Transistor Network Generation in Combinational Logic Circuits

Naveen J[1], Jabez Daniel V D M[2]
[1]PG Scholar, [2]Assistant Professor, ,Electronics and Communication Department
Dr.Sivanthi Aditanar College of Engineering
Tiruchendur,India

Abstract

Optimizing transistor network represents an efficient way of improving VLSI circuits. Existing optimization methods are not actually the most effective way to generate optimized network. The method of reducing only number of literals in a given Boolean expression does not actually reduce the transistor count. This paper proposes a graph-based method for minimizing the number of transistor count that compose a network. This proposed method can provide networks with minimum transistor count with irredundant sum-of-products expression as the input. Experimental results produced by the proposed method shows a substantial reduction in the number of transistor used in the networks.

Keywords— *Digital circuit, factorization, graph theory, transistor network.*

I. INTRODUCTION

In digital VLSI design the power dissipation and area of circuits are mainly related to the number of transistors used in the circuit [1]-[3]. Hence, transistor network optimization is an efficient way when designing custom gates to be used in the libraries [4],[5]. Numerous methods have been proposed in the literature for generating and optimizing transistor networks. The most basic method used for transistor network optimization is factoring the Boolean expressions where only series–parallel (SP) transistor network can be produced [8]-[11]. On the contrary, graph-based methods can produce both Series-Parallel and non-Series-Parallel (NSP) structures which can provide considerable amount of transistor count reduction.

Reducing the number of literals in Boolean expressions does not actually provide the most efficient transistor networks. Hence, existing factorization methods and graph-based methods may not be able to reduce the transistor count to a greater extent. The way of decreasing the transistor count in logic gates leads to reduction in circuit area and power consumed in digital logic circuits.

In this paper, we propose a method for enhancing the reduction of transistors using both series-parallel (SP) and non-SP (NSP) arrangements that performs OR and AND operations.

Using only graph-based methods can reduce the transistor count upto a certain limit. In this paper, we propose a graph-based method followed by efficient factorization method which enhance the transistor network to reduce the transistor count to greater extent than other existing optimization techniques.

II. PRELIMINARIES

In this work, the goal is to produce a compact transistor network for a given Boolean function. For a better understanding of the proposed methods, some short-term definitions are presented in the following.

A literal is a variable of a Boolean function *f* or its complement, *e.g. c* or *!c*, whereas a *cube* is a product of literals. An irredundant sum-of-products (ISOP) is a sum-of-products (SOP) where either a literal or a cube cannot be removed without changing the function of Boolean expression.

A series-parallel network (SP) is a network where the transistors are in either series or in parallel arrangements. It can also contain both series and parallel arrangements combined.

A non-series-parallel network (NSP) is a network where the transistors have a bridge connection between series and parallel arrangements. To simplify the discussions in the following sections only the Pull Down network is considered.

The combining of transistors enhances the transistor count reduction in logic gates thereby reducing the power dissipation. Hence, the transistor merging is applied to the non-critical circuit paths.

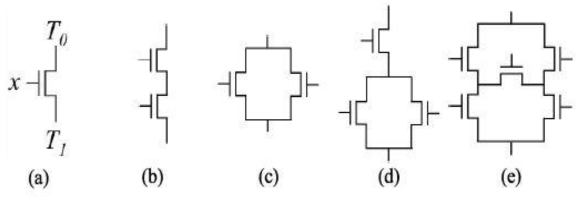


Fig 1.(a) MOS Transistor, (b) Series Network, (c) Parallel Network, (d) Series-Parallel, (e) Non-Series-Parallel.

III. PROPOSED WORK

In this section, we present a novel graph-based method to reduce the transistors used in the network structure. The proposed method progressively reduce the cubes from an ISOP *F* by applying different heuristics to find out the network structure that provide a starting point to perform the logic sharing and consider some constraints to maximize the reduction of transistor count.

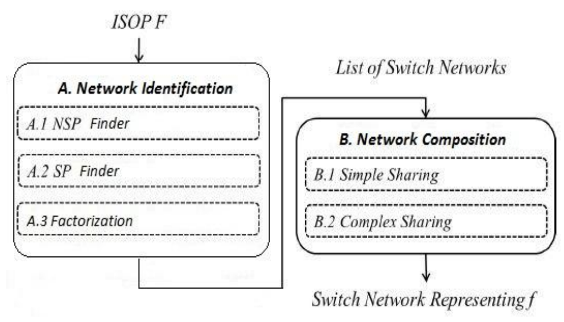


Fig. 2. Execution Flow of the Proposed Method.

Initially, we apply the NSP Finder procedure over *F* in order to find partial transistor networks. Cubes used to achieve such a network structure are removed from *F*. This removal leads to a simpler ISOP *F1*. Now the SP Finder procedure is performed over *F1* in order to find any SP transistor networks. Cubes used to achieve such a network structure are removed from *F1*. This removal leads to a simpler ISOP *F2*. In this order, we apply the algebraic Factorization over *F2* to produce factored forms that improves the transistor count reduction. Finally, the Network Composition step receives the partial networks and comprises them into a single network by applying edge-sharing technique. Another important aspect is that the proposed graph-based method can either enable or disable one of the following steps NSP Finder, SP Finder, Factorization. As the proposed approach is heuristic, it is appropriate to try the three combinations to enable or disable these steps and select the best network among them.

A. Non- Series-Parallel Finder:

An ISOP *F* with *m* cubes is an undirected graph *G* = (*V*,*E*), where vertices in *V* = {*v1, v2, …, vm*} represent different cubes of *F*. An edge *e* = (*vi*,*vj*) ∈ *E*, being *i* ≠ *j*, exists if, and only if, *vi* ∩ *vj* ≠ ∅. Such edge *e* is labeled *vi* ∩ *vj*. In order to identify Non-Series-Parallel network structures, the combination of *m* cubes are taken four at a time. The method tries to generate network for each combination of four cubes. To confirm that the generated network produces a Non-Series-Parallel network, two rules must be checked:

Rule 1 – Let the set of edges Ev connected to the vertex *v* ∈ *V*. For each cube *v* ∈ *V*, all literals from *v* must be mutual throughout the edges *e* ∈ *Ev*. The rule is contented i only if the following equation results the logic value 1:

$$\prod_{v\in V}\left(\left(\bigcup_{e\in E_v} e\right)=v\right).$$

Rule 2 – The generated network must be symmetric to the graph shown in Fig. 3(a), which is referred as Non-Series-Parallel network.

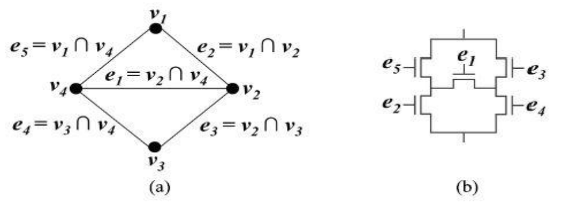


Fig.3. (a) Non-Series-Parallel Network Template, (b) Resulting Transistor Network.

For each network structure found, the proposed method maps the edges from the network, seen in Fig. 3(a). Then the edge reordering is performed over the template graph which leads to the transistor network shown in Fig. 3(b). The worst case time complexity of the Non-Series-Parallel Finder step is bounded by the cost of performing 4-combination of *m* cubes from *F*.

B. Series-Parallel Finder:

Let *F*1 be the Boolean expression that represents all the cubes of *F* that were not used to construct the transistor networks in the Non-Series-Parallel finder step. In order to find Series-Parallel network, a combination of *m*1 cubes are taken four at a time from *F*1. To confirm that the generated network produces a Series-Parallel network, Rule 1 and the following Rule 3 must be verified.

Rule 3 – The generated network must be symmetric to the graph shown in Fig. 4(a). Similarly to the previous step, the Series-Parallel finder step must perform transformations in order to achieve a Series-Parallel network. First, the edges shown in Fig. 4(a) are mapped to a template graph, as shown in Fig. 4(b). Then the edge reordering is performed over the template graph which leads to the transistor network shown in Fig. 4(c).

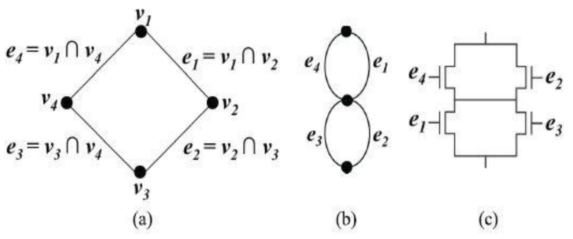
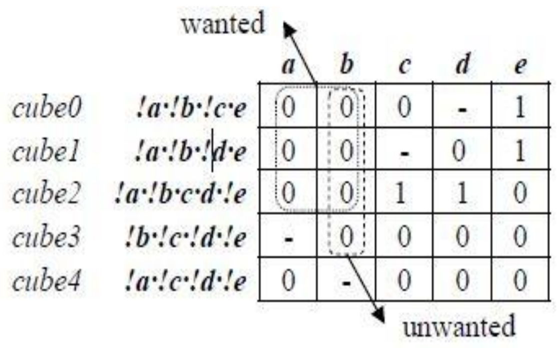


Fig. 4.(a) SP Network Template, (b) Auxiliary Template graph, (c) Resulting Transistor Network.

C. Factorization:

This section presents a empirical approach to perform naïve algebraic factorizations to the Boolean function considering a constraint to prioritize factorizations that improves the transistor count reduction. In the proposed approach, cubes that share minimum of two literals is taken. For instance, consider the matrix shown in Fig. 5, which is obtained from the following Boolean function:

F = (!a · !b · !c · e) +(!a · !b · !d · e)+(!a · !b · c · d · !e) +(!b · !c · !d · !e) + (!a · !c · !d · !e) (1)



		a	b	c	d	e
cube0	!a·!b·!c·e	0	0	0	-	1
cube1	!a·!b·!d·e	0	0	-	0	1
cube2	!a·!b·c·d·!e	0	0	1	1	0
cube3	!b·!c·!d·!e	-	0	0	0	0
cube4	!a·!c·!d·!e	0	-	0	0	0

wanted
unwanted

Fig. 5. Matrix Obtained from (1), which Represents Cubes Relationship.

D. Network Composition:

This Network Composition module is used to combine partial network structures generated during previous steps in order to construct the final network. This step receives the logic function *F* and a list of partial transistor networks from previous steps and combine them into a single network structure. All the partial networks are generated by the NSP Finder, SP Finder and Factorization steps. The final network is produced by combining all the partial networks in parallel then a logic sharing is performed to remove redundant terms.

E. Simple Sharing:

The simple sharing step performs the edge sharing technique presented in [13]. Actually, the method pass through the transistor network structure searching for similar transistors. The network is then restructured so that only one common node is between similar transistors. In some cases, the similar transistors must be exchanged in the network structure in order to share a common node.

F. Complex Sharing:

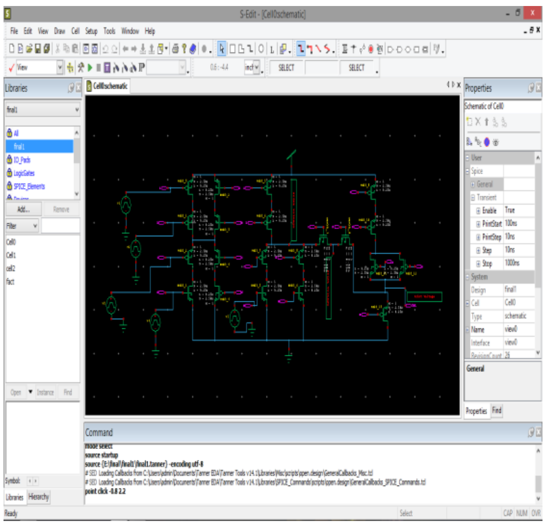
The complex sharing step receives the network structure from the previous step and attempts to perform further optimizations. Few times, where a common node is not directly found due to the position of the transistors in the network. In order to improve the switch sharing, direct Series-Parallel compressions are performed. A fascinating characteristic of the proposed method is that it can find and generate Series-Parallel and Non-Series-Parallel network structures by applying logic sharing. In few cases, Non- Series-Parallel structures produces significant reduction in the final transistor count.

IV. SIMULATION RESULTS

In this proposed work, the logic functions can be implemented using less number of transistors by graph based methods. The edges connecting some intermediate nodes of two (or more) paths are called bridges. Consider a complex Boolean function represented as follows,

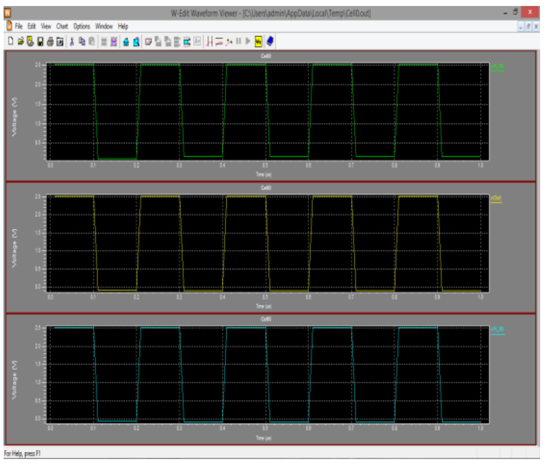
F = (!a ·!b · !c ·e)+(!a · !b · !d · e)+(!a · !b · c · d · !e) +(!b · !c · !d · !e) +(!a · !c · !d · !e) + (!a · !b · !c · d) + (a · b· c · !d) + (!a · !b · c · !d) + (a · b · !c · !d) (2)

The proposed method for Boolean function shown in (2) is designed in TANNER EDA tool . The network design shown in Fig.6 (a) is drawn in S-Edit (Schematic Editor) of Tanner EDA. Then, the circuit is simulated and the output waveforms are viewed in Waveform Editor, W-Edit as shown in Fig.6 (b).



S-Edit - [Cell0:schematic]
File Edit View Draw Cell Setup Tools Window Help
Libraries
Properties
Command

(a)



(b)

Fig. 6. (A)Schematic of the Boolean Function Using the Proposed Method in S-Edit Tool, (B) Matrix *M* Obtained From (1), Which Represents Cubes Relationship.

TECHNIQUE USED	TRANSISTOR COUNT
Factoriztion	27
[16]	25
Proposed Method	21

Table 1. Comparison of Proposed Method with Various Approaches for Boolean Function (2).

V. CONCLUSIONS

It is known that the reduction in transistor count improves the circuit performance and area of digital circuits. This paper presented a graph-based approach to generate optimized transistor networks. As demonstrated through experimental results, the proposed method provided significant reduction in transistor count to achieve compact networks. When the exact factorization is unknown or cannot be easily found, the proposed graph-based approach tends to provide better results than the existing related techniques. The proposed method results in a reduction of transistor count when compared to previous approaches and able to deliver bridge networks.

REFERENCES

[1] Y.-T. Lai, Y.-C. Jiang, and H.-M. Chu, "BDD decomposition for mixed CMOS/PTL logic circuit synthesis," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), vol. 6. May 2005, pp. 5649–5652.

[2] H. Al-Hertani, D. Al-Khalili, and C. Rozon, "Accurate total static leakage current estimation in transistor stacks," in Proc. IEEE Int. Conf. Comput. Syst. Appl., Mar. 2006, pp. 262–265.

[3] T. J. Thorp, G. S. Yee, and C. M. Sechen, "Design and synthesis of dynamic circuits," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 1, pp. 141–149, Feb. 2003.

[4] A. I. Reis and O. C. Andersen, "Library sizing," U.S. Patent 8 015 517, Jun. 5, 2009.

[5] R. Roy, D. Bhattacharya, and V. Boppana, "Transistor-level optimization of digital designs with flex cells," Computer, vol. 38, no. 2, pp. 53–61, Feb. 2005.

[6] M. Rostami and K. Mohanram, "Dual-vth independent-gate FinFETs for low power logic circuits," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 30, no. 3, pp. 337–349, Mar. 2011.

[7] M. H. Ben-Jamaa, K. Mohanram, and G. De Micheli, "An efficient gate library for ambipolar CNTFET logic," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 30, no. 2, pp. 242–255, Feb. 2011.

[8] M. C. Golumbic, A. Mintz, and U. Rotics, "An improvement on the complexity of factoring read-once Boolean functions," Discrete Appl. Math., vol. 156, no. 10, pp. 1633–1636, May 2008.

[9] E. M. Sentovich et al., "SIS: A system for sequential circuit synthesis," Dept. Elect. Eng. Comput. Sci., Univ. California, Berkeley, Berkeley, CA, USA, Tech. Rep. UCB/ERL M92/41, May 1992.

[10] M. G. A. Martins, V. Callegaro, L. Machado, R. P. Ribas, and A. I. Reis, "Functional composition and applications," in Int. Workshop Logic Synthesis Tech. Dig. (IWLS), Jun. 2012, pp. 1–8. [Online]. Available: http://www.inf.ufrgs.br/logics/

[11] M. G. A. Martins, L. S. da Rosa, Jr., A. B. Rasmussen, R. P. Ribas, and A. I. Reis, "Boolean factoring with multi-objective goals," in Proc. IEEE Int. Conf. Comput. Design (ICCD), Oct. 2010, pp. 229–234.

[12] L. S. da Rosa, Jr., F. S. Marques, F. R. Schneider, R. P. Ribas, and A. I. Reis, "A comparative study of CMOS gates with minimum transistor stacks," in Proc. 20th Annu. Conf. Integr. Circuits Syst. Design (SBCCI), Sep. 2007, pp. 93–98.

[13] V. N. Possani, R. S. de Souza, J. S. Domingues, Jr., L. V. Agostini, F. S. Marques, and L. S. da Rosa, Jr., "Optimizing transistor networks using a graph-based technique," J. Analog Integr. Circuits Signal Process., vol. 73, no. 3, pp. 841–850, Dec. 2012.

[14] D. Kagaris and T. Haniotakis, "A methodology for transistor-efficient supergate design," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 15, no. 4, pp. 488–492, Apr. 2007.

[15] J. Zhu and M. Abd-El-Barr, "On the optimization of MOS circuits," IEEE Trans. Circuits Syst. I, Fundam. Theory Appl., vol. 40, no. 6, pp. 412–422, Jun. 1993.

[16] V. N. Possani, V. Callegaro, A. I. Reis, R. P. Ribas, F. S. Marques, and L. S. da Rosa Jr., "Graph-based transistor network generation method for supergate design," IEEE Trans. on Very Large Scale Integration (VLSI) Systems, vol.24, no.2, pp.692-705, Mar. 2015.

[17] V. N. Possani, R. S. Souza, J. S. Domingues Jr., L. V. Agostini, F. S. Marques, and L. S. da Rosa Jr., "Optimizing transistor networks using a graph-based technique," Journal of Analog Integrated Circuits and Signal Processing, vol.73, no.3, pp.841-850, Dec. 2012.

[18] Sreelekshmi S., Pooja S. Mohan, " Area Efficient Architecture for TCAM Using Hybrid Partitioned SRAM," International Journal of Electronics and Communication Engineering (IJECE), vol. 2, no. 7, July 2015.