54x54-bit Radix-4 Multiplier based on Modified Booth Algorithm

Ki-seon Cho, Jong-on Park, Jin-seok Hong, Goang-seog Choi
Storage Solution Group, DM R/D Center
Samsung Electronics Co., Ltd. 416, Maetan 3 dong, Paldal gu, Suwon, Korea
82-31-200-4764
kiseoncho@samsung.com

ABSTRACT

In this paper, we describe a low power and high speed multiplier suitable for standard cell-based ASIC design methodologies. For the purpose, an optimized booth encoder, compact 28-2, 27-2, …, and 10-2 compressors, and XOR based adder are proposed. While the whole design is coded in Verilog-HDL language and implemented through commercially available EDA tool chain, the implementation gives comparable results to full custom designs [1][2]. Realistic simulations using extracted timing parameters from the layout show that the propagation time of a critical path is 3.25ns at 2.5V on a 0.18um process technology, which is almost 21% faster than the conventional multiplier [2].

Categories and Subject Descriptors

B.7.1 [**Integrated Circuits**]: Types and Design Styles – *Advanced technologies, Microprocessors and microcomputers, Algorithms implemented in hardware.*

General Terms

Algorithms, Performance, Design.

Keywords

Multiplier, Booth encoder, Wallace tree, Compressor, Adder

1. INTRODUCTION

Nowadays, fast parallel multipliers are important for high speed and low power signal processing systems and much effort has been devoted to techniques for their construction [3][4]. For the techniques, Booth algorithm, PTL (Pass-transistor logic) [5][6], Wallace tree, and carry look-ahead adder have been proposed. While the Booth algorithm innovatively reduces the partial products half, PTL contributes to making high speed macro-cells. However, PTL does not prevail due to increasing wire delay as the fabrication process goes to more deep-submicron.

In order to implement these structures, full custom design method was preferred to standard cell-based ones. These, however, have low flexibility and long design and verification time. In an attempt to tackle these issues, behavioral modeling in hardware description languages and synthesis and auto P&R using commercial EDA tools are researched [7][8].

This paper describes a low power and high speed multiplier suitable for standard cell-based ASIC design methodologies. So, a high speed booth encoder, 28-2 compressor based on a full-adder, and 108-bit conditional select adder based on XOR with a separated carry generation block are devised as components. The high speed booth encoding algorithm simplifies the modified booth algorithm and reduces the number of gate counts about 35%, compared to the conventional CMOS one [9]. Further, a novel compression method using 28-2, 27-2, 26-2, …, and 10-2 compressors are proposed to replace commonly used 4-2 or 9-2 compressors. Finally, fourteen conditional sum modules and one separated block carry generation block are combined to make a 108-bit conditional select adder. With these structures, we followed commercially available EDA tool chains starting from behavioral codes in Verilog-HDL.

The architecture of the 54x54-bit multiplier is described in Section 2, the circuit design of the booth encoder based on modified Booth algorithm, comparators, and conditional sum adder in section 3, 4 and 5, comparisons of the proposed design methods and conventional design methods in section 6, and finally the simulation results in Section 7.

2. ARCHITECTURE

The block diagram of a proposed 54×54-bit multiplier is depicted in figure 1. It employs a booth encoder block, compression blocks, and an adder block. The partial products are rapidly generated by the proposed booth encoder and the partial product generator (PPG), and summed by proposed compressors. The two summed results are added together in the 108-bit adder. The whole architecture is same as the conventional multipliers.

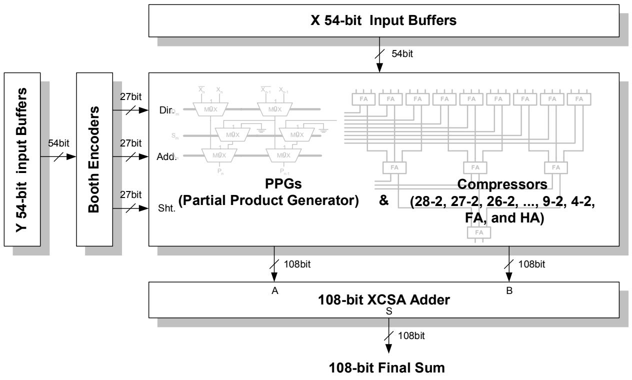

Figure 1. Proposed 54x54-bit multiplier architecture

GLSVLSI'03, April 28-29, 2003, Washington, DC, USA.

3. BOOTH ENCODER AND PPG MODULE

After the modified Booth algorithm was proposed by McSorley [10], the adoption of booth encoder has been generalized to reduce the number of partial products in multiplication. In our design, we have created a new encoder based on the modified Booth algorithm. Booth function basically has three basic operators, which we call ‘direction’, ‘shift’, and ‘addition’ operator. The direction operator determines which multiplicand is taken among normal (X) and inversed (~X) ones. For example, if each triplet of multiplier (Y) is “101”, we should take the two’s complement of X. But, it is very difficult to implement. Thus, the partial product can take the inversed bits of X and “+1” is implanted in the compressed module. The shift operator means a shift one position to left. If each triplet of multiplier (Y) is “011” or “100”, we should shift one position to left. The addition operator means an addition of X to the partial product. We propose a new expression to simplify booth encoding as follows.

$$Direction = Y_{m+1}\,;$$

$$Shift \quad = Y_{m-1}\cdot (Y_{m+1} \oplus Y_m) + Y_{m-1}\_bar \cdot (Y_{m+1} \oplus Y_m)\,;$$

$$Addition \;= Y_{m-1} \oplus Y_m\,;$$

But, the shift operator has two XOR terms, thus, circuits are complex. We redefine the shift expression to remove one XOR term as follows. The truth table can be written in table 1 and a booth encoder and a PPG are depicted in figure 2.

$$Shift \quad = Y_{m+1} \oplus Y_m\,;$$

Table 1. Truth table for booth encoding (- : don’t care)

Ym+1 Ym Ym-1	Booth Op.	Dir.	Sht.	Add.
0 0 0	0x	0	0	0
0 0 1	1x	0	-	1
0 1 0	1x	0	-	1
0 1 1	2x	0	1	0
1 0 0	-2x	1	1	0
1 0 1	-1x	1	-	1
1 1 0	-1x	1	-	1
1 1 1	-0x	1	0	0

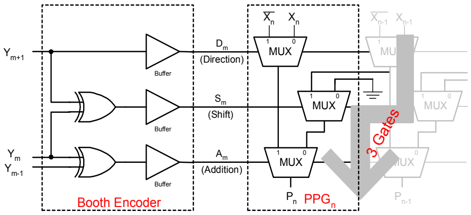

Figure 2. Booth encoder and PPG

The proposed booth encoder is organized only with two XORs and a PPG with three multiplexers. Therefore, the whole structure to obtain partial products is simplified more than conventional designs.

4. COMPRESSION MODULE

Generally, the Wallace tree with 4-2 and 9-2 compressors are used to compress partial products from PPG [6][7]. In a case of the 54x54-bit multiplier, 27 rows and 108 columns of partial products from PPG have formed a trapezoid. For practical purposes, it has 28 rows and 108 columns to handle a two’s complement for ±2x booth operations. To obtain a two-bit (carry & sum) compression result through using a compressor, conventional designs had adopted the Wallace tree with 4-2 and 9-2 compressors. There are three improving issues that are the number of full-adder, critical path, and complexity of sign extension [11]. Therefore, we propose 28-2, 27-2, …, and 10-2 compressors. Especially, the last issue is very difficult to solve with 4-2 or 9-2 compressors. The proposed compressors have a carry compensation port P[0] to compensate for a carry coming from a former compression block. Using the proposed compressors gives an improved critical path and the reduced number of full-adders. It is shown in the following figure 3, figure 4, and figure 5.

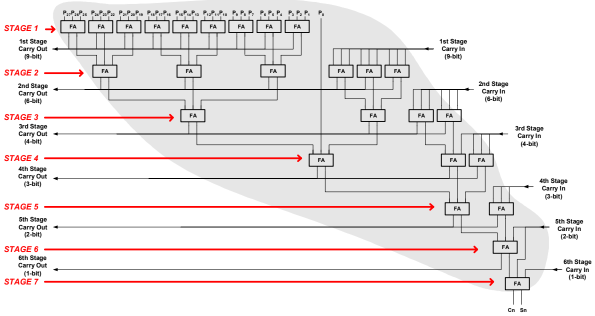
STAGE 1
1st Stage Carry Out (9-bit)
1st Stage Carry In (9-bit)
STAGE 2
2nd Stage Carry Out (6-bit)
2nd Stage Carry In (6-bit)
STAGE 3
3rd Stage Carry Out (4-bit)
3rd Stage Carry In (4-bit)
STAGE 4
4th Stage Carry Out (3-bit)
4th Stage Carry In (3-bit)
STAGE 5
5th Stage Carry Out (2-bit)
5th Stage Carry In (2-bit)
STAGE 6
6th Stage Carry Out (1-bit)
6th Stage Carry In (1-bit)
STAGE 7
FA
Cn Sn

Figure 3. 28-2 compressor for 28 rows of partial product

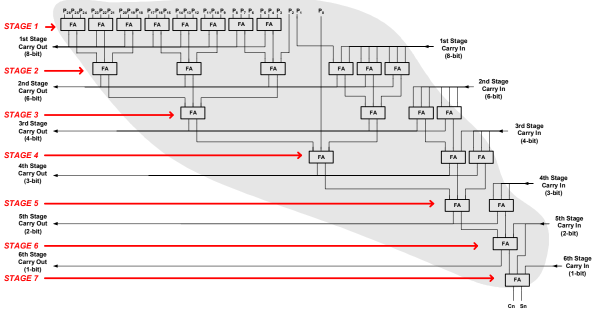
STAGE 1
1st Stage Carry Out (8-bit)
1st Stage Carry In (8-bit)
STAGE 2
2nd Stage Carry Out (6-bit)
2nd Stage Carry In (6-bit)
STAGE 3
3rd Stage Carry Out (4-bit)
3rd Stage Carry In (4-bit)
STAGE 4
4th Stage Carry Out (3-bit)
4th Stage Carry In (3-bit)
STAGE 5
5th Stage Carry Out (2-bit)
5th Stage Carry In (2-bit)
STAGE 6
6th Stage Carry Out (1-bit)
6th Stage Carry In (1-bit)
STAGE 7
FA
Cn Sn

Figure 4. 27-2 compressor for 27 rows of partial product

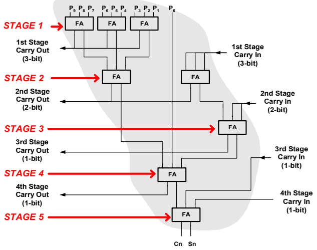
STAGE 1
1st Stage Carry Out (3-bit)
1st Stage Carry In (3-bit)
STAGE 2
2nd Stage Carry Out (2-bit)
2nd Stage Carry In (2-bit)
STAGE 3
3rd Stage Carry Out (1-bit)
3rd Stage Carry In (1-bit)
STAGE 4
4th Stage Carry Out (1-bit)
4th Stage Carry In (1-bit)
STAGE 5
FA
Cn Sn

Figure 5. 10-2 compressor for 10 rows of partial product

The left side of the proposed compressors compress partial products from PPG (P[max:0]), and the right side carries from the former compressor block. Therefore, a 28-2 compressor composed of only 26 full-adders and a critical path of 7 stages. The carry-in/out from or to the proposed compressor are summarized in the table 2. The figure 6 illustrates a composition technique in the sign extension region. The remnant carry out the of former compressor block is connected to the first input of the current compressor (P[0]).

Table 2. carry-in/out of proposed compressors

Input	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2
stage 1	9	8	8	8	7	7	7	6	6	6	5	5	5	4	4	3	3	2	1	3	2	2	2	1	1	1	0
stage 2	6	6	5	5	5	4	4	4	3	3	3	2	2	2	1	1	3	3	3	2	2	1	1	1	1		
stage 3	4	4	4	3	3	3	2	2	2	1	1	1	3	3	3	3	2	2	2	1	1	1	1	1			
stage 4	3	3	3	3	3	3	3	3	3	3	3	3	2	2	2	2	1	1	1	1	1	1					
stage 5	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1								
stage 6	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1											
stage 7	1	1	1	1	1	1	1	1	1	1	1	1															
output sum & carry	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

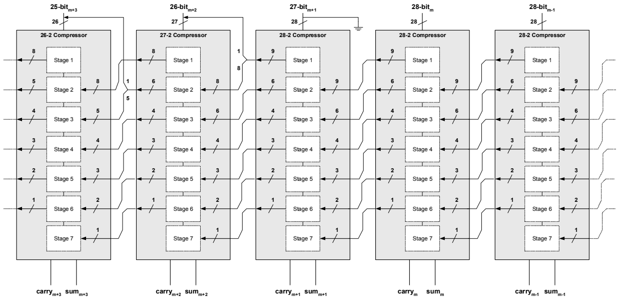

Figure 6. Composition of the sign extension region is illustrate

5. ADDITION MODULE

As we know, the final adder module to sum two rows of 108-bit is very important in multipliers [4][12]. We propose an improved adder based on XOR to minimize gate counts and critical path. Fourteen XCSA (XOR based conditional select adder) blocks and a separated carry generation block are combined to make our proposed adder. Each modularized XCSA consists of an 8-bit sum generator and a carry generator. The carries of each XCSA are transmitted to the BCGB (block carry generation block). The following expressions are describes how to determine a sum and a carry by XOR (A, B).

$$Sum \;\; = A_m \oplus B_m\,;$$

$$Carry = if\,((A_m \oplus B_m) == 1) \qquad then\ cout = cin\ ;$$

$$else\ if\,((A_m \oplus B_m) == 0) \;\; then\ cout = A_m\ ;$$

The whole architecture of the proposed 108-bit adder is shown in the figure 7 and the sum/carry generator in figure 8. The figure 9 depicts the data path delay of the proposed adder and the conventional adder [1].

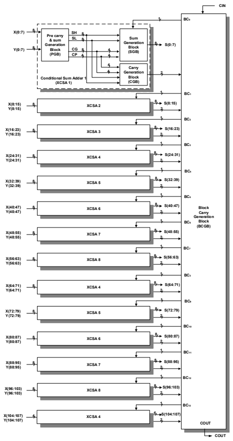
CIN
Pre carry & sum Generation Block (PGB)
Sum Generation Block (SGB)
Carry Generation Block (CGB)
Conditional Sum Adder (XCSA 1)
XCSA 2
XCSA 3
XCSA 4
XCSA 5
XCSA 6
XCSA 7
XCSA 8
XCSA 4
Block Carry Generation Block (BCGB)
COUT

Figure 7. Proposed 108-bit adder architecture

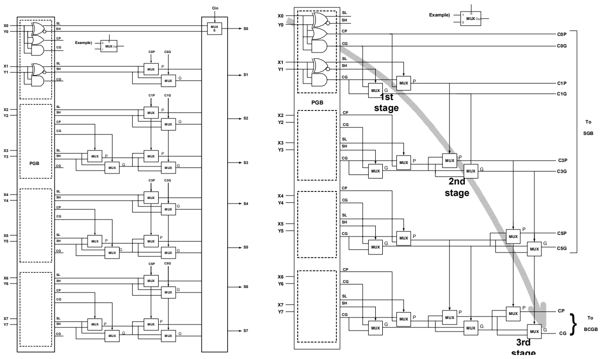

Figure 8. Proposed 8-bit sum/carry generator in XCSA

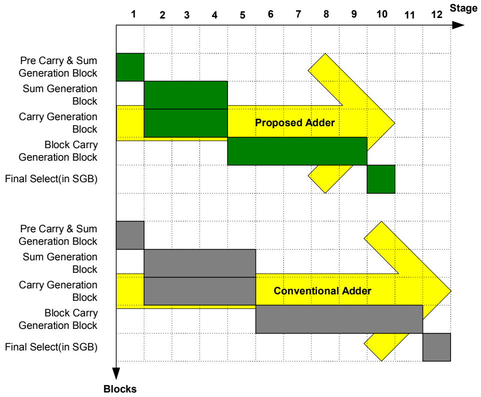

Figure 9. Data path of proposed adder and conventional one

6. COMPARISONS

6.1 Synthesis Conditions

The Verilog-HDL language was used to describe conventional and proposed architectures of full-adder, multiplexer, compressor structures, and data path architecture. To retain the structure in the logic synthesis, we applied the commands - *set_dont_use, set_dont_touch* to the Synopsys dc-compiler.

6.2 Booth Encoder and PPG module

The comparison of the proposed multiplier and the conventional ones in terms of booth encoder and PPG is summarized in the table 3. The gate count of booth encoder and PPG is decreased about 25%, and the critical path about 50% compared to conventional designs. The simulation result is 0.72ns in the delay time even with twice driving capacity input buffers being used.

Table 3. Comparison of booth encoder and PPG

		Ohkubo [1]	Goto [2]	Proposed
gate count	Booth encoder	38	50	26
	PPG	28	20	24
Critical path (gate)		6	5	3
Delay time		1.1ns	-	0.68ns

6.3 Compression module

We reduced the critical path 13% down by one full-adder to 7 stages and the number of full-adders down by two full-adders to 26 full-adders, as shown in table 4. The propagation time of the critical path is 1.45ns.

Table 4. Comparison of compressors for 28 rows

	Ohkubo [1]	Goto [2]	Proposed
full-adder count	28	28	26
Critical path(FA/gate)	8/16	8/16	7/14
Delay time	1.9ns	-	1.45ns

6.4 Addition Module

The table 5 shows that we reduced gate counts and improved critical path more effective than conventional adders. The number of gate counts reduces by about 200 gates (14%) to 1050 and the improved critical path by two or three stages to 10. The propagation time of the critical path is 1.12ns.

Table 5. Comparison of 108-bit adder

		Ohkubo [1]	Goto [2]	Proposed
gate count		1208	1162	1050
Critical path (gate)	Sum	4	5	3
	Carry	4	3	3
	Block carry	5	4	4
	Total	13	12	10
Delay time		1.4ns	-	1.12ns

6.5 Summary

The table 6 shows that summarized comparison results of the comparison conventional multipliers with proposed multiplier. The result of gate counts does not reach our expectation. The reason is mainly from the fact that the compressor occupies the most part of a multiplier [2].

Table 6. Comparison of Multiplier

	Ohkubo [1]	Goto [2]	Proposed
Gate Count	24050	23166	22080 (5% ↓)
Critical Path (gate)	35	33	28 (15% ↓)
Delay Time	4.4ns	4.1ns	3.25ns (21% ↓)

7. SIMULATION RESULTS

The propagation time of a critical path is 3.25ns at 2.5V on a 0.18um process technology and is shown in figure 10. Figure 11 shows a layout from Apollo.

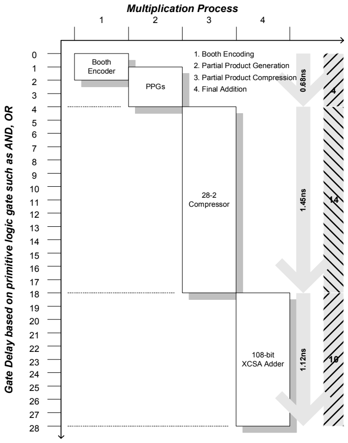

Figure 10. Multiplication time of 54x54-bit multiplier

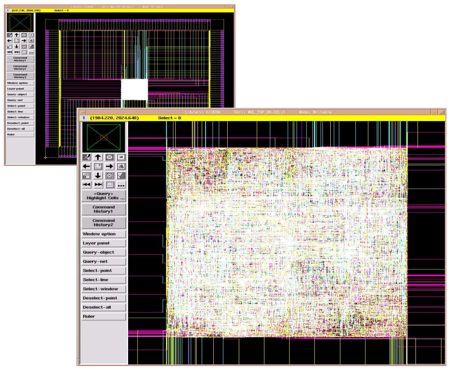

Figure 11. Layout of 54x54-bit multiplier (Apollo)

8. CONCLUSIONS

The multiplier is a major element to determine the DSP performance. Especially, the current mobile appliances are require chips with high speed operation and low power dissipation. Thus, we proposed the novel booth encoder, the 28-2 compressor, and the XOR based adder to create high performance multipliers. While our experimental implementation shows comparable results to manually drawn designs, we believe that improvements in EDA tools are also critical in achieving better outcomes.

9. REFERENCES

[1] N. Ohkubo, et al., "A 4.4ns CMOS 54x54-b Multiplier Using Pass -Transistor Multiplexer", *IEEE J. of Solid-State Circuits,* vol. 30, no. 3, pp. 251-257, Mar., 1995.

[2] G. Goto, et al., "A 4.1-ns Compact 54×54-b Multiplier Utilizing Sign-Select Booth Encoders", *IEEE J. of Solid-state Circuits*, vol. 32, no. 11, pp. 1676-1681, Nov. 1997.

[3] A.P. Chandrackasan and R.W. Brodersen, *Low power digital CMOS CMOS Design*. Norwell, MA: Kluwer, 1995.

[4] J. F. Cavanagh, "*Digital Computer Arithmetic - Design and Implementation*", Mc-Graw-Hill, pp. 137-235, sep., 1986.

[5] K. Yano, et al., "A 3.8-ns CMOS 16x16-bit Multiplier Using Compl- ementary Pass-transistor Logic", *IEEE J. of Solid-State Circuits,* vol. 25, no. 2, pp. 388-395, Apr., 1990.

[6] R. Zimmermmann, et al., "Low-Power Logic Styles: CMOS Versus Pass-Transistor Logic", *IEEE J. of Solid- state Circuits*, vol. 32, no. 7, pp. 1079-1090, July, 1997.

[7] Verilog-HDL by Cadence co., ltd.

[8] Chip Synthesis by Synopsys co., ltd.

[9] Hagihara. Y, et al., “A 2.7ns 0.25um CMOS 54x54 b multiplier”, Solid-State Circuits Conference, 1998, Digest of Technical papers, 45th ISSCC 1998 IEEE International, pp. 296-297, 5-7 Feb 1998.

[10] O. L. McSorley, “High Speed Arithmetic in Binary Computers”, Proc. IRE, Vol 49, pp 67-71, Jan. 1961

[11] Electrical and computer Engineering, Oklahoma state university.
http://ee.okstate.edu/Courses/Descriptions/6263.htm.

[12] Gustavo a. Ruiz, "Evaluation of three 32-bit CMOS adders in DCVS Logic for Self-Timed Circuits", *IEEE IEEE J. of Solid-State Circuits*, Vol.33, no.4, pp.604-613, April, 1998.