Chung Yuan Journal
Vol. 26, No. 3, 1998, pp.71-76

具函數性考量之技術映射

林 佑 政*

中原大學電子研究所資訊組
台灣省中壢市普仁 22 號

謝 財 明**

中原大學資訊工程研究所
台灣省中壢市普仁 22 號

(Received: June 23, 1998; Accepted: August 27, 1998)

摘 要

在現場可程式化閘陣列設計流程中，技術映射(Technology Mapping)是一項重要的步驟。以往的研究，大多只是考量電路的結構性，不考量其函數性，利用圖形理論的方法來解決技術映射問題。不同於以往的研究，本論文提出同時考慮電路函數性與結構性的演算法。實驗結果證明，在同樣的輸入電路下，本演算法較以往的研究在 LUT 的個數上有 4.5%的改善、時間延遲方面也有約 1.4%的改善、而在 LUT 間的連線方面則有 3.8%的改善。

關鍵詞：現場可程式化閘陣列、技術映射、函數性考量。

壹、前 言

近年來，在數位線路設計中，現場可程式化閘陣列(Field Programmable Gate Array，FPGA)〔1〕〔2〕〔3〕〔28〕常被用以製作系統雛形。查表式現場可程式化邏輯閘陣列(Lookup Table based FPGA, LUT-based FPGA)中含有若干個 LUT。一個具有 k 個輸入之 LUT 可實踐任何有 k 個變數之布林函數(boolean function)稱爲 k-LUT。在組合電路設計方面，將原電路用若干個 LUT 加以實踐之過程，我們稱之爲技術映射(Technology Mapping)，此類問題依其最佳化目標的不同，大致可分爲三類：

第一類，以降低在映射後的等價線路所含 LUT 個數爲主要目標，如 Murgai 等人所提出之 MIS-pga〔21〕〔22〕、Frances 等所提出之 Chortle 及 Chortle-crf〔15〕〔16〕、Karplus 之 Xmap〔18〕、Woo 之 VISmap〔27〕。第二類，以降低在映射後的等價線路線路時間延遲（delay）爲主要目標，亦即改進線路性能(performance)，包括 Murgai 等人所提之方法〔20〕， Chen、Cong 及 Ding 等人所提之以圖形理論爲基礎之演算法〔8〕〔9〕〔11〕。第三類，以提高在映射後的等價線路之可繞性（routability）爲主要目標。如 Bat、Hill〔5〕及 Schlag 等〔26〕所提之方法。第四類，綜合上述兩個或兩個以上之目標，以期求得一多目標間之均衡(trade-off) 〔25〕〔12〕。這些方法在某些特定問題（如 tree circuit）均可獲最佳解〔15〕〔16〕。

然而，以上所提及之演算法，多數是以圖形爲基礎(graph-based)所作的邏輯閘分解，並無保留原電路的函數資訊，因此所得到的結果就無法考慮到電路原有的函數特性。爲了得到較佳的技術映射解，我們應對原電路的函數資訊做適當的保留與分析，再來做邏輯閘的分解工作。以達到我們降低電路延遲及降低 LUT 使用個數的目的。以往的研究，大多是考量電路的結構性，不考量其函數性，利用圖形理論的方法來解決技術映射問題，先將電路轉化成包含節點(node)與邊(edge)的圖形，再利用如最大流量(max flow)一類的演算法將圖形做適當的技術映射。如 Cong 等的 FlowMap〔11〕即是典型的例子，而 FlowMap 最大的限制在於任何節點的扇入數都不得大於一固定的常數 k，因此一般做法是將電路先分解成 2-bounded 的電路，再套用這些演算法。但這些步驟皆未考慮到轉化後的圖形它在原始電路中所代表的函數意義，因此在本研究中，我們提出一具函數性考量的技術映射演算法。

貳、問題描述

爲求得好的技術映射解，首先我們所要考慮的問題是，我們必須能分析電路的結構及函數，針對電路的特性來作處理，在 LUT-Based 的映射問題中，我們可將如圖 1(b)使用一個 2-LUT 及一個 3-LUT 之映射電路以圖 1(a)之使用

* E-Mail: linyu@cad.ice.cycu.edu.tw
** E-Mail: hsieh@mbox.cycu.edu.tw

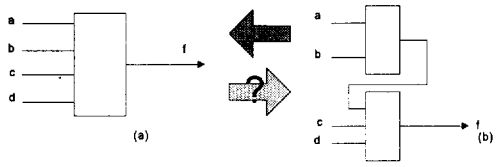

圖 1　並非任何 LUT 皆可任意分解

一個 4-LUT 之映射電路來取代，但圖 1(a)的電路卻不一定能化爲圖 1(b)之電路。我們考慮一個函數。

f = adc + b，顯然，f 可如圖 1(a)，以一個 4-LUT 來實現，但卻不能把 LUT 任意的分解成如圖 1(b)的形式，同理，邏輯閘也無法任意地分解。因此在許多的例子中，雖然分解後能得到比分解前時間延遲更短的映射解，但並不是所有的邏輯閘或 LUT 皆可任意的分解。而這個盲點卻常被人所忽視，舉例說明，圖 2 是由 Habib 及 Xu〔17〕中所舉的一個範例，圖 2(a)爲一個有迴路的循序電路，此電路中 f1、f2 爲暫存器，P1、P2、P3、P4 爲主要輸入端。圖 2(b)爲映射到 LUT 的結果，我們可看到此電路的迴路(loop)部份經過兩個 LUT，也就是說時間延遲是 2。而圖 2(c)的迴路部份僅經過一個 LUT，也就是說時間延遲僅爲 1。如果一個 4-LUT 能隨意分解成兩個 3-LUT，當然時間延遲即可輕易降低，但是由上述的例子我們知道一個 K-LUT 並不能任意分解成 K_1-LUT、K_2-LUT、K_3-LUT、…、K_m-LUT，其中 $K_1+K_2+\ldots+K_m = K$。也正因爲如此，雖然圖 2(c)的映射解有較短的時間延遲，但是在大多數的情況下，我們是無法將電路如此輕易分解的。

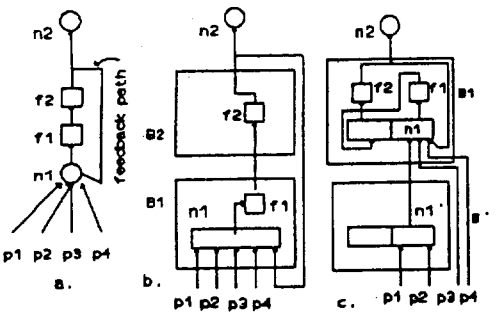

圖 2　邏輯閘或 LUT 之分解[17]

參、新演算法

一個電路必定是由多種不同的邏輯閘所組成，如 AND、OR、NAND、NOR、XOR、NOT、…等等，在這些基本邏輯閘中，AND、OR、XOR 等少數邏輯閘具有可任意分解與合併的特性。根據 generalized DeMorgan's theorem，

$$(a_1 + a_2 + \ldots + a_n)' = a_1' * a_2' * \ldots * a_n'$$

$$(b_1 * b_2 * \ldots * b_n)' = b_1' + b_2' + \ldots + b_n'$$

若電路是一個樹狀結構的電路，如圖 3(a)，我們可以將電路中所有含有反向的邏輯閘依以上述的定理做處理，而將在電路內所有有反向的邏輯閘(NOT、NAND、NOR)的反向函數都調整至主要輸入端(PI)，如圖 3(b)：

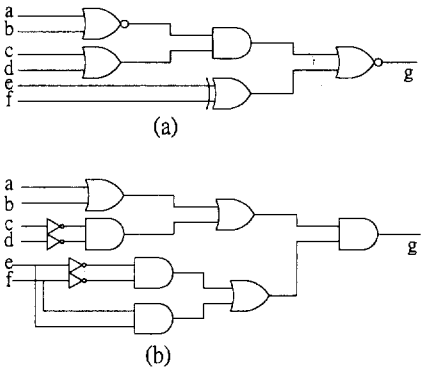

圖 3　電路中反向閘之移動

將電路內的所有反向閘推向主要輸入端的目的在於使電路中的邏輯閘皆可任意分解與合併，就樹狀結構的電路而言，除了主要輸入端仍可能有反向閘外，其餘部份皆僅留下由 AND、OR 邏輯閘所構成的電路結構。因此我們可得到一等價且單純的電路。

將反向閘移往主要輸入端後，接下來我們所要處理的，就是將相鄰具有相同的邏輯函數的邏輯閘作合併的動作，舉例來說，若節點 v 的函數爲 AND，而在其前行節點的部份若也爲 AND，我們可以將這兩個邏輯閘做合併成一個節點，如圖 4 所示。

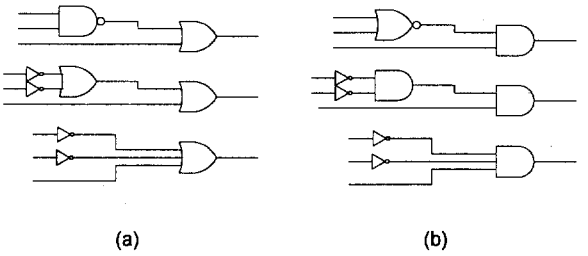

圖 4　邏輯閘的合併

在圖 4 中，我們先將所有的反向閘移往主要輸入端後，我們可以將同樣函數的邏輯閘合併。如圖 4(a)，反向閘移往

主要輸入端後，則電路中的兩個 OR 邏輯閘便可合併；如圖 4(b)，則是將反向閘移往主要輸入端後，再將兩個 AND 邏輯閘合併。

合併相同函數節點的目的是爲了要重新考慮其邏輯閘分解的方式，因爲輸入的電路可能無法符合我們的實際要求，若我們只考慮原始電路的結構，是無法得到最佳值的。所以我們應該還要考慮如何做函數化的分解，重新考慮節點函數性的分解後，使分解出來的電路符合我們的要求，最後再根據不同需求而套用適當的技術映射演算法。

此外，組合電路可能會有多重扇出的情況，對於此情況我們可以兩種方式來解決：第一種方法是僅將暫存器推至多重扇出的地方便停止，並不強迫暫存器要全移至主要輸入端的地方。第二種方法爲以節點複製（node replication）的方法來解決，則可保證暫存器必可推至主要輸入端。

在本論文中，將我們的演算法命名爲 SFC-Preprocess (Structure-Functionality Consideration Preprocess)。以下我們舉一範例說明本演算法的流程。

[例一]

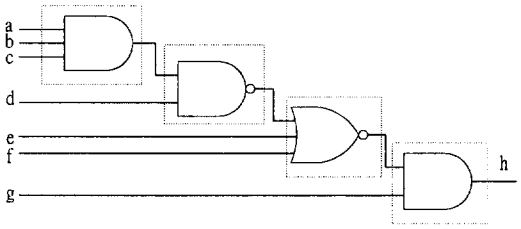

圖 5　原始邏輯電路

圖 5 爲一個 3-bounded 的單純邏輯電路，我們將其以 3-LUT 來做技術映射，則需要四個 LUT 且時間延遲爲 4。在不考慮電路函數性的情況下，這個映射解已無法再有任何的改進。

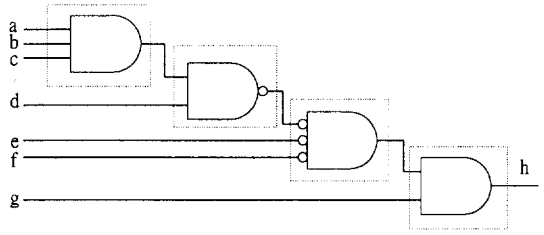

圖 6　反向閘往主要輸入端移動後之電路

接著，將反向閘往主要輸入端移動，如圖 6 所示，然後再將可合併之邏輯閘做合併的動作，如圖 7 所示。

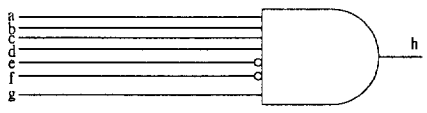

圖 7　邏輯閘合併後之電路

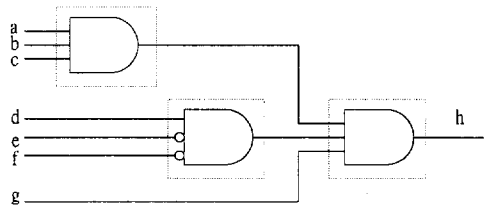

圖 8　映射後之 LUT 電路

最後，由於電路只含單純 AND 邏輯閘，因此可自由合併與分解，經恰當地分解後，重新映射至 3-LUT 中，則我們可得到一個僅需三個 LUT 且時間延遲爲 2 的映射解。

Algorithm SFC-Preprocess + FlowMap

```
T：list of non-PO nodes in topological order；
U：list of all nodes in topological order；
E：list of all edges；
k：the fanin number of LUT;
/*phase 1: push invert gate */
for   all edges of E   do
      if invert gate exists on the edge
            weight(e)=1
      else
            weight(e)=0
      endif
endfor
while   T is not empty   do
      remove the first node t from T：
      FI ={e | e∈fanin edge of t}
      FO={e | e∈fanout edge of t}
      if   |FI|>0
            if   weight(e)=1 for all e∈FO
                  for all e∈FO
                        let weight(e)=0 if weight(e)=1;
                        let weight(e)=1 if weight(e)=0;
                  endfor
            endif
      endif
endwhile

/* phase 2:merge gate with the same type */
while   U is not empty   do
      remove the first node u from U：
      V={v | v is a fanin node of u};
            while   V is not empty   do
                  remove the first node v from V
                  if    weight of edge between u,v is zero and type
                        of u,v is the same do
                        fanin(u)=fanin(u)∪fanin(v);
                        discard node v;
                  endif
            endwhile
endwhile

/* phase 3: Nodes decomposition */
for all node v
      if   |fanin(v)|>k
```

```
            decompose node
        endif
endfor

/* phase 4:Mapping */
FlowMap Mapping;

        end-algorithm
```

設 m=|V|(節點數)，n=|E|(邊數)，k 爲 LUT 最大輸入端個數，我們的演算法的時間複雜度(time complexity)在第一、二步驟爲 O(m+n)，也就是 topological sort 的複雜度，第三步驟 FlowMap 的複雜度是 O(kmn)，因此整體而言，我們的流程時間複雜度爲 O(kmn)。但我們的第三步驟是隨使用者需求而允許有所變更的，如 FlowMap-r、chortle、VISmap 等或自行發展之映射演算法。

針對經我們處理所得到的電路而言，因電路中的邏輯閘皆可任意地分解。故其技術映射深度與面積最佳解的下限可由以下幾項 lemma 得到。

Lemma 1：並不是所有任意邏輯閘皆可任意分解。

[證明]：舉一反例證明，考慮一 3-inputs 的邏輯閘，設其函數爲 f(a,b,c) = ab + ca + bc ，若我們欲將其分解爲兩個 2-inputs 的邏輯閘所構成的電路，則無法達到。

Lemma 2：當電路中的邏輯閘皆是 AND、OR、XOR 時，則邏輯閘可任意地分解。

[證明]：設一邏輯閘之函數爲 $f(x_1,x_2,...x_i,...x_p)$，則當邏輯閘爲 AND、OR、XOR 時：因爲 AND、OR、XOR 均爲滿足結合律之運算，故 $f(x_1,x_2,...x_i,...x_p) = f(x_i,f(x_1,x_2,...,x_{i-1},x_{i+1},...x_p))$，故此邏輯閘必可任意分解。

例如，當 p=3，即變數個數爲 3 個時，則

abc=a(bc)=(ab)c=b(ac)；
a+b+c=a+(b+c)=(a+b)+c =b+(a+c)；
a⊕b⊕c=a⊕(b⊕c)=(a⊕b)⊕c=b⊕(a⊕c)。

定理 1：

對一個 n-輸入、單一輸出的邏輯閘或 LUT，若我們將其分解成 k-輸入（k<n）、單一輸出的邏輯閘或 LUT，則分解後的個數至少爲 $\left\lceil \frac{n-1}{k-1} \right\rceil$，而深度則爲 $\lceil \log_k n \rceil$。

[證明]：當一邏輯閘分解時，以分解成平衡樹的結構可得到深度的最佳化。一 PI 個數爲 n，分支個數爲 k 的平衡樹其深度爲 $\lceil \log_k n \rceil$。而以圖 9 的架構我們即可使 LUT 或邏輯閘的 fanin 端使用率達到最高。設 a_n 表示 n-輸入、單一輸出的邏輯閘(或 LUT)分解成 k-輸入、單一輸出的邏輯閘(或 LUT)個數。由圖 9 可得遞迴方程式：

$a_n = a_{n-k+1} + 1$ (n>k) 且 $a_1=a_2=...=a_k=1$，

解此遞迴式，得 $a_n = \left\lceil \frac{n-1}{k-1} \right\rceil$。

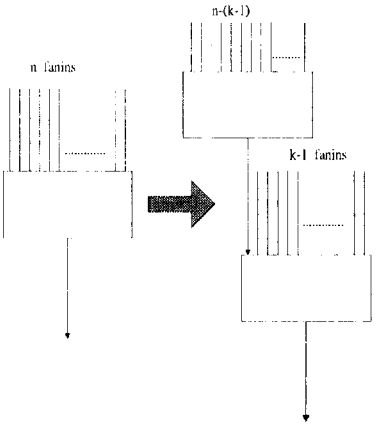

圖 9　使邏輯閘或 LUT 的 fanin 端使用率達到最高的分解架構

肆、實驗結果

我們已成功地在 Sun SPARC-10 工作站上以 C 語言實現我們的 SFC-Preprocess (Structure-Functionality Consideration Preprocess)演算法。給定一個一般的布林函數電路作爲輸入，我們首先利用 DAG-Map 所提及的 DMIG 及 SIS 中的 tech_decomp 將電路分解成 2-bounded 的電路，以保證電路中皆爲單純邏輯閘，然後使用 SFC-Preprocess 與特定之技術映射演算法將對電路做映射的動作。在不失一般性的情況下，我們選擇 k=5 當作 k-LUT 的 k 值。在選擇 benchmark 方面，我們以 MCNC〔29〕的 benchmark 來測試與比較我們的結果。

Table 1 是我們目前所測試過的電路，node 個數是指電路中邏輯閘的個數，edge 數是指邏輯閘之間的連線數，depth 則是原電路的深度。

由於 FlowMap 只接受 k-bounded 的輸入電路，而原始電路並不是 k-bounded 的，因此 FlowMap 需有對節點分解的前置處理。FlowMap 的時間複雜度爲 O(kmn)；k 爲 LUT 最大扇入數，m 爲電路中節點數，n 爲電路中的邊數，由於我們的實驗複雜度取決於 FlowMap，因此經我們處理後的電路節點及邊數如果會增加過度，則整體的效率便有可能會被拖累。然而實驗證實，經 SFC-Preprocess 所輸出電路節點數與邊線個數，僅比原始電路增加約 0.03%，所以我們的方法在時間複雜度上頗具效率。

Table 1　經過前處理的 initial 2-bounded circuits

Circuit	Initial 2-bounded circuits		
	#LUTs	Delay	#edges
9symml	234	12	468
Alu2	671	23	1342
Alu4	1166	28	2332
Apex6	666	14	1332
Apex7	293	12	587
c499	414	18	828
c880	354	22	700
Count	127	19	254
Duke2	885	9	1770
Rot	1288	21	2589
Vg2	796	9	1592
X1dc7a	2113	11	4233
X2dc7a	62	5	124
Z4ml	248	8	496

Table 2　實驗結果比較

Circuit	FlowMap(SIS)			SFC_Preprocess +FlowMap		
	#LUTs	Delay	#edges	#LUTs	Delay	#edges
9symml	73	5	275	70	5	279
Alu2	307	9	1171	307	9	1171
Alu4	552	9	2105	552	9	2105
Apex6	267	5	1061	267	5	1061
Apex7	122	4	426	122	4	426
c499	74	4	280	74	4	280
c880	161	7	604	74	7	193
Count	43	5	158	35	5	139
Duke2	374	4	1355	374	4	1355
Rot	541	8	1907	541	8	1907
Vg2	290	4	1080	290	4	1080
X1dc7a	806	5	2882	746	5	2828
X2dc7a	20	3	72	17	2	66
z4ml	45	3	198	38	3	174
Total	3675	75	13574	3507	74	13064
	1	1	1	0.9542	0.9866	0.9624

在 Table 2 中，我們以 SIS 中的 FlowMap 與 SFC_Preprocess+FlowMap 相比較，由表中可知，無論在 LUT 的個數、LUT 的連線個數或是時間延遲方面，本方法皆有不錯的改善。

伍、未來的工作

在後續工作裡，我們所要做的是找尋更佳的節點分解演算法，以期在合理的時間複雜度內，達到所要求面積與深度的技術映射解，並考慮將此演算法應用至循序電路上。在循序電路方面，本演算法可結合循序電路映射演算法中常見的時序重置(retiming)、節點複製(replication)等技巧，在不增加過多的額外時間複雜度的前提下，試著考慮電路之函數性，以得到更佳之映射解。

陸、參考文獻

1. Actel, ACT2 Data Sheet, The Sunnyvale, Calif., ACTEL Corporation, 1991.
2. Actel, ACT3 Data Sheet, The Sunnyvale, Calif., ACTEL Corporation, 1992.
3. Altera, Data Book, The San Jose, Calif., ALTERA Corporation, 1993.
4. K. Bartlett, G. Borriello, and S. Raju, "Timing Optimization of Multiphase sequential Logic," IEEE Trans. on Computer-Aided Design, pp. 51-62, 1991.
5. N. Bhat and D. Hill, "Routable Technology Mapping for FPGAs," ACM/SIGDA Workshop on FPGAs, pp. 143-148, 1992.
6. S. D. Brown, R. J. Francis, J. Rose, and Z. G. Vranesic, Field-Programmable Gate Arrays, Kluwer Academic Publishers, 1992.
7. S. T. Chakradhar, S. Dey, M. Potkonjak, and S. G. Rothweiler, "Sequential Circuit Delay Optimization Using Global Path Delays," Proc. 30th ACM/IEEE Design Automation Conf., pp. 483-489, 1993.
8. 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.
9. 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.
10. J. Cong and Y.Ding, "On Nominal Delay Minimization in LUT-Based FPGA Technology Mapping," Integration, the VLSI Journal, Vol. 18, pp. 73-94, 1994.
11. J. Cong and Y. Ding," FlowMap: 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.
12. 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.
13. 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.
14. S. Dey, F. Brglez, and G. Kedem, "Partioning Sequential Circuits for Logic Optimization," Proc. Intl. Conf. on Computer Design, pp. 70-76, 1991.

15. R. J. Francis, J. Rose, and K. Chung. "Chortle: A Technology Mapping Program for Lookup Table-Based field Programmable Gate Arrays." Proc. of the Design Automation Conference, pages 613-619, 1990.
16. R. J. Francis, J. Rose, and Z. Vranesic. "Chortle-crf: Technology Mapping of Lookup Table-Based FPGAs for Performance. "Proc. of the Design Automation Conference, Page 227—233, 1991.
17. S.Habib and Quan Xu,"Technology Mapping Algorithm for Sequential Circuits Using Lootup Table Based FPGAs." IEEE Design and Test of Computers,pp 164-167,1995
18. K. Karplus, "Xmap: A Technology Mapper for Table-lookup FPGAs," Proc. 28th ACM/IEEE Design Automation Conf. , pp. 240-243, 1991.
19. G. De Micheli, "Synchronous Logic Synthesis: Algorithms for Cycle-Time Minimization," IEEE Trans. on Computer-Aided Design, Vol. 10, pp. 63-73, 1991.
20. 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.
21. 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.
22. 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.
23. P. Pan and C. L. Liu, "Optimal Clock Period FPGA Technology Mapping for Sequential Circuits," Proc. 33th ACM/IEE Design Automation Conference, pp. 720-725,1996.
24. P. Pan and C. L. Liu, "Technology Mapping of Sequential Circuit for LUT-based FPGAs for Performance," Proc. ACM 4th Intl. Symposium on Field-Programmable Gate Arrays," pp.58-64,1996.
25. P. Sawkar and D. Thomas, "Area and Delay Mapping for Table-look-up Based Field Programmable Gate Arrays," Proc. 29th ACM/IEEE Design Automation Conf., pp. 368-373, 1992.
26. 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, pp. 13-26, 1994.
27. 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
28. Xilinx,The Programmable Gate Arrays Data Book. Xilinx, San Jose,CA，1993
29. Saeyang Yang, Logic Synthesis and Optimization Benchmark User Guide Version 3.0 .January 15,1991

Technology Mapping with Functionality Considerations

YU-CHUNG LIN

Department of ElectronicEnineering
Chung Yuan Christian University
Chung-Li, Taiwan 32023, R.O.C.

TSAI-MING HSIEH

Department of Information & computer Engineering
Chung Yuan Christian University
Chung-Li, Taiwan 32023, R.O.C.

ABSTRACT

All existing technology mapping algorithms, which were proposed to find a minimum-depth mapping solution, consider only the structure of a given circuit, and do not consider the functionality of the circuit. In contrast to the conventional algorithms, we proposed a new technology mapping algorithm with functionality consideration for depth minimization and reduction of the number of LUT's.

Key words: *Field Programmable Gate Array (FPGA), Technology Mapping, functionality consideration.*