Sum-of-Products & Parallel VLSI Transistor-Level by CMOS Logic Gates

Mr. N. Md. Bilal[1] Shasi Rekha Talari[2]
[1]Associate Professor [2]PG-Scholar
[1,2]Department of Electronic & Communication Engineering
[1,2]SVR Engineering College Nandyal

Abstract— VLSI digital design, the signal delay propagation, power dissipation, and area of circuits are strongly related to the number of transistors (switches). Hence, transistor arrangement optimization is of special interest when designing standard cell libraries and custom gates. Switch based technologies, such as CMOS, FinFET, and carbon nanotubes, can take advantage of such an improvement. Transistor network optimization represents an effective way of improving VLSI circuits. This paper proposes a novel method to automatically generate networks with minimal transistor count, starting from an irredundant sum-of-products expression as the input. The method is able to deliver series–parallel (SP) and non-SP switch arrangements, improving speed, power dissipation, and area of CMOS gates.

Key words: sum-of-products (SOP) -parallel (SP), VLSI, Transistor-level, and CMOS logic gates

I. INTRODUCTION

IN VLSI digital design, the signal delay propagation, power dissipation, and area of circuits are strongly related to the number of transistors (switches) [1]. Hence, transistor arrangement optimization is of special interest when designing standard cell libraries and custom gates [4]. Switch based technologies, such as CMOS, FinFET [6], and carbon nanotubes [7], can take advantage of such an improvement. Therefore, efficient algorithms to automatically generate optimized transistor networks are quite useful for designing digital integrated circuits (ICs). Several methods have been presented in the literature for generating and optimizing transistor networks. Most traditional solutions are based on factoring Boolean expressions, in which only series–parallel (SP) associations of transistors can be obtained from factored forms [8]. On the other hand, graph-based methods are able to find SP and also non-SP (NSP) arrangements with potential reduction in transistor count.

Despite the efforts of previous works, there is still a room for improving the generation of transistor networks

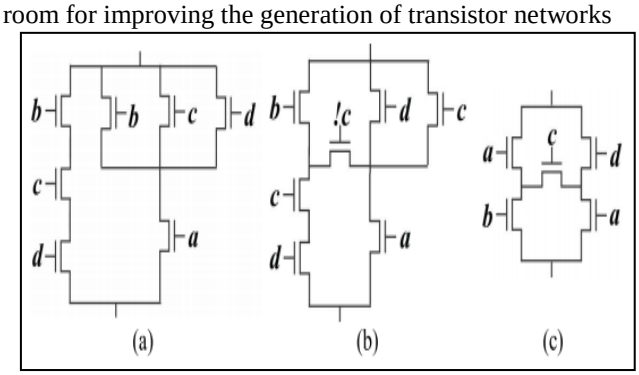

Fig. 1: Transistor networks corresponding to (1). (a) SP solution from factored form. (b) NSP from existing graph-based generation methods. (c) Optimum NSP solution.

For instance, consider a given function represented by the following equation:

$$F = a \cdot b + a \cdot c + a \cdot d + b \cdot c \cdot d. \quad (1)$$

For this function, factorization methods are able to deliver the SP network shown in Fig. 1(a), comprising seven transistors. Existing graph-based methods, in turn, are able to provide the NSP solution shown in Fig. 1(b), also with seven transistors. However, the optimal arrangement composed of only five transistors, as shown in Fig. 1(c), is not found by any of these methods [8].

II. IMPLEMENTATION

A. Transistor Stacks Bounding

Switch networks can be exploited by switch-based technologies, which present some restrictions or guidelines to be followed by designers. For example, in the conventional CMOS design technology, the maximum number of stacked transistors is usually limited to four. Such restriction is done in order to avoid performance degradation. Notice that there is a lower bound on the stacked transistors in switch networks. This lower bound corresponds to the minimum decision chain (MDC) property of the represented Boolean functions [19]. In this sense, an interesting feature to control (or to limit) the number of stacked transistors was included in our method. The method can operate in two execution modes, bounded and unbounded, as described below.

1) Bounded Mode

In this execution mode, a bound variable is used as reference to control the maximum number of transistors in series. The bound value must be equal or greater than the number of literals of the smallest cube from F, i.e., the maximum number of literals in a single cube. When the method is running in the bounded mode, the kernel identification module accepts only switch networks in which maximum stacked transistors do not exceed the bound value. Hence, the networks satisfying such a bound are added to the list S of found networks. This control is also performed during the network composition module when applying switch sharing, since it can increase the transistor stack.

2) Unbounded Mode

When running in the unbounded mode, there is no restriction of transistor stacking, i.e., the bound variable is not considered. Basically, just the total transistor count of the network is taken as metric cost. Hence, there are cases that the networks generated through the unbounded mode result fewer transistors when compared with bounded solutions. Moreover, these different modes are quite useful to explore the tradeoff between circuit area and performance.

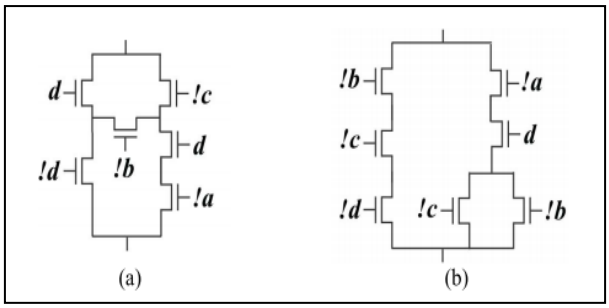

Fig. 2:

Example 12:

In order to demonstrate these two execution modes, consider the following equation:

$$F = !a \cdot !b \cdot d + !a \cdot !c \cdot d + !b \cdot !c \cdot !d. \quad (13)$$

The network shown in Fig. 21(a) is obtained when running the method in the unbounded mode. Notice that, in this network, the size of the transistor stack $!a \cdot d \cdot !b \cdot d$ is >3, which is the number of literals of the smallest cube from (13). When running the method in the bounded mode, it is possible to ensure transistor stacks with at most three devices, as shown in Fig. 21(b). On one hand, the bounded network presents an overhead of one transistor in comparison with the unbounded solution. On the other hand, the bounded solution has smallest transistor stacks. In this sense, one can consider the bounded solution when targeting performance or the unbounded solution for smaller area.

III. Experimental Work

A. *Logic Synthesis*

An irredundant sum-of-products (SOP) expression have been designed in Verilog described at the RTL level, synthesized with XILINK ISE 14.1 technology library

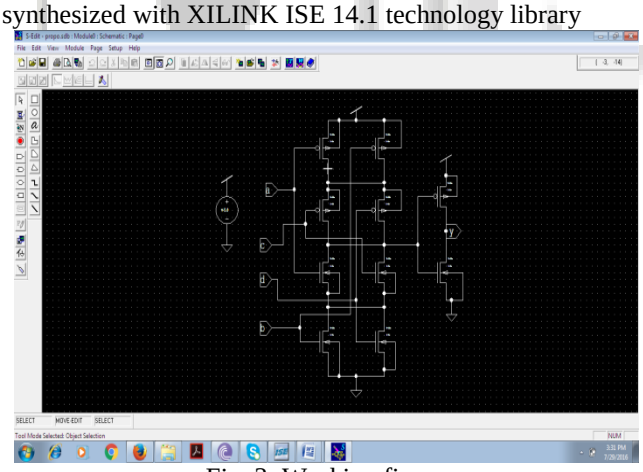

Fig. 3: Working figure

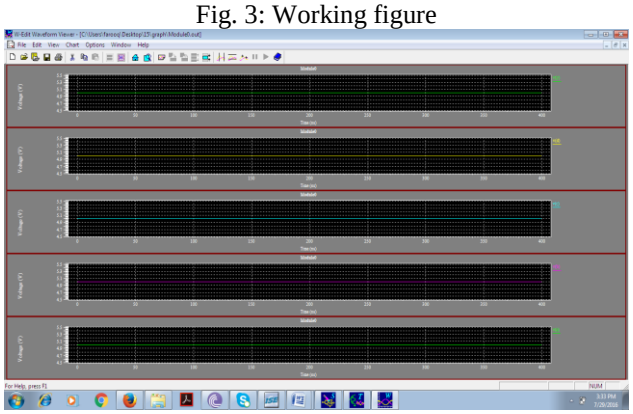

Fig. 4:

IV. Conclusion

This paper described an efficient graph-based method to generate optimized transistor (switch) networks. Our approach generates more general arrangements than the usual SP associations. Experimental results demonstrated a significant reduction in the number of transistor needed to implement logic networks, when compared with the ones generated by existing related approaches. It is known that the transistor count minimization in CMOS gates may improve the performance, power dissipation, and area of digital ICs. In a general point-of-view, the proposed method produces efficient switch arrangements quite useful to be explored by different IC technologies based on switch theory.

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. CircuitsSyst. (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.-AidedDesign 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 readonce Boolean functions," Discrete Appl. Math., vol. 156, no. 10, pp. 1633–1636, May 2008.

