Rectangular Styled Wallace Tree Multipliers

Mounir Bohsali (mounir@eecs.berkeley.edu)
Michael Doan (mydoan@uclink.berkeley.edu)

Abstract
Non-regularities in the construction of traditional Wallace tree multipliers result in a large amount of wasted area when laid out in a rectangular shape. Itoh et.al. [1] proposes a modified Wallace tree construction that would save most of the wasted area in the multiplier layout. This modified Wallace tree construction is investigated and evaluated. A comparison between the critical path and wiring overhead between the tradition and the modified Wallace trees is presented. A 25X12 bits Wallace tree multiplier is also designed in a 0.18μm process.

Motivation
In response to the high demand of fast three-dimensional computer graphics, high-speed floating-point processing, etc…, dedicated multiplier units are incorporated in almost every digital signal processor today as well as low-cost general purpose microprocessors. To achieve high speed, the Wallace tree architecture combined with the modified Booth recoding technique is used since it adds the partial products in parallel in a tree-like fashion. Because of its non-regularity, the layout of Wallace tree multipliers suffers from a large wasted area (e.g. 30%) when laid out in a rectangular area which is the preferred form of layout when many different blocks are to be interfaced with each other. Since cost is a fourth power function of area, the dead area problem of Wallace tree multipliers should be solved in order to realize a low cost solution, especially when the multipliers units occupy a large potion of the chip. Since the wiring of Wallace trees is considerably complex, any solution to the dead area problem should not add considerably to the wiring overhead in order not to increase the design time in today's highly competitive market.

Background
Any multiplier can be divided into three stages: Partial products generation stage, partial products addition stage, and the final addition stage.
In the first stage, the multiplicand and the multiplier are multiplied bit by bit to generate the partial products. In this stage, a second-order Booth encoding algorithm is usually used instead to reduce the number of partial products to half.
The second stage is the most important, as it is the most complicated and determines the speed of the overall multiplier. This project will be focused on the optimization of this stage, which consists of the addition of all the partial products. If speed is not an issue, the partial products can be added serially, reducing the design complexity. However, in high-speed designs, the Wallace tree construction method is usually used to add the partial products in a tree-like fashion in order to produce two rows of partial products that can be added in the last stage. Although fast, since its critical path delay is proportional to the logarithm of the number of bits in the multiplier, the Wallace tree introduces other problems such as wasted layout area and increased complexity, which we will elaborate on shortly.

In the last stage, the two-row outputs of the tree are added using any high-speed adder such as look-ahead adder to generate the output result.

Wallace Tree Construction

There exist a handful of ways to construct the Wallace Tree. One method considers all the bits in each column at a time and compresses them into two bits (a sum and a carry). A second method, which is used in this project, considers all the bits in each fours rows at a time and compresses them in an appropriate manner (see Figure 1). In Figure 1, the Wallace tree uses 4:2 compressor, 3:2 compressors, full adders and half adders to compress a 12 partial products tree. Each 4:2 compressor takes in four bits from the same position "j", one bit from the previous position "j-1" (which is the carry our of the compressor in the previous position) and outputs one sum bit in the position "j" and two carryouts into the next position "j+1".

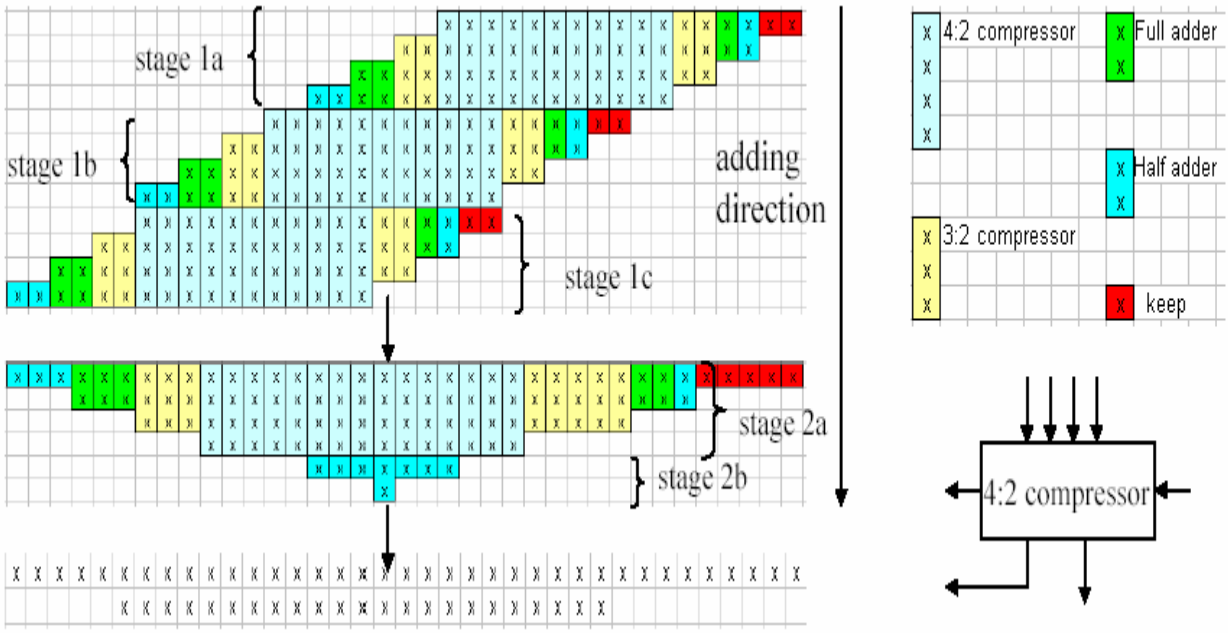

Figure 1. Traditional Wallace Tree construction

Problem Statement

The superior performance of the Wallace tree comes at a heavy price: non-regularity and wasted area.

In a conventional tree, the partial products in the tree are added in one direction (e.g. top to bottom) with the number of compressors or adders and wires needed increasing as the adding stage moves forward. Moreover, the compression process differs at each position because of the variability in the number of partial products to be added/compressed.

Unfortunately, because of its non-regularity, it becomes almost impossible to layout a Wallace tree multiplier in a rectangular shape without wasting a significant amount of chip area. In a conventional Wallace tree, every partial product is added in one direction from top to bottom. Therefore, the number of compressors or full adders increases as the adding stage moves downward, and the layout becomes wider at the bottom than at the top. Thus, when incorporated in a bigger design, area will be wasted. This translates into larger overall area, and thus higher costs.

Figures 2a and 2b below show the increasing width of the adding stages as one moves forward in the tree resulting in a dead area. In the figure, each horizontal line corresponds to one stage of compressors/adders in the Wallace tree. When the tree is laid out in a rectangle, the dead area if more than 50% is the layout is directly mapped from the schematic (Figure 2a).

Even though placing the adding/compression stages on one side of the rectangle as shown in Figure 2b saves some area, the problem is still very significant. In this case, more than 38% of the overall rectangular area is wasted.

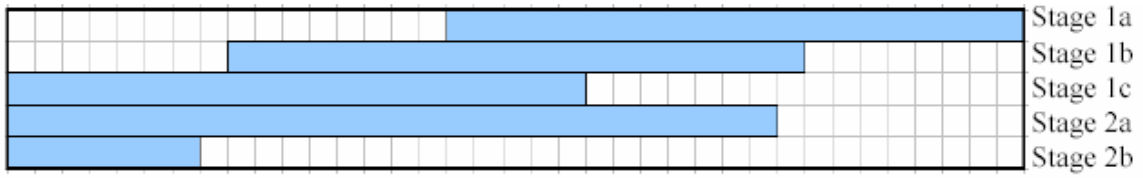

Figure 2a. Sketch of layout of conventional Wallace tree multiplier before arrangement

Figure 2b. Sketch of layout of conventional Wallace tree multiplier after arrangement

Solution

To solve the dead area problem, Itoh et.al. [1] proposed a new tree construction method (see in Figure 2) that results in considerably decreased dead area when the block is laid out in a rectangular shape. The solution states that if the partial products are divided into two groups around the middle of the tree, and the partial products within each group are added using the conventional method but with the partial products of one block are added from top to bottom, and those of the second are added in the opposite direction, a considerable amount of wasted area can be saved when the left and right trees are made to interface each other. The following explains the above procedure.

First, the partial products bits are divided into two separate trees called left tree and right tree. The right tree consists of the least significant bits and the left tree consists of the most significant bits. Figure 3 shows this procedure for a 12x17 bits tree. The right tree is then compressed in a Wallace tree fashion using 4:2 and 3:2 compressors and full and half adders in an upward direction, and the left tree is added in a similar fashion but in a downward direction. Figure 4 illustrates this procedure.

As shown in Figure 5, when the layouts of the two separate tree are brought together, they fit into each other like a puzzle and only a small amount of area is wasted. Figure 7 shows how much layout area reduction can be ideally achieved for 12x16, 16x16, and 20x16 Wallace trees.

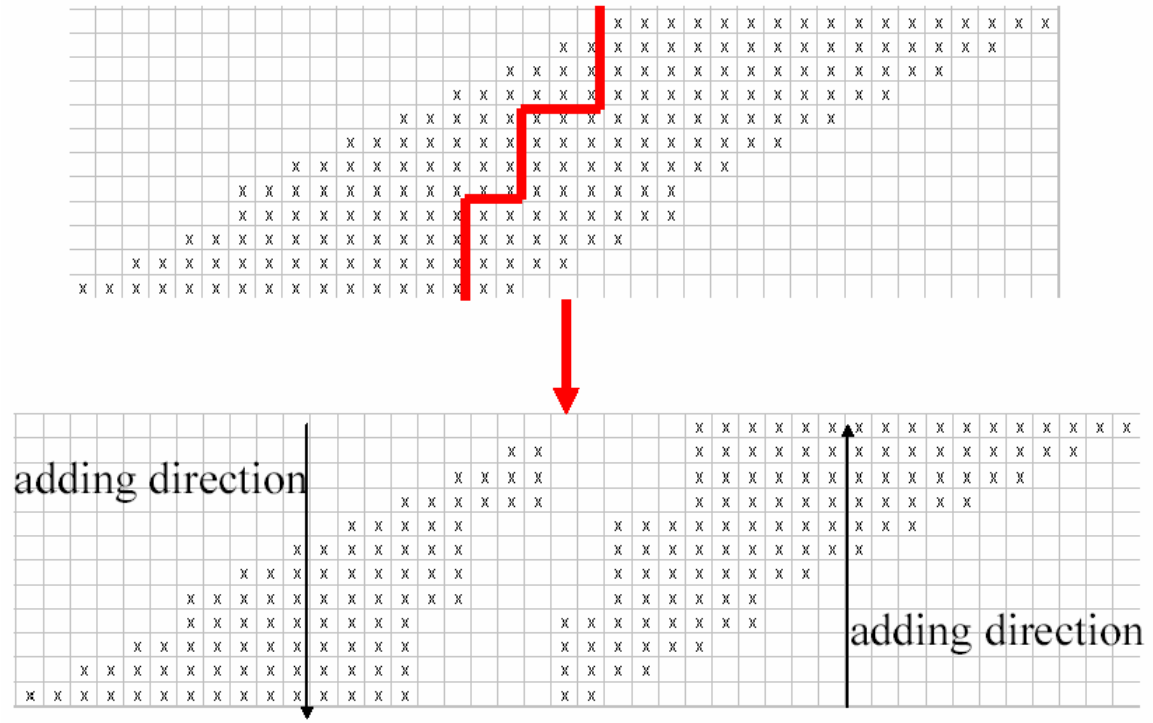

Figure 3. Splitting of Wallace tree into two trees

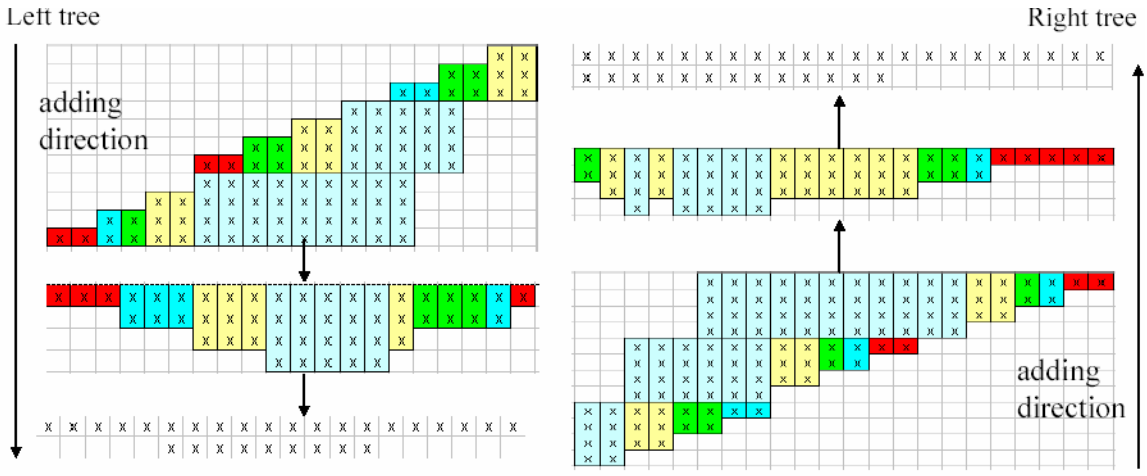

Figure 4. Wallace tree construction of the two modified sub trees.

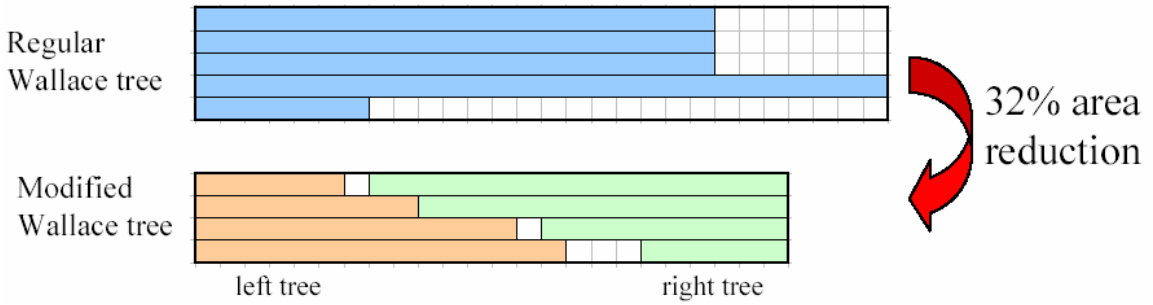

Figure 5. sketch of layout of conventional and modified Wallace trees

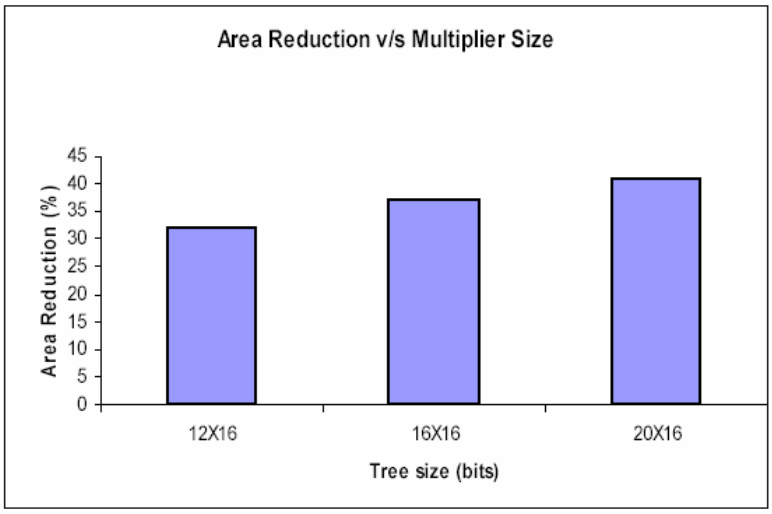

Figure 6. Area reduction using the modified Wallace tree construction

Critical Path Analysis:
Fortunately, although not the goal of this modification, the critical path of the modified Wallace tree is found to be about 10% smaller than that of a traditional tree for a 12x17 bits tree. Figures 7a and 7b show that the critical path of a traditional Wallace is 132 gates whereas

that of the modified tree is 120 gates.

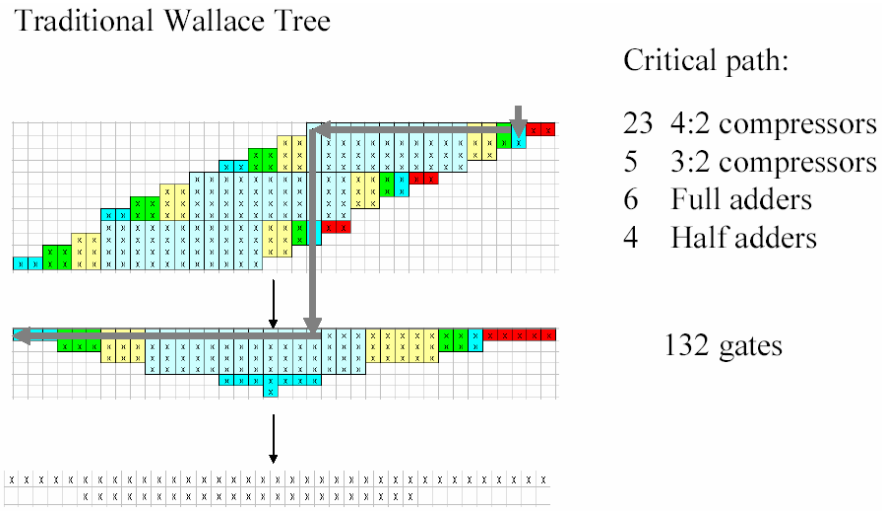

Figure 7. Critical path of conventional Wallace tree.

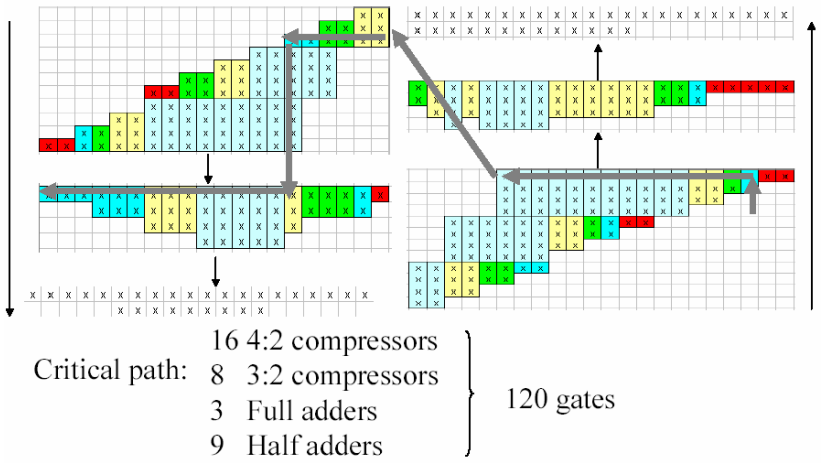

Figure 8. Critical path of modified Wallace tree.

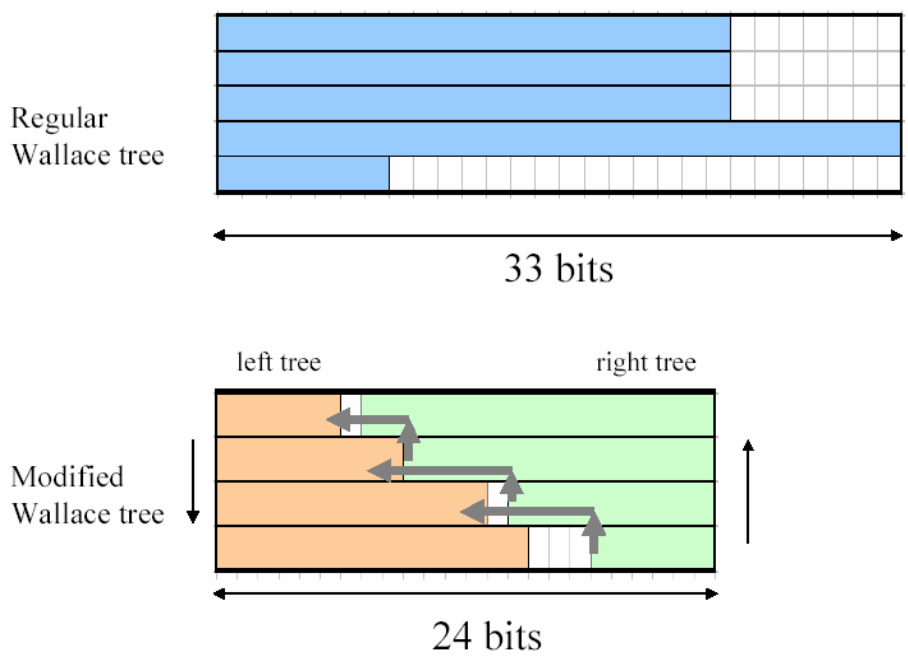

Figure 9. Wiring overhead of conventional and modified Wallace trees.

Wiring Analysis:
As shown in Figure 9, the proposed method reduces the width of the layout from 33 bits to 24 bits, which reduces considerably the length of wires needed. This will eventually reduces the parasitic capacitances inherent in the wiring. However, the wiring of the proposed tree is slightly more complex because if requires that many of the right tree rows to be routed one position up then into the left tree rows.

Multiplier Design:
A 16x22 bits multiplier is designed in a 0.18µm process with 1.8V supply. The critical path of the Wallace tree is simulated to be 5.4ns.

The multiplier block diagram is shown in Figure 10. The Booth array takes in a 16 bits multiplicand and a 22 bits multiplier and generates 12 partial products of 17 bits each. The partial products are then added in a Wallace tree fashion as show in Figure 1. The Booth encoder array consists of 12 cells called PP_generator. Each of these cells consists of a booth decoder with three inputs and three outputs and 17 cells called PP_generate_1bit. The booth decoder and the PP_generate_1bit cells implement the equation presented in the Figure.
Figures 11, 12, 13, and 14 show the gate level diagrams of the 4:2 compressor, full adder, booth encoder, and the PP_generate_1bit cell respectively.

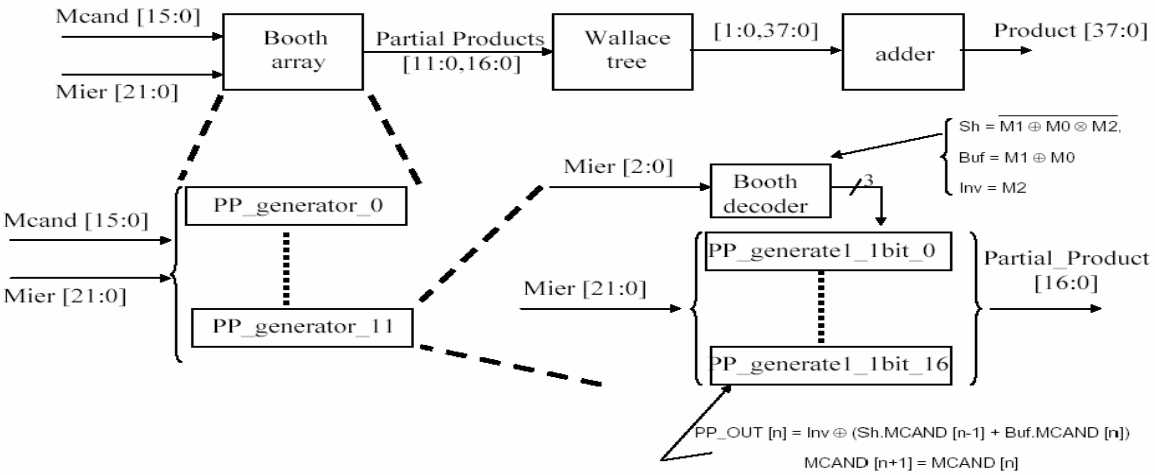

Figure 10. Block diagram of Wallace tree multiplier.

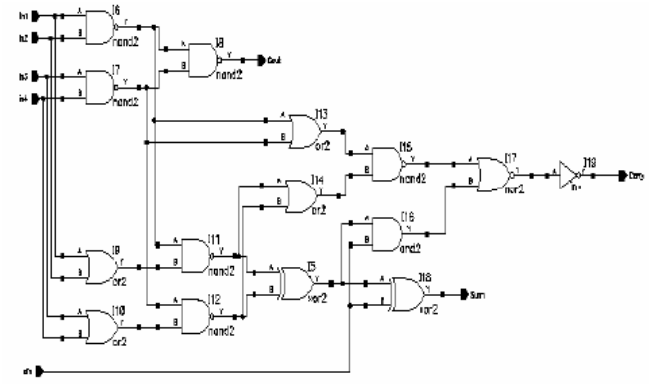

Figure 11. Schematic of 4:2 compressor

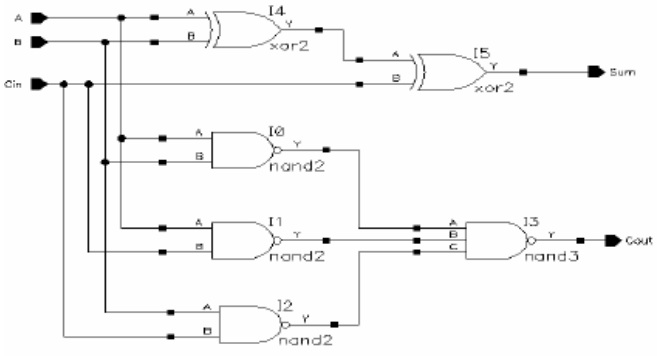

Figure 12. Schematic of full adder

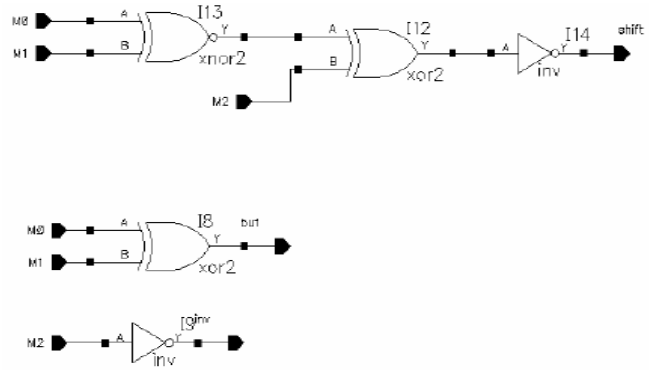

Figure 13. Schematic of booth decoder

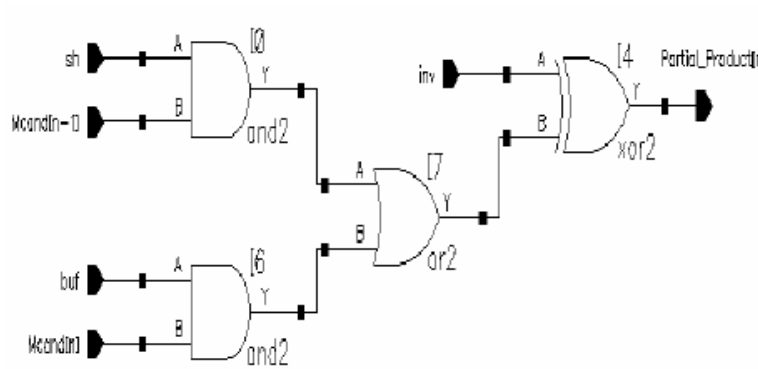

Figure 14. Schematic of PP_generate_1bit

Conclusion

The wasted area problem of large Wallace tree multipliers can be solved using a new method of tree construction. The modified tree has a slightly smaller critical path but a slightly larger wiring overhead. A 25x12 bits Wallace tree multiplier was designed in 0.18μm

References

[1] Niichi Itoh, Yuka Naemura, Hiroshi Makino, Yasunobu Nakase, Tsutomu Yoshihara, and Yasutaka Horiba, "A 600-MHz 54 x 54-bit Multiplier with Rectangular-Styled Wallace Tree", JSSC, vol.36, no. 2, February 2001.

[2] Gensuke Goto, Tomio Sato, Masao Nakajima, and Takao Sukemura, " A 54x54-b Regularly Structured Tree Multiplier", JSSC, col. 27, no. 9, September 1992.

[3] Siraram Vangal et.al. "A 5GHz Floating Point Multiply-Accumulator in 90nm Dual Vt CMOS", ISSCC 2003.

[4] Robert Montoye, et.al. "A Double Precision Floating Point Multiply", ISSCC 2003.

[5] Niichi Itoh, Yuka Naemura, Hiroshi Makino, Yasunobu Nakase, "A Compact 54x54 Multiplier with Improved Wallace-Tree Structure", 1999 Symposium on VLSI Circuits Digest of Technical Papers.

[6] Pascal C.H. Meier, Rob A. Rutenbar, L. Richard Carley, "Exploring Multiplier Architecture and Layout for Low Power", IEEE 1996 Custom Integrated Circuits Conference.

[7] S. Shah, A.J. Al-Khalili, D. Al-Khalili, "Comparison of 32-bit Multipliers for Various Performance Measures", The 12th International Conference on Microelectronics, Tehran, Oct. 31-Nov. 2, 2000.