Logical Effort to study and Compare the performance of VLSI Adders.

Satyajit Anand
Electronics & Communication Engineering
Mody Institute of Technology & Science
Lakshmangarh-332311, Rajasthan, India
satyajitanand45@gmail.com

P.K Ghosh
Electronics & Communication Engineering
Mody Institute of Technology & Science
Lakshmangarh-332311, Rajasthan, India
pkghosh_ece@yahoo.co.in

Manoj Kumar
Electronics & Communication Engineering
Mody Institute of Technology & Science
Lakshmangarh-332311, Rajasthan, India
talkto_manoj2000@yahoo.co.in

Gaurav Dhiman
Electronics & Communication Engineering
Mody Institute of Technology & Science
Lakshmangarh-332311, Rajasthan, India
ggrdhiman@gmail.com

***Abstract*-CMOS logic gates are basic building blocks for VLSI adder's circuits. The delay through these gates is related to their sizes and terminal loads. Logical effort is a technique, which gives insight about proper sizing of CMOS logic gates to have the minimum achievable delay. In this paper, we discuss first the technique of logical effort; three common architectures for VLSI adders are sized using logical effort to get the minimum possible delay. Simulated results are used to design fast CMOS circuits. A comparison between delays for these structures is presented according to simulation results in 32 nm standard CMOS process.**

***Keywords*- Static-CMOS Xor, Mux Gate, optimization, Logical Effort.**

I. Introduction

CMOS logic gates are basic building blocks for gate circuits. The delay through these gates is related to their sizes and their terminal loads. Logical effort is a technique, which gives insight about proper sizing of CMOS logic gates to have the minimum achievable delay. Different Configuration of VLSI Adders are sized using logical effort to get the minimum possible delays. The paper is organized as follows. Section II describes the theory of logical Effort. Section III describes the theory of multi-state circuits; Logical effort adder's structure is discussed in section IV. Finally, simulation results are shown in section V.

II. Logical Effort

The method of logical effort is founded on a simple model of the delay through a single MOS logic gate. The model describes delays caused by the capacitive load that the logic gate drives and by the topology of the logic gate. Clearly, as the load increases, the delay increases, but delay also depends on the logic function of the gate. Inverters, the simplest logic gates, drive loads best and are often used as amplifiers to drive large capacitances. Logic gates that compute other functions require more transistors, some of which are connected in series, making them poorer than inverters at driving current. Thus a NAND gate must have more delay than an inverter with similar transistor sizes that drives the same load. The method of logical effort (LE) quantifies these effects to simplify delay analysis for individual logic gates and multi-stage logic networks.
The first step in modeling delays is to isolate the effects of a particular integrated circuit fabrication process by expressing all delays in terms of a basic delay unit τ. Thus we express absolute delay d_{abs} as the product of a unit-less delay of the gate, and the delay unit d that characterizes a given process [1]:

$$d_{abs} = d\tau \quad (1)$$

Unless otherwise indicated, we will measure all times in units of τ which is about 50 ps in a typical process. The delay incurred by a logic gate is comprised of two components, a fixed part called the parasitic delay, p and a part that is proportional to the load on the output of the gatet, called the effort delay or stage effort$f$. The total delay, measured in units of τ, is the sum of the effort and parasitic delays. Thus

$$d = f + p \quad (2)$$

The effort delay depends on the load and the properties of the logic gates driving the load. We introduce two related terms for these effects: the logical effort g, captures properties of the logic gate, while the electrical effort h, characterizes the load. The effort delay of the logic gate is the product of these two factors.

$$f = gh \tag{3}$$

The logical effort captures the effect of the logic gate topology on its ability to produce output current. It is independent of the size of the transistors in the circuit. The electrical effort describes how the electrical environment of the logic gate affects performance and how the size of the transistors in the gate determines its load-driving capability. The electrical effort is defined by

$$h = {C_{out}}/{C_{in}} \tag{4}$$

where C_{out} is the capacitance that loads the logic gate and C_{in} is the capacitance presented by the logic gate at one of its input terminals. Many CMOS designers also call electrical effort as fanout. Combining Eqs. 2 and 3, we obtain the basic equation that models the delay through a single logic gate, which in time scale is

$$d = gh + p \tag{5}$$

This equation shows that logical effort g and electrical effort h both contribute to delay in the same way. This formulation separates g, h, p and τ the four contributions to delay. The process parameter τ represents the speed of the basic transistors. The parasitic delay p expresses the intrinsic delay of the gate due to its own internal capacitance, which is largely independent of the size of the transistors in the logic gate. The electrical effort h combines the effects of external load C_{out} with the sizes of the transistors in the logic gate. A typical value of the parasitic delay p_{inv} of an inverter is 1.0 as shown in Table 2. The logical effort, g expresses the effects of circuit topology on the delay free of considerations of loading or transistor size. Logical effort is useful because it depends only on circuit topology. Logical effort values for a few CMOS logic gates are shown in Table 1. Logical effort is defined such that an inverter has a logical effort of unity. This unit-less form signifies that all delays are measured relative to the delay of a simple inverter. An inverter driving an exact copy of it experiences an electrical effort of unity. Because the logical effort of an inverter is defined to be one, an inverter driving an exact copy of it will therefore have also an effort delay of one. Fig.1 shows the architectures of Inverter, NAND and NOR gates.

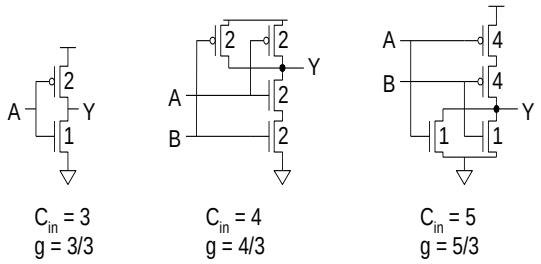

Fig.1. Simple gates architecture for (a) Inverter (b) two i/p Nand gate and (c) two i/p Nor gate

III. Multi-stage circuits

In a logic circuit, typically there are some cascade stages. In such a case optimal number of stages and proper size of each stage are important issues in determining the total delay of the circuit. Using logical effort we can calculate these parameters to achieve the best propagation delay for the whole circuit. In a multi-stage logic circuit, each stage has its own logical effort value. The total logical effort in a given path of the circuit can be called as the path logical effort (G) and can be calculated as [1][3][4][7][6]

$$G = \prod g_i \tag{6}$$

where g_i is logical effort of $i-th$ stage. Also the electrical effort of the path can be defined as

$$H = \frac{C_{out}}{C_{in}} \tag{7}$$

where C_{out} and C_{in} are the output and input capacitances of the path, respectively. For a multi-stage logic circuit, another parameter should be defined in order to cover fan-out of each stages. When a logic gate has other gates connected to its output which are not in the target path, a fraction of the output current is directed along the path while the rest is directed off that path. Here we define the branching effort (b) as

$$b = \frac{C_{on}+C_{off}}{C_{off}} \tag{8}$$

where C_{on} is the load capacitance along the path and C_{off} is the capacitance of the connections that lead off the path. If there is no branching, the branching effort is one. For a given path the branching effort of the path (B) is the product of the branching efforts of the gates (b_i) along that path. Then

$$B = \prod b_i \quad (9)$$

Finally the path effort (F) can be defined as

$$F = GBH \quad (10)$$

Now we can find the optimal value of the stage effort in order to have minimum delay along a given path. The path delay (D) is the sum of the delays of each stage and it can be divided into two parts consisting of the path effort delay (D_F) and the path parasitic delay (P) as

$$D = \sum d_i = D_F + P \quad (11)$$

where the path effort delay is

$$D_F = \sum g_i h_i \quad (12)$$

and the path parasitic delay is

$$P = \sum P_i \quad (13)$$

It can be proved that the path delay is least when each stage has the same stage effort. It means that the minimum delay is achieved when the stage effort is

$$f = g_i h_i = F^{1/N} \quad (14)$$

This is the most important concept in logical effort optimization. Using (14) the minimum achievable delay is

$$D = NF^{1/N} + P \quad (15)$$

Using (14) and the definition of the electrical effort we can determine the transistor sizes of gates along a path. To do that we start from the end of the path and move backward and apply the following capacitance transformation

$$C_{in} = \frac{g_i C_{out}}{f} \quad (16)$$

To get more insight about the technique we give here an example. A multi-stage logic network consisting of NAND gates is shown in Fig. 2. The last stage drives a capacitance, which is 4.5 times larger than input capacitance of the NAND gate at the input stage [2]. The objective here is to optimize the size of the gates to achieve minimum delay along the path from A to B. According to table 1, the logical effort of a NAND gate is 4/3 then the path logical effort is $G = (4/3)^3$. The branching effort for the first stage is $\frac{y+y}{y} = 2$ and for the second stage the value is $\frac{z+z+z}{z} = 3$. Then the path branching effort is $B = 2x3 = 6$. The path electrical effort is $H = \frac{4.5C}{C} = 4.5$. Thus $F = GBH = 64$ and the minimum delay is $D = 3(64)\left(\frac{1}{3}\right) + 3(2p_{inv}) = 18$ delay units. To achieve this delay we should have equal stage efforts. For 3 stages, the stage effort should be $(64)^{1/3} = 4$. To calculate the transistor sizes we start from the output using (16), z = 4.5C× (4/3)/4 = 1.5C. For the second stage y = 3z×(4/3)/4 = z = 1.5C. To check the correctness of the calculations we can use (16) for the first stage as well, which gives 2y×(4/3)/4 = (2/3)y = C, as given in the design specification. According to these calculations, the second stage should have 1.5 times larger transistors than the first stage and the last stage should have the same size as the second stage to achieve 18 units of delay [13].

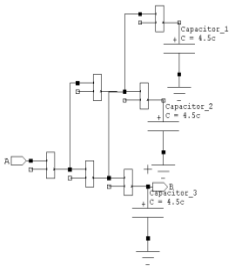

Fig.2: A multi-stage logic network

Table 1: Logical effort of static CMOS gates. (γ = 2,where γ is the ratio between PMOS and NMOS transistor size.)

Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				
NAND		4/3	5/3	6/3	(n+2)/3
NOR		5/3	7/3	9/3	(2n+1)/3
XOR, XNOR		4	12	32	

Table 2: Estimates of parasitic delay of different logic gate types assuming simple layout styles. A typical value of p_{inv} the parasitic delay of an inverter is 1.0.

Gate type	Number of inputs				
	1	2	3	4	n
Inverter	1				p_{inv}
NAND		2	3	4	np_{inv}
NOR		2	3	4	np_{inv}
XOR,		4	6	8	$2np_{inv}$

IV. Logical Effort of adder structures

In these structures, static CMOS gates have been used [12]. Figs.3 and 4 show the static implementation of XOR and MUX, Assuming $\gamma = 2$, all PMOS transistors in Figs. 3 and 4 should be two times larger than NMOS transistors to have equal rise and fall times. Also in the rest of our discussion we will assume that sizes of the transistors are in a way that the worst-case resistance for pull-up and pull-down network is the same as a minimum size inverter. In this case, all NMOS transistors in Figs. 3 and 4 are 2 times larger than a minimum size NMOS transistor. According to these assumptions, Logical Effort per input in both gates is 2.

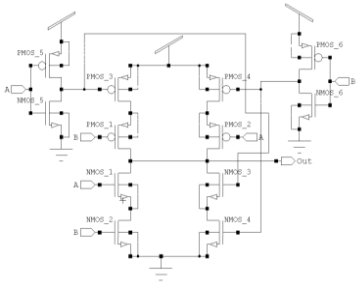

Fig.3 Static CMOS -XOR Gate

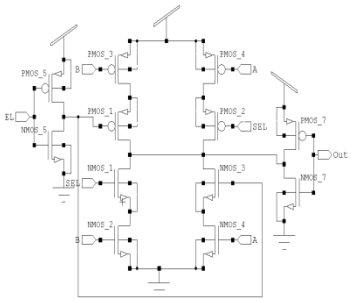

Fig.4 Static CMOS- MUX Gate

Fig. 5 shows a structure consisting of 4 XOR gates and two multiplexer [9].

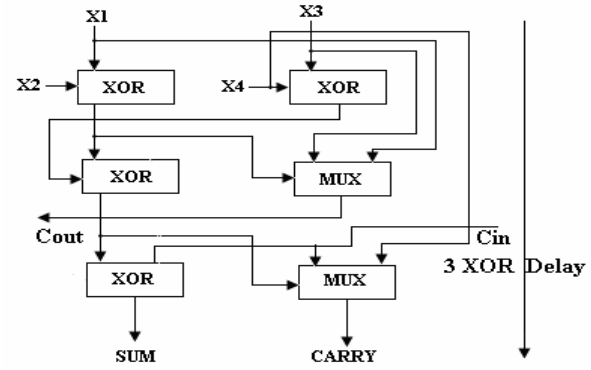

Fig. 5. Configuration 1 taken from [9]

Second configuration is introduced in [8] and is shown in Fig. 6. It uses only multiplexers.

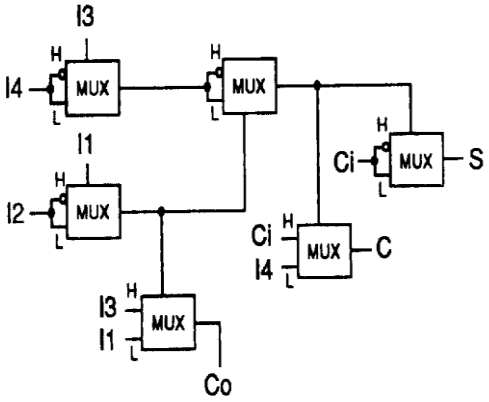

Fig. 6. Configuration taken from [8]

In [9], a structure consisting of 6 XOR gates and three multiplexer are introduced (shown in Fig. 7).

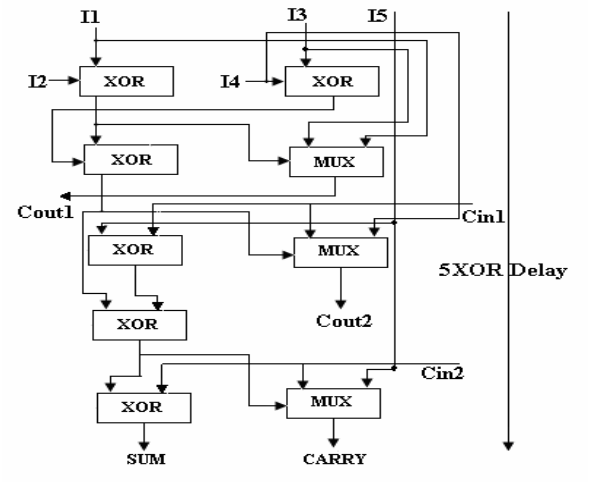

Fig. 7. Configuration taken from [9]

V. **Simulation Results**

Simulation results of the above mentioned configurations are shown in table 3. Optimization based on logical effort reduces the delay in all the configurations.

Table 3 Simulation results of three configurations before and after optimization

Design	Delay before optimization (in sec)	Delay after optimization using Logical Effort(in sec)
Configuration 1	1.8557e-010	1.2159e-010
Configuration 2	1.3033e-010	1.2874e-010
Configuration 3	3.9298e-010	3.7772e-010

Delay plots for the above three configurations of adders are shown in Fig.8.

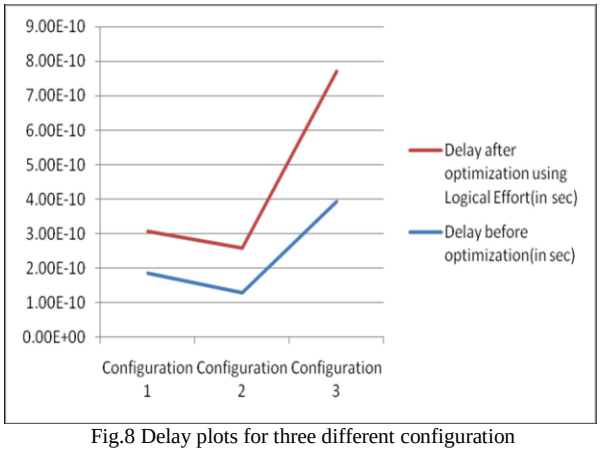

Fig.8 Delay plots for three different configuration

CONCLUSION

Use of Logical Effort methods for performance comparison of three different adder configurations were presented with wire capacitance included. Computed results are found consistent with simulation and are encouraging that configuration 1 are faster than configuration 2 & 3 in CMOS logic gates.

ACKNOWLEDGEMENT

We are thankful to the Dean, FET and HOD, Department of Electronics & Communication Engineering for providing us necessary permission to carry out this work.

REFERENCES

1. Sutherland, B. Sproull, D. Harris, "Logical Effort Designing Fast CMOS circuit", Morgan Kaufmann Publisher, 1999.

2. Yingtao Jiang et.al,"A Novel Multiplexer-Based Low-Power Full Adder", IEEE Transactions on Circuits and Systems -II: Express Briefs, vol. 51, no. 7, July 2004.

3. M. J. Sebastian smith, "application specific integrated circuits", Addision- Wesely 1997.

4. S.K.Mathew et al,"A 4GHz 130nm Address Generation Unit with 32-bit Sparse–tree Adder Core,"2002 Symposium on VLSI (2008), pp.281-285 April.

5. Kabbani, "Modeling and optimization of switching power dissipation in static CMOS circuits", IEEE Computer Society Annual Symposium on VLSI (2008), pp. 281–285 April.

6. J. Park et al., "470ps 64-Bit Parallel Binary Adder," 2000 Symposium on VLSI circuits Digest of Technical Papers.

7. H. Q. Dao, V. G. Oklobdzija, "Application of Logical Effort Techniques for Speed optimization and Analysis of Representative Adders," 35th Annual Asilomar conference on Signals, Systems and Computers, Pacific Grove, California, Nov 4–7, 2001.

8. V. G. Oklobdzija, "High-Performance System Design: Circuits and Logic", IEEE Press, 1999.

9. N. Ohkubo et, "A 4.4 ns CMOS 54×54 multiplier using pass-transistor multiplexer", in IEEE JSSC, vol. 30, No. 3, pp. 251-257, Mar. 1995.

10. K. Prasad and K. K. Parhi, "Low-power 4-2 and 5-2 compressors", in processing signals systems and computers, vol. 1, pp. 129-133, Nov. 2001.

11. Victor Adler and EBY G. Friedman, "Delay and Power Expressions for a CMOS inverter Driving a Resistive-Capacitive Load," Analog Integrated circuit and signal processing, 14, 29–39 (1997).

12. Chip-Hong Chang, Jiangmin Gu and Mingyan Zhang,"Ultra Low-Voltage Low-Power CMOS 4-2 and 5-2 Compressors for Fast Arithmetic Circuits," in IEEE transaction on circuit and systems-I, Vol. no. 51, 10, October 2004.

13. S. Anand and P. K. Ghosh, "Optimization and comparison of 4-stage inverter, 2-i/p Nand, 2-i/p Nor Gate by using Logical Effort," AIP Conf. Proc. November, 2010, Volume 1324, pp. 356-359