一種用於循序及組合模組設計之循序電路技術映射的新演算法†

A New Approach for Technology Mapping of Sequential Circuits in Sequential and Combinational Modules Designs

謝財明
Hsieh Tsai-Ming
hsieh@cchp01.cc.cycu.edu.tw

林志澤
Lin Chih-Tse
james@cad.ice.cycu.edu.tw

陳永福
Chen Yung-Fu
yfc@cad.ice.cycu.edu.tw

中原大學資訊工程研究所
中壢市普仁里 22 號
Department of Information and Computer Engineering, Chung-Yuan Christian University
22, Pu Jen Li, Chung Li, Taiwan, R.O.C.

摘 要

在本篇論文中，我們提出了一個新演算法，可適用於不同的 FPGA 架構。論文中，將此方法應用在 Actel 的 Mux-based 架構，經由實驗結果得知，我們的新演算法能達到模組數的最佳化，同時在時間複雜度上比[12][2]所提的方法有顯著的改進。本演算法更可應用到具有迴路的循序線路。

關鍵字：多工器為基礎的現場可程式閘陣列，循序電路，技術映射，時序重置，線路複製，反向邊，循序模組(S 模組)，組合模組(C 模組)。

Abstract

In this paper, we proposed a new algorithm which is suitable for different FPGA architectures, and we apply this algorithm to Actel's MUX-based architecture. Extensive experiments show that our algorithm works very well. A solution with minimum number of module can be obtained with a lower time complexity then that of existed algorithms[12][2]. The proposed algorithm can be applied to sequential circuits with loop.

Key Words: MUX-based FPGAs, sequential circuits, technology mapping, retiming, replication, back edge, sequential module(S module), combinational module(C module).

一、前 言

在近年來的數位線路設計上，現場可程式閘陣列(Field Programmable Gate Array)，簡稱 FPGA， 扮演了一個很重要的角色。由於它方便修改的特性符合電子工業所必須具有的快速生產期，因此，FPGA 在電路的設計上，用來當產品設計的雛型(Prototype)，是一個相當實際的架構。比起 MPGA (Mask Programmable Gate Array)的架構來說，FPGA 的好處有：較低的製作成本及較短的製作時間。此二項因素對於變化日益快速的資訊工業來說，是極重要的目標也是極重大的改變。因此，採用 FPGA 來設計數位電路，不僅可以獲得相當的好處，也是未來設計電路必行的趨勢。

除此之外，現行一些設計電路的電腦輔助設計工具(CAD TOOLS)，其介面及流程都相當簡單而且容易學習，因此，在人員訓練的成本上，亦節省下大筆費用。有了優秀的硬體架構後，當然也必須要有良好的設計軟體來加以輔佐， 如此更能相得益彰。

在 CAD TOOL 的系統當中，其流程大致上分為兩大部份：(1)邏輯設計(logical design)及(2)實體設計(Physical design),邏輯設計包含了有邏輯合成(logical synthesis)及技術映射(technology mapping)等步驟，而實體設計包含了有置放(Placement)及繞線(Routing)等步驟。在本篇論文中，將探討在此流程中的一些技術映射(Technology mapping)演算法。這些演算法的目的是在將設計者用硬體描述語言(Hardware description Language)或用其他的型式所設計出來的電路，對其做映射及最佳化。而通常最佳化的目標大致可分為以下幾類：(1)延遲(delay)的最佳化 (2)面積(Area)的最佳化(3)增加可繞度等等。

1、 就延遲的最佳化來說，也就是要縮短電路的時序週期，相關的論文有 R. Murgai 等人所提的方法[18][17], K. C. Chen [3]，P. Pan & C. L. Liu[20][21]，J. Cong , Y. Ding 等人所提之以圖形理論為基礎之演算法 [7][3][4][5]以及 C. E. Leiserson & J. B. Saxe[15][14]採用的時序重置(retiming)的方法。

2、 就面積的最佳化來說，也就是將電路中所包含的 FPGA 個數減少。在 LUT 方面，其相關的論文則包括有 R. Murgai 等人提出的 Mis-fpga [19]和[16]，及 R. J. Frances 等人提出的 Chortle[10]及 Chortle- crf[11] , K. Karplus 所提之 Xmap[13]，及 N. S. Woo 所提之以 edge visability 為基礎之方法[23]。在 MUX-based 方面，有 D. Kagaris & S. Tragoudas[12]所用的花費流量網路(cost flow network)的方法，求得最小模組數，複雜度為多項式時間及 Y. P. Chen & D. F. Wong[2]所用的 ILP 方法，求得最小模組數，複雜度為指數時間。另外還有 Mis-fpga[19]則是 LUT 和 MUX 兩種架構上均可適用的。

† 本論文之研究，接受國科會經費補助，計劃編號：NSC87-215-E033-004 。

3、就增加可繞度來說，相關的論文有 N. Bat & D. Hil [1]及 Schlag[22]等所提之方法。

另外還有一些研究是結合上述三種目標為考量，以期求得一多個目標間的平衡(Trade off) [9][6][8]。這些問題在某些特定問題上(如：tree circuit)均可獲得最佳解，而值得一提的是 Cong 及 Ding 的 FlowMap 方法[7]。對一 k-bounded 的組合電路，在單位延遲模式(Unit delay model)下，即每一 K-LUT 具有一固定延遲而與其所實現之函數無關，FlowMap 能求得一個具有最小延遲之最佳解。

在本篇論文中，將提出一種新的演算法，並且適用於不同的 FPGA 架構上，包括有 LUT-based 及 MUX(Multiplexor)-based 等架構。但本論文探討的則是在 MUX-based 架構。在[2]中 Y. P. Chen 及 D. F. Wang 提出一整數線性規劃的方法，求得模組數的最佳解，而 D. Kagaris & S. Tragoudas 所提出的一時間複雜度為多項式時間複雜度(Polynomial time complexity)的演算法[12]。此演算法是採用了一種花費流量網路(cost flow-network)的觀念，再加上 Leiserson 和 Saxe 的時序重置(retiming)，然後去求得一個最小數量的循序及組合模組數 (minimum Sequential & Combinational Modules)的電路。本論文中所提的新演算法，其在時間複雜度上又再加以改進，以期用更小的時間複雜度，來達成模組數的最小化。同時本方法也將可以處理有回路(Loop)存在的循序電路，更進一步的，巢狀回路(nested loop)的循序電路也將獲得解決。在本方法的演算法中，我們也將會用到時序重置(retiming)，線路複製(re-plication)等方法，其詳細的方法及步驟，我們將在後面做詳細的介紹。

二、問題描述

在本節中，我們將對問題做描述。首先我們將電路轉換成圖形(Graph)的型式，如 $G = (V \cup I \cup O, E)$，其中的集合 V 的每一元素代表的是在電路中的每一個邏輯模組(Logic Module)，I 代表的是主要輸入端之集合，O 則是主要輸出端之集合。假如一個模組 u 的主要輸入端或是輸出端散出到 k 個其他的模組或是主要輸出端 $v_1, v_2, \ldots v_k$，我們則把它描述成從 u 連向外的 k 個邊$(u,v_1),(u,v_2),\ldots,(u,v_k)$。每一個邊上則有權重 $w(u,v)$，權重代表的是原始電路的設計

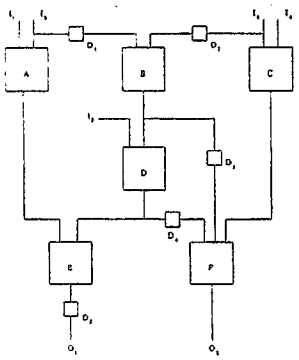

Fig.1 原始電路

中，點 u 和點 v 之間連接經過了幾個正反器(flip-flop)。如 Fig.1 的原始電路經過轉換成圖形則變成了 Fig.2。

假設在多重散出(multi-fanout)的情形下，經過時序重置後，一個點 $u \in V \cup I$ 在它的向外邊上有時序重置的權重 $w'(u,v_1)$、$w'(u,v_2),\ldots,w'(u,v_k)$，經由暫存器分享(Register Sharing)，我們可以知道實際所需要的暫存器個數是 $\max\{ w'(u,v_i), i=1,2,\ldots,k\}$。假如點 u ($u \in V$) 是一個模組，然後我們則要分辨下列情況：假如對每一個 $w'(u,v_i)$，$1 \le i \le k$，如果 $w'(u,v_i) \ge 1$，就可以將一個暫存器組合到這個模組 u，變成一個循序模組，然後所需要的模組數便成為 1 個循序模組再加上 $(\max\{w'(u,v_i)\}-1)$個循序模組。如此一來，在這個情況下，對點 u 來說，它的總共模組數則可以用下式來表示：$M_u = 1 + (\max\{ w'(u,v_i) \} - 1) = \max\{ w'(u,v_i) \}$。

另外一種情況，則是至少有一 $w'(u,v_i)$為 0，因此這樣就沒有暫存器可以被組合到這個模組中，而總共所需要的模組數則是 1 個組合模組(映射到 u 的)再加上 $\max\{ w'(u,v_i) \}$個循序模組(映射到各邊上的暫存器)，也就是如下式所列 $M_u = 1 + \max\{w'(u,v_i)\}$。若是 u 是一個主要輸入端($u \in I$)，當然也是不能將暫存器跟它組合，所以總共的模組數則是 $M_u = \max\{w'(u,v_i)\}$。

我們最主要的目的是經由時序重置的方法後，我們對每一個點 $u \in V \cup I$ 所得到總共模組數$\sum_{u \in V} M_u$希望是最小的。藉由這樣的問題陳述，我們提出了新演算法，它將針對循序電路的時序重置以及線路複製等方法，提出一個加以組合改進的方法，除了上述的目標加以達成外，我們的方法在時間複雜度上也改進了許多。

三、新演算法的方法

在本節中，我們將介紹本論文所提出的新演算法。新演算法的方法及步驟大致如下：

*第一步驟：*將電路轉換成 DAG(Directed Acyclic Graph)，以便我們的演算法在 DAG 的結構上操作。

*第二步驟：*對 DAG 做出拓樸順序(topological order)，依照拓樸順序將暫存器由主要輸出端或虛擬

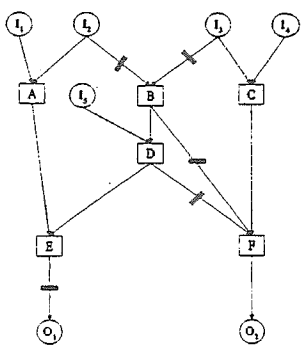

Fig.2 轉換成圖形

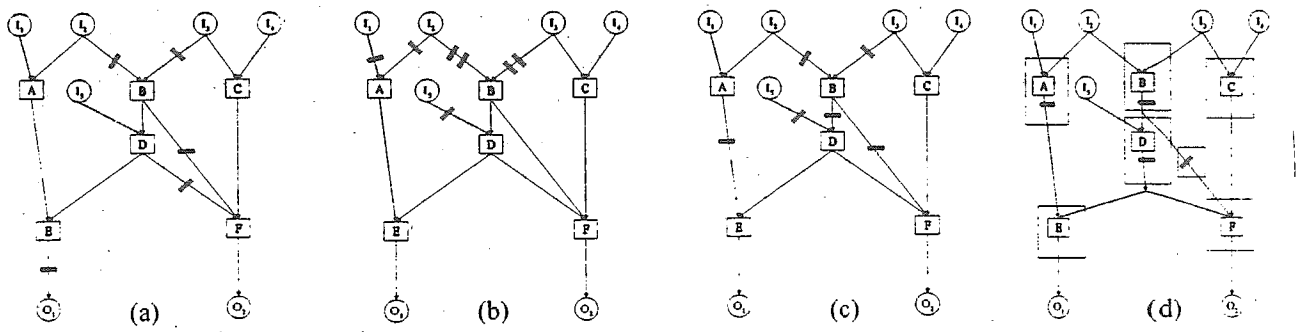

Fig.3 (a)原始電路, (b)經由 Backward Retiming 所有的暫存器均推至主要輸入端, (c) Forward Retiming 之過程, (d)最後的映射結果。

主要輸出端漸漸推向主要輸入端或是虛擬主要輸入端，此過程為"後向時序重置"(backward retiming)。在此過程中，若是遇到有多重散出的情形時，而且散出邊上的暫存器數目不相同時，我們就必須要採用線路複製(replication)的方法，才能將暫存器都順利的推向主要輸入端或是虛擬主要輸入端。

*第三步驟：*接著我們會將第二步驟的結果寫成測試電路(benchmark)的檔案型式，我們採用的是Blif(Berkeley Logic Interchange Format)的檔案格式。此 Blif 檔案再經過 SIS 的 Act_map 後，即是一個稍微經過最佳化的電路。此時電路中的暫存器都還是在主要輸入端或是虛擬主要輸入端上。只是對組合電路的部份做最佳化罷了。

*第四步驟：*然後我們再將暫存器依照 BFS(Breadth First Search)的造訪(traverse)順序來將暫存器推向主要輸出端或是虛擬主要輸出端，此過程為"前向時序重置(forward retiming)"。然而在此過程中，我們不做任何線路複製(replication)，只是做時序重置。然後每對一個模組做了時序重置後，我們則試著做合併(merge) 的動作，以便將線路複製所產生多出來的模組儘量的再合併回去，並且將其下面的一個暫存器和這個模組組合成一個循序模組。依此動作我們則可以將暫存器放到適當的位置，使得模組數可以得到最佳的改善。如果說這一個步驟還沒有辦法求得更佳的模組數，我們則可以先不要組合成循序模組，先儘量將暫存器前向時序重置到主要輸出端或虛擬主要輸出端，然後再做一次後向時序重置，將模組和暫存器組合成一個循序模組。如此便可以得到一個最佳的解。我們將以下列例子說明我的演算法。

[*例一*] 如 Fig.3(a)所示圓圈代表 I/O，方框則代表 Logic Module，▮ 代表暫存器。而其中的每一個模組有可能代表著單一個閘(gate)，也可能代表著包含多個閘。首先我們的新演算法對於這樣的圖形，則是把所有的暫存器都以後向時序重置的方法將其全部推到主要輸入端或是虛擬主要輸入端，但是其中若是碰到多重散出的情形時，若其散出的暫存器的個數有不相同的話，我們就採用線路複製的方法來使得暫存器可以被順利推到主要輸入端或虛擬主要輸入端。其結果如 Fig.3(b)所示。而在此例中並沒有用到線路複製的方法。在將暫存器都推到主要輸入端或虛擬輸入端之後，我們對每個邏輯模組(Logic Module)依照其 BFS 的追蹤順序，若是它的所有散入都有暫存器存在，我們就將其最小共同擁有數量的暫存器做前向的時序重置，而在每一個前向時序重置的動作之後，我們就將這個模組和其下面的一個暫存器結合為一個循序模組(S. module)。其過程和結果則如 Fig.3(c)至 Fig.3(d)所示。總共是 7 個模組，4 個循序模組，3 個組合模組。其所獲得的結果是和 D. Kagaris & S. Tragoudas 的方法是相同的。但是在時間複雜度上，我們的方法則是降低了一些。從 O(nm) 降到 O(nk)，其中的 n 代表電路中模組的個數，m 代表 n 個模組中彼此間的連接，k 代表對所有 n 個模組中，其最大的散出數，k 通常遠小於 m。□

[*例二*] 若有一電路如 Fig.4(a) 所示，首先我們將電路中所有的暫存器都以"後向時序重置"的方法，都推到主要輸入端或虛擬輸入端上，其結果如 Fig.4(b) 所示。而在這個例子中我們將必需要用到線路複置(replication)的方法，才能將具有多重散出狀況時的暫存器往上推。在經過了後向時序重置的步驟後，我們將結果再經由 SIS 系統來做 Act_map，所得到的結果則如 Fig.4(c)所示。最後我們再進行前向時序重置。在做前向時序重置時，我們可以有兩種不同的方式可以進行，第一種就是如例一中所用的方法，在進行 BFS 的追蹤時，同時進行前向時序重置，接著做合併的動作，在完成單一個模組的時序重置及合併後，就把一個暫存器和一個模組，組合成一個循序模組，然後進行下一個模組的時序重置。第二種方法則是進行前向時序重置時，先不要組合成循序模組，先儘量的把暫存器都推到靠近主要輸出端或虛擬輸出端的地方。但是我們則不用複製線路的方法，只要符合時序重置的條件時，我們就進行前向時序重置。等所有暫存器都推到靠近主要輸出端或虛擬主要輸出端，我們再進行一次後向時序重置，這時候就將一個暫存器和一個模組，組合成一個循序模組，再做後向時序重置。□

由例二我們可以看出來，經過線路複製和合併，使得在電路中，暫存器可以移動、放置的地方機會增加了許多，因為這樣的原因，我們所得到的模組

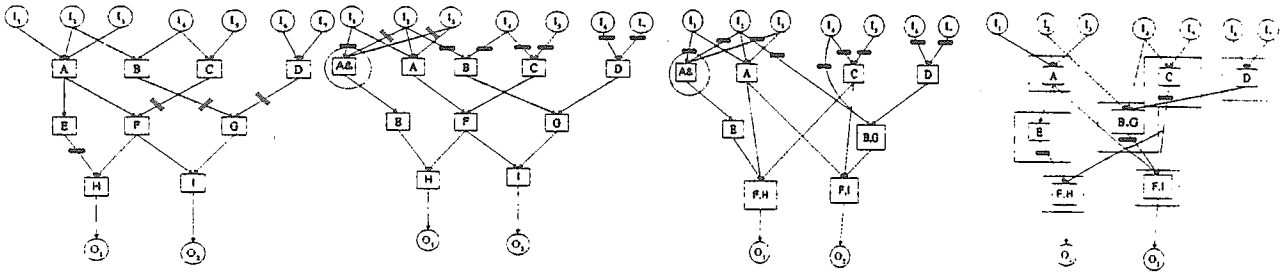

Fig.4 (a)原始電路, (b)暫存器均推至主要輸入端, (c)經過 SIS 的 Act_map, (d)最後映射結果。

數則可以有所減少。Fig.4(d)為本演算法最後映射的結果。如原本由 D. Kagaris & S. Tragoudas 的方法，需要 3 個循序模組，5 個組合模組，總共要 8 個模組。但是用本論文的新演算法所得的結果則是只要 3 個循序模組，4 個組合模組，總共則是共要 7 個模組。因此在這個例子中，我們的新演算法則是獲得比較佳的結果。然而這其中最主要的原因在於我們有做了線路複製，將暫存器都推到了主要輸入端或虛擬主要輸入端的地方後，我們再把結果經過 SIS 做 Act_map 的最佳化，然後再重新安排暫存器的放置位置，如此便可以得到一個較優於 D. Kagaris & S. Tragoudas 的結果。

本演算法對於具有回路(loop)的電路例如 Fig.5(a)，則是先將在圖形中所謂的前向暫存器(forward FFs)的地方，即如 D1 的地方，先對其做後向時序重置，將暫存器推到後向暫存器或反向邊(backward FFs or backward edge)的地方即停止，然後再將反向邊的地方切斷，如下 Fig.5(b)至 Fig.5(c)所示。並且每切斷一個後向邊，即造一組虛擬的輸入端和輸出端，所得到的結果則如 Fig.5(d)所示。然後同樣的我們在將結果經過 SIS 的 Act_map 後，結果和 Fig.5(d)是相同的。接下來則是將暫存器重新分配。在 Fig.5(e)中，則是對電路做前向時序重置，最後把切斷的反向邊再重新的連接回來。最後所得到的結果則如 Fig.5(f)所示。因此這個電路所需要的模組數則為，2 個循序模組，2 個組合模組，總共則是 4 個模組。

由上述例子來看，我們的演算法在模組數的減少上，若是加上線路複製的方法之後，並且暫存器都推到主要輸入端或虛擬主要輸入端的話，我們所得到的結果在模組數的減少上是有明顯的改進，並且還能處理具有回路(loop)的複雜電路，甚至更複雜的巢狀回路(nested loop)電路也都將可以解決。因此我們就提出了例三來說明一下如何解決巢狀回路(nested loop)的問題。

[*例三*] 若有一電路如 Fig.6(a)所示，由於在切斷回路的方法上，我們採用的是一邊時序重置，一邊切斷回路。將暫存器都推往主要輸入端或是虛擬主要輸入端，推到反向邊的時候，我們就將反向邊切斷，並且將暫存器放置在虛擬主要輸入端。如此重覆直到沒有回路為止。過程及結果如 Fig.6(a)至 Fig.6(d)。接著就將暫存器再重新分配，進行前向時序重置，即如 Fig.6(e)所示。然而在進行前向時序重置時，就將暫存器和組合模組合併起來，如此我們就可以得到一個相當好的解。其結果如 Fig.6(f)所示。□

新演算法如下：

step 1： transfer the given circuits into a DAG and get the topological order of nodes in the DAG.

step 2： According the topological order of nodes, process each node :
    for each node *v* :
    if the fanouts of *v* have differents weights then do
        replication and backward retiming
    else do
        backward retiming
    /* After step 2, we got the circuit which has all registers on the primary or pseudo primary input lines. */

step 3: Write retiming result circuit obtained in step 2 to a BLIF file so that it can be used as the input of Act_map. Then Act_map maps the combinational part of the circuit.

step 4: Get the BFS traverse order, do forward retiming for each node according this order.

step 5: Merge every register immediately following a node (C module) to get a new S module. Findly we got the mapping solution.

四、時間複雜度的比較

我們知道新演算法的時間複雜度，最主要是決定於 step 2。因為在這個步驟中，做了取得拓樸順序(topological order),依此順序來進行後向時序重置，然而在做後向時序重置的同時，必需同時判斷是否進行線路複製。所以當最差的情況(worst case)時，我們要對每一個點做線路複製，並且要做時序重置。至於要複製多少個模組，則決定在其散出邊上的暫存器個數的種類多寡(k)。例如在一線路上某一模組其散出邊上有 0 個暫存器，有 1 個暫存器，有 3 個暫存器的話，則暫存器個數的種類則有三種，所以對這個模組來說，總共要複製兩個相同的模組，加上原本的共三個。所以其時間複雜度則為 $O(nk)$。其中的 n 代表的是模組的個數，k 則是代表暫存器個數的種類。演算法中若是中途要經過 SIS 的 Act_map 的話，時間複雜度則必需要決定於 SIS 中的 Act_map 的時間複雜度。而我們的方法在時間複雜度上比起 D. Kagaris & S. Tragoudas 的方法其時間複雜度為

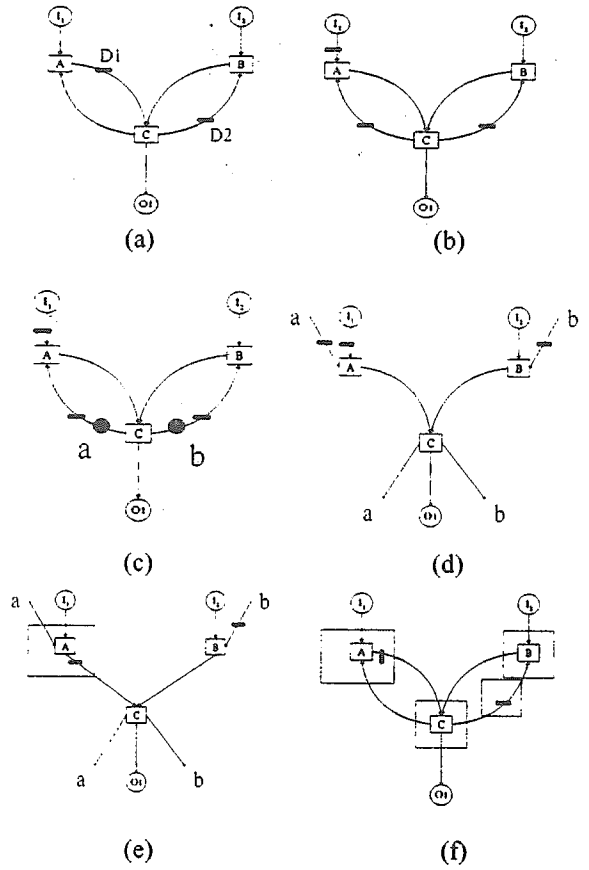

Fig.5 (a)原始電路, (b)對點 A 做後向時序重置, (c) 推到反向邊即停止, (d)切斷迴路並造成虛擬輸出入端, (e) 做前向時序重置並組合暫存器, (f)最後映射結果。

O(nm)來說，因為 k 通常小於 m，因此我們的時間複雜度是優於他們的。

由以上的分析看來，若是我們要以時間複雜度為比較的話，我們的結果在模組數上，可以得到最小的模組數解。這跟先前的研究是相同的模組數。若是我們要在模組數上面有所突破，我們則必須要透過一些其他的技術映射方法，藉由這些技術映射的方法來考慮其函數性，將所需要的模組數減少。由於我們的方法會產生一個夠大的組合電路，因此在考慮函數性的映射或是減少模組數上面，相信一定會有良好的結果。

五、實驗結果及比較

在本章中我們將列出我們的一些實驗的結果。藉由實驗結果的觀察，我們可以得知，新演算法執行後會有不錯的結果。我們已經將演算法以 C 語言撰寫，作業系統為 SUN OS，執行的機器為 SUN SPARC 10。輸入則是 MCNC[24]的測試電路，輸出則是計算出所需要的時間及所需要的循序模組數和組合模組數。以下則是列表說明經過我們的新演算法所做出來的結果。其中 S.代表循序模組，C.代表組合模組，# of module 則為總共的模組數(S.+C.)。經由與原電路的比較或跟經過 SIS 的 Act_map 比較，我們可以知道已將暫存器組合到組合模組中成為循序模組了。

在 Table.1 中，我們分成了四個欄位來比較。第(1)欄代表原始電路，經過 Act_map 處理後得到第(2)欄，第(2)欄結果經過新演算法的 back_FFs+forward_FFs 處理後得到第(3)欄，第(2)欄結果依序經過 back_FFs、Act_map、forward_FFs 處理後得到第(4)欄，第(2)欄結果經過 Act_map 兩次處理後，得到第(5)欄。

六、結論及未來展望

在本篇論文中我們提出了一個新演算法，對 Actel 的 Act 系列的架構電路而言此演算法以 $O(nk)$的時間複雜度，就可以求得最佳化的模組數。由實驗結果 Table.1 知，本論文不止在時間複雜度上有很好的結果，且又能進一步減少原線路所需模組數。在 step1、step2 中我們藉由時序重置、線路複製、切斷回路等方法，我們可以得到一較完整的組合電路。若在 step3 中先將所得之組合電路映射到以查表式 FPGA(LUT-base FPGA)為基礎的電路，則本演算法所述觀念可直接應用於查表式 FPGA 技術映射問題。

七、參考文獻

[1] N. Bhat and D. Hill, "Routable Technology Mapping for FPGAs," *ACM/SIGDA Workshop on FPGAs*, pp. 143-148, 1992.

[2] Y. P. Chen and D. F. Wong, "On Retiming for FPGA Logic Minimization," *IEEE Intl. Conf. on Computer Design*, pp.394-397,1994.

[3] K. C. Chen, J. Cong, Y. Ding, A. B. Kahng, and P. Trajmar," DAG-MAP: Graph-Based FPGA Technology Mapping for Delay Optimization," *IEEE Design and Test of Computer*, Vol. 10, pp. 7-20, 1992.

[4] J. Cong and C. Wu, "An Improved Algorithm for Performance Optimal Technology Mapping with Retiming in Lookup-Table Based FPGAs," *UCLA-CSD 960012, Technique Report*. 1996.

[5] J. Cong and Ding, "On Nominal Delay Minimization in LUT-Based FPGA Technology Mapping," *Integration, the VLSI Journal*, Vol. 18, pp. 73-94, 1994.

[6] J. Cong and Y. Ding, "Beyond the Combinatorial Limit in Depth Minimization for LUT-Based FPGA Designs," *Proc. IEEE Intl. Conf. on Computer-Aided Design*, pp. 110-114, 1993.

[7] J. Cong and Y. Ding," Flow Map: An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table Based FPGA Designs," *IEEE Trans. on Computer-Aided Design*, Vol. 13, pp. 1-11, 1994.

[8] J. Cong and Y. Ding," On Area/Depth Trade-off in LUT-based FPGA Technology Mapping," *IEEE Trans. on VLSI Systems*, Vol 2, pp. 137-148, 1994.

[9] J. Cong and Y.-Y. Hwang, "Simultaneous Depth and Area Minimization in LUT-based FPGA Mapping," in *UCLA Computer Science Department Technical Report* CSD-950001, Los Angeles, CA(January 1995).

[10] R. J. Francis, J. Rose, and K. Chung, "Chortle: A Technology Mapping For Lookup Table-Based Filed Programmable Gate Arrays," *Proc. 27th ACM/IEEE Design Automation Conf.*, pp. 613-619, 1990.

[11] R. J. Francis, J. Rose, and Z. Vranesic, "Chortle-crf: Fast Technology Mapping for Lookup Table-Based FPGAs ," *Proc.*

28th ACM/IEEE Design Automation Conf., pp. 227-233, 1991.

[12] D. Kagaris and S. Tragoudas, " A fast Algorithm for Minimizing FPGA Combinational and Sequential Modules," ***ACM Trans. on Design Automation of Electronic Systems***, pp.341-351 ,vol.1,No.3,July,1996.

[13] K. Karplus, "Xmap: A Technology Mapper for Table-lookup FPGAs," ***Proc. 28th ACM/IEEE Design Automation Conf.***, pp. 240-243, 1991.

[14] C. E. Leiserson, and J. B. Saxe, "Retiming Synchronous Circuitry, " ***Algorithmica,*** Vol. 6, pp. 5-35 , 1991.

[15] C. E. Leiserson, F. M. Rose, and J. B. Saxe, "Optimizing Synchronous Circuitry by Retiming," ***Proc. 3rd Caltech Conf. on VLSI,*** pp. 87-116, 1983.

[16] R. Murgai, N. Shenoy, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Improved Logic Synthesis Algorithms for Table Look Up Architectures." ***Proc. IEEE Intl. Conf. on Computer-Aided Design,*** pp. 564-567, 1991.

[17] R. Murgai, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Performance Directed Synthesis for Table Look Up Programmable Gate Arrays," ***Proc. IEEE Intl. Conf. on Computer-Aided Design,*** pp. 572-575, 1991.

[18] R. Murgai, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Sequential Synthesis for Table Look Up Programmable Gate Arrays," ***Proc. 30th ACM/IEEE Design Automation Conf.***, pp. 224-229, 1993.

[19] R. Murgai, Y. Nishizaki, N. Shenoy, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Logic Synthesis for Programmable Gate Arrays," ***Proc. 27th ACM/IEEE Design Automation Conf.***, pp.620-625, 1990.

[20] P. Pan and C. L. Liu, "Optimal Clock Period FPGA Technology Mapping for Sequential Circuits." ***33th ACM/IEE Design Automation Conference***, pp. 720-725,1996.

[21] P. Pan and C. L. Liu, "Technology Mapping of Sequential Circuit for LUT-based FPGAs for Performance," ***ACM 4th Intl. Symposium on Field-Programmable Gate Arrays***," pp.58-64,1996.

[22] M. Schlag, J. Kong, and P. K. Chan,"Routability-Driven Technology Mapping for Lookup Table-based FPGA's," ***IEEE Trans. On Computer-Aided Design***, Vol. 13-26,1994

[23] N. S. Woo,"A Heuristic Method for FPGA Technology Mapping Based on the Edge Visibility,"Proc. ***28th ACM/IEEE Design Automation Conf.***, pp.248-251, 1991

[24] S. Yang, "Logic Synthesis and Optimization Benchmark User Guide Version 3.0", *Microelectronics Center of North Carolina, P.O. Box 12889, Research Triangle Park, NC 27709*, Jan. 15, 1991.

[25] 李志宏,"一種用於以查表式為基礎之 FPGA 循序電路設計中新的技術映射演算法", *中原大學資訊工程學系碩士學位論文*, 民國八十六年.

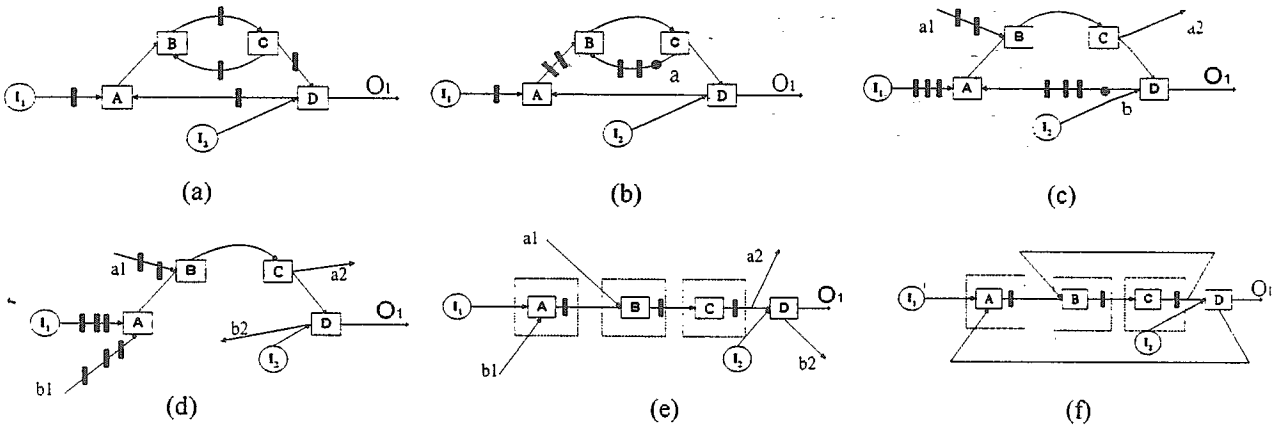

Fig.6 (a)原始電路, (b)對點 B 做後向時序重置, (c)對點 A 做後向時序重置, (d)切斷迴路並造成虛擬輸出入端, (e)前向時序重置結果, (f)連接切斷點。

	(1)	(2)	(3)	(4)	(5)
測試電路	Initial S.+C.= # of modules.	Act_map S.+C.= # of modules	Act_map, back_FFs, forward_FFs S.+C.= # of modules	Act_map, back_FFs Act_map, forward_FFs S+C.= # of modules	Act_map, Act_map S+C.= # of modules
S27.blif	3+10=13	3+7=10	3+4=7	3+4=7	3+7=10
Dk15.blif	2+27=29	2+27=29	2+25=27	2+25=27	2+27=29
DK14.blif	3+38=41	3+38=41	3+35=38	3+35=38	3+38=41
Shiftreg.blif	3+13=16	3+5=8	4+2=6	4+2=6	3+3=6
Test1.blif	5+6=11	5+5=10	4+3=7	4+3=7	5+5=10
Test2.blif	3+6=9	3+5=8	5+2=7	5+2=7	3+5=8
Test3.blif	3+9=12	3+6=9	3+4=7	3+4=7	3+6=9
Test4.blif	4+9=13	4+8=12	3+5=8	3+4=7	4+8=12
Total	26+118=144	26+101=127	27+80=107	27+79=106	26+99=125
comparison	1	0.882	0.743	0.736	0.868

Table.1 實驗結果及比較