Low-Power Array Multipliers with Transition-Retaining Barriers

Abstract

This paper presents a method for reducing the power dissipation of array multipliers. It is based on the insertion of transition-retaining barriers *(TRB) to decrease the propagation of the* useless *transitions in the array that only dissipate power and do not contribute to the calculation of the final result. With three TRBs we can achieve a reduction in power dissipation of 30% in a 32×32-bit array multiplier with an increase in area and delay of 3.9% and 7.6% respectively.*

1 Introduction

Recent developments in microelectronics technology allow circuits with more and more functionality to be integrated in just one chip. Nowadays, portable applications are not only wrist clocks or calculators but computers, multi-media terminals or mobile telephones. These new applications require high computational speed and, at the same time, suffer from power dissipation constraints. Dissipation of a chip grows, among other factors, with the level of integration and the computational speed.

Multipliers are basic arithmetic circuits for many DSP applications. Among multipliers, parallel multipliers are the fastest and they are widely used in order to achieve high execution speed. Parallel multipliers present a good regularity but also a high dissipation that increases dramatically with the operand bit-width. Techniques for decreasing the dissipation of this type of circuits are thus suitable and in some cases necessary if power dissipation is a major issue.

Multipliers are classified into two basic types: sequential and combinational. The combinational ones are further classified into two subtypes [7]:

- *parallel multipliers*: all partial products are generated in parallel and afterwards a multi-operand adder is used to obtain the final result. The delay of a parallel multiplier is $O(log\ n)$, being n the bit-width of the input data.

- *array multipliers*: these multipliers do not use two separate circuits for partial product generation and accumulation. A single cell is used in an array structure where the new partial products and the partial accumulation are produced simultaneously. The delay of an array multiplier is $O(n)$.

Schemes for parallel multipliers have been studied for decades trying to reduce the number of logical levels of the multiplier. Because there is less freedom in the design of array multipliers, fewer research has been made. Basically, all the efforts have been directed to array multipliers for signed digits [1, 12].

Parallel multipliers are faster than array multipliers because they reduce the number of logical levels but they also suffer from a bad layout regularity, a drawback that does not make them appealing to VLSI design.

The layered structure of array multipliers makes them suitable for applying reducing-power techniques. A layered structure, where each layer is excited from the very beginning, creates many *useless* signal transitions that only dissipate power and do not contribute to the calculation of the final result.

These transitions can be prevented, thus reducing the overall power dissipation. A method is to latch some paths inside the array multiplier. The latch prevents a useless transition from generating more transitions. In other words, it prevents the *snow-ball* effect.

This paper is organized as follows. In Section 2, previous studies in the area of low-power multipliers are presented. In Section 3, the power dissipation model is explained. In Section 4, the potential power dissipation reduction in the array multipliers is analyzed. In Section 5, the *transition-retaining barrier* (TRB) technique is proposed. In Section 6, the implementation of the TRBs is described. Finally, the results obtained are presented in Section 7. In Section 8, some conclusions are presented.

2 Previous Work

The design of arithmetic circuits aiming at minimizing power dissipation has basically focused on multipliers [14, 18, 11, 10, 9, 8, 13]. Other arithmetic circuits have been rarely studied [4, 6].

In [18], a comparison is presented between the design of a static CMOS multiplier and the design using Complementary Pass-Logic (CPL) [17]. Due to the fewer transistors used in the CPL design and their lower input capacitance, a 28% of power dissipation reduction is achieved with a frequency of operation of 100 MHz and a power supply of 4V.

In [14], an architecture to implement the data compression in a 52×52-bit Booth multiplier based on tally functions is described. The authors derive a reduction in power of 35% in the compression stage of the multiplier compared with that of a conventional multiplier.

The characterization of unnecessary activity in an array multiplier has been studied in [11]. It is shown that the unnecessary activity in a 16×16-bit array multiplier is the 50% of the total activity.

In [10], the switching activity of multipliers is studied as a function of the multiplication algorithm, the compressor circuitry and the circuit design style. A 8×8-bit modified Booth multiplier is designed where the activity is estimated to be reduced by a factor of 2.

Finally, in [9, 8, 13] a similar technique as the one proposed in this paper is studied. In [9], latches are inserted in a 16x16-bit radix-4 Booth multiplier to reduce the switching activity. The authors claim that a 40% of power savings is achieved with a frequency of operation of 50 MHz and a power supply of 3V. In [8], the influence of transition activity (due to combinational logic, flip-flops and the clock line) is analyzed in array multipliers and an optimal level of pipelining is hinted. In [13], a design of a Booth encoded array multiplier is presented that uses dynamic CMOS logic together with a self-timed evaluate signal in such a way that each carry-save adder within the array evaluates only after all of its inputs have stabilized. No power reduction due to this technique is reported.

3 Power Dissipation Model

In CMOS circuits there are three sources of power dissipation: *signal transitions*, *direct-path short-circuit current* and *leakage current* [3]. In static CMOS, signal transitions (*switching activity*) dominates the total power dissipation due to the charging of capacitors.

Under the following assumptions:

- the switching activity is the only source of power dissipation
- the only capacitance switched is the gate output capacitance
- a signal is charged to power supply or to ground, but not to an intermediate value,

we can model the average power dissipation of a CMOS gate with the following equation:

$$P_{avg} = \frac{1}{2} \frac{C_{load}\ V_{dd}^2\ E(transitions)}{T_{cyc}}$$

where P_{avg} is the average power dissipation, C_{load} is the load capacitance, V_{dd} is the power supply, T_{cyc} is the global clock period and $E(transitions)$ is the number of gate output transitions per clock cycle. In a combinational circuit, we can redefine $E(transitions)$ without loss of generality as the number of gate output transitions per operation.

4 Potential power reduction in array multipliers

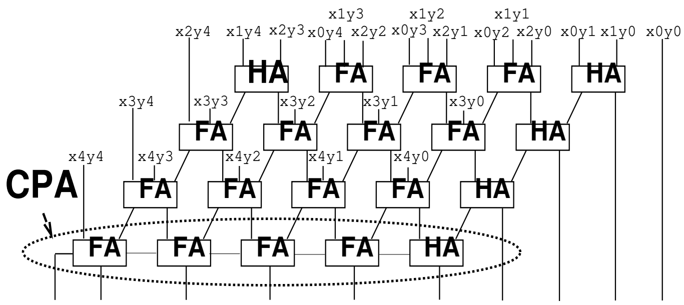

Figure 1: 5×5-bit array multiplier.

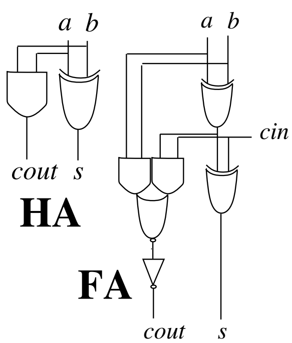

Figure 2: **(a)** FA and **(b)** HA structure.

The array multiplier structure studied is depicted in Figure 1. This array multiplier computes the $2n$-bit product of two n-bit unsigned numbers. The basic building blocks or cells of the array multiplier are full-adders (FA) and half-adders (HA) (see Figure 2).

Figure 3 plots the logic transitions per operation that take place in the connections of the FA and HA in a 16×16-bit array. Each block in the figure represents the sum of the s and $cout$ signal transitions of a FA or HA.

The layout of the figure has been rotated to allow an appropriate observation of the transitions at each cell. The tallest bars of the plot correspond to the carry-propagate adder (CPA).

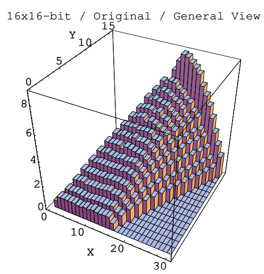

Figure 3: Sum of s and $cout$ signal transitions per operation in a 16×16-bit array multiplier.

The average power dissipation of a CMOS circuit depends linearly on the number of signal transitions, as it has been shown in Section 3. Thus, *useless* signal transitions (i.e. transitions that only dissipate power and do not contribute to the calculation of the final result) should be prevented.

In the layered structure of array multipliers, where each layer is excited from the very beginning, many useless transitions are generated. These transitions, in their turn, generate more useless transitions, thus creating a *snow-ball* effect. We observe in Figure 3 the consequence of this effect. The deepest layers of the array are the most power consuming.

The potential power reduction in array multipliers increases dramatically with the operand bit-width. Consider Figure 4. Both useful and total number of signal transitions for an 8×8, 16×16, 32×32 and 64×64-bit array multipliers are shown. has been used. It is clear that the larger the array multipliers is, the larger is the possibility for reducing the power dissipation.

It is interesting to notice that the potential for reducing power consumption in parallel multipliers (e.g. Wallace) is lower than in array multipliers. The reason is that parallel multipliers present less generation of useless transitions due to the more balanced paths to the basic blocks that compose the multiplier. We have observed that, for example, the number of signal transitions in a 32x32-bit array multiplier doubles the number in its Wallace counterpart.

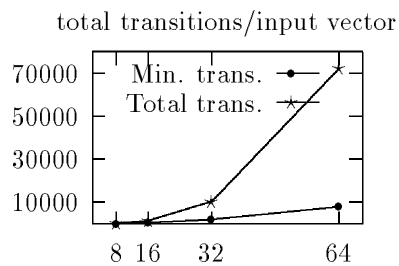

Figure 4: Useful and total number of signal transitions for an 8×8, 16×16, 32×32 and 64×64-bit array multipliers.

5 Transition-Retaining Barriers

A strategy to avoid useless transitions in data-path circuits it to insert self-timed controlled latches along the paths where these transitions occur.

A possible solution is to latch every output of each cell. This configuration has some drawbacks that make it not very appealing: area, delay and power dissipation increase due to the latches.

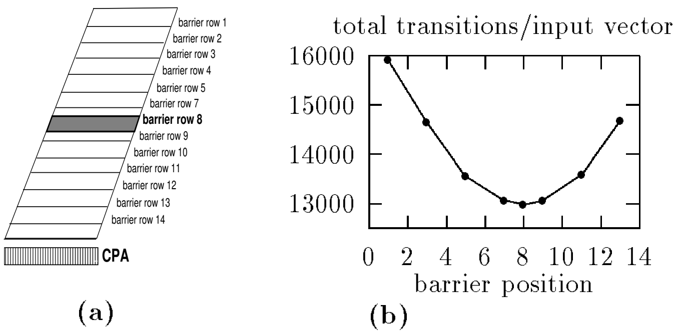

Figure 5: **(a)** Possible insertion locations for a TRB and **(b)** Transitions per input vector for different insertion locations of one TRB in the 16×16-bit array.

A trade-off is to find, for instance, a barrier of latches that divides the array into two parts (the upper and bottom halves). Thus, the useless transitions that are generated in the upper half of the array will not cross the barrier until the barrier switches into transparent mode. Henceforth, this barrier of latches is called a *transition-retaining barrier* (TRB).

For VLSI arrays, a regular implementation of this barrier is sought. The easiest way to insert latch barriers in the array is by taking advantage of the layered structure of the array.

The enabling signal of the latch barriers is generated by delaying an input signal (for example, the clock signal). The delay should be no longer than the critical path from the inputs of the array to the barrier (thus not increasing the delay of the multiplier) and as similar as possible to the critical path (thus no useless transitions are allowed to cross the barrier too early).

It is obvious that the best location for the array is somewhere in the middle of the array because we balance the useless transitions between the two partitions. Figure 5(b) shows an analysis of the number of transitions per input vector in the 16×16-bit array with different locations for the TRB. The optimum partition is in row 8 of Figure 5(a).

We can evaluate also the power dissipation if two or more TRBs are inserted. Similarly, the optimum is obtained when the array is divided into equal parts.

6 Implementation of the TRB

Since the cost of latches may be too high (both in area and power dissipation), other structures that perform the same transition-retaining function at the outputs of the multiplier cells have to be devised.

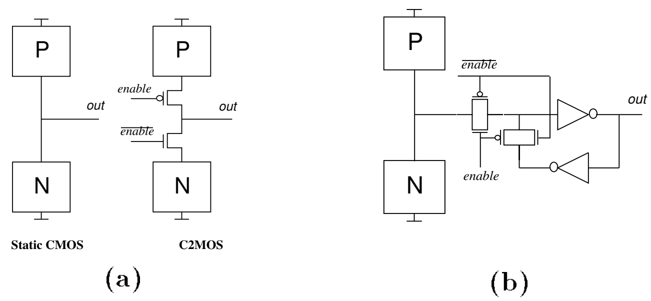

Figure 6: **(a)** Difference between Static CMOS and C^{2}MOS and **(b)** Avoiding direct path currents.

The designer can choose between a static or dynamic implementation of the latch. The most appropriate dynamic latch structure in terms of area is the Clocked-CMOS (or C^{2}MOS) latch [17]. The area overhead consists of four transistors per cell (two for each output of the full-adder cell). Figure 6(a) depicts the C^{2}MOS latch. Related work with power dissipation using C^{2}MOS has been presented in [15, 2].

The main problem with the use of C^{2}MOS structures is that capacitive coupling may arise at the *out* node in Figure 6(a), causing direct path currents in the gates fed by this node. To avoid this problem, the static structure depicted in Figure 6(b) should be used.

The delay mentioned in Section 4 is implemented now as a chain of inverters. We distinguish 4 types of delay-cells:

- TA simulates the delay of the first layer of the array

- TB simulates the delay of a full-adder
- TC simulates the delay of a full-adder and controls the TRB and
- TD is the same as TC but has no other fanout delay-cell

Figure 7 depicts the layout of a 4x4-bit array with two TRBs where the four types of delay-cells can be observed. All the controlling signals are obtained by delaying the multiplier input signal *start*.

The number of inverters in each delay-cell depends on the delay of the inverter, the input capacitance of the inverter and the capacitance of the C^2MOS signals ($enable$ and $\overline{enable}$ of Figure 6(b)). The number of inverters in TC and TD delay-cells also depends on the operand bit-width of the array.

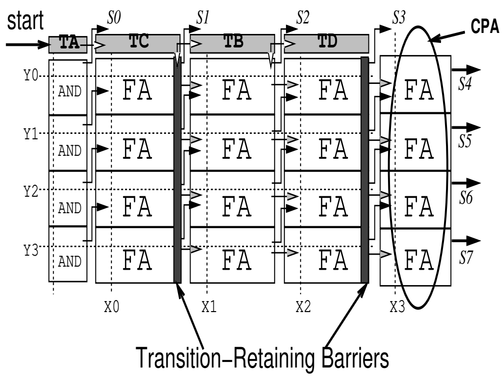

Figure 7: 4x4-bit array with two TRBs. The four types of delay-cells are shown

7 Results

Four configurations of array multipliers have been designed for the evaluation of the power dissipation, area and delay. The original array multiplier is compared to designs with one, two and three TRBs. When n TRBs are inserted, the array multiplier is divided into $n+1$ equal parts.

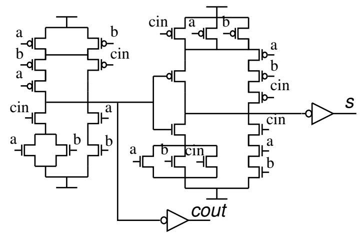

Figure 8: Design of a FA.

Array Multiplier	*Area (mm^2)*	*Delay (ns)*	*Energy (nJ/op.)*	*Energy Improv.*
8×8-bit original	0.78×1.49	20.81	1.87	-
8×8-bit 1 TRB	0.88×1.51	21.69	1.98	-5.6%
8×8-bit 2 TRB	0.88×1.53	23.29	1.99	-6.0%
16×16-bit original	1.56×2.96	42.00	12.18	-
16×16-bit 1 TRB	1.66×2.98	43.25	11.29	7.3%
16×16-bit 2 TRB	1.66×3.00	44.06	10.63	12.7%
16×16-bit 3 TRB	1.66×3.01	49.61	9.68	20.5%
24×24-bit original	2.35×4.44	60.73	35.71	-
24×24-bit 1 TRB	2.45×4.46	64.32	32.16	9.9%
24×24-bit 2 TRB	2.45×4.47	66.19	29.46	17.5%
24×24-bit 3 TRB	2.45×4.49	69.48	26.58	25.5%
32×32-bit original	3.13×5.91	84.47	79.93	-
32×32-bit 1 TRB	3.23×5.92	86.32	69.54	13.0%
32×32-bit 2 TRB	3.23×5.94	88.19	62.35	22.0%
32×32-bit 3 TRB	3.23×5.96	91.48	56.00	30.0%

Table 1: Comparison between the original design and the new designs with TRBs.

The configurations are implemented in a Sea-of-Gates (SoG) design style using the Ocean package [5]. The full-adder cell is implemented as in [17] (see Figure 8). The evaluation of the power dissipation has been obtained with a switch-level simulator [16].

The scenario of the simulation is the following:

- input patterns are randomly generated with probability of logical 1 at the primary inputs being 0.5
- all inputs to the multiplier arrive at the same time and
- between one pseudo-random input vector and the following one there is enough time for the outputs to settle.

The results are presented in Table 1. In the 8$\times$8-bit designs with one and two TRBs, the overhead due to the additional logic generates more power dissipation than it reduces. Hence, the overall power dissipation is higher than in the original design.

A trade-off exists between the delay of the new designs and their power dissipation. If the delay of the delay-cells is greater than the delay of the full-adder cell, the multipliers will have less power dissipation but will be slower. Thus, the increase in the delay in the designs with barriers is due to the additional transistors of the C^2MOS logic and the longer delay of the delay-cells.

We have the best result in power dissipation reduction when three TRBs are inserted. But it is important to notice that the greatest decrease in power dissipation is achieved when inserting one TRB with regard to the original design rather than when inserting two TRBs with regard to inserting one TRB.

The increase in area when inserting TRBs is small, and more negligible as the bit-width increases (there is an increase of 15.3% when inserting three TRBs in the 16$\times$16-bit and just a 7.6% in the 32$\times$32-bit with also three TRBs). The overhead in power dissipation, area and delay due to the TRBs has been taken into account in the results in Table 1.

Figure 9 shows how the transitions in the array are reduced when inserting one and two TRBs.

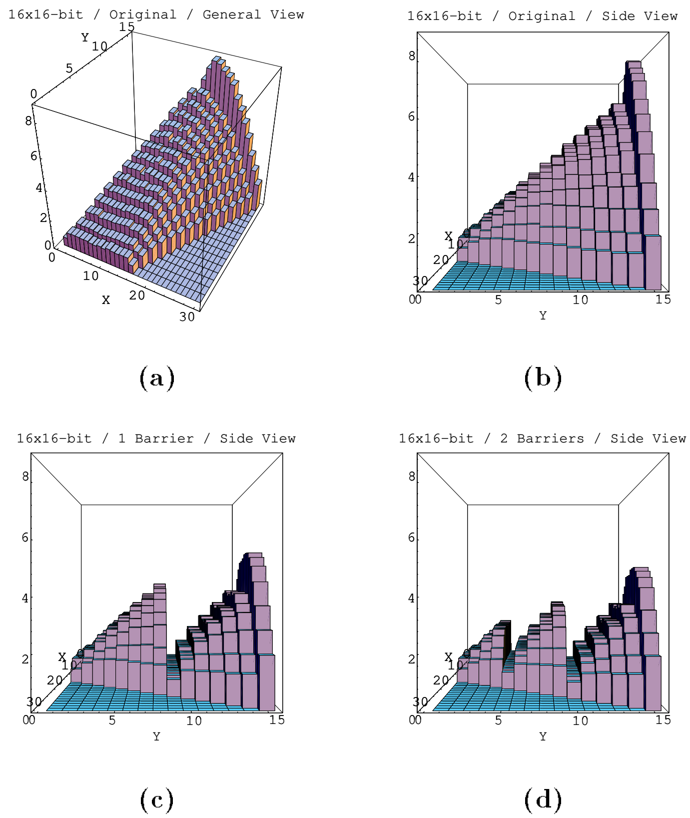

Figure 9: Sum of s and $cout$ signal transitions per input vector in a 16×16-bit array multiplier. **(a)** General view of the design with no TRBs, **(b)** Side view of the design with no TRBs, **(c)** Side view of the design with one TRB and **(d)** Side view of the design with two TRBs.

8 Summary and Conclusions

The technique of inserting *transition-retaining barriers* (TRB) in order to decrease the propagation of the *useless* transitions in array multipliers has been effective in reducing the power dissipation of this type of circuits. With three TRBs we can achieve a reduction in power dissipation of 30% in a 32×32-bit array multiplier with an increase in area and delay of 3.9% and 7.6% respectively.

The final multiplier remains combinational and highly regular. This implies that the technique presented in this paper can be easily integrated in existing CAD tools for the generation of array multipliers.

Acknowledgments

This work has been partially supported by CICYT TIC95-0419 and Dept. d'Ensenyament de la Generalitat de Catalunya.

The authors would like to thank the valuable comments of the reviewers.

References

[1] C. Baugh and B. Wooley. A two's complement parallel array multiplication algorithm. *IEEE Trans. on Comp.*, C-22(12):1045–1047, Dec. 1973.

[2] M. Borah, R. Owens, and M. Irwin. High-throughput and low-power DSP using clocked-CMOS circuitry. In *Int. Symp. on Low Power Design*, pages 139–144, Apr. 1995.

[3] A. Chandrakasan, S. Sheng, and R. Broderssen. Low power CMOS digital design. *IEEE Trans. on SSC*, 27(4):473–483, Apr. 1992.

[4] M. Ercegovac and T. Lang. Reducing transition counts in arithmetic circuits. In *Proc. Int. Symp. on Low Power Electronics*, pages 64–65, Oct. 1994.

[5] P. Groeneveld and P. Stravens. Ocean: the Sea-of-Gates design system. Technical report, Delft Univ. of Tech., 1993.

[6] U. Ko, P. Balsara, and W. Lee. Low-power design techniques for high-performance CMOS adders. *IEEE Trans. on VLSI Syst.*, 3(2):327–333, June 1995.

[7] I. Koren. *Computer Arithmetic Algorithms*. Prentice-Hall, 1993.

[8] J. Leijten, J. van Meerbergen, and J. Jess. Analysis and reduction of glitches in synchronous networks. In *Proc. EDAC*, Mar. 1995.

[9] C. Lemmonds and S. Shetti. A low power 16 by 16 multiplier using transition reduction circuitry. In *Proc. Int. Workshop on Low Power Design*, pages 139–142, Apr. 1994.

[10] V. Moshnyaga and K. Tamary. A comparative study of switching activity reduction techniques for the design of low-power multipliers. In *Proc. Int. Symp. on Circuits and Syst.*, pages 1560–1563, Apr. 1995.

[11] M. Ortega and J. Figueras. Extra power consumed in static CMOS circuits due to unnecessary logic transitions. In *IV Int. Design Automation Workshop (RUSSIAN WORKSHOP'94)*, June 1994.

[12] S. Pezaris. A 40ns 17-bit by 17-bit array multiplier. *IEEE Trans. on Comp.*, C-20(4):442–447, Apr. 1971.

[13] G. Sobelman and D. Raatz. Low-power multiplier design using delayed evaluation. In *Proc. Int. Symp. on Circuits and Syst.*, pages 1564–1567, Apr. 1995.

[14] M. Song and K. Asada. Design methodology for low power data compressors based on a window detector in a 54$\times$54 bit multiplier. In *Proc. Int. Symp. on Circuits and Syst.*, pages 1568–1571, Apr. 1995.

[15] Y. Susuki, K. Nogami, and M. Kakumu. Clocked CMOS calculator circuitry. *IEEE JSSC*, SC-8(6):462–469, Dec. 1973.

[16] A. van Gerenden. SLS: An efficient switch-level timing simulator using min-max voltage waveforms. In *Proc. VLSI 89 Conf.*, pages 79–88, Aug. 1989.

[17] N. Weste and Eshragian. *Principles of CMOS VLSI Design: A systems Perspective*. Addison-Wesley, 2nd edition, 1993.

[18] K. Yano, T. Yamanaka, T. Nishida, M. Saito, K. Shimohigashi, and A. Shimizu. A 3.8-ns CMOS 16x16-b multiplier using complementary pass-transistor logic. *IEEE JSSC*, 25(2):388–395, Apr. 1990.