Information Sciences Letters

Volume 2
Issue 3 *Sep. 2013*

Article 5

2013

A New network multiplier using modified high order encoder and optimized hybrid adder in CMOS technology

Pooya Asadi
Department of Computer, Varamin-Pishva Branch, Islamic Azad University, Varamin, Iran,
p_asadi@iauvaramin.ac.ir

Inf. Sci. Lett. **2**, No. 3, 159-164 (2013)

Information Sciences Letters
An International Journal

http://dx.doi.org/10.12785/isl/020305

A New network multiplier using modified high order encoder and optimized hybrid adder in CMOS technology

Pooya Asadi *

Department of Computer, Varamin-Pishva Branch, Islamic Azad University, Varamin, Iran

Received: 3 May. 2013, Revised: 3 Aug. 2013, Accepted: 5 Aug. 2013
Published online: 1 Sep. 2013

Abstract: In this paper, a new low power, high speed network multiplier is presented. For increasing performance of multiplier, a novel modified high-order encoder is proposed. Previous encoders have complicated hardware and their ability to decrease number of input operands is low. Presented encoder uses high-order algorithm and therefore reduces number of partial products efficiently. A new hybrid adder is presented which uses ideas of carry lookahead adder and ripple carry adder to modify final adder architecture. Previous carry lookahead adders have large carry network and their ability to decrease noise margin is low. It uses a DCVS carry network, which provides high speed and less wiring problems in compare with previous algorithms. Proposed hybrid adder has major effect on multiplier efficiency. A new network array is presented which uses high performance modules. Using a new algorithm, this study reduces critical path of tree multiplier. A novel counter is implemented which uses less transistor count and less power consumption in compare with previous algorithms. This counter uses pass transistor technology. Transistor connections, in mentioned counter are implemented in a new efficient way. The new multiplier has better efficiency in different electronic and algorithmic parameters in compare with previous implementations.

Keywords: adder, CMOS, computer architecture, multiplier

1 Introduction

The desire for high-speed calculation has been increasing with developing computer systems. Higher frequency is the key to increasing system efficiency, especially in DSP and graphics. One of the most essential and important operators is network multiplier. High-speed multipliers are required for improving of pipeline methods (Asadi, 2012). This study describes a 64*64-bit network multiplier that is implemented for high-speed systems such as graphic and arithmetic processors. One of the important factors is improved network architecture. Another factor is the VLSI layout, which has been obtained by using 60 nm CMOS second-level-metal implementation. By using three metal wirings, compact layout structure has been designed with minimum wiring effects. In the following sections, details of the multiplier architecture are proposed. One of the design methods for high-speed multipliers is to design high performance network and reduce the number of processing steps. It is well known that both the modified high-order encoder (Wang, et. al, 2011) and the network structure are important in reducing processing steps. In the Dadda network array, it is important that it is the most suitable scheme in decreasing propagation steps in a tree regardless of its complicated wiring implementation (Kuang, et. al, 2009). The Dadda network has been widely used in different operators, particularly those with less than 64 bit (Chen, et. al, 2012). Multiplier architecture is shown in Figure 1. It uses four to counter chain in different rows. Different contributions have been presented for different parts of Figure 1.

2 Pass transistor counter using DCVS technology

The minimum number of transistors for producing the C_{out} output is three and it uses 10 transistors, but it lacks the current wiring issue. The new gate is based on low-power

* Corresponding author e-mail: p_asadi@iauvaramin.ac.ir

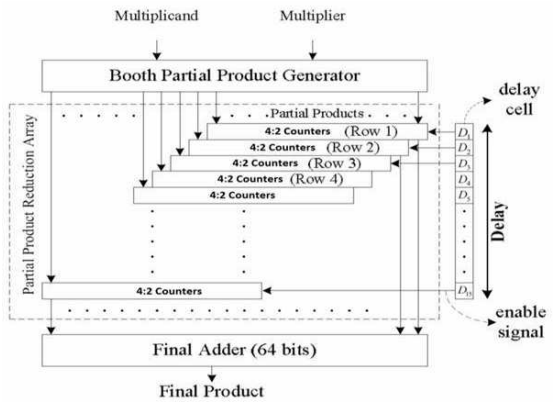

Fig. 1: Multiplier architecture

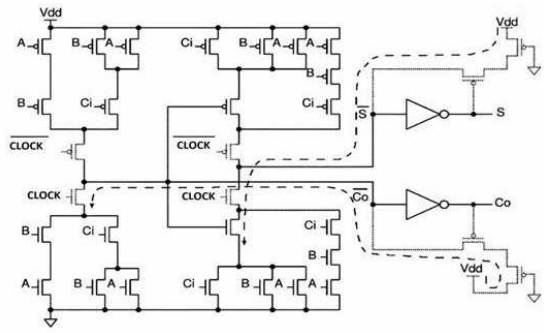

Fig. 2: Mirror CMOS counter

DCVS logic method, as shown in Figure 2 (Kuang, et. al, 2010). Its logic equation is given by

$$Z = X \cdot Y = \sum_{j=0}^{15} \left(\sum_{i=0}^{15} (X_i Y_j) 2^{(i+j)} \right)$$

This logic has obtained the efficiency of low power DCVS technology, which has been presented in (Seidel, et. al, 2005). It is effective in performance for pass transistor technologies. Its ability against current wiring and transistor technology such as complicated noise margin enables it to work consistently at low current and 60 nm transistor technology. It has been studies to synthesize the counter module as a single module in simulation (Seo, et. al, 2010). The efficiency of the architecture will be reduced radically and high power consumption at low-supply current increases. For this reason, the counter modules cannot be chained without additional gates connected to the outputs of each module (Asadi, 2012). This will be further synthesized. The circuit architecture, which is made of four chained counter modules, is shown in Figure 2. This architecture simulates the gates like regular operators and binary counters that use compressor modules as the building structure. All the needed input-signal to output signal transitions are computed in the test equations.

The idea of using higher radix compressors develops itself into the use of four to two and higher such as 32:6 (Juang, et. al, 2005). The advantage of using higher radix compressors is mainly in more compression and regular structure, while a disadvantage is the fact that due to their technology, performance of their use reduces with the size of the compressor. The architecture of the four to two compressor is shown in Figure 2. In the first step, we have four to two compressors decreasing the number of operands to four, which is summed into two with one step of seven to three compressors. Our conclusion is that we can design a vertical chain in the same time as horizontal, and we know that this is the best way we can use. In the case of counters being used instead of a compressor, the size will be calculated by the time it takes to chain sum output. We have selected a 4-bit counter for our simulation. However, horizontal latency of the circuit is three XNOR gates, which is better than that of the six to two counter. A partial product input including 4-bit counters is presented in Figure 3. Critical path for a 64-bit multiplier network using 6-bit counters is calculated between five and seven XNOR delays relating on how fast sum output can be chained. The signal output from the multiplier network using 6-bit counters is shown in Figure 3.

3 A hybrid adder with carry lookahead modules

A hybrid adder is a combination of the new lookahead method and skip blocks. The basic idea is to take the entire algorithm of a carry-lookahead adder and change the individual modules from carry-skip kind to hybrid type. Once again, we do not require to make two forms of a carry select block. The top line of hybrid adder would calculate c_i as $c_i = s_i + g_i + 1$. Sum bit calculation logic in a hybrid adder is $s_i = p_i + (h_{j,k-1} + c_{pl} \cdot g_{jl})$, where g_{jl} are the block-produce and block chain signals for the bit positions 1 through (k-1) in jth module. Although we have used carry lookahead blocks in this presentation. Let us design a block-skip method for the hybrid adder. Let c_{i0} present the carry at the ith bit location with carry-in $c_{i0} = 0$ ($c_{i0} = 1$). This indicates a reasonable way to calculate c_{i1} from c_{i0}. How can we incorporate this method into a hybrid adder with carry-propagation synthesis modules? Notice that it shows that for any bit location i, $c_{il} = c_{il-1} + c_{i0} \cdot p_i$. The calculation of p_i needs an additional circuit and gate per bit block. Each bit part includes the following logic

$$S = P \oplus Ci$$
$$Co = PCi + P'A$$
$$\overline{S} = PCi + P'Ci'$$

Inf. Sci. Lett. **2**, No. 3, 159-164 (2013) / www.naturalspublishing.com/Journals.asp

Table 1: Comparison of different code's generator

	Present Study	(Muralidharan, et.al, 2011)	(Cho, et. al, 2011)
Technology (*nm*)	60	60	60
Transistor counts	520	890	970
Multiplication time (ns)	6.2	9.3	8.7
Chip Area (mm^2)	12	19	17
Power Diss. (mW)	14.5	18.9	23.4

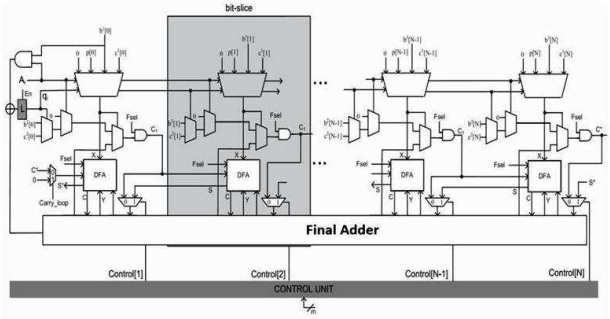

Fig. 3: Lookahead adder network

C_{pb} is the carry signal of the most-significant bit of the previous module. Table 1 shows comparison of different code's generator. Note that since each module is calculating with module carry-out of one, the carry into the LSB location of each module is 0, i.e., $s_i = p_i + c_{p??}$ and $c_i = g_i$ when it is the least-significant-bit of a module. Note that $s_i = p_i + c_i - 1$, but c_{i-1} is $(\overline{c_{pb}}.g^0_{i-1}) + (c_{pb}.g^1_{i-1})$. A decreased form for this equation c_{i-1} is $(\overline{c_{pb}}.g^i_{i-3}) + (c_{pb}.g^i_{i-3})$, this is $(c_{pb}.g^1_{i-1})$which is equal to $(c_{pb}.g^1_{i-1})$. For each module, we also require to determine the final value of the sum-in from the most-significant bit location, which is used to calculate from two copies of the next module. Let c_{bl} show the sum-out from the higher bit position of the lth module, for 1<=l<=k (there are k units). Once again, the calculation logic needed c_{bl} where it is the module generate for the lth unit for 2<=l<=k.

$$B = |d[B] + 1|_{2^n+1}$$
$$= \left| (1 + b_0 - 2b_1) + \sum_{i=1}^{\lfloor n/2 \rfloor} (b_{2i-1} + b_{2i} - 2b_{2i+1}) 2^{2i} \right|_{2^n+1}$$

The basic Wallace carry chain unit is presented in Figure 4. Outputs are obtained from the appropriate points in the network. In order to make the Wallace carry propagation as fast as possible, each transistor chain is

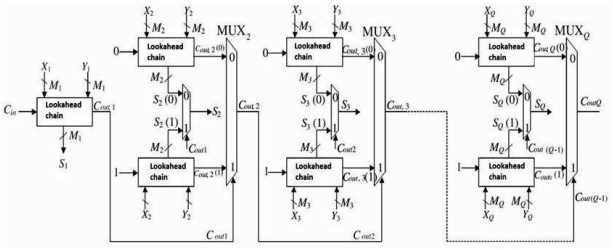

Fig. 4: Hybrid carry network

designed to approximately result the bit module where it is substituted. The second step (and higher level if required) P_{cc} units are substituted in left holes in the expression of second step P_{cc} units to decrease the width of the wiring. Some equations of the Wallace carry chain module are possible since not all of the inputs were used in all of the blocks. In the basic Wallace carry propagation presented in Figure 4 the expression of chain transistors used to compute the block chain differs from the line of transistors used for chain the produce signals only in the wiring of P_0. This removes the intermediate inputs. The carry-in shown in Figure 4 contains three extra transistors for calculating carry out. This carry out expression is used once at the top of the array. Multiplier architecture is shown in Figure 1. It uses four to two counter chain in different rows. Different contributions have been presented for different parts of Figure 1.

4 The Network propagation tree

The counter floor plan is presented in Figure 4. Metal three is used for long vertical lines carrying the input signals, the bit chain and produce inputs, the computed carries, and the outputs. Metal two runs horizontally, and is used for local signals. The third module is a stack of three bit V_{cc} cells connected in pairs to form seven bit carry lookahead modules. A ONE carry is input to each block in this row. The fourth row consists of propagate carry modules with a carry in of ZERO. Next are the exclusive-NOR gates for computing the sums from the chain carries and the bit sums. The forth row is the carry array. The last row consists of counters, which select between the seven bits sum outputs. The idea of this multiplier is based on the architecture of a low-power multiplier with modified Booth algorithm (Cho, et. al, 2011). A Booth encoder using counter array requires nearly half the size of a conventional array multiplier and decreases delay. One method to modify this circuit structure is to improve the final chain adder structure, thereby decreasing the carry delay. In our circuit simulation, a dynamic counter with P_{cc} is used (Figure 4). This calculates the data flow of the multiplier in a modified new method. The final multiplication output is

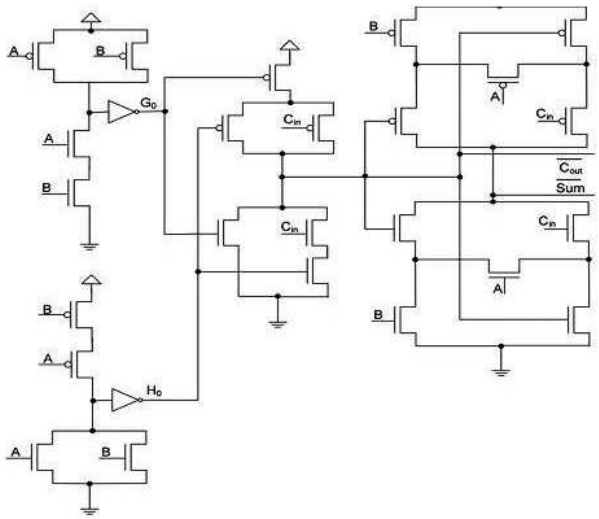

Fig. 5: Three to two compressor

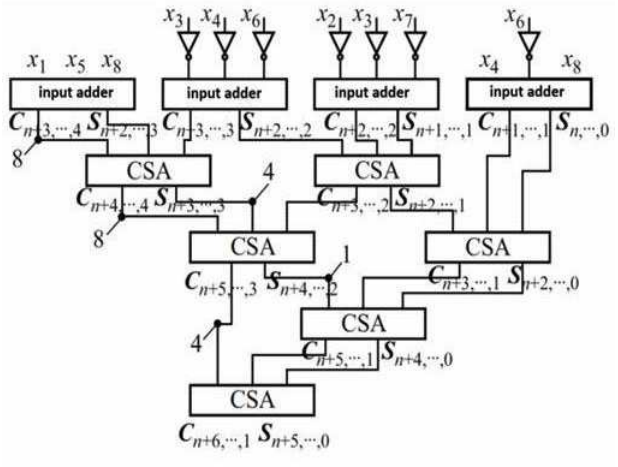

Fig. 6: High-speed summation tree

computed in the subsequent synthesis phase. This multiplication algorithm incorporates some timing constraints since there are nearly N/2 dynamic signal and N static sum delays. Circuits simulated on the gate synthesis have the following timing constraints

$$t_{-1} = \begin{cases} 1 & -\mu\,(x_{n-1}+y_{n-1}) < 0 \\ 0 & -\mu\,(x_{n-1}+y_{n-1}) \geqslant 0 \end{cases}$$

This includes a multiplication frequency at 300 MHz for arithmetic processor. The area of the 32-bit * 32-bit multiplier takes 5.6 mm^2. In this way, it is more efficient to compute the P_{cc} counter than the dynamic adder array. A comparison with a new implementation for low power multiplier structure is presented in Figure 4 (Chen, et. al, 2012). A 64-bit *64-bit multiplier and a 108-bit adder have been simulated in arithmetic chip. The chip implements a part of the dynamic counter cell. Simulation results of both counter and modules have been verified with 60nm CMOS technology. The array adds eight trees and six compressors (eight-expressions). It will run with 300-MHz arithmetic processor.

5 Efficient array elements

The circuits for CMOS counters and XOR gates are well known (Muralidharan, et.al, 2011), and the standard implementations were used for these modules with the dimensions computed to provide the appropriate current output. The XNOR gate and compressor modules effectively included three gates in one module. These kinds of module were used often in implementation and incorporating three gates in one module consumed less

Table 2: Comparison between 64×64 bit multipliers

Multipliers	Present Study	(Wang, et. al, 2011)	(Nakamoto, et. al, 2011)
Technology (*nm*)	60	60	60
Transistor counts	6400	9200	11250
Multiplication time (ns)	17.9	23.4	19.8
Chip Area (mm^2)	62	79	82
Power Diss. (mW)	36.3	48.5	42.9

area than three separate modules. The exclusive-NOR gate proved more complicated implementation task.

The structure of the design obtains high speed and low power. The traditional DCVS exclusive-NOR gate was rather slow and is high-power consumption unit, so a variation on the high speed CMOS exclusive-NOR was used. The algorithm of the circuit is such that the internal connection driving the output gate could overshoot V_{dd} by high noise. This issue may be explained by simulating the voltages across the gate-output capacitances (C_{gd}) of the p-channel transistors n_1 and n_2, as shown in Figure 5. When both signals change from low to high, the n-channel transistors p1 and p2 will shut down and removes the V_{int} node. The voltage between the gate-output capacitances increases at V_{dd}, but the gate input is now near the V_{dd} rather than one circuit so the node V_{int} increases above V_{dd}. The maximum power occurs if the three signals change simultaneously and increases as the time between the three rising sides is increased. The results of a Cadence simulation of the exclusive-NOR module are presented in Figure 5. V_{int} is clearly seen to increase V_{dd}; it is therefore important to connect a guard ring into this module in order to remove the possibility of occurring noise problems. As stated previously, all the modules used in this implementation incorporated signal strength in any case, so decreased noise problems would have taken effect. As mentioned

Inf. Sci. Lett. **2**, No. 3, 159-164 (2013) / www.naturalspublishing.com/Journals.asp

earlier, pass transistors based counters are inherently high speed implementations and consume less power. If the noise margin can be enhanced, they tend to be suitable alternatives in array architecture structures. In order to modify the noise margin, additional gates are needed at each output of the counters. With applying only one buffer at the output, total power consumption can be decreased while keeping a sufficient output capability. The adder inserted circuit can be designed using mentioned expressions. According to it, another buffer is needed to get a converted signal in B. If the converted signal is used as an input, the counter implementation can be optimized. In the presented design, the converted signal of C_i is used as an input. The rewritten counter logical functions are

$$AB = \left(2^{N/2}C + D\right)\left(2^{N/2}E + F\right)$$
$$= 2^{N}(CE) + 2^{N/2}(CF + DE) + DF$$

The implementation has one converted input and one complementary output, but in a compressor, there are seven inputs and three outputs. If this issue cannot be solved, buffers are required to be inserted to make the circuit module correctly. Two kinds of pass transistors based counters can be implemented. In every counter, both sum and the converted output of sum can be achieved by applying different interconnections. Either carry or the converted sum is available depending on the signals. XNOR and adder are basic modules for a transmission gate based counter, as shown in Figure 5. Efficient XNOR and adder circuits are presented in Figure 6 and

$$c_i = y_{i-1} + x_{i-1}\left(y_{i-2} + x_{i-2}\left(\ldots\left(y_1 + x_1 y_0\right)\ldots\right)\right)$$

The chained counter architecture is the most reliable way to design a multiplier, which uses a set of counters with shifted signals as shown in Figure 6. All counters have one input connecting to the multiplicand and the other signal connecting to its previous step with three bit shifting. Obviously, the larger partial generator hardware is, the longer counter propagate will computed. As an example with eight partial generators presented in Figure 6, eight counters are used. It is easy to see that the number of counter increases linearly with increasing number of partial generators. Therefore, the size of multiplier increases.

6 Conclusion

In this paper, a new network multiplier is presented which uses high-order encoder and optimized hybrid adder in CMOS technology. Three modifications have been implemented in new multiplier in compare with previous algorithms. A new network array is presented. For partial product reduction step of algorithm different tree and array methods are presented. Tree algorithms have high speed but they have complicated hardware. Array algorithms have large hardware but have regular structure. Network array presented in this paper provides regular wiring, less hardware complexity and high speed in compare with previous implementations. First step of a multiplier algorithm is partial product generation. Conventional implementation of this step uses two strategies. Low order encoders use small hardware but they have low speed. High-order encoders have large and complicated hardware but they have high ability to decrease number of partial products. In this paper, a novel high-order encoder is proposed which has high-speed, reduced hardware complexity and high ability to decreasing number of partial products in compare with previous algorithms. A new hybrid adder is presented which uses combination of two conventional algorithms. Carry lookahead adder has a complex carry network and ripple carry adder has a long critical path. With combining these algorithms this paper presented a new hybrid adder with efficient carry network and decreasing noise problems. Presented multiplier has increased speed at 10 percent, reduces power at 12 percent and decreases transistor count at 8 percent in compare with previous algorithms. These comparisons have been shown in Table 2.

References

[1] Asadi, P., A New Optimized Tree Structure in High-Speed Modified Booth Multiplier Architecture, American Journal of Scientific Research, **52**, 48-56 (2012).
[2] Muralidharan, R., et al, Radix-8 Booth Encoded Modulo 2ˆn-1 Multipliers with Adaptive Delay for High Dynamic Range Residue Number System, IEEE Transactions on Circuits and Systems I: Regular Papers, **58**, 982-993 (2011).
[3] Chen, Y. H., et. al, A High-Accuracy Adaptive Conditional-Probability Estimator for Fixed-Width Booth Multipliers, IEEE Transactions on Circuits and Systems I: Regular Papers, **59**, 594-603 (2012) .
[4] Cho, K. J., et. al, Fixed-Width Modified Booth Multiplier Design Based on Error Base on Error Bound Analysis, Multimedia, Computer Graphics and Broadcasting, **263**, 248-256 (2011).
[5] Juang, T. B., et. al , Low-Error Carry-Free Fixed-Width Multipliers with Low-Cost Compensation Circuits, IEEE Transactions on Circuits and Systems II: Express Briefs, **52**, 299-303 (2005).
[6] Kuang, S. R., et. al, Design of Power-Efficient Configurable Booth Multiplier, IEEE Transactions on Circuits and Systems I: Regular Papers, **57**, 568-580 (2010).
[7] Kuang, S. R., et. al, Modified Booth Multipliers with a Regular Partial Product Array, IEEE Transactions on Circuits and Systems II: Express Briefs, **56**, 404-408 (2009).

[8] Nakamoto, R., et. al, 4-bit SFQ Multiplier Based on Booth Encoder, IEEE Transactions on Applied Superconductivity, **21**, 852-855 (2011).
[9] Seidel, P. M., et. al, Secondary Radix Recodings for Higher Radix Multipliers, IEEE Transactions on Computers, **54**, 111-123 (2005).
[10] Seo, Y. H., et. al, A New VLSI Architecture of Parallel Multiplier-Accumulator Based on Radix-2 Modified Booth Algorithm, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, **18**, 201-208 (2010).
[11] Wang, J. P., et. al, High-Accuracy Fixed-Width Modified Booth Multipliers for Lossy Applications, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, **19**, 52-60 (2011).