A Delay Improved Gate Level Full Adder Design

PADMANABHAN BALASUBRAMANIAN[§] and NIKOS E. MASTORAKIS[¶]
[§]School of Computer Science,
The University of Manchester,
Oxford Road, Manchester M13 9PL,
UNITED KINGDOM.
padmanab@cs.man.ac.uk
[¶]Department of Computer Science,
Military Institutions of University Education, Hellenic Naval Academy,
Piraeus 18539,
GREECE.
mastor@hna.gr

Abstract: - The binary full adder module is an important element present in processor data paths. The speed of computation that can be achieved in a processor data path is usually governed by the operating speed of the basic full adder. In fact, the full adder logic forms the basis of essential arithmetic operations like multiplication and division. Even subtraction in two's complement form is realized in a straightforward fashion using a linear cascade of full adders. Three novel gate level solutions, namely XNM, XNAIMC and XAC based full adder designs were presented in our earlier work. In this article, we present a succinct description of a further delay improved version. For a 32-bit **a**dder/**s**ubtractor **m**odule (ASM) constructed using the ripple carry configuration and incorporating the proposed **X**OR, **O**R, **A**ND and **c**omplex gates (XOAC) based full adder, corresponding speed improvement of 18.7%, 9.4% and 2.9% was achieved over the realizations utilizing the XNM, XNAIMC and XAC based full adders respectively. Comprehensive comparison with similar implementations encompassing only different gate level full adder designs further substantiate the speed efficiency of the proposed adder, all targeting the highest speed corner of the 65nm STMicroelectronics CMOS process.

Key-Words: - Combinational logic, Full adder, High performance, Standard cells, and Deep submicron design.

1 Introduction

A 1-bit full adder block consists of three input bits (say, *a*, *b* and *cin*) and produces two outputs (say, *sum* and *cout*), where '*sum*' refers to the summation of the input bits, *a*, *b*, and *cin*, where *cin* is the input carry from a least significant stage to the present adder stage. The overflow/output carry from this adder stage is represented as '*cout*'.

A number of full custom transistor level designs have been proposed for the full adder functionality [1] – [10], optimizing any or all of the design parameters – speed, power and area. In this article, our main discussion is on the proposed delay improved (better speed) gate level design for the full adder logic, utilizing the elements (cells) of a commercial standard cell library [11].

The remainder of this paper is organized as follows. Section 2 elucidates the motivation behind the proposed full adder design and gives the details. Section 3 reports the simulation results obtained. Concluding remarks are given in the next section.

2 Proposed XOAC Based Full Adder

The truth table of a 1-bit full adder is given in Table 1 and the fundamental equations governing the full adder's sum and carry outputs are listed following it. The sum output corresponds to the exclusive-OR operation performed on all the input bits, while the carry output basically conforms to majority logic, in that, if any of the two inputs have a similar logic state, then the output carry assumes the same state.

Table 1. Truth table of a binary full adder

Inputs			**Outputs**	
a	***b***	***cin***	***sum***	***cout***
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

$$Sum = a \oplus b \oplus cin \quad (1)$$

$$Cout = a \bullet b + b \bullet cin + a \bullet cin \quad (2)$$

The above Boolean equations for *sum* and *cout* can be implemented in a straightforward manner using only two gates of the standard cell library, as shown in fig. 1.

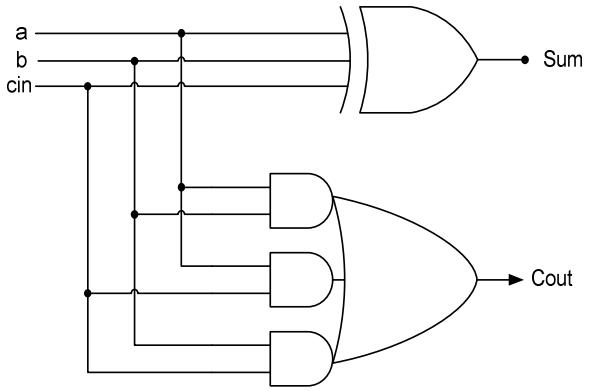

Fig. 1. Minimum gates based full adder design

It can be noticed from the above diagram that the sum logic has been realized using a single XOR3 gate, while the output carry logic has been implemented using a complex gate, namely AO222 cell. Though the above implementation may be optimal for the sum logic, it would not be so from the perspective of the carry output signal, as the carry signal (*cout*) has to propagate through the AO222 cell in every stage of the cascade, unlike the *sum* output, consequently experiencing higher propagation delay.

A high speed path for the carry output signal is facilitated in the full adder shown below, which is the XAC based full adder design presented in [15].

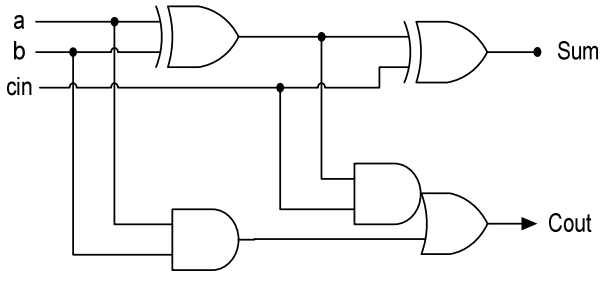

Fig. 2. XAC based full adder realization [15]

The equations corresponding to the above full adder are given by (3) and (4). Although the carry signal has to traverse through only the AO12 cell in every stage of the cascade, in case of carry-ripple architecture, the delay corresponding to the generation of the output carry signal in the least significant adder stage would be primarily the sum of the propagation delays of XOR2 and AO12 gates. Nevertheless, the design benefits from lesser area occupancy as there is some logic sharing between the sum and carry outputs.

$$Sum = (a \oplus b) \oplus cin \quad (3)$$

$$Cout = a \bullet b + (a \oplus b) \bullet cin \quad (4)$$

In order to minimize the propagation delay of the carry signal corresponding to the least significant adder stage, with a view to effect a slight further enhancement in speed, a minor logic optimization has been performed. As a result, the new *sum* and *cout* expressions are represented by (5) and (6) respectively. The resulting synthesized full adder functionality is depicted by fig. 3. This is referred to as the proposed XOAC based full adder design in this paper. It exploits the advantages inherent in the adder designs portrayed by fig. 1 and fig. 2.

$$Sum = a \oplus b \oplus cin \quad (5)$$

$$Cout = a \bullet b + (a+b) \bullet cin \quad (6)$$

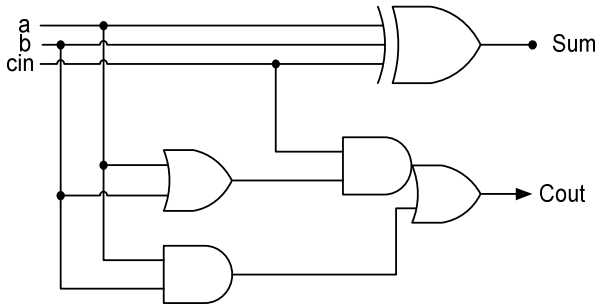

Fig. 3. Proposed XOAC based full adder

3 Evaluation using a 32-bit ASM – Simulation Mechanism and Results

Many different gate level full adder designs given in [4], [7] – [13] are taken up for evaluation, along with the multi-level NAND gates based full adder and the minimum gates based full adder designs. A 32-bit on-demand add/subtract module (ASM) [16] is used as the platform for analyzing the delay efficiency of the various full adder designs. An *n*-bit ASM, in its rudimentary form, consists of a cascade of full adder modules connected in a ripple carry fashion, as shown in fig. 4.

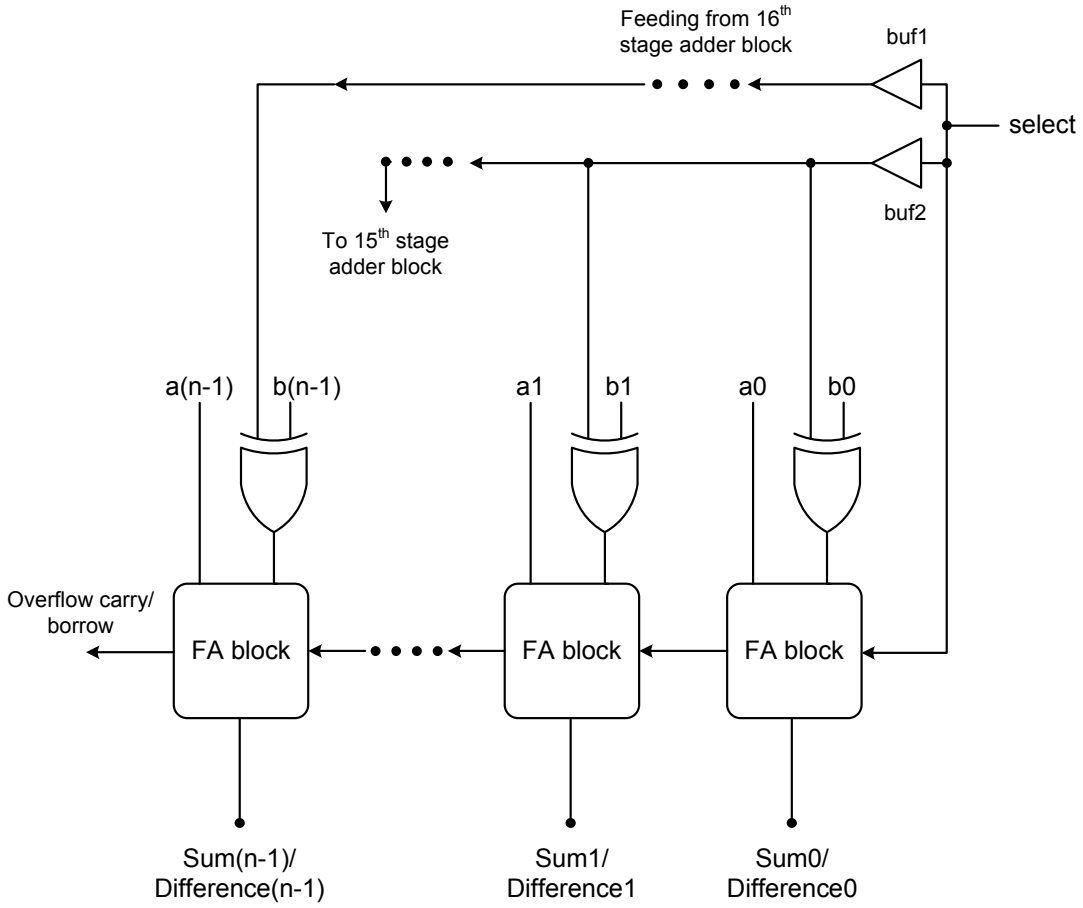

Fig. 4. An *n*-bit on-demand Adder/Subtractor module utilizing ripple carries architecture

The ASM performs either binary addition or two's complement binary subtraction based on the value of the *select* signal. If the *select* input is asserted low, then the ASM becomes equivalent to an *n*-bit carry-ripple adder, with the carry input to the least significant adder being a logic zero, producing sum and carry outputs. On the other hand, if the *select* input is asserted high, then the ASM performs binary subtraction in two's complement form producing difference and borrow outputs, with the one's complement of the subtrahend obtained through inversion using the bit-wise XOR operation and a binary '1' added to the least significant bit by means of a high carry input to the least significant full adder (FA) block. The *select* input needs buffering so as to be able to drive all the XOR gates, connected to all the full adders in the cascade; otherwise it would lead to timing violations. Therefore, the select input is first fed to two small buffer cells, namely *buf1* and *buf2* and then routed to the various XOR gates, as shown in fig. 4.

4.1 Simulation Mechanism

The different full adder designs were all described in cell level Verilog HDL, so that the physical implementation would be in exact conformity with the logical description. They were then instantiated to implement a 32-bit ASM, using the topology shown in fig. 4. The simulations were performed targeting the highest speed corner of all the best case electrical specifications of the 65nm STMicroelectronics bulk CMOS process [11]. The simulation results mentioned in the next sub-section purely reflect the performance and area metrics of the combinatorial logic and does not account for any sequential components. This sets the tone for a fair comparison of various full adders. Nevertheless, the target library is inherently optimized for low power.

Static timing analysis and cells area estimation were performed within the Synopsys PrimeTime environment. All the adder's inputs possess the driving strength of the minimum sized inverter in

Table 1. Relative increase in area occupancy for the different gate level 1-bit full adder designs in comparison with that of the single-bit full adder module present in the commercial library

1-bit full adder realization	Relative increase in area occupancy
Minimized SOP based full adder [12]	3.83×
Multi-level NAND based full adder [13]	2.22×
Half adders based full adder [13]	1.78×
Minimum gates based full adder	1.44×
Full adder embodying logic sharing [14]	2.67×
MUXes based full adder [4]	2.56×
XNOR-XNOR based full adder [7] – [9]	1.33×
XOR-XOR based full adder [7] – [9]	1.33×
Centralized full adder [8] [9]	1.78×
Shannon's theorem based full adder [10]	3.17×
XNM based full adder [15]	1.33×
XNAIMC based full adder [15]	1.72×
XAC based full adder [15]	1.56×
Proposed design – XOAC based full adder	1.83×

the library, while their outputs possess fanout-of-4 drive strength. Appropriate wire load was selected automatically for timing evaluation purpose.

4.2 Results and Inferences

The relative increase in area consumption for the different gate level 1-bit full adder designs, in comparison with the area occupied by a single-bit full adder block made available as part of the commercial library is listed in Table 1. It can be found that amongst all the full adder designs, the XNOR-XNOR, XOR-XOR and XNM based full adder realizations suffer from the least area expense (1.33×). Also, when compared to the proposed XOAC based full adder, they exhibit 27.3% lesser area consumption. However, speed is of higher significance than area.

Table 2 highlights the critical data path delay (or longest data path delay) for the 32-bit ASM implementation, incorporating various gate level full adder blocks, for the case of binary subtraction. The figures mentioned in brackets indicate the increase in maximum path delay for the different realizations in comparison with that implemented using the proposed XOAC based full adder module.

It can be seen that the proposed XOAC based full adder has contributed to the least critical path delay for the 32-bit ASM realization, featuring a maximum operating speed of 584.80MHz. This is considerably higher than the operating speed of the 32-bit ASM, implemented using the commercial library's full adder module by 34.5%. The ASM realization utilizing our earlier proposed XAC based full adder is competitive, in that, it reports a maximum operating frequency of 568.18MHz.

Table 3 gives the total cells area metric for the ASM implementations embodying different gate level full adder designs. The figures mentioned within brackets in the second column of Table 3 indicate the corresponding relative increase in area for the ASM implementations comprising the respective full adder modules, over that utilizing the commercial library's full adder. The ASM based on the commercial library's full adder module has the least area occupancy; mainly due to the observation made from Table 1. Though the minimum gates based full adder design might be interesting to note at the logic level (as it requires the least number of gates), it is 1.3× expensive than the former.

Table 2. Delay results corresponding to the highest speed corner of the 65nm CMOS process (V_{dd} = 1.35V, $T_{Junction}$ = -40°C)

On-demand 32-bit Adder/Subtractor module	**Critical path delay (ns)**
Minimized SOP based full adder [12]	3.05 (78.36%)
Multi-level NAND based full adder [13]	3.21 (87.72%)
Half adders based full adder [13]	2.48 (45.03%)
Minimum gates based full adder	2.81 (64.33%)
Full adder embodying logic sharing [14]	3.22 (88.30%)
MUXes based full adder [4]	2.04 (19.30%)
XNOR-XNOR based full adder [7] – [9]	1.95 (14.04%)
XOR-XOR based full adder [7] – [9]	1.95 (14.04%)
Centralized full adder [8] [9]	2.04 (19.30%)
Shannon's theorem based full adder [10]	2.92 (70.76%)
Commercial library's full adder [11]	2.30 (34.50%)
XNM based full adder [15]	2.03 (18.71%)
XNAIMC based full adder [15]	1.87 (9.36%)
XAC based full adder [15]	1.76 (2.92%)
Proposed design – XOAC based full adder	1.71

Table 3. Area metric for the 32-bit ASM implemented using various full adder modules

On-demand 32-bit Adder/Subtractor module	**Total cells area (μm^2)**
Minimized SOP based full adder [12]	1285.44 (2.94×)
Multi-level NAND based full adder [13]	802.88 (1.84×)
Half adders based full adder [13]	669.76 (1.53×)
Minimum gates based full adder	569.92 (1.30×)
Full adder embodying logic sharing [14]	936.00 (2.14×)
MUXes based full adder [4]	902.72 (2.07×)
XNOR-XNOR based full adder [7] – [9]	536.64 (1.23×)
XOR-XOR based full adder [7] – [9]	536.64 (1.23×)
Centralized full adder [8] [9]	669.76 (1.53×)
Shannon's theorem based full adder [10]	1085.76 (2.49×)
Commercial library's full adder [11]	436.80
XNM based full adder [15]	536.64 (1.23×)
XNAIMC based full adder [15]	653.12 (1.50×)
XAC based full adder [15]	603.20 (1.38×)
Proposed design – XOAC based full adder	686.40 (1.57×)

Among all the 32-bit ASM realizations, the implementations consisting of XNOR-XNOR, XOR-XOR and XNM based full adder blocks occupy the least area (only 1.23× more area) in comparison with that incorporating the commercial library's full adder. The ASM comprising of the proposed XOAC based full adder reportedly occupying 27.9% more area than these. Nevertheless, from the performance perspective, it is notably superior to the ASM constructed using XNOR-XNOR/XOR-XOR based and XNM based full adders by 14% and 18.7% respectively.

5 Conclusions

A delay optimized gate level full adder design has been presented in this work. For a 32-bit ASM constructed based on the carry/borrow-ripple configuration and incorporating the proposed XOAC based full adder, corresponding speed improvement of 18.7%, 9.4% and 2.9% was achieved over the realizations utilizing the XNM, XNAIMC and XAC based full adders respectively.

It is to be noted that the proposed full adder has facilitated a 32-bit ASM implementation, which is faster than that incorporating the commercial library's full adder by a significant margin of approximately 35%, which underlines the performance advantage enjoyed by the proposed XOAC full adder design. The simulations have all been performed targeting the efficient best case electrical specification of the 65nm standard cell library (inclusive of wire load information), as it is suited for facilitating high speed realizations. This paper adds value to the existing research on a fundamental data path element design.

References:

[1] N. Zhuang and H. Wu, A New Design Of The CMOS Full Adder, *IEEE Journal of Solid-State Circuits*, Vol. 27, No. 5, May 1992, pp. 840-844.

[2] N.H.E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design – A Systems Perspective*, 2nd Edition, Addison-Wesley Publishing, MA, USA, 1993.

[3] R. Shalem, E. John and L.K. John, A Novel Low-Power Energy Recovery Full Adder Cell, *Proc. ACM Great Lakes Symposium on VLSI*, pp. 380-383, 1999.

[4] M. Margala, Low-Voltage Adders For Power-Efficient Arithmetic Circuits, *Microelectronics Journal*, Vol. 30, No. 12, December 1999, pp. 1241-1247.

[5] A.M. Shams, T.K. Darwish and M.A. Bayoumi, Performance Analysis Of Low-Power 1-Bit CMOS Full Adder Cells, *IEEE Trans. on VLSI Systems*, Vol. 10, No. 1, February 2002, pp. 20-29.

[6] M. Zhang, J. Gu, C.H. Chang, A Novel Hybrid Pass Logic With Static CMOS Output Drive Full Adder Cell, *Proc. IEEE Intl. Symposium on Circuits and Systems*, pp. 317-320, 2003.

[7] Y. Jiang, A. Al-Sheraidah, Y. Wang, E. Sha and J.-G. Chung, a Novel Multiplexer-Based Low-Power Full Adder, *IEEE Trans. on Circuits and Systems – II: Express Briefs*, Vol. 51, No. 7, July 2004, pp. 345-348.

[8] S. Goel, S. Gollamudi, A. Kumar and M. Bayoumi, On The Design Of Low-Energy Hybrid CMOS 1-Bit Full Adder Cells, *Proc. 47th IEEE Intl. Midwest Symposium on Circuits and Systems*, vol. II, pp. 209-212, 2004.

[9] S. Goel, A. Kumar and M.A. Bayoumi, Design of Robust, Energy-Efficient Full Adders for Deep Submicrometer Design Using Hybrid-CMOS Logic Style, *IEEE Trans. on VLSI Systems*, Vol. 14, No. 12, December 2006, pp. 1309-1321.

[10] C. Senthilpari, A.K. Singh and K. Diwakar, Design Of A Low-Power, High-Performance, 8×8 Bit Multiplier Using A Shannon-Based Adder Cell, *Microelectronics Journal*, Vol. 39, No. 5, May 2008, pp. 812-821.

[11] STMicroelectronics CORE65LPLVT_1.10V Version 4.1 Standard Cell Library, *User Manual and Databook*, July 2006.

[12] R.K. Brayton, A.L. Sangiovanni-Vincentelli, C.T. McMullen and G.D. Hachtel, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, MA, USA, 1984.

[13] M. Morris Mano, *Digital Design*, 3rd Edition, Prentice-Hall Inc., NJ, USA, 2002.

[14] J.P. Uyemura, *CMOS Logic Circuit Design*, 1st Edition, Springer, 1999.

[15] P. Balasubramanian and N.E. Mastorakis, High Speed Gate Level Synchronous Full Adder Designs, *WSEAS Transactions on Circuits and Systems*, Vol. 8, No. 2, February 2009, pp. 290-300.

[16] M. A. Karim and X. Chen, *Digital Design: Basic Concepts and Principles*, CRC Press, Boca Raton, FL, USA, 2007.