실시간 멀티미디어 시스템을 위한 새로운 고속 병렬곱셈기

조 병 록† · 이 명 옥††

요 약

본 논문에서는 고속 병렬 곱셈기에서 속도향상을 위해 부분 곱을 가산하는 과정에 구성되는 CSA(Carry Selèct Adder) 트리에 새로운 압축기를 적용한 새로운 첫 번째 부분 곱가산(First Partial product Addition : FPA)을 제안하여 기존의 전가산기를 이용한 병렬가산기보다 부분곱을 계산하는 속도를 약 20% 개선할 수 있게 했다. 새로운 회로는 새로운 FPA 구조를 사용하여 최종 합 CLA 비트를 N/2로 줄인다. 2.5v 0.25um CMOS 기술을 이용하여 제작된 16×16 곱셈기는 5.14nS의 곱셈 고속을 얻었다. 이 곱셈기의 구조는 파이프라인 설계에 용이하며 고성능을 낸다.

New High Speed Parallel Multiplier for Real Time Multimedia Systems

Byung Lok Cho† · Mike Myung-Ok Lee††

ABSTRACT

In this paper, we proposed a new First Partial product Addition (FPA) architecture with new compressor (or parallel counter) to CSA tree built in the process of adding partial product for improving speed in the fast parallel multiplier to improve the speed of calculating partial product by about 20% compared with existing parallel counter using full Adder. The new circuit reduces the CLA bit finding final sum by N/2 using the novel FPA architecture. A 5.14*nS* of multiplication speed of the 16×16 multiplier is obtained using 0.25㎛ CMOS technology. The architecture of the multiplier is easily opted for pipeline design and demonstrates high speed performance.

키워드 : Parallel Multiplier, ASIC, Full Custom Design, SoC, IP, Multimedia Communication System, FPA, CSA, CMOS, Pipelining Low Power

1. Introduction

The most important problem in digital signal processing is the multiplication and addition requiring fast operation. The processing is done by repeated operations of multiplication and addition like DFT (Discrete Fourier Transform), Convolution, Correlation etc. As multimedia systems are more complexed and important, it is necessary to design fast multiplier in the multimedia systems though much studies on multiplier has been done. The algorithm applied to fast multiplier proposes the methods which can reduce the number of partial product to *N/2* (Radix-4) by using modified Booth algorithm [1] utilizing multi-bit recording, and use Wallace tree [2] architecture to reduce delay of partial product addition, and utilizes CSA tree [3]. The algorithm presents the way of reducing propagation delay and proposes simple and regular architecture suitable for VLSI design [3]. Also, using 4 : 2 compressors recently uses the algorithm in reducing the numbers of partial products. The algorithm adopts the way of adding the final two of partial products ultimately by using fast adder CLA (Carry Lookahead Adder) [4-6]. This paper proposes new parallel counter architecture, and the method of reducing *N/2* delay for the bit of the final CLA, which has the largest delays by calculating sub-partial product obtained first during operation in advance.

† 정 회 원 : 순천대학교 전자공학과 교수
†† 정 회 원 : 동신대학교 전기전자정보통신공학부 교수
논문접수 : 2002년 10월 14일, 심사완료 : 2003년 10월 24일

2. A New Development of Fast Parallel Multiplier

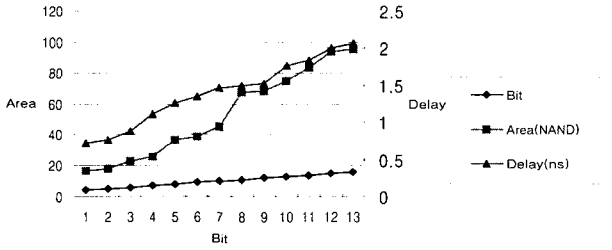
2.1 Architecture of new compressor

The principle of reducing n-number of partial products to construct partial summed CSA is similar to parallel counter counters the number of '1' in each bit string of partial products. In other words, parallel counter calculates the sum of '1' in each string to minimize propagation delay in adding partial product. This is suitable for the architecture of parallel counter. By definition of binary, if the result of sum in parallel counter is 2-bit, the largest number that can be represented in binary is 3. If the results are 3 bit, the number is 7. The largest number M that can be expressed is represented as equation (1),

$$M = \sum_{i=1}^{N} 2^{i-1} \ (i = 0, 1, 2, 3, \cdots) \qquad (1)$$

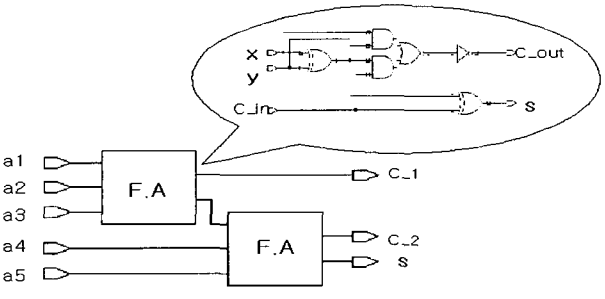
where M represents the bit of each string. M increases with N. Improvement of VLSI design technology makes the ratio of route delay higher than gate delay. Therefore, di-

vision of whole partial product to reduce the numbers routed can be of great help for performance improvement. Maximized use of the largest number M that can be represented can minimize the numbers routed from block to block. (Figure 1) shows area and delay depending on the bits of each string in parallel counter. The demerit of parallel counter is that the area and delay increased as the number of bits increased. In the (Figure 2), S refers to the sum value of its bit string. C_1 refers to Carry moving to the string next to bit, and C_2 refers to carry moving to the string next to 2-bit.

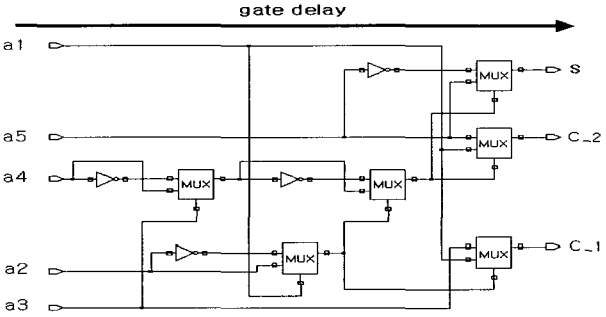


(Figurer 1) Area and delay depending on the bits of parallel counter(N : M compressor)

The architecture of 4 : 2 compressor is composed of two full adders as shown in (Figure 2) (a) where one of full adders has three gates delay and two full adders have six gates delay. (Figure 2) (b) shows only three mux delays after the gates of the full adder is replaced by a combination of multiplexors as expressed in equation (2).



(a) Full-adder



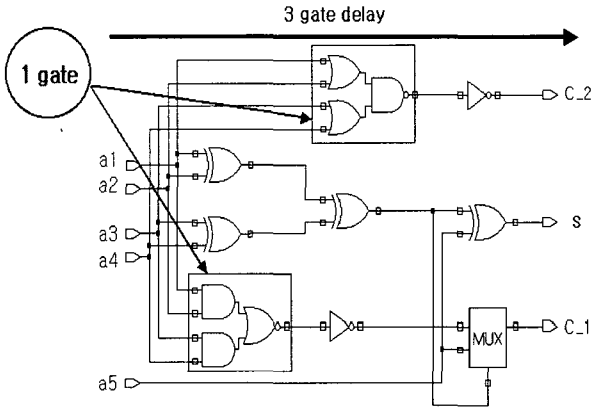
(b) Mux-based 4 : 2 compressor [4]

(Figure 2) 4 : 2 compressor model using

$$
\begin{aligned}
s1 &= (a1 \oplus a2) \oplus a3 \\
S &= (a4 \oplus a5) \oplus s1 \\
C\_1 &= a4\ a5 + (a4 \oplus a5)\ s1 \\
C\_2 &= a1\ a2 + (a1 \oplus a2)\ a3
\end{aligned} \tag{2}
$$

The architecture proposed in this paper can reduce gate delay by calculating only C_1 moving to next string first instead of calculating C_2(Carry) if n(number of partial product) is 5 when calculating Carry moving to next string. The largest number that can be represented in 3-bit is $2^1 + 2^1 + 2^0 = 5$. This makes the logic calculating carry simple and reduces delay as expressed in equation (3) whose symbols are indicated in (Figure 3).

$$
\begin{aligned}
g &= [(a1 \oplus a2) \oplus (a3 \oplus a4)] \\
h &= a1\ a2 + a3\ a4 \\
S &= a5 \oplus g \\
C\_1 &= g\ a5 + g'\ h \\
C\_2 &= [(a1 + a2)(a3 + a4)]
\end{aligned} \tag{3}
$$



(Figure 3) Proposed new 4 : 2 compressor

〈Table 1〉 4 : 2 Comparisons for compressor delay using 2.5v 0.25㎛ CMOS Cell Based technology

Performance Comparison	FA based 4 : 2 compressor [(Figure 2) (a)]	Mux based 4 : 2 compressor [(Figure 2) (b)]	New 4 : 2 parallel counter [(Figure 3]
Delay [ns]	0.87	0.85	0.63
Power [uW]	293.6801	317.9863	321.0827

(Figure 3) shows that C_1 is the carry moving to next end in calculating carry by applying RB Adder used for RB (Redundant Binary) operation whose expression belongs to Radix-2 SD family representing digit set -1, 0, 1 [7-9]. Note that our proposed circuit shows a good result with three buffers at the output stage of the compressor circuit though the 4 : 2 compressor has many fan-ins. Binary number of two bits can be expressed as binary number, however it is not suitable for high speed parallel multiplier architecture since RB-to-Binary converter is necessary, *i.e.*, RB number

must be transformed into binary number, which results in additional hardware and propagation delay. C_2 does not calculate carry moving forward by 2-bit as parallel counter does, but carry moving forward by 1-bit. Therefore C_2 reduces four or five partial products to three and finds 3 gate delays. Compared with the (Figure 2) (a), one gate delay takes over 3 gate delays. When it passes through two full adders and one half adder, it goes through 7 gate delays. Therefore, delay gains of about 30% can be obtained in adding partial product. <Table 1> represents a performance comparison of delay for three types of 4 : 2 compressor, based on simulation results using 0.25㎛ CMOS standard cell library and process parameters in the primary level at the supply voltage of 2.5v. Our suggested RBM 4 : 2 compressor as shown in (Figure 3) indicates outperformance of 20% delay reduction compared with full adder 4 : 2 compressor in (Figure 2) (a) and mux-based 4 : 2 compressor in (Figure 2) (b) though power consumption by the proposed 4 : 2 parallel counter is slightly larger than conventional due to more larger current consumption of the complex CMOS gates. Note that the simulations are completed with 0.25㎛ CMOS process technology in HSPICE tool.

2.2 High Speed Parallel Multiplier with New FPA

The method used for improving multiplication speed is Booth algorithm reducing partial product to N/2. This method has been used so far in nearly all-fast multipliers. The methods of using Wallace tree and 4 : 2 compressors are used in adding partial product [4, 5, 10, 11]. Also, as for complement on 2, sign extension elimination method is used for reducing sign extension. CLA (Carry Lookahead Adder) and CSA (Carry Select Adder) are used for adding last two partial products [4, 6]. The key to improving speed is how much efficiently partial product is added, and how much fast last large bit is added. These two factors have effects on the speed of multiplier. Especially, as the CLA speed accounts for about one-third of the speed of whole multiplication, designing fast adder is very important. As bit increases, circuit of parallel counter gets complex. Its area and delay are increased in a linear way and efficiency in area or delay decreases (Figure 1). However, the reduction of partial Products by dividing them in 3bits or 5bits is most efficient. Conventional Booth encoder is shown in <Table 2> and its equations are expressed in equation (4).

〈Table 2〉 Booth encoder for Z = X Y case

X2i + 1	X2i	X2i - 1	Q(i)	Y	Neg	PY
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	1	0	1
0	1	1	2	0	0	1
1	0	0	-2	0	1	0
1	0	1	-1	1	1	0
1	1	0	-1	1	1	0
1	1	1	0	0	0	0

$$
\begin{aligned}
&Y = X_{2i} \oplus X_{2i-1} \\
&Neg = X_{2i+1} \cdot (X_{2i} \cdot X_{2i-1})' \\
&PY = X_{2i+1}' \cdot (X_{2i} + X_{2i-1}) \\
&X = -X_{N-1} 2^{N-1} + \sum_{i=0}^{N-2} X_i 2^i \\
&\quad = \sum_{i=0}^{N/2-1} (-2X_{2i+1} + X_{2i} + X_{2i-1}) \cdot 2^{2i} \\
&Z = X \cdot Y = \sum_{i=0}^{N/2-1} Y \cdot Q(i) \cdot 2^{2i} \qquad (4)
\end{aligned}
$$

X : Multiplier, Y : Multiplicand, and Z : Product

Here the partial products which are generated in the case of adding '1' into LSB of the partial product when it is being transformed as two's complement are expressed as equation (5), which is a case of encoding extension elimination method and Q(i) = {-2, -1, 0, 1, 2}

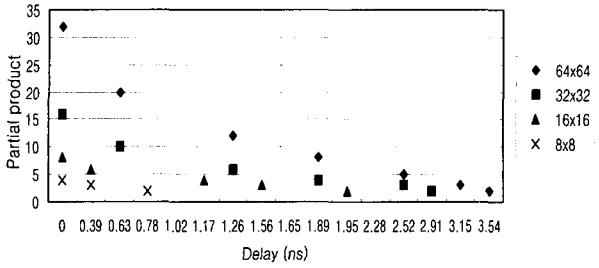
If Y = '*1*' and PY = '*1*' and Neg = '*0*',

$$
\begin{aligned}
&P_k = \left\{ (2^N Y_{N-1})' + \sum_{i=0}^{N-1} 2' y_i \right\} g\, 2^{2k} \\
&\text{and } PY = \text{'0'} \text{ and } Neg = \text{'1'} \\
&P_k = \left\{ (2^N Y_{N-1} + (\sum_{i=0}^{N-1} 2' y_i) \right\} g\, 2^{2k} \\
&\text{if } Y = \text{'0'} \text{ and } PY = \text{'0'} \text{ and } Neg = \text{'1'} \\
&P_k = \left\{ 2^N Y_{N-1} + \sum_{i=0}^{N-1} 2' y_i + (2^0)' \right\} g\, 2^{2k} \\
&\text{and } PY = \text{'0'} \text{ and } Neg = \text{'1'} \\
&P_k = \left\{ 2^N Y_{N-1} + \sum_{i=0}^{N-1} 2' y_i \right\} g\, 2^{2k} \qquad (5) \\
&\textit{where, } ()' \textit{ is complement,} \\
&\qquad k = \frac{N}{2} - 1 (k = 0, 1, 2, 3 \cdots)
\end{aligned}
$$

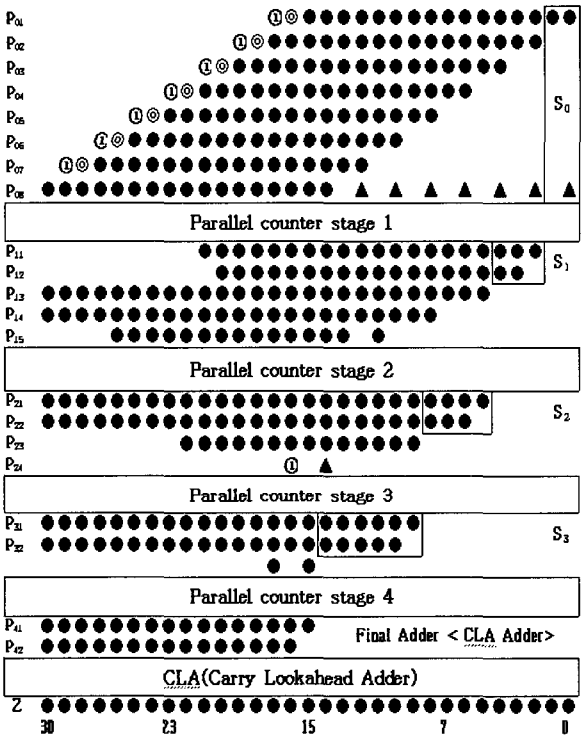
(Figure 5) shows the process of calculating partial product in the case of 16×16. Looking the process of calculation of partial products, the partial products divided in 3bits and 5bits passes through first parallel counter stage based on our proposed new 4 : 2 compressor, and the results of sub-bit of partial product are obtained first whenever partial product passes through each stage. As such lower bit adder possessing 3-6bits first can calculate results obtained, it is possible to calculate the results of sub-partial product while upper partial products are passing through 4 : 2 compressor stage. The bits of upper partial products passed through last stage are 16.

(Figure 4) shows a delay comparison for parallel counter for N = 8, 16, 32, 64 of N×N multiplier. Here each mark represents parallel counter stage depending on the N. It is found that delay difference between N = 32 and N = 64 needs one more stage, which means 0.63*nS* delay. Parallel counter

stage based on N-bit of multiplier and multiplicand is stage number + 1 as N becomes 2N.



(Figure 4) Delay comparison of parallel counter for N×N multiplier where the delays for 5 : 3 and 3 : 2 parallel counter are 0.63*nS* and 0.39*nS*, respectively



(Figure 5) Calculation process using parallel counter in case of 16×16

Where ● : partial product of '0' or '1'-bit, ◎ : complement bit by elimination of sign-bit extension, ▲ : LSB of the partial product for 2's complement + '1'-bit when $Q(k) = \{-1, -2\}$. Lower partial product sums of $S_0 \sim S_3$ as shown in (Figure 5) are expressed in equation (6) as the below :

$$S_0 = \sum_{i=0}^{1} P^i_{00} \cdot 2^i + N^0$$

$$S_1 = \sum_{i=0}^{2} P^i_{10} \cdot 2^i + \sum_{i=0}^{1} P^{i+1}{}_{11} \cdot 2^{i+1} + S^2{}_0$$

$$S_2 = \sum_{i=0}^{3} P^i_{20} \cdot 2^i + \sum_{i=0}^{2} P^{i+1}{}_{21} \cdot 2^{i+1} + S^3{}_1$$

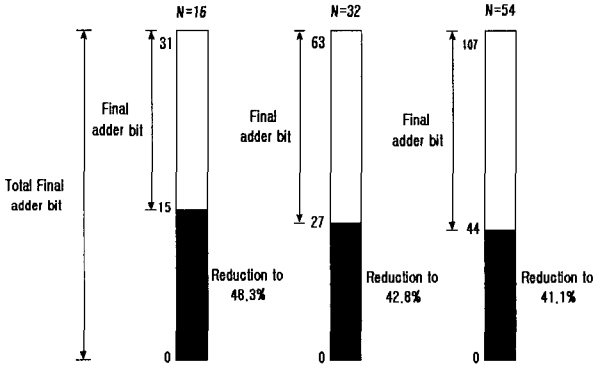
$$S_3 = \sum_{i=0}^{5} P^i_{30} \cdot 2^i + \sum_{i=0}^{4} P^{i+1}{}_{31} \cdot 2^{i+1} + S^4{}_2$$

A general formula can be expressed as

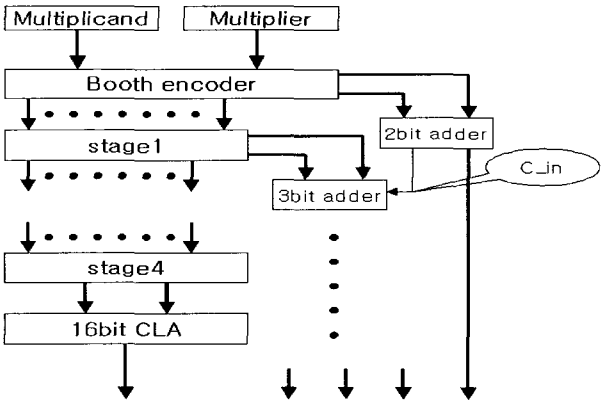
$$S_n = \sum_{i=0}^{n'} P^i_{n0} \cdot 2^i + \sum_{i=0}^{n'-1} P^{i+1}{}_{n1} \cdot 2^{i+1} + S^{n+1}{}_{n-1}$$

$$(n = 1, 2, 3, \cdots \cdots), \qquad (6)$$

Where n : parallel counter stage, n′ : two MSBs of the partial product, S_0 : sum of lower bit in the first generated partial product, S_1, S_2, S_3 : sum of lower bit in the partial product out of each CSA tree stage, P^k_{ji} : k is CSA tree stage, j is a row of k^{th} CSA tree stage, i is a j^{th} bit of the partial product, and N^0 : Negative of <Table 2> for Booth encoder. The upper partial products reached first are calculated in advance by CLA, and then they are selected by sub-calculation values to output final results. Using 16-bit CLA instead of 32-bit CLA can reduce effectively the delay by final adder. (Figure 6) shows a gain of the final adder bit when N = 16, 32, 54, 64 of N×N multiplier is experimented. It is found that the final adder bit can be reduced as N = 16 : 48.3%, N = 32 : 42.8%, N = 54 : 41.1%. Thereby (Figure 7) shows an architectural block diagram of high speed parallel multiplier with proposed new FPA based on aforementioned calculation process in (Figure 5).



(Figure 6) A gain of the final adder bit with FPA applying N = 16, 32, 54, 64 of N×N multiplier

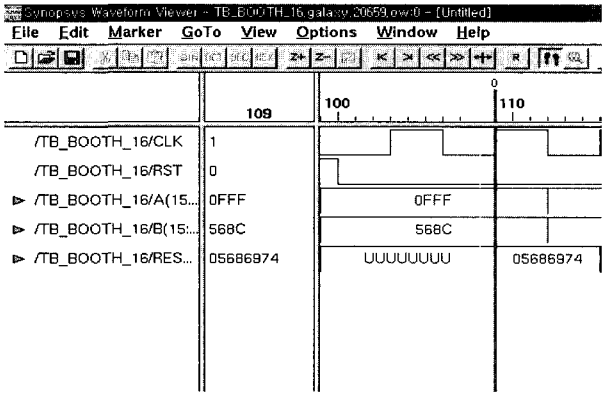


(Figure 7) Block diagram of fast parallel multiplier with proposed new FPA

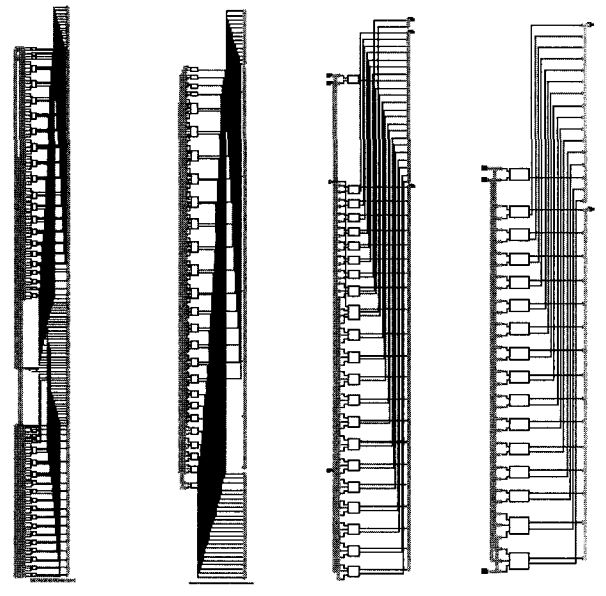
(Figure 7) shows block diagram of fast parallel multiplier with proposed new FPA.

3. Results and Discussion

The system Synthesis and function simulations are done by Synopsys CAD tool using 0.25㎛ CMOS standard cell[12]. (Figure 8) confirms a logic simulation result for 16×16 multiplier applying a new FPA algorithm with new compressor as expected.



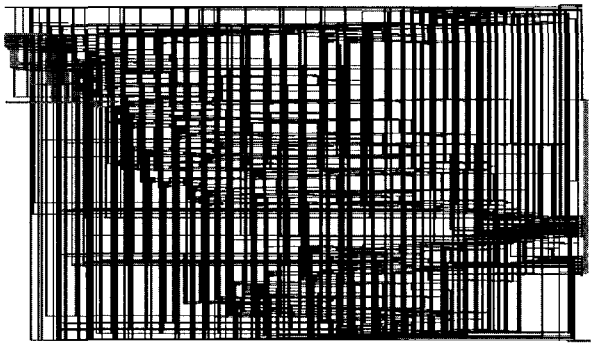
(Figure 8) A logic simulation result for our suggested 16×16 multiplier using new FPA



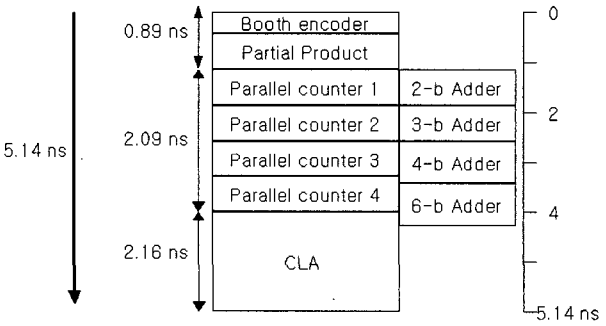
(a) stage 1 (b) stage 2 (c) stage 3 (d) stage 4

(Figure 9) Circuit diagrams of each stage of parallel counter

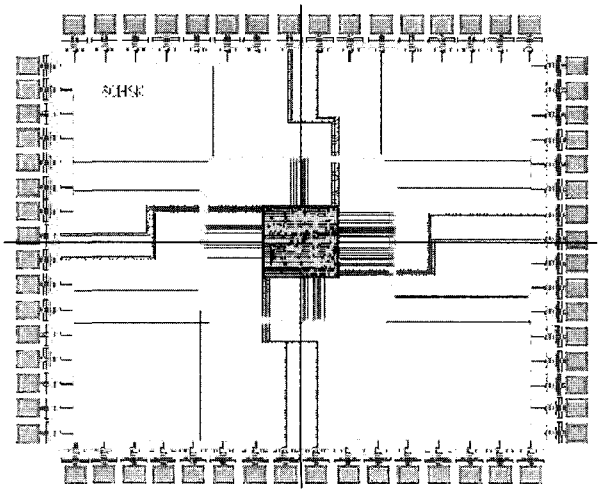
(Figure 8) are the synthesized results for circuit diagrams of each stage of parallel counter and the 16×16 multiplier, respectively. The final synthesized multiplier demonstrates about 5000 gates as an optimal design where (Figure 9) and (Figure 10) are synthesized with 2.5v, 0.25㎛ CMOS Cell Library [12]. Further, (Figure 11) shows the delay(nS) required in each stage up to final sum in Booth Decoder. The n-b Adder portion in the middle section is the block adding in advance partial product and the sub-partial product generated first in each stage of parallel counter. The values are calculated before upper partial products are calculated. (Figure 12) is a microphotography layout of proposed 16×16 multiplier using 0.25㎛ CMOS Cell technology. The chip size is just 2×2(㎟) with 0.35×0.39(㎟) of core area, and the system has about 5000 gate counts.



(Figure 10) A synthesized 16X16 Multiplier with FPA



(Figure 11) Delay times[*ns*] for each block of proposed 16×16 multiplier using 0.25㎛ CMOS Cell technology



(Figure 12) A microphotography layout of proposed 16X16 multiplier using 0.25㎛ CMOS Cell technology

There is delay of about 5.14*nS* in multiplication 16×16 bits. Also, each stage of parallel count has similar delays. Design with division of several stages can ensure high speed performance. It is expected to have higher performance if the multiplier is designed with pipelining architecture.

4. Conclusion

In this paper, we proposed new parallel counter to reduce the area and delay of parallel counter, which increases with increased bits of parallel counter We have succeeded in reducing delay by about 20%, which is required to reduce partial products down to 4 : 2 compressor. We proposed the method of carrying out prior addition of sub-partial products generated first in advance. Therefore we have succeeded in shortening addition time by reducing the bits of CLA by 50%,which adds final partial products. The final adder bit can be reduced as N = 16 : 48.3%, N = 32 : 42.8%, N = 54 : 41.1% for N×N multiplier. If the method proposed in this paper is applied to multiplier with high bits, the process of reducing partial products or the addition time of final CLA is shortened. Therefore the architecture by the proposed method is suitable for multimedia system needing real time processing [13].

References

[1] A. D. Booth, "A signed binary multiplication technique," *Quarter. J. Mechs. Appl. Math.*, Vol.4, No.2, pp.235-240, 1951.

[2] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. on Electron. Comput.*, Vol.EC-13, pp.14-17. 1964.

[3] Yong Seok Lee, Geun Seon Hong, and Yong Deak Kim, "Astudy on the design scheme of CSA array for the speed up of array multiplier," *J. IEEK*, Vol.31 B, No.5, pp.73-80, 1994.

[4] Norio Ohkubo, Makoto Suzuki, Toshinobu Shinbo, Toshiaki Yamanaka, Akibior Shimizu, and Katuro Sasaki, "A 4.4ns CMOS 54 54-b multiplier using pass-transistor multipiexer," *IEEE J. Solid-State Circuits*, Vol.30, No.30, pp.251-257, Mar., 1995.

[5] Gensuke Goto, Atuski Inoue, and Ryoichi Ohe, "A 4.1-ns compact 54 54-b multiplier utilizing sign-select booth encoders," *IEEE J. Solid-State Circuits*, Vol.32, No.11, pp. 1676-1681, Nov., 1997.

[6] Deuk-kyung Kim, Kyung-Wook Shin, Yong-Surk Lee, and Moon-Key Lee, "Area-time complexity analysis for optimal design of multipbit recoding parallel multiplier," *J. IEEK*, Vol.32 A, No.5, pp.71-80, 1995.

[7] N. Takagi, H. Yasuura and S. Yajima, "High-speed VLSI multiplication algorithm with a redundant binary addition tree," *IEEE Trans. on Comput.*, Vol.C-34, No.9, pp.789-796, Sep., 1985.

[8] Kyung-Wook Shin, "A high-speed complex multiplier based on redundant binary arithmetic," *J. IEEK*, Vol.34 C, No. 2, pp.29-37, 1997.

[9] B. I. Park, *et al.*, "A Regular Layout structured Multiplier based on Weighted Carry-Save Adders," *Proc. IEEE International Conference on Computer Design*, pp.243-248, Oct., 1999.

[10] Ahmad A. Hiasat, "New efficient structure for a Modular Multiplier for RNS," *IEEE* Trans. on Comput., Vol.49, No.2, pp.170-174, Feb., 2000.

[11] Young-in Kim and Jin-Ho Cho, "An architecture for 32 x 32 bit high speed parallel multiplier," *J. IEEK*, Vol.31 B, No.10, pp.67-71, 1994.

[12] IDEC, IDEC Cell Library Data Book. *IDEC-C221*, June, 2000.

[13] M. Lee, *et al*, "IPs and SoC Design with Ultra-Low Power Real-Time Embedded 3-D Multimedia Chip," CAD 및 VLSI 설계연구회 학술발표대회, 제1권, pp.47-52, 2002.

조 병 록

e-mail : blcho@sunchon.ac.kr

1987년 성균관대학교 전자공학과(공학사)

1990년 성균관대학교 대학원 전자공학과 (공학석사)

1994년 성균관대학교 대학원 전자공학과 (공학박사)

1987년~1988년 (주)삼성전자 종합연구소

1994년~현재 순천대학교 전자공학과 조교수

관심분야 : 디지털 통신이론, 디지털 통신시스템 ASIC 설계, 무선멀티미디어용 고속 모뎀 설계, 무선망 성능분석

이 명 옥

e-mail : mikelee@dsu.ac.kr

1981년 전북대학교 정밀기계공학과(공학사)

1983년 AME, Arizona State University (BS)

1985년 CME, Arizona State University (MNS)

1988년 CME, Arizona State University (PhD)

1992년~1994년 일본 동경대학교 Industrial Research Fellow

1986년~1996년 Motorola Inc.(USA, Arizona주 Mesa시 연구소 및 일본 동경지점 Senior Member of Technical Staff)

1996년~1997년 ETRI 양자소자연구실, 초빙교수

1997년~1997년 미국 캘리포니아주 Univ. of California at Berkeley(UCB), Visiting Professor

1998년~1998년 일본 동경대학교, Visiting Professor

1999년~2002년 (주)하이칩스 부사장 & 연구소장

1996년~현재 동신대학교 전기전자정보통신공학부 교수

2000년~현재 (주)하이멥스텔레콤 기술이사

관심분야 : 광자시스템용 Opto-ULSI 프로세서 Co-Design, 멀티미디어 통신시스템 IP 설계 및 SoC, 영상통신 Video/Audio 압축 기술, 지능형 BioMEMS 설계, Linux 기반 HW-SW기술