 由前述說明可知，本發明的貢獻為：

(1)以全域觀點找出填補數值：習知技術中的貪婪啟發式演算方法係以區域性觀點，因此有很大機率會無法修完電路中的所有保持違規(hold violations)。而，本發明會計算每個邏輯閘的扇出錐(fanout cone)所能填補的彈性範圍，然後根據得到的結果，以全域觀點的方式來決定填補數值。

(2) 粗質延遲填補(coarse-grained delay padding)與細質延遲填補(fine-grained delay padding)：由於元件庫(cell library)所能提供的延遲是固定的，因此本發明提出在粗質延遲填補(coarse-grained delay padding)中，使用備用元件(spare cell)，而細質延遲填補(fine-grained delay padding)使用虛擬金屬(dummy metal)，因為虛擬金屬(dummy metal)可以依需要來調整大小。

(3) 後佈局階段(post-layout stage)執行短路徑填補(short path padding)：習知技術在處理這個問題時，是在邏輯重新合成階段決定填補數值，因此習知技術的時序報告(timing report)可能有一些不正確，同時可以使用的填補資源還不明確，因此本發明在後佈局階段(post-layout stage)階段實現延遲填補。

【0062】 綜上所述，根據以上的觀察，本發明首先在一填補數值決定步驟(Padding Value Determination)中試著減少填補數值並且決定其位置。接著在負載/緩衝配置步驟(load/buffer allocation)做分配來滿足填補數值決定步驟決定的填補數值。在填補數值決定步驟中，本發明以全域觀點計算每個邏輯閘的扇出錐(fanout cone)所能填補的彈性範圍，並依此決定每個邏輯閘的填補數值。在負載/緩衝配置步驟中，為了達成填補數值決定步驟決定的填補數值，本發明提出了在粗質延遲填補步驟(coarse-grained delay padding)及細質延遲填補步驟(fine-grained delay padding)。因為元件庫(cell library)能提供的延遲是固定的值，所以粗質延遲填補(coarse-grained delay padding)中，使用備用元件(spare cell)。剩下的部分，由於虛擬金屬(dummy metal)可以視為額外的資源並且可以被調整，所以，而細質延遲填補(fine-grained delay padding)使用虛擬金屬(dummy metal)。

【0063】 上述實施例僅係為了方便說明而舉例而已，本發明所主張之權利範圍自應以申請專利範圍所述為準，而非僅限於上述實施例。

【符號說明】

【0064】

陰影拴鎖器110　　組合邏輯120

主正反器130

【生物材料寄存】

國內寄存資訊【請依寄存機構、日期、號碼順序註記】

國外寄存資訊【請依寄存國家、機構、日期、號碼順序註記】

【序列表】(請換頁單獨記載)

104年11月4日修正替換頁(本) P.1-5
第 102107593 號修正頁

申請專利範圍

1. 一種工程變更之保持時間修復方法，其係對一已佈局及繞線設計藉由塞入最少的電容來填補完該已佈局及繞線設計的短路徑，該方法包含：

一填補數值決定步驟，接收該已佈局及繞線設計，依據一元件庫、一時序限制及一時序分析報告，以決定並輸出該已佈局及繞線設計之每一個閘所需填補的數量與位置；以及

一負載/緩衝配置步驟，依據一備用元件資訊、一虛擬金屬資訊、及該已佈局及繞線設計之每一個閘所需填補的數量與位置，以填補完該已佈局及繞線設計的短路徑；

其中，該填補數值決定步驟包含：

一填補資源收集步驟，依據該備用元件資訊及該虛擬金屬資訊，以收集該已佈局及繞線設計之一個閘其扇出網絡的一有界盒子中的對應備用元件資訊及對應虛擬金屬資訊；

一扇出填補靈活性計算/靈活性檢查步驟，計算該已佈局及繞線設計之一個閘的扇出填補靈活性$P_F(i)$；

一填補值決定步驟，依據該一個閘的該扇出填補靈活性$P_F(i)$以計算該一個閘的一填補值；

一判斷步驟，其判斷該已佈局及繞線設計之每一個閘的短路徑之保持違反是否均已解決或是違反已無法再刪減，若否，則重回扇出填補靈活性計算/靈活性檢查步驟；以及

一填補值再精鍊步驟，若判定該已佈局及繞線設計之每一個閘的短路徑之保持違反均已解決或是違反已無法再刪減時，以反向拓樸次序計算該已佈局及繞線設計之一個閘的再精鍊填補值，以再刪減該已佈局及繞線設計之一個閘的填補值；

其中，該負載/緩衝配置步驟包含：

一找尋備用元件候選者步驟，對該已佈局及繞線設計之需要進行填補的閘/走線產生備用元件候選者，其係將每個需要進行填補的閘的扇出網絡的一有界盒子中的對應備用元件規劃為需要進行填補的閘能夠使用的資源；

一備用元件選擇步驟，其係將每個需要進行填補的閘/走線選擇最佳的次集合加總解，填補完該已佈局及繞線設計的短路徑；以及

一虛擬金屬配置步驟，利用虛擬金屬插置，填補完該已佈局及繞線設計的短路徑；

其中，該已佈局及繞線設計係以一有向圖K{K = (G,E)}表示，該有向圖的每個節點g_i($g_i \in G$)係為該已佈局及繞線設計中的一個閘，並在該節點g_i上以D(i)表示對應於該閘的閘延遲，每一條邊e(i,j){e(i,j) ∈ E}表示連著該閘的導線，節點g_i的輸出端之建立到達時間A(i)係以下列公式表示：

$$A(i) = \max_j\{A(j)|e(j,i) \in E\} + D(i),$$

節點g_i的輸出端之建立需要時間R(i)係以下列公式表示：

$$R(i) = \min_k\{R(i,k)|R(i,k) = R(k) - D(k), e(i,k) \in E\},$$

節點 g_i 的輸出端的保持到達時間 a(i)係以下列公式表示：

$$a(i) = \min_j\{a(j)|e(j,i) \in E\} + D(i),$$

節點 g_i 的輸出端之保持需要時間 r(i)係以下列公式表示：

$$r(i) = \max_k\{r(i,k)|r(i,k) = r(k) - D(k), e(i,k) \in E\}$$

。

2. 如申請專利範圍第 1 項的工程變更之保持時間修復方法，其中，邊 e(i,j){e(i,j) ∈ E}紀錄由節點 g_i 指向節點g_j 的建

立邊緣鬆弛 S(i,j)係以下列公式表示：

$$S(i,j) = R(i,j) - A(i),$$

節點$g_i \in G$所記錄的建立節點鬆弛S(i)係以下列公式表示：

$$S(i) = \min_j\{S(i,j)|e(i,j) \in E\} = R(i) - A(i),$$

邊$e(i,j) \in E$紀錄由節點g_i指向節點g_j的保持邊緣鬆弛H(i,j)係以下列公式表示：

$$H(i,j) = a(i) - r(i,j),$$

節點$g_i \in G$所記錄的保持節點鬆弛H(i)係以下列公式表示：

$H(i) = \min_j\{H(i,j)|e(i,j) \in E\} = a(i) - r(i)$。

3. 如申請專利範圍第 2 項的工程變更之保持時間修復方法，其中，每個節點 g_i的安全填補數值 $P_{saf}(i)$係以下列公式表示：

$P_{saf}(i) = \min\{S(i), |\min\{0, H(i)\}|, P_{\max}(i)\}$。

4. 如申請專利範圍第3項的工程變更之保持時間修復方法，其中，節點$g_i \in G$所記錄的扇出填補靈活性$P_F(i)$係以下列公式表示：

$$P_F(i) = \begin{cases} 0, & g_i \in PO \text{ or } H(i) \geq 0; \\ \min\{0, \min_{e(i,j)\in E}\{H'(i,j)\}\} - H(i), & otherwise. \end{cases}$$

當中，$H'(i,j) = H(i,j) + P_F(j) + P_{saf}(j)$。

5.如申請專利範圍第4項的工程變更之保持時間修復方法，其中，該一個閘的該填補值係以下列公式表示：

$P(i) = \max\{P_{saf}(i) - P_F(i), 0\}$。

6.如申請專利範圍第5項的工程變更之保持時間修復方法，其中，一反向填補值$P_{rev}(i)$係以下列公式表示：

$$P_{rev}(i)=\begin{cases} P(i), & if\ g_i\ has\ only\ one\ hold\ violating\ fanin; \\ 0, & otherwise. \end{cases}$$

一新增安全填補數值 $P_{add}(i)$係以下列公式表示：

$$P_{add}(i) = \min\{S(i), P_{\max}(i) - P(i)\}.$$

該再精鍊填補值 $P_{ref}(i)$係以下列公式表示：

$$P_{ref}(i) = P(i) + \min\{P_{add}(i), \min_j\{P_{rev}(j)|H(i,j) < P(i)\}。$$

7.如申請專利範圍第6項的工程變更之保持時間修復方法，其中，該填補值再精鍊步驟更包含：

一走線填補值步驟，計算一走線e(i,j)的一走線填補值，以再刪減該已佈局及繞線設計之一個閘/走線的填補值，該走線填補值P(i,j)係以下列公式表示：

$$P(i,j) = \min\{S(i,j), |\min\{0, H(i,j)\}|\}。$$

8.如申請專利範圍第7項的工程變更之保持時間修復方法，其中，該備用元件選擇步驟中，如果發生衝突的情況，將有衝突情況的進行填補的閘/走線集合，並且以該進行填補的閘/走線的次集合加總解的數量來做從小到大的排序，最後再以此順序，讓每個該進行填補的閘/走線選擇最佳的次集合加總解。

9.如申請專利範圍第8項的工程變更之保持時間修復方法，其中，於該虛擬金屬配置步驟中，如果發生衝突的情況，以最大流量網絡方法分配虛擬金屬。

圖式(請見下頁)

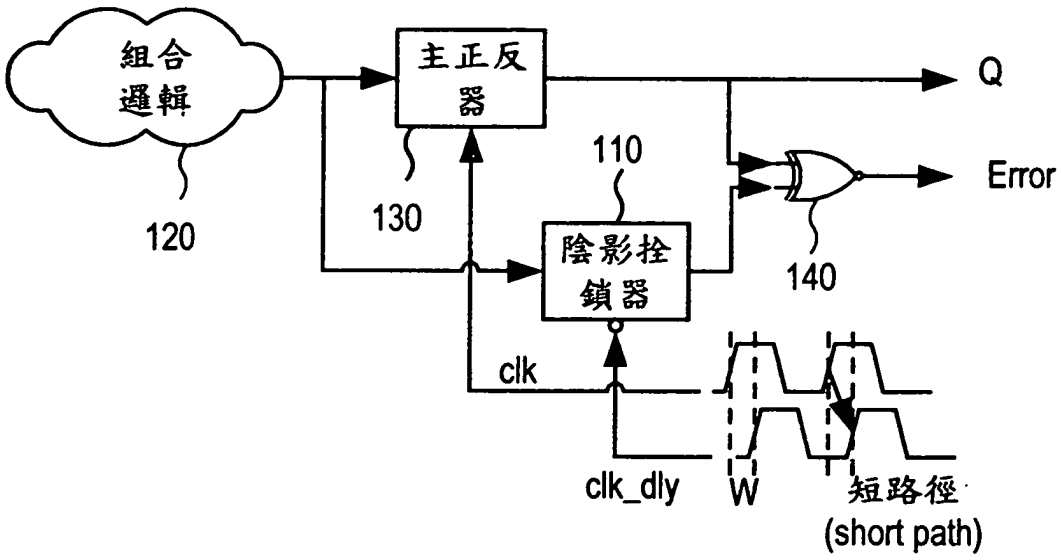
組合
邏輯
主正反
器
Q
Error
110
120
130
陰影拴
鎖器
140
clk
clk_dly
W
短路徑
(short path)

圖 1

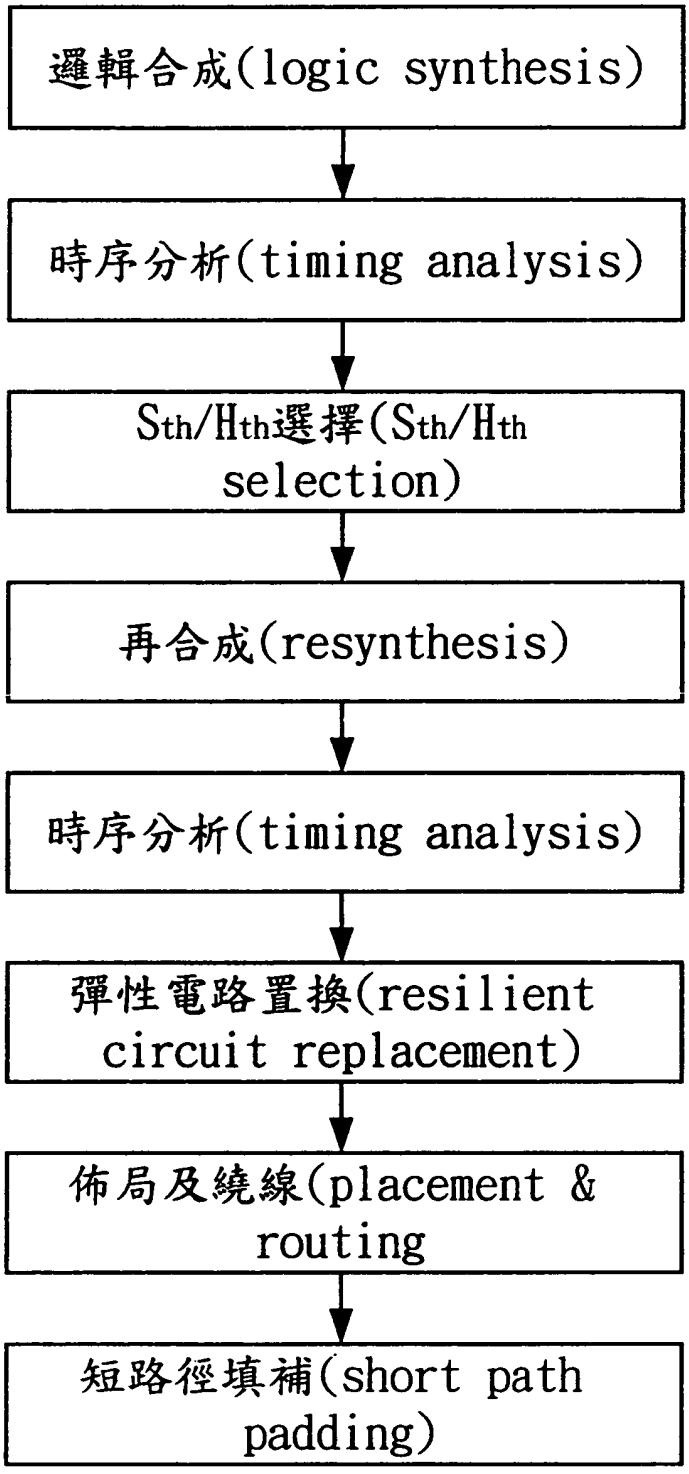
邏輯合成(logic synthesis)
時序分析(timing analysis)
Sth/Hth選擇(Sth/Hth selection)
再合成(resynthesis)
時序分析(timing analysis)
彈性電路置換(resilient circuit replacement)
佈局及繞線(placement & routing
短路徑填補(short path padding)

圖 2

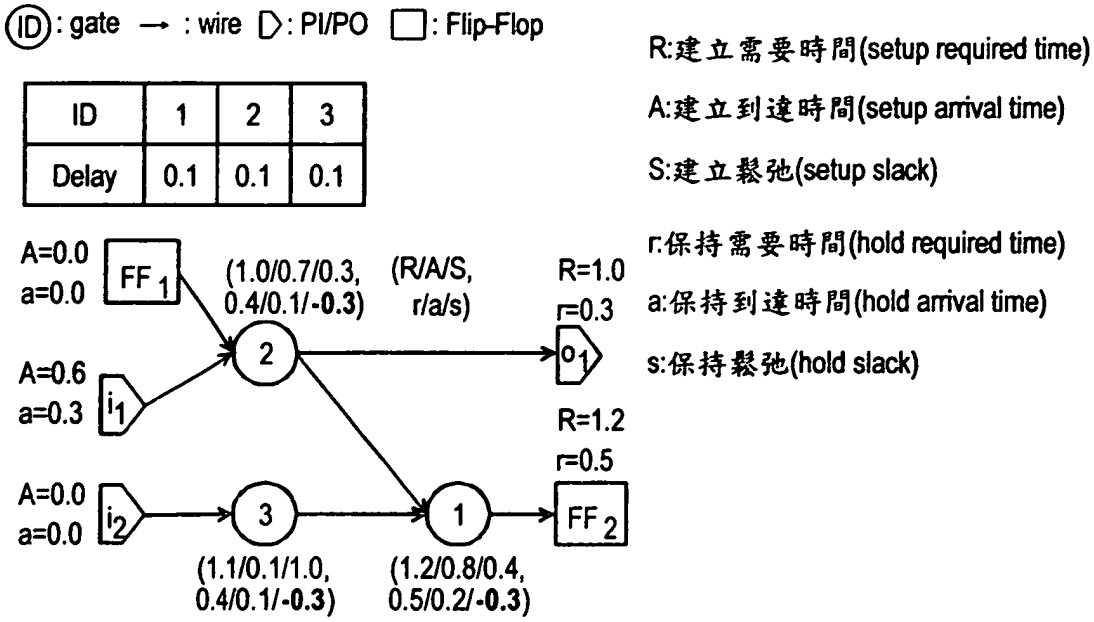
(ID): gate → : wire D: PI/PO □: Flip-Flop
R:建立需要時間(setup required time)
A:建立到達時間(setup arrival time)
S:建立鬆弛(setup slack)
r:保持需要時間(hold required time)
a:保持到達時間(hold arrival time)
s:保持鬆弛(hold slack)
ID 1 2 3
Delay 0.1 0.1 0.1
A=0.0
a=0.0
FF1
(1.0/0.7/0.3,
0.4/0.1/-0.3)
(R/A/S,
r/a/s)
R=1.0
r=0.3
A=0.6
a=0.3
i1
2
o1
R=1.2
r=0.5
A=0.0
a=0.0
i2
3
1
FF2
(1.1/0.1/1.0,
0.4/0.1/-0.3)
(1.2/0.8/0.4,
0.5/0.2/-0.3)

圖 3A

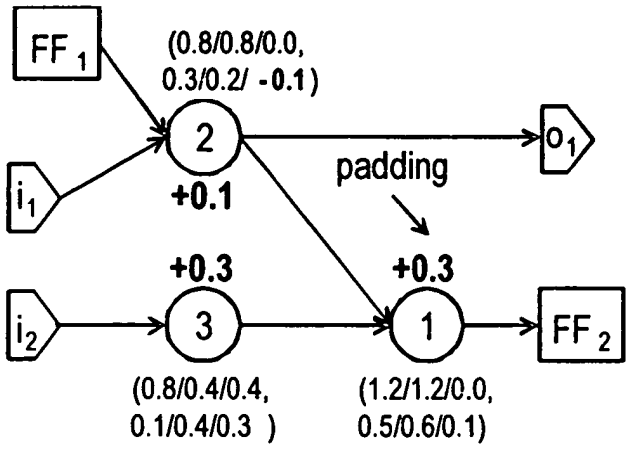
FF1
(0.8/0.8/0.0,
0.3/0.2/ -0.1)
2
o1
padding
i1
+0.1
+0.3
+0.3
i2
3
1
FF2
(0.8/0.4/0.4,
0.1/0.4/0.3)
(1.2/1.2/0.0,
0.5/0.6/0.1)

圖 3B

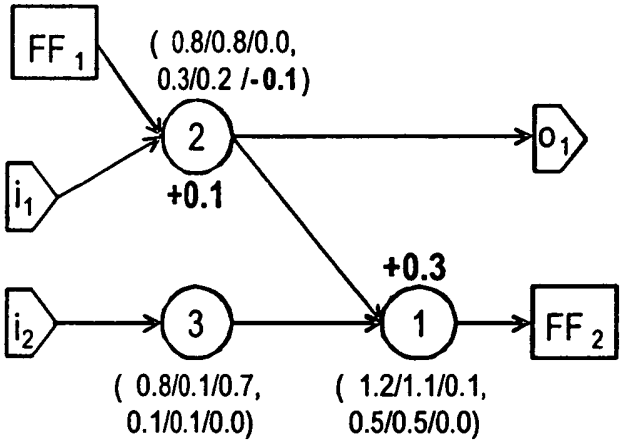
FF 1
(0.8/0.8/0.0,
0.3/0.2 /-0.1)
2
o1
i1
+0.1
+0.3
i2
3
1
FF 2
(0.8/0.1/0.7,
0.1/0.1/0.0)
(1.2/1.1/0.1,
0.5/0.5/0.0)

圖 3C

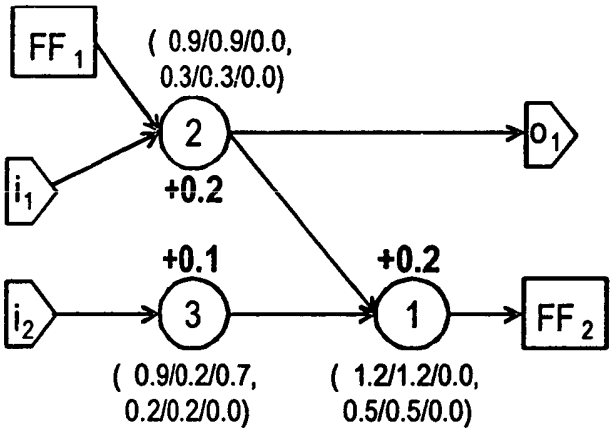
FF 1
(0.9/0.9/0.0,
0.3/0.3/0.0)
2
o1
i1
+0.2
+0.1
+0.2
i2
3
1
FF 2
(0.9/0.2/0.7,
0.2/0.2/0.0)
(1.2/1.2/0.0,
0.5/0.5/0.0)

圖 3D

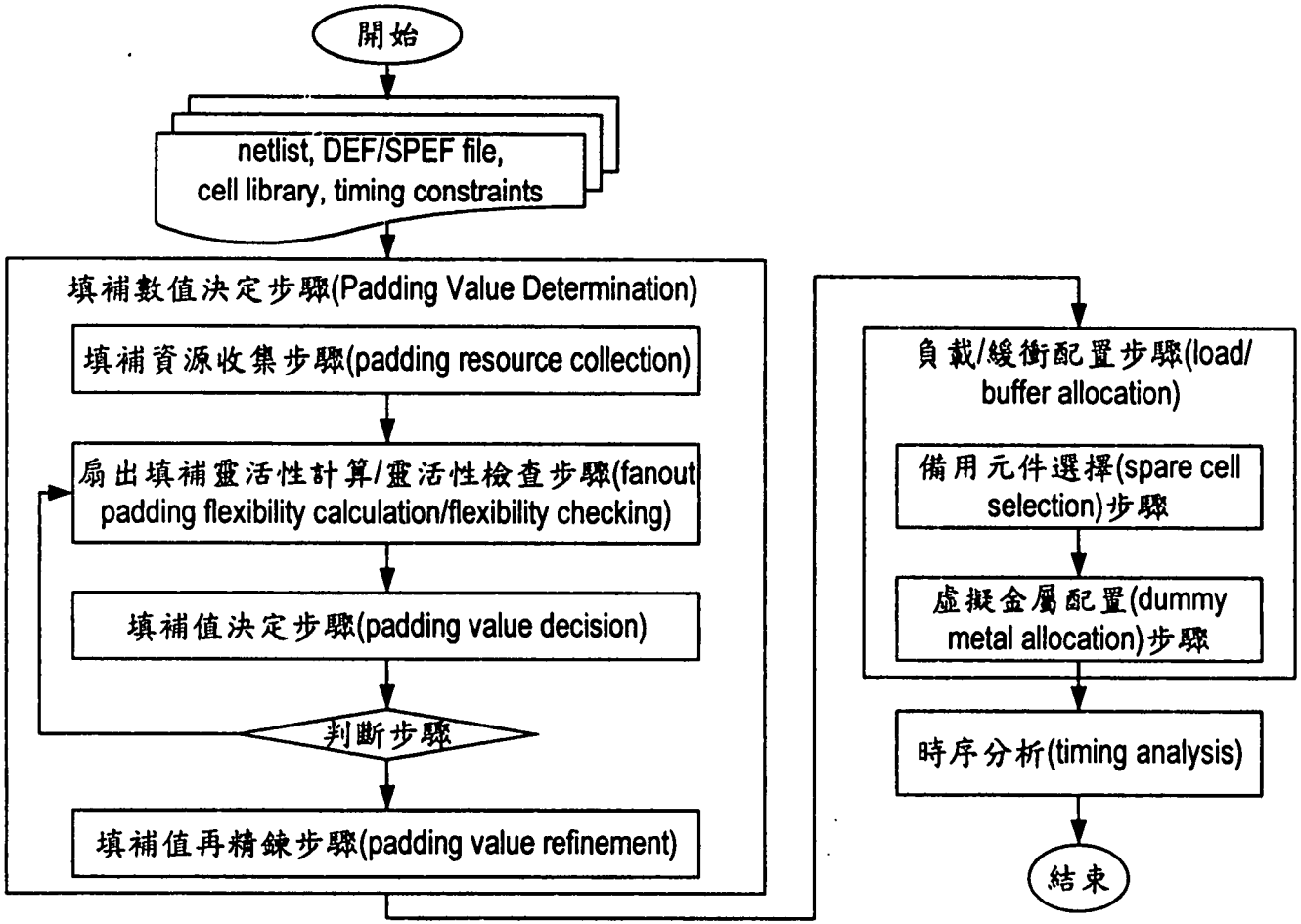
開始
netlist, DEF/SPEF file,
cell library, timing constraints
填補數值決定步驟(Padding Value Determination)
填補資源收集步驟(padding resource collection)
扇出填補靈活性計算/靈活性檢查步驟(fanout padding flexibility calculation/flexibility checking)
填補值決定步驟(padding value decision)
判斷步驟
填補值再精鍊步驟(padding value refinement)
負載/緩衝配置步驟(load/buffer allocation)
備用元件選擇(spare cell selection)步驟
虛擬金屬配置(dummy metal allocation)步驟
時序分析(timing analysis)
結束

圖 4

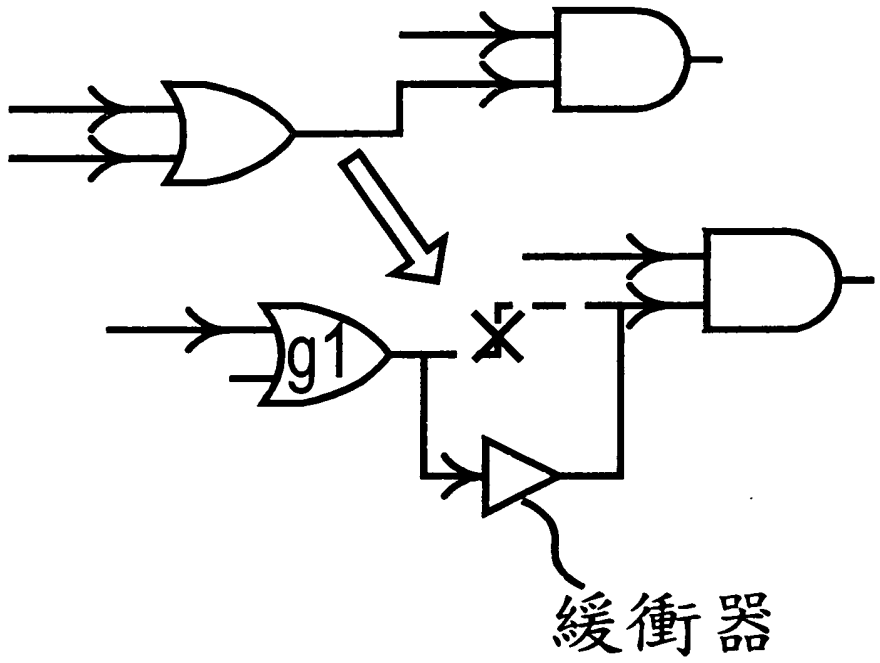

圖 5A

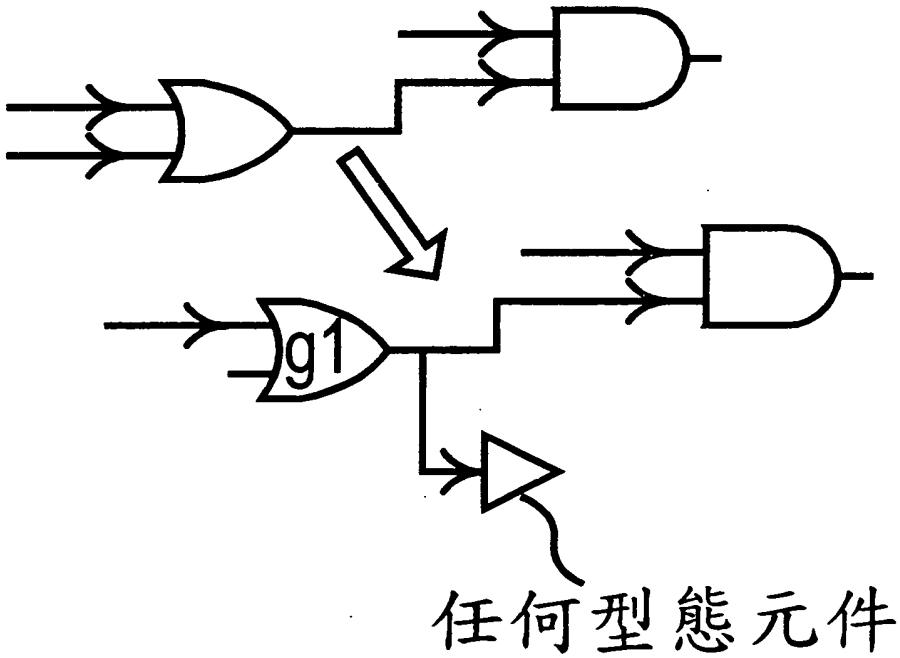

圖 5B

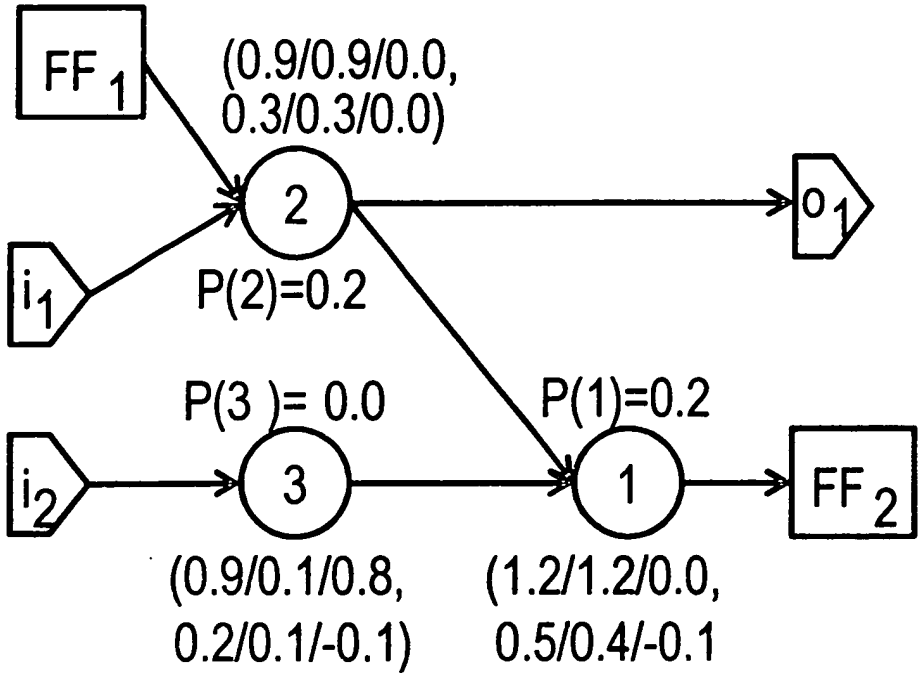
FF 1
(0.9/0.9/0.0,
0.3/0.3/0.0)
2
o 1
i 1
P(2)=0.2
P(3)= 0.0
P(1)=0.2
i 2
3
1
FF 2
(0.9/0.1/0.8,
0.2/0.1/-0.1)
(1.2/1.2/0.0,
0.5/0.4/-0.1

圖 6A

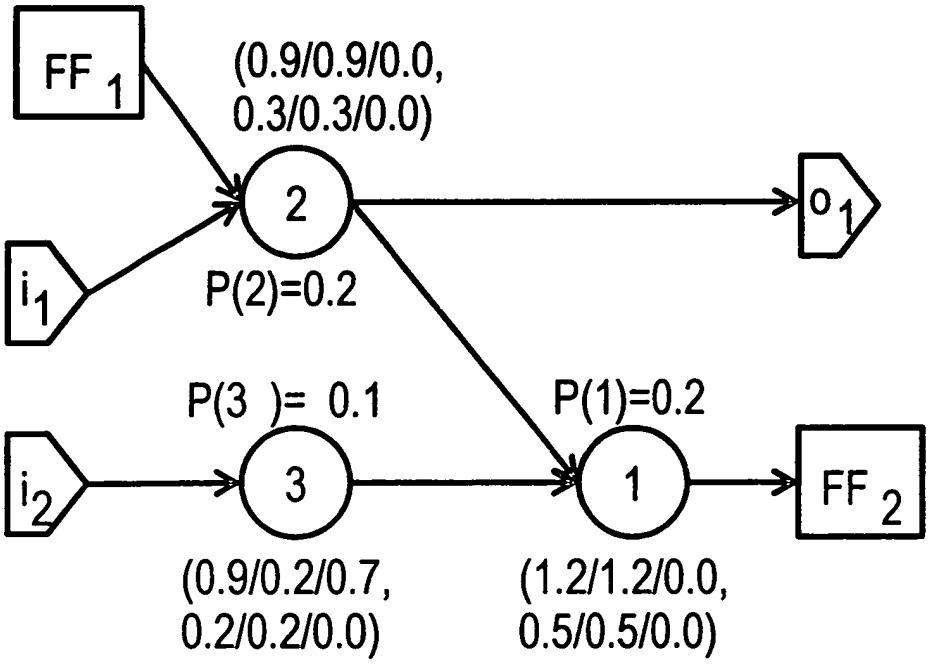
FF 1
(0.9/0.9/0.0,
0.3/0.3/0.0)
2
o 1
i 1
P(2)=0.2
P(3)= 0.1
P(1)=0.2
i 2
3
1
FF 2
(0.9/0.2/0.7,
0.2/0.2/0.0)
(1.2/1.2/0.0,
0.5/0.5/0.0)

圖 6B

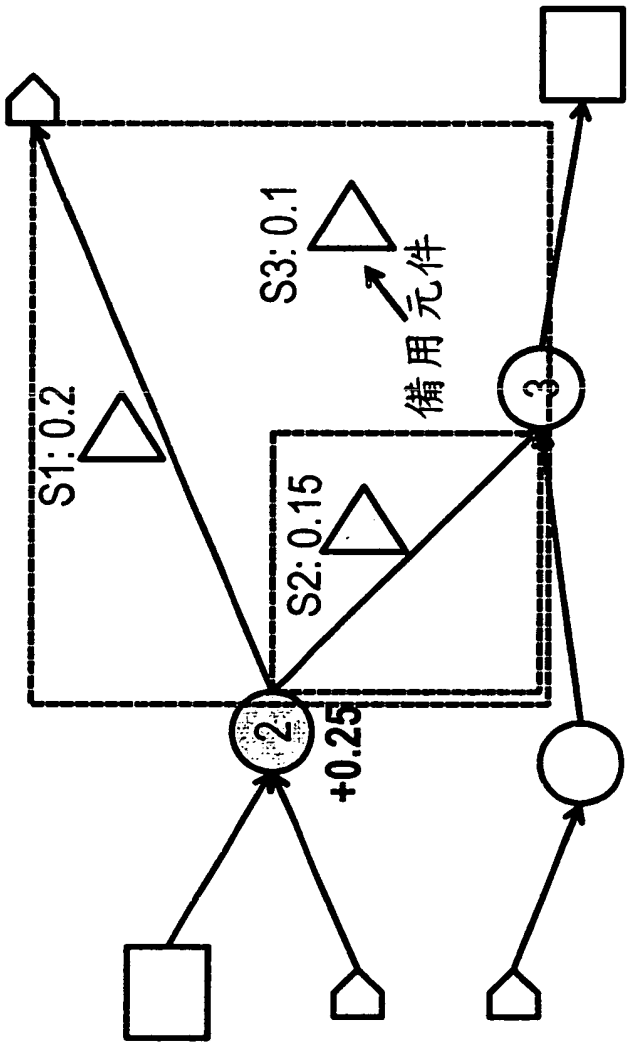

圖 7

		填補數值(padding value)					
		0.00	0.05	0.10	0.15	0.20	0.25
備用元件 (spare cells)	∅	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅
	$\{s_1\}$	0.00 ∅	0.00 ∅	0.00 ∅	0.00 ∅	0.20 $\{s_1\}$	0.20 $\{s_1\}$
	$\{s_1, s_2\}$	0.00 ∅	0.0 ∅	0.00 ∅	0.15 $\{s_2\}$	0.20 $\{s_1\}$	0.20 $\{s_1\}$
	$\{s_1, s_2, s_3\}$	0.00 ∅	0.0 ∅	0.10 $\{s_3\}$	0.15 $\{s_2\}$	0.20 $\{s_1\}$	0.25 $\{s_2, s_3\}$

圖 8

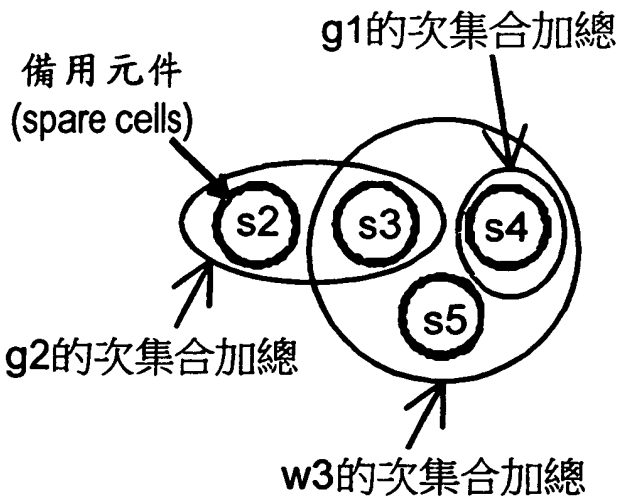
g1的次集合加總
備用元件
(spare cells)
s2
s3
s4
s5
g2的次集合加總
w3的次集合加總

圖 9

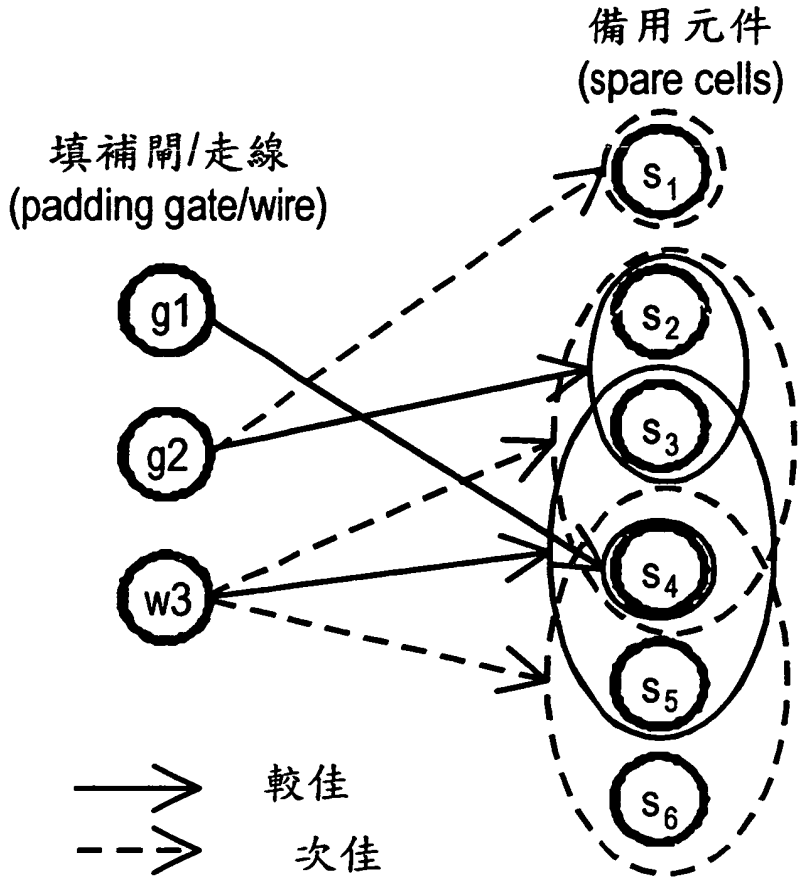
備用元件
(spare cells)
填補閘/走線
(padding gate/wire)
g1
g2
w3
s1
s2
s3
s4
s5
s6
較佳
次佳

圖 10

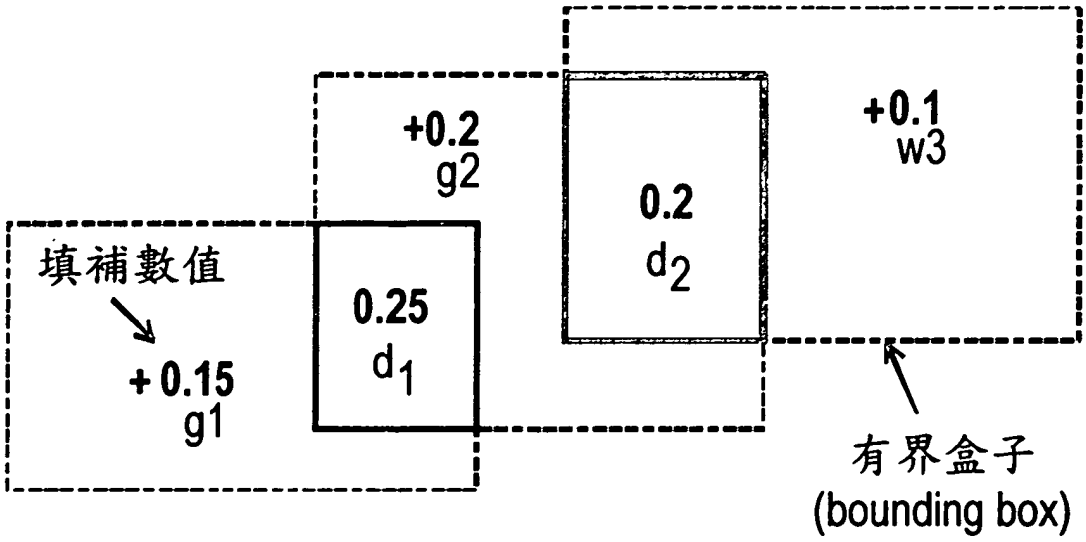
+0.2
g2
0.2
d2
+0.1
w3
填補數值
0.25
d1
+0.15
g1
有界盒子
(bounding box)

圖 11

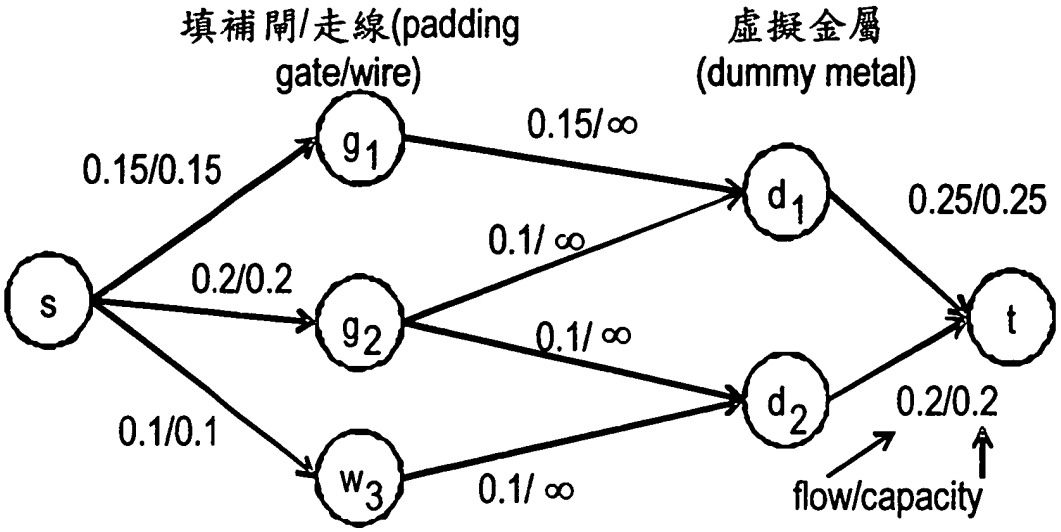
填補閘/走線(padding gate/wire)
虛擬金屬
(dummy metal)
0.15/0.15
0.15/∞
0.25/0.25
0.1/∞
0.2/0.2
0.1/∞
0.1/0.1
0.2/0.2
0.1/∞
flow/capacity
s
g1
g2
w3
d1
d2
t

圖 12

Circuit	#Gate	#FF	#SFF	Conservative clock period (ns)	THS (ps)
s1196	301	19	1	1.0	152.5
s1423	486	74	45	1.0	4,916.9
s5378	739	162	37	1.0	3,852.8
s9234	555	132	24	1.0	1,929.0
s13207	748	213	14	1.0	371.2
s15850	428	128	29	1.0	2,114.6
s38584	7,890	1,159	194	3.4	108,759.0
des_perf	51,349	8,808	1,190	2.9	583,579.0
b19	72,872	5,541	2,737	3.8	1,481,800.0

圖 13

Circuit	LP				Greedy 1: Largest setup first [12]				
	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	Runtime (s)	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	#Ite.	Runtime (s)
s1196	0.0	0.0	152.5	0.03	0.0	0.0	338.4	5	0.02
s1423	0.0	0.0	4,126.8	0.15	0.0	43.5	4,968.7	56	0.45
s5378	0.0	0.0	3,661.3	0.06	0.0	79.3	4,678.1	60	0.79
s9234	0.0	0.0	1,459.6	0.05	0.0	12.0	1,878.3	33	0.30
s13207	0.0	0.0	371.2	0.03	0.0	34.5	762.3	22	0.21
s15850	0.0	0.0	1,305.1	0.03	0.0	0.0	1,662.8	43	0.25
s38584	0.0	0.0	108,476.0	6.96	0.0	0.0	225,026.0	780	115.19
des_perf	0.0	0.0	583,579.0	60.59	0.0	19,270.5	795,022.0	3,414	6,401.86
b19	NA	NA	NA	timeout	0.0	108,078.0	6,128,710.0	21,902	37,473.30

圖 14A

Circuit	Greedy 2: Most path sharing first [11][12][13]					本發明: PushPull				
	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	#Ite.	Runtime (s)	TNS_1 (ps)	THS_1 (ps)	Padding delay (ps)	#Ite.	Runtime (s)
s1196	0.0	0.0	173.8	3	0.01	0.0	0.0	152.5	1	0.02
s1423	0.0	43.5	4,802.8	35	0.24	0.0	0.0	4,746.9	2	0.05
s5378	0.0	79.3	4,555.4	43	0.58	0.0	0.0	3,722.7	2	0.08
s9234	0.0	12.0	2,158.5	27	0.26	0.0	0.0	1,647.5	1	0.05
s13207	0.0	34.5	621.5	18	0.21	0.0	0.0	371.2	2	0.07
s15850	0.0	0.0	2,161.5	41	0.27	0.0	0.0	1,510.8	2	0.04
s38584	0.0	85.8	130,703.0	512	80.37	0.0	0.0	143,764.1	2	1.04
des_perf	0.0	3,864.1	595,088.0	2,257	4,202.46	0.0	0.0	583,579.0	2	9.46
b19	0.0	65,746.7	1,729,230.0	6,220	10,978.40	0.0	0.0	1,675,688.0	4	21.68

圖 14B

Circuit	LP+Mapping [6]			本發明: PushPull		
	TNS_2 (ps)	THS_2 (ps)	Runtime (s)	TNS_2 (ps)	THS_2 (ps)	Runtime (s)
s1196	0.0	0.3	0.03	0.0	0.0	0.33
s1423	0.0	826.4	0.09	0.0	0.0	0.43
s5378	0.0	302.3	0.05	0.0	0.0	0.54
s9234	0.0	631.3	0.04	0.0	0.0	0.41
s13207	0.0	161.2	0.03	0.0	0.0	0.61
s15850	0.0	352.0	0.02	0.0	0.0	0.38
s38584	0.0	29,970.7	0.58	0.0	0.0	1.96
des_perf	0.0	51,146.0	41.99	0.0	0.0	12.36
b19	NA	NA	NA	0.0	0.0	43.33

圖 15