# Glitch Reduction In Low Power TG-Multiplier

V. V. M.Krishna[1], M. Venkata Subbarao[2] and Dr. V. Venkatarao[3]
[1] *Assoc. Professor, Department of ECE, Tirumala Engineering College, Narasaraopet, A.P, India*
[2] *Asst. Professor, Department of ECE, Tirumala Engineering College, Narasaraopet, A.P., India*
[3] *Professor, Department of ECE, Narasaraopet Engineering College, Narasaraopet, A.P, India*

## Abstract

*Since multiplication is one of the most critical operations in many computational systems, there have been many algorithm proposed to perform multiplication, each offering different advantages and having tradeoffs in terms of speed, circuit complexity, area and power consumption. This paper focuses an algorithm of a multiplexer-based multiplication method, an efficient algorithm which is applicable to low power applications. And it has been proved that the multiplexer-based multiplier out performs the modified Booth multiplier both in speed and power dissipation by 13% to 26%, due to small internal capacitance. After analyzing the performance characteristics of conventional multiplier types, it is observed that the one designed using multiplexer-based multiplication algorithm is more advantageous, especially when the size of the multiplied numbers is small. In order to verify the superiorities of this algorithm, we performed an implementation, in which the bit size of the multiplicand and the multiplier is comparably large.*

## 1. Introduction

In the signal processing offered in modern audio applications, multipliers are certainly among the most power-hungry elaboration units. At the same time, they are very frequently used components in application-specific integrated circuits (ASICs) and fundamental blocks in digital signal processors (DSPs) [2].Being rather complex combinational modules with numerous unbalanced reconvergent paths, multipliers suffer particularly from spurious switching activity generation and propagation [1], which can even dominate the total dynamic consumption. While trying to optimize the efficiency of multipliers, many works in the past [3]–[5] investigated only the basic constitutive cell, namely the full-adder. This way of proceeding overlooks the previously-mentioned relevant aspect of glitch propagation and does not take wire parasitics into account either.

The easiest solution to reduce spurious activity propagation is certainly pipelining. Yet, the large power and area overheads due to the introduction of flip-flops (FFs) limit its use to high speed implementations, as in [6]. Apart from that, three fundamental approaches have been proposed in the literature so far to abate glitch generation and propagation in parallel multipliers, namely:

1) Shortening full-adder chains;
2) Equalizing internal delays;
3) Aligning sum and carry signals.

The first technique consists in rearranging the full-adder cells in order to carry out the same operation within shorter paths [7]. The advantage is that fewer glitches are generated and propagated when this can be done with no extra logic, as in a Wallace tree.

In the second technique, the delays of the internal signals are equalized by redesigning the full adders [8]. The efficiency is generally dependent on parasitic and process variations.

The third technique consists in the alignment of the internal signals by means of self-timed circuits. The independent delay line triggers special cells that implement the functionality of both a full-adder and a latch [9]. These circuits present superior glitch suppression. However, large energy overhead and strong process dependence represent a heavy burden.

Two more general techniques for glitch suppression, which do not specifically address multiplier architectures, have been proposed. The first one acts on transistor sizes to adjust the cell delays, in order to balance reconverging paths, hence reducing glitch generation. The second implements a special resistive cell to increase internal ramp times.

Compared to these two low-power strategies, the hereby introduced technique presents the following advantages:

1) It limits the area increase, which is relevant in;
2) It can do without large consuming transistors, needed by;
3) It is more robust to process and voltage variation.

This confirms the relevant power efficiency of the Wallace tree over other traditional structures, by presenting a comprehensive study on the spurious activity propagation. The effect of transistor sizing is also evaluated: in low-frequency low-voltage applications, minimum-size devices

decrease the switching capacitance without leading to large crossover currents.

Based on these results, new multiplier architecture is introduced, called TG-Multiplier that reduces spurious activity further compared with both traditional and recent architectures. At the same time, TG-Multiplier has positive effects on leakage reduction and it is robust to process variation and voltage scaling, without imposing any overhead in terms of energy. The introduced technique combines static CMOS with transmission gates that abate glitches via resistance–capacitance (RC)-equivalent low-pass filtering.

This work confirms the relevant power efficiency of the Wallace tree over other traditional structures, by presenting a comprehensive study on the spurious activity propagation. The effect of transistor sizing is also evaluated: in low-frequency low-voltage applications, minimum-size devices decrease the switching capacitance without leading to large crossover currents. Based on these results, a new multiplier architecture is introduced, called TG-Multiplier, that reduces spurious activity further compared with both traditional and recently published architectures. At the same time, TG-Mult has positive effects on leakage reduction and it is robust to process variation and voltage scaling, without imposing any overhead in terms of energy. The introduced technique combines static CMOS with transmission gates that abate glitches via resistance–capacitance (RC)-equivalent low-pass filtering. Additionally, it guarantees limited overhead of propagation delay and area, hence finding potential application in low-frequency portable devices, such as hearing aids.

The reminder of this paper is organized as follows. Section 2 introduces the multiplexer based multiplication algorithm. Section 3 shows the circuit structure of multiplexer based multiplier. Section 4 compares recently published with traditional architectures. In the same section, the new multiplier TG-Mult is introduced. Measurements are presented in Section 5. After the discussion of the results in Section 6, Section 7 draws the conclusions. Eventually, the Appendix shortly describes a new practical methodology to determine spurious activity through simulations.

## 2. Multiplexer Based Algorithm

The algorithm is a different version of effective parallel multiplication, so one bit of the multiplier and the multiplicand are processed in each step. The multiplicand and the multiplier are interchangeable since the algorithm is symmetric. Parallel implementation of the algorithm results in a smaller circuit by means of area and it provides faster addition of partial products. Thus, its circuitry complexity is almost the same as the implementations based on Modified Booth's algorithm, but the multiplication time is considerably faster. These advantages are valid for both positive numbers and numbers in two's complement form.

Consider the multiplication of two n-bit numbers X and Y, where

$$X = X_{n-1}X_{n-2}\ldots\ldots\ldots X_2X_1X_0 = \sum_{J=0}^{n-1} X_J\, 2^J$$

$$Y = Y_{n-1}Y_{n-2}\ldots\ldots\ldots Y_2Y_1Y_0 = \sum_{J=0}^{n-1} Y_J\, 2^J$$

As derived in , based on these two equalities, the numbers $X_{n-1}$ □ and $Y$□$n$-1 can be defined as

$$X_{n-1} = X_{n-2}\ldots\ldots\ldots X_2X_1X_0 = \sum_{J=0}^{n-2} X_J\, 2^J$$

and $X = X_{n-1} + 2^{n-1}X_{n-1}$

$$Y_{n-1} = Yn_{-2}\ldots\ldots\ldots Y_2Y_1Y_0 = \sum_{J=0}^{n-2} Y_J\, 2^J$$

and $Y = Y_{n-1} + 2^{n-1}Y_{n-1}$

Thus, the product P of X and Y can be written as

$P = X . Y$
$P = \{2^{n-1}X_{n-1} + X_{n-1}\} . \{ 2^{n-1}Y_{n-1} + Y_{n-1}\}$
$P = 2^{2n-2} X_{n-1}Y_{n-1}+2^{n-1}\{ X_{n-1}Y_{n-1}+ X_{n-1}Y_{n-1}\}+X_{n-1}Y_{n-1}$

By the definition of $P_{n-1} = X_{n-1} Y_{n-1}$ and $P_j = X_j Y_j$ where $X_j$ and $Y_j$ represent the $j^{th}$ least significant of X and Y, the product P can be written as

$P_J = X_J Y_J$
$= 2^{2J-2}X_{J-1}Y_{J-1}+2^{J-1}\{ X_{J-1}Y_{J-1}+X_{J-1}Y_{J-1}\} + X_{J-1} Y_{J-1}$
$= 2^{2J-2} X_{J-1} Y_{J-1} + 2^{J-1} \{ X_{J-1} Y_{J-1}+X_{J-1} Y_{J-1}\} + P_{J-1}$

Hence,

$$P = \sum_{J=0}^{n-1} X_J Y_J\, 2^{2J} + \sum_{J=1}^{n-1} \{ X_J Y_J + X_J Y_J \}.2^j$$

$$P = \sum_{J=0}^{n-1} X_J Y_J\, 2^{2J} + \sum_{J=1}^{n-1} \{ Z_J\, 2^J \}$$

Where $Z_J = X_J Y_J + X_J Y_J$

From the above equation we can say the multiplication can be devided into two operations

•Grouping of partial products are distinguished by connecting them with solid lines

•Folding the array along the line of symmetry gives the final form of the algorithm.

The values of Zj, which is dependent to xj and yj, are shown in Table 1.

Table 1: Truth Table for $Z_J$

| $X_J$ | $Y_J$ | $Z_J$ |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | $X_J$ |
| 1 | 0 | $Y_J$ |
| 1 | 1 | $X_J + Y_J$ |

It's easy to find out that *Z j* requires addition operation only when both of the multiplied bits are equal to 1. In order to perform addition, we use carry propagate adders, as in Figure 1 where the sum and carry values of $X_0 \square Y_0$ , $X_1 \square Y_1$ , $X_2 \square Y_2$ and $X_3 \square Y_3$ are shown respectively.

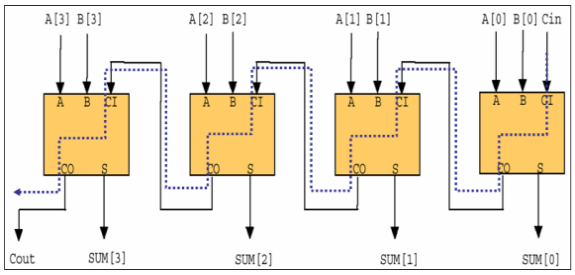

Figure 1 Four-bit carry propagate adder

In these steps, $S_j = S_j\, S_{j-1} S_{j-2} \ldots\ldots\ldots\ldots S_1 .S_0$ and at each step only $S_j$ and $C_{j+1}$ values are new; the remaining bits of $S_j$ are formed in the previous j-1 steps. Thus $S_j$ can be written as

$$\mathbf{S_j = S_{j-1} + X_{j-1} + Y_{j-1}}$$

## 3. Multiplexer Based Multipliers

Realization of the equations derived above is possible by using multiplexer based multipliers; with a 4-to-1 multiplexer where *x j* and *y j* are the control bits.In the parallel realization of the algorithm, the terms $Z_j 2^j$ and $X_j\, Y_j\, 2^{2J}$ are produced in the jth row of multiplexers, and are subsequently summed. The Implementation of the terms $Z_j 2^j$ and $X_j\, Y_j\, 2^{2J}$ is as shown in Figure 2.

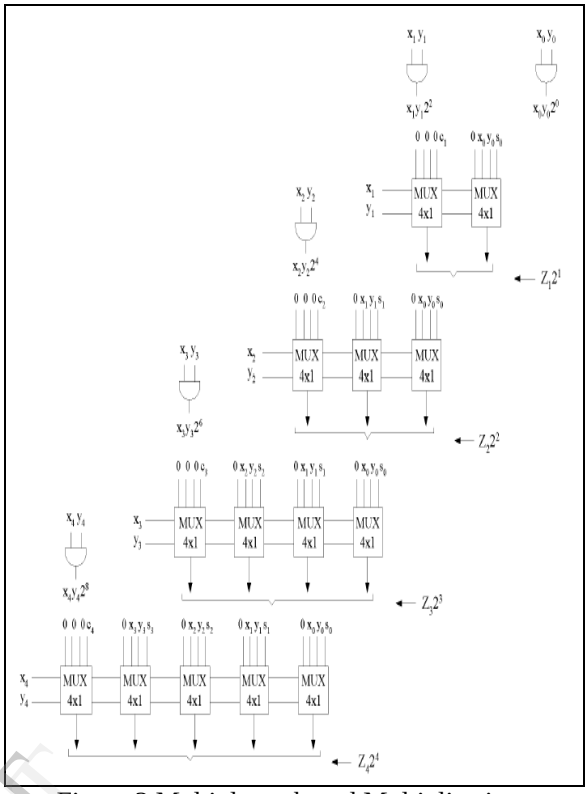

Figure 2 Multiplexer-based Multiplication Algorithm

The multiplexer based multiplier array has two different types of cells. The first type, Cell1 which is shown in Figure 3, consists of a 4-to-1 multiplexer and a full-adder. At the i[th] row of the array, Xj and Yj bits are transmitted to (n *-1-i)*□first type cells. As well, they are broadcast to j+1 diagonally placed cells of the array in the total architecture.

Cell II which is shown in Figure 3, consists of: two full-Adders which produces new bits Sj = Si and $C_{J+1}$ and two AND gates. Bits si along with Xj and Yj are broadcast to all first type cells of the i[th] row of the array. Bits $C_{J+1}$ are propagated to the next second type cells where *Si*, Xj and Yj bits are transmitted to all first type cells in the i[th] row of the array.

By using two AND gates, the term Xj , Yj which is required to find out the product according to equation, and $X_j\, Y_j\, C_j$ value, which enters the other full-adder cell to form S*out* and C *out* are found. Finally, S *out* and C *out* bits are sent to two-bit carry-look ahead adders to form the product. The reason of using a carry-look a head adder is to perform the addition with the possible smallest complexity and the fastest operation speed.

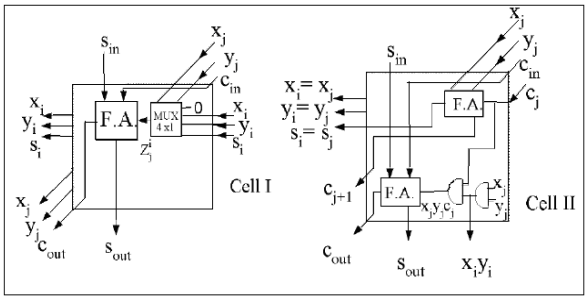


Figure 3 Circuit structures of Cell-I and Cell-II blocks

In the j th diagonal row of the array, j first type cells and one second type cell exist, which makes n(n-1)/2 *first* type and n second type cells in the total architecture, as shown in Figure 4.

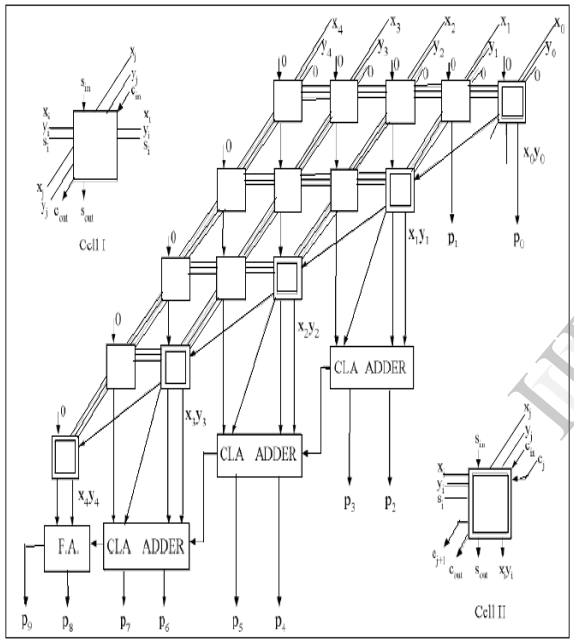


Figure 4 Multiplexer-based parallel multiplier.

## 4. *Comparison of Various Multipliers*

The circuit complexity of the given algorithm, by means of gate and transistor count, is given in Table 2. It can be formulated that the total multiplier contains

$\frac{n(n-1)}{2}$ 4-to-1 multiplexers, $\frac{n(n-1)}{2}+2n+1$ full adders, $n-2$ two-bit-CLA cells and $2n$ AND-gates.

Based on the number of transistors of various circuit types, given in Table 2, the total number of gates and transistors of various multiplier types are presented in Table 3

Table 2: Number of gates and transistors for various types of circuits

| CIRCUIT | GATES | TRANSISTORS |
|---|---|---|
| Half Adder | 5 | 10 |
| Full Adder | 12 | 26 |
| 2-to-1 MUX | 3 | 6 |
| 4-to-1 MUX | 5 | 16 |
| XOR | 4 | 6 |
| 2-Bit CLA | 23 | 50 |

Table 3: Circuit complexity comparison of various multipliers

| Type of Multiplier | Number of Cells | Number of Gates | Number of Transistors |
|---|---|---|---|
| Array | $n^2$ | $13n^2$ | $30n^2$ |
| Counter Cell | $n(n+1)/2$ | $13(n^2+3n-2)$ | $30(n^2+3n-2)$ |
| Modified Booth's Algorithm | Cell A: $(n+1)^2/2$<br>Cell B: $(n+1)/2$ | $10n^2+23n+13$ | $21n^2+52n+31$ |
| MUX Based Multiplier | Cell I: $n(n-1)/2$<br>Cell II: $n$<br>CLA: $n$ | $8.5n^2+16.5n-22$ | $21n^2+37n-99$ |

Pekmestzi's multiplier also has the smallest operation delay. The multiplication time of the structure is equal to T= (n+1) $\Gamma_{FA}$ , $\Gamma_{FA}$ being the delay of a full-adder; under the assumption of carry propagation delay of a two-bit carry-look ahead adder is also $\Gamma_{FA}$ □. This delay is significantly small when compared to the operation delay of the conventional array multiplier, which is T= (2n-1) $\Gamma_{FA}$ .

Even though the carry propagate adders are replaced with carry-look ahead adders in order to perform a faster addition and to shorten the multiplication time, the total delay is found to be $\Gamma_{FA}(n+\log_2 n)$ , which is still slower than that of the structure in Figure 3.

Table 4: Operation time comparison of various multipliers

| Type of Multiplier | Operation Time (with FA's) | Operation Time (with CLA's) |
|---|---|---|
| Array | $(2n-1)\tau_{FA}$ | $\tau_{FA}(n+\log_2 n)$ |
| Counter Cell | $(n+1)t$ | $(n+1)t$ |
| Modified Booth's Algorithm | $3n\tau_{FA}/2$ | $n\tau_{FA}/2+\log_2 n\,\tau_{FA}$ |
| MUX Based Multiplier | $(n+1)\tau_{FA}$ | $(n+1)\tau_{FA}$ |

## 5. Measurement Results

Measurements of dynamic power confirm the results of transistor-level simulations (Table) in terms of relative benefits, although simulated

results tend to underestimate the measured consumption (accuracy ranges from 15% to 30%). According to measurements, the Wallace-tree multiplier dissipates about 15% less energy than the reference CSM. Further 24% savings are possible by implementing minimum-size transistors. Yet, the proposed TG-Mult is by far more efficient than all the other architectures.

Table 5: Performance Results

| Multiplier type | Adder type | Prop.delay(ns) | Power in (µw) |
|---|---|---|---|
| 4x4 bit Booth multiplier | CMOS 28-T | 1.41ns | 0.260 mw |
| | TG 18-T | 1.22ns | |
| | CLA | 1.19ns | |
| 4x4 bit multiplexer based multiplier | CMOS 28-T | 1.09ns | 61.980 µw |
| | TG 18-T | 0.928ns | |
| | CLA | 0.912ns | |

## 6. Results and Discussions

Aconductive transmission gate acts, in first approximation, as a resistance , the value of which is dependent on the working region of the two transistors. In the given technology, the resistance ranges from about 15 k to almost 60 k for a TG with minimum-size transistors. By connecting it to a typical node load of 10 fF, an equivalent *RC* filter with a time constant of few hundreds of picoseconds results. When several transmission gates are cascaded, the time constant easily reaches a few nanoseconds, enough to filter out the majority of glitches. The following two reasons allow TG-Mult to be robust against leakage:

1) The implementation of minimum-size devices;

2) The reduction of the number of -to-ground paths.

## 7. Conclusion

Multiplier energy efficiency is the result of careful tradeoffs among several, often contrasting factors, from architectural down to transistor level. The new multiplier structure introduced in this work (TG-Mult) succeeds in reducing spurious switching activity significantly without compromising the benefits with energy-hungry add-on subcircuits. Transmission gates combined with level-restoring static CMOS gates suppress glitches via *RC* low-pass filtering, while preserving unaltered driving capabilities.

Measurements point out 43% energy savings over a regular Wallace architecture and more than 24% compared to a Wallace featuring minimum-size devices. Additionally, simulations show 34% energy savings over Chong [1] and 56% over Leapfrog [9]. The price to be paid is a delay overhead, which appears acceptable in low-frequency audio applications. An exhaustive overview is given in Fig. 6: TG-Mult provides a new and significantly better Pareto-optimal circuit. The EDP is the second best after Wallace_minsize. A limited area overhead compared to the traditional Wallace architecture should also be considered.

The authors believe that the proposed combination of transmission gates with level-restoring static CMOS gates could save power in other combinational blocks and in a larger variety of applications.

## 8. Refernces

[1] Flavio Carbognani, Felix Buergin, Norbert Felber, Hubert Kaeslin, *Member, IEEE*, and Wolfgang Fichtner, *Fellow, IEEE* "Transmission Gates Combined With Level-Restoring CMOS Gates Reduce Glitches in Low-Power Low-Frequency Multipliers" IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 16, No. 7, July 2008.

[2] John P. Uyemura " CMOS Logic Circuit Design" Springer international edition-2005

[3] John P.Uyemura " Chip Design for Submicron VLSI" Thomson India Edition-2007.

[4] Neil H.E.Weste,David Harris,Ayan Banerjee "CMOS VLSI DESIGN" Third edition PEARSON Education INDIA EDITION-2006.

[5] Douglas A.Pucknell,Kamran Eshraghian "BASIC VLSI DESIGN"Prentice Hall of INDIA PVT LTD THIRD EDITION-2005.

[6] V. D. Agrawal, "Low Power Design by Hazard Filtering," *Proc. 10tth International Conference on VLSI Design*, (1997), pp. 193-197.

[7] V. D. Agrawal, M. L. Bushnell, G. Parthasarathy and R. Ramadoss, "Digital Circuit Design for Minimm Transient Energy and a Linear Programming Method," *Proc. 12tth International Conference on VLSI Design*, (1999),

[8] A. P. Chandrakasan and R. W. Brodersen, *Low Power Digital CMOS Design*. Boston: Kluwer Academic Publishers, (1995).

[9] F. Gao and J. P. Hayes, "Total Power Reduction in CMOS Circuits via Gate Sizing and Multiple Threshold Voltages," *Proc. Design Automation Conference*, (2005), pp. 31-36.