World Applied Sciences Journal 37 (2): 110-117, 2019
ISSN 1818-4952
© IDOSI Publications, 2019
DOI: 10.5829/idosi.wasj.2019.110.117

An Improved Circuit for Signed Number Modified Booth Multiplier

[1]Manan Joshi, [2]D.S. Chauhan and [3]B.K. Kaushik

[1]Uttarakhand Technical University, Dehradun, India
[2]GLA University, Mathura, India
[3]Indian Institute of Technology, Roorkee, India

Abstract: This paper presents a circuit for low power Radix-4, 32 × 32 bit multiplier with an improved performance. The proposed design overcomes the drawbacks of the earlier multipliers by using Hybrid-CMOS Logic style and CMOS transmission gate and CMOS NAND and NOT Gate based circuits and a novel D Flip Flop to perform all the operations. The proposed multiplier is compared with existing multipliers. Based on the simulation results, overall reduction of 15.3% in PDP, 7.97% in power consumption and 7.96% in critical path delay is observed for 32 bit multiplier.

Key words: Power delay product · CMOS logic · Transmission gate logic · Carry propagate adder · Partial product generator · Modified booth algorithm · Radix-4 multiplier · Analog circuit design

INTRODUCTION

Demand for increasingly portable yet high performance multimedia and communication products imposes stringent constraints on the power consumption of internal circuits. Among them the multipliers perform one of the most frequently encountered arithmetic operations in digital signal processors for embedded applications. Reduction in power consumption of multiplier circuit thus is desired.

Multiplication process consists of the following major steps: 1) recoding and generating partial products 2) reducing the partial products by partial product reduction schemes 3) adding the remaining rows of partial products by using a carry propagate adder to obtain the final product. A number of techniques have been developed to reduce power consumption by improving these steps.

Performance comparison review of 32 bit multiplier designs [1, 2] helps in focussing on improving the partial products generation in the modified booth algorithm. This leads to improvement in delay and power consumption. Considerable research work focuses on approaches that use modified algorithms, architectures to reduce power consumption. A lot of effort has been put into reducing spurious switching activity by balancing internal paths by a combination of architectural and transistor level optimisation techniques. Initially power consumption of functional units of multipliers [3] was minimized by partially guarded computation. A multiplier [4] extended the carry free approach to generate product for data with large dynamic range. This helped in reducing glitches and thus save power. A low power multiplier [5] designed to minimize switching activities of partial products is proposed. Glitch power minimization [6] by selective gate freezing is used to reduce switching activity. There-after, circuit-block switch-off scheme [7] was proposed for fast power efficient multiplier. A low power a carry free array multiplier [8, 9] that uses a left to right leapfrog (LRLF) structure, an upper/lower splitting structure, using on the fly conversion [10] in parallel with the linear reduction is proposed. Column bypassing technique [11] is proposed for improving speed. A multi precision Dynamic Voltage scaling Multiplier [12] with operands Scheduler aggressively reduces power consumption and is meant for error tolerant applications. A CMOS multiplier using Complementary Pass-Transistor Logic [13] is proposed. Circuit techniques were developed in [14] and these include the use of CMOS to give low power and high performance and the use of new complementary pass transistor logic-transmission gate full adder. A low power multiplier with the spurious power suppression techniques [15] and a low-power low-area multiplier based on shift and add architecture [16] are developed.

Corresponding Author: Manan Joshi, Uttarakhand Technical University, Dehradun, India.

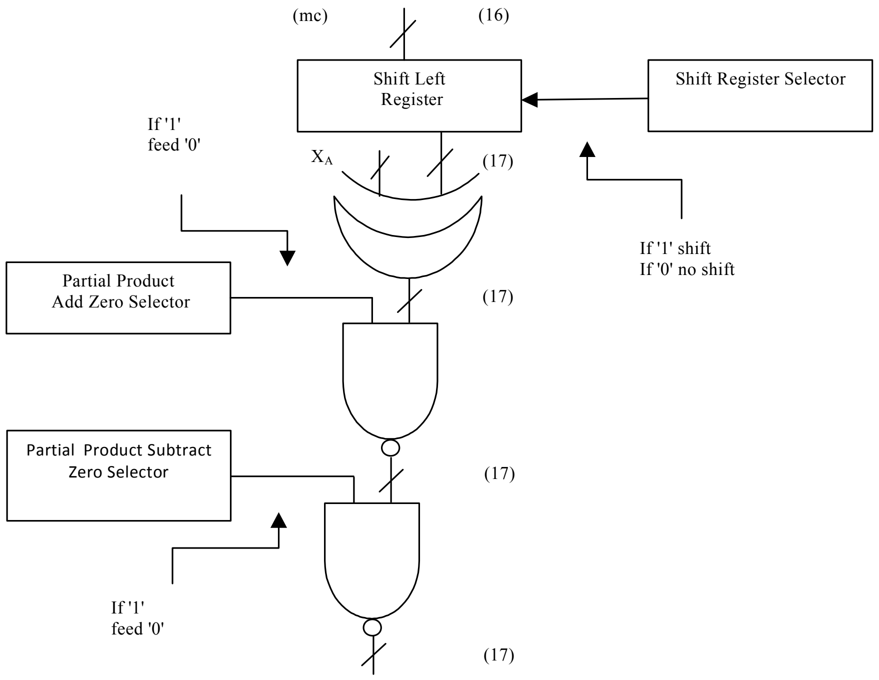

Fig. 1: Partial Product Generator

Multiplier [17] uses a regular partial product array using Modified Booth algorithm to reduce power consumption, while a shift add multiplication approach [18] is applied to obtain a low voltage micro power asynchronous multiplier though at the cost of speed. A high speed word level finite field Multiplier [19] in F_2^m using redundant representation meant for low precision multiplication applications and a ROM based logic design for low power 16 bit multiplier [20] is proposed for high precision applications. A new VLSI architecture of parallel multiplier accumulator based on radix-2 Modified Booth Algorithm [21] improves speed and power consumption. Algorithm, structure and circuits were analysed [22] and sign generation and 4-2 compressors used to minimise switching activities. An efficient sign extension scheme [23] to process the sign bits by allowing the multiplier to bypass processing sign extensions to reduce power consumption was proposed. A reorganised booth encoded carry save adder array [24] was implemented to reduce power consumption in the multiplier. Modified radix - 4 Booth encoders [25] to generate partial products that are summed by counters in an array with reducing sum and carry vectors was developed that reduced power consumption in the multiplier.

A multiplier [26] that employed a bottom up temporal tilting approach is used to design a leapfrog array to minimise spurious transitions and thus save power. Power aware scalable pipelined booth multiplier [27] is proposed. A proposed multiplier [10] implemented multiply-accumulate operation in multiplication time to perform multiplication at a faster rate. Another multiplier [28] meant for applications where approximate results can be tolerated uses "on the fly rounding". Dynamic power reduction is also achieved by monitoring the effective dynamic range of the input operands so as to disable unused sections of the multiplier and/or truncate the output product at the cost of reduced precision. Such a method can be used in sensor applications and artificial neural networks where operations in lower precisions are most frequently required.

The rest of the paper is organised as follows. Section II presents the proposed 8 × 8 bit multiplier with its functionality. Section III compares the proposed 8 × 8 bit multiplier with architectures proposed earlier [13-15]. Finally, conclusions are drawn in section IV.

The Proposed Multiplier: The proposed multiplier uses the radix - 4 booth algorithm to perform 32 × 32 bit multiplication. Fig. 1 shows the block diagram of the booth multiplier. It consists of carry propagate adder with carry look ahead generator, a mechanism to generate partial products and a register to hold the final product.

Partial Product Generation: A register holds the multiplicand and is part of the partial products generator. The partial product generator generates the partial

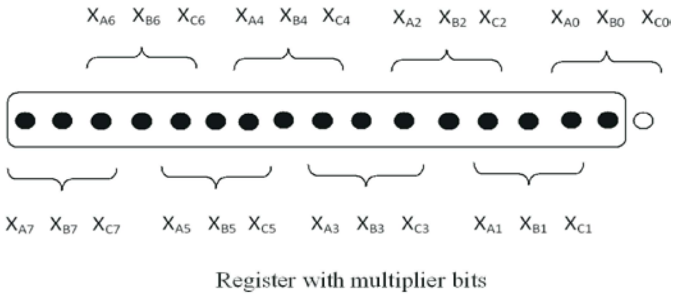
X_{A6} X_{B6} X_{C6}
X_{A4} X_{B4} X_{C4}
X_{A2} X_{B2} X_{C2}
X_{A0} X_{B0} X_{C0}
X_{A7} X_{B7} X_{C7}
X_{A5} X_{B5} X_{C5}
X_{A3} X_{B3} X_{C3}
X_{A1} X_{B1} X_{C1}
Register with multiplier bits

Fig. 2: Obtaining X_A, X_B and X_C from register with multiplier bits

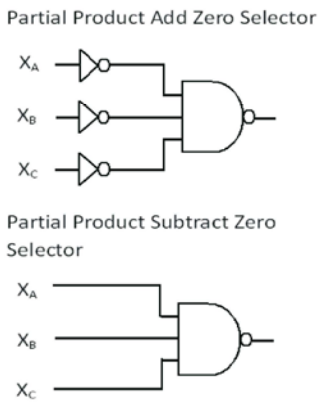
Partial Product Add Zero Selector
X_A
X_B
X_C
Partial Product Subtract Zero Selector
X_A
X_B
X_C

Fig. 3: Partial Product Add Zero and Subtract Zero Selector

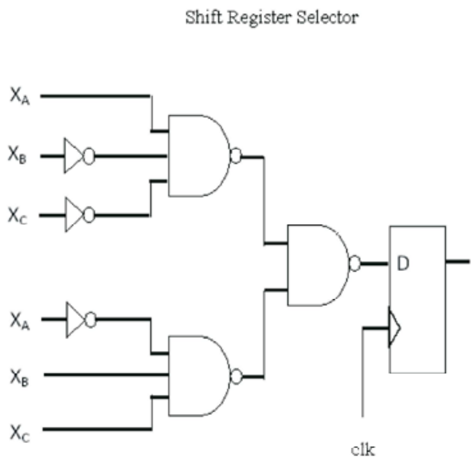

Fig. 4: Shift Register Selector

Table 1: Booth Multiplier Scheme

SELECT LINE (Encoding)	PARTIAL PRODUCTS (Operations)
000	Add Zero
001	Add Multiplicand
010	Add Multiplicand
011	Add 2*Multiplicand
100	Subtract 2*Multiplicand
101	Subtract Multiplicand
110	Subtract Multiplicand
111	Subtract Zero

products and it consists of a shift left register, an array of XOR gates, two arrays of NAND gates, shift register selector and partial product add zero selector and subtract zero selector. The input to shift register selector, add zero selector and subtract zero selector are X_A, X_B and X_C and they are obtained from the multiplier bits. The manner in which the bits X_A, X_B and X_C are obtained is shown in the Fig 2 below.

Table I below shows the radix - 4 booth recoding scheme and gives the partial products with the corresponding select lines.

Radix - 4 Booth Multiplier Scheme: Fig. 3 shows the block diagram of the partial products generator. Signals X_A, X_B, X_C are the least significant bits in the universal shift register. These signals form the select lines of the partial products generator. The multiplicand is represented by (mc).

The eight bit multiplicand is shifted left whenever the partial product is to be doubled based on the select line encoding. When shift register selector output is one multiplicand is shifted left. Array of XOR gates inverts the multiplicand whenever X_A is one. The AND gate array is used to give a zero output whenever partial product zero selector's output is one. Finally the partial product is obtained as the output of AND gate array. The select line encoding is obtained from the multiplier.

Fig 4 shows the structure of the partial product zero selector. The inputs X_A, X_B and X_C are derived from the register that holds the multiplier. Fig. 5 shows the generation of these bits. Fig. 6 shows the structure of the shift register selector. The same inputs X_A, X_B and X_C that are applied to partial product zero selector are also given to the shift register selector. The circuit of these two selectors are obtained from the Radix - 4 Booth Multiplier scheme given in Table I.

The Addition Process: Addition process is achieved with the help of carry propagate adder and look ahead carry generator.

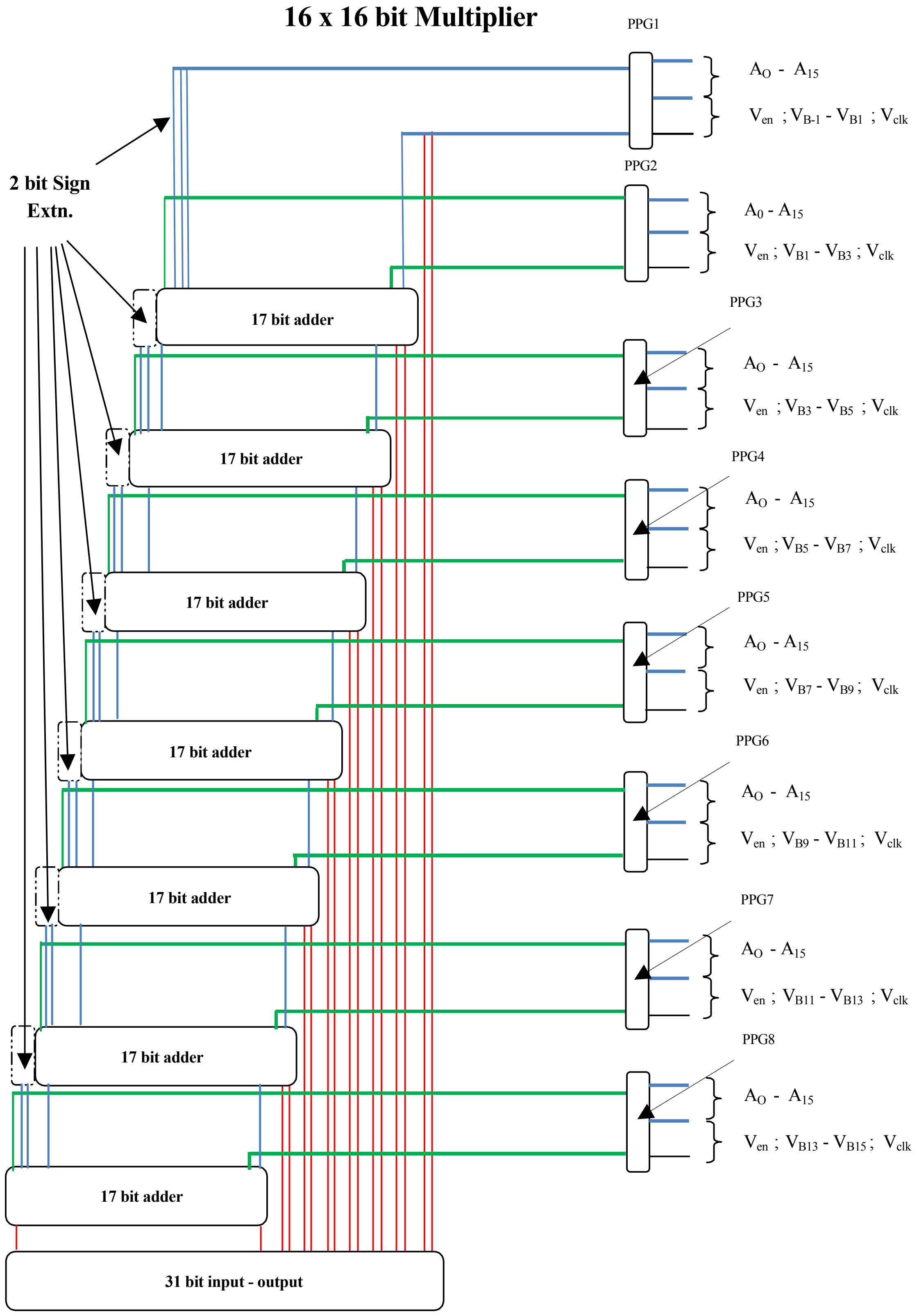
16 x 16 bit Multiplier
PPG1
A_O - A_{15}
V_{en} ; V_{B-1} - V_{B1} ; V_{clk}
PPG2
A_0 - A_{15}
V_{en} ; V_{B1} - V_{B3} ; V_{clk}
PPG3
A_O - A_{15}
V_{en} ; V_{B3} - V_{B5} ; V_{clk}
PPG4
A_O - A_{15}
V_{en} ; V_{B5} - V_{B7} ; V_{clk}
PPG5
A_O - A_{15}
V_{en} ; V_{B7} - V_{B9} ; V_{clk}
PPG6
A_O - A_{15}
V_{en} ; V_{B9} - V_{B11} ; V_{clk}
PPG7
A_O - A_{15}
V_{en} ; V_{B11} - V_{B13} ; V_{clk}
PPG8
A_O - A_{15}
V_{en} ; V_{B13} - V_{B15} ; V_{clk}
2 bit Sign Extn.
17 bit adder
17 bit adder
17 bit adder
17 bit adder
17 bit adder
17 bit adder
17 bit adder
31 bit input - output

Fig. 5: Structure of Proposed Multiplier

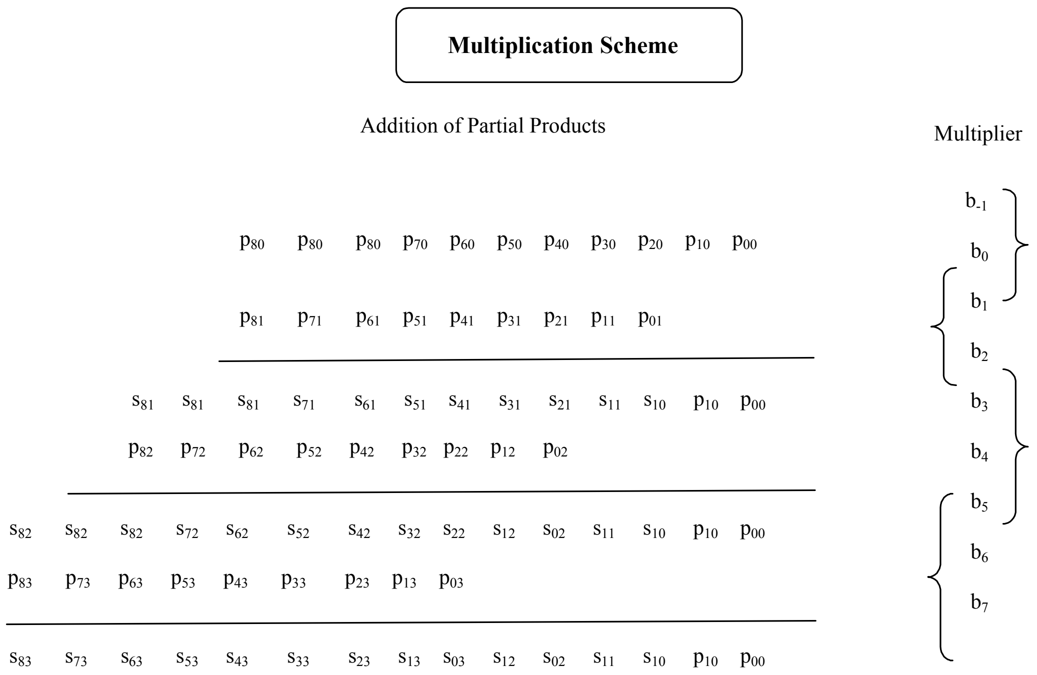
Multiplication Scheme
Addition of Partial Products
Multiplier
p80 p80 p80 p70 p60 p50 p40 p30 p20 p10 p00
p81 p71 p61 p51 p41 p31 p21 p11 p01
s81 s81 s81 s71 s61 s51 s41 s31 s21 s11 s10 p10 p00
p82 p72 p62 p52 p42 p32 p22 p12 p02
s82 s82 s82 s72 s62 s52 s42 s32 s22 s12 s02 s11 s10 p10 p00
p83 p73 p63 p53 p43 p33 p23 p13 p03
s83 s73 s63 s53 s43 s33 s23 s13 s03 s12 s02 s11 s10 p10 p00
b-1 b0 b1 b2 b3 b4 b5 b6 b7

Fig. 6: Partial Product Addition Scheme

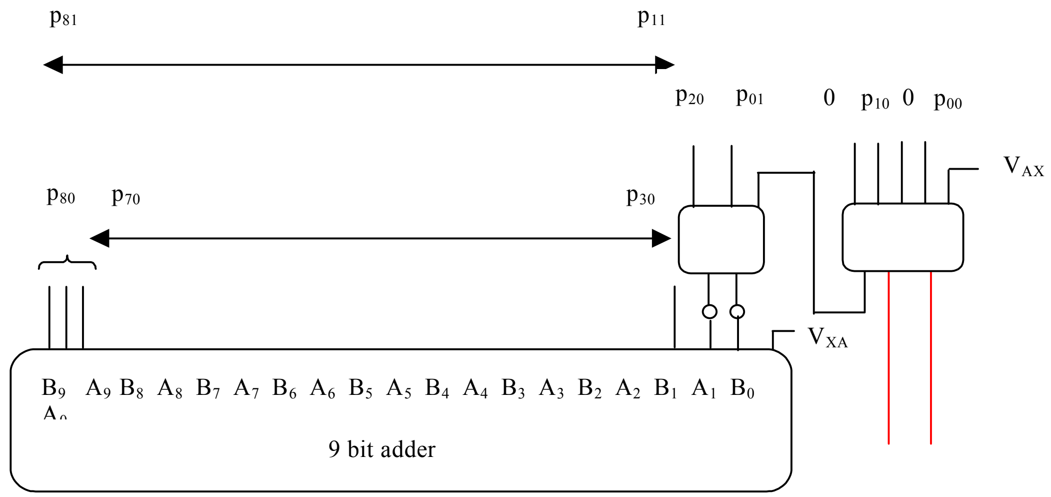

Fig. 7: Addition of first two Partial Products

C_0 = input carry
$C_1 = G_0 + P_0C_0$; $C_2 = G_1 + P_1G_0 + P_1P_0C_0$
$C_3 = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_0$
$C_4 = G_3 + P_3C_3$; $C_5 = G_4 + P_4G_3 + P_4P_3C_3$
$C_6 = G_5 + P_5G_4 + P_5P_4G_3 + P_5P_4P_3C_3$
$C_7 = G_6 + P_6C_6$; $C_8 = G_7 + P_7G_6 + P_7P_6C_6$
$C_9 = G_8 + P_8G_7 + P_8P_7G_6 + P_8P_7$

where $P_i = A_i + B_i$ (carry propagate) and $G_i = A_iB_i$ (carry generate). The sum and carry are generated using the expressions given below.

Sum: $S_i = P_i$ XOR C_i and
Carry: $C_{i+1} = G_i + P_iC_i$

The XNOR circuit [29] using transmission gates only followed by an inverter is used to generate the sum. To perform the multiplication process the three-three bits of the multiplier are used to generate partial product from the multiplicand. Fig. 7 shows this scheme.

Fig. 8 shows that a zero is appended as an LSB to the multiplier. Then partial products are generated as explained earlier. Curly braces indicate generation of

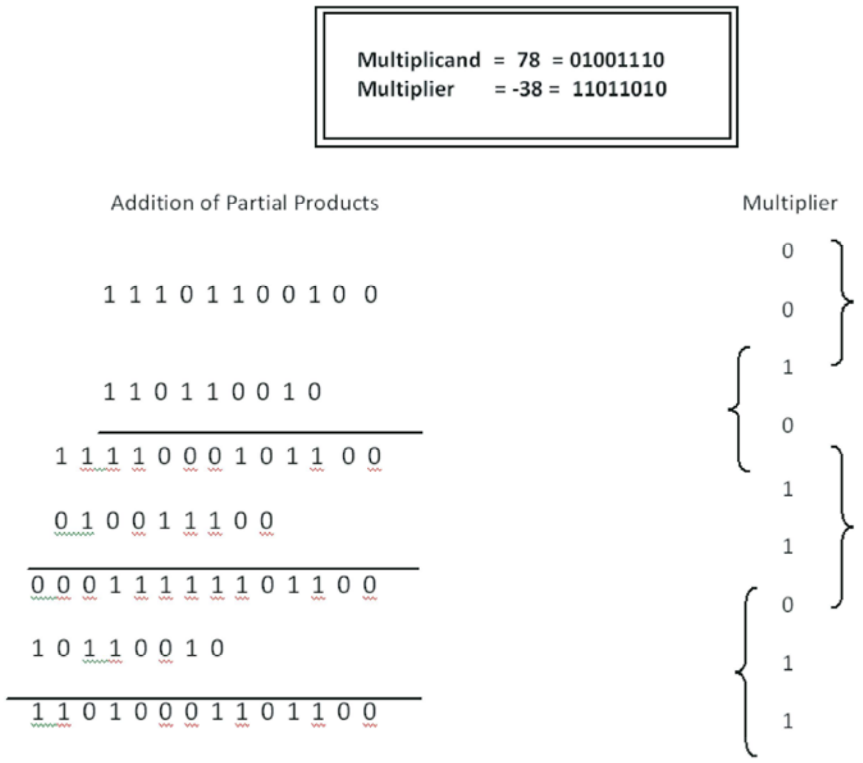

Fig. 8: Multiplication Results

X_A, X_B and X_C. Eight rows of partial products generated using partial product generator are shown. In general there will be (n)/2 partial products, where 'n' is the operand length. These bits are then added to get the final product.

The Overall Multiplication Process: Fig. 5 gives the overall structure of the 16 bit multiplier. The output of the partial product generator is given to seventeen bit adder. The first two LSBs do not form part of the addition process and go directly to the output. The MSB of the first partial product generator is given as the last two input bits to the adder. This performs the sign extension process. Fig. 6 explains the requirement of this sign extension in the signed multiplication process.

Fig 8 is used to demonstrate the signed multiplication process using an example. The dashed block at the MSB of the seventeen bit adder indicates the sign extension process wherein the MSB of the previous adder is given to the last three MSBs to the next seventeen bit adder. This can be done because output from carry look-ahead adder is from an inverter. The high drive capability of inverter enables the sign extension process. The extra gates that are employed so as to generate a regular partial product array are thus avoided. Hence a lot of saving in terms of number of gates results which in turn saves crucial power. The red lines indicate that the first two LSBs of adder are sent directly to the output. The carry input to each adder is the corresponding X_A used for the generation of the partial product. For the addition of X_{A0} modification of first seventeen bit adder is done as shown in Fig. 7. An 8 bit multiplier has a 9 bit partial product, a 16 bit multiplier has a 17 bit partial product and a 32 bit multiplier has a 33 bit partial product.

RESULTS AND DISCUSSIONS

The proposed multiplier is verified by means of Tanner version 13.0 using 0.18μm technology. The circuit is simulated at a power supply of 1.8V and a frequency of 250MHz. The energy consumption of the proposed multiplier is less as compared to the earlier reported multipliers. The 8 bit, 16 bit and 32 bit multiplier are designed and simulated. The simulation results show an improvement of in power consumption and critical path delay over the earlier reported multipliers. The percentage improvement is shown in the Table III below.

Table II: Performance comparison with the existing multipliers

n	Multiplier Name	Power (mW)	Delay (ns)	% Δ (Power)	% Δ (Delay)
8	Proposed	0.450	3.49	7.97	8.39
	Kuang *et al.* [30]	0.489	3.81	20.5	4.27
	Yeh *et al.* [31]	0.615	3.98	6.81	2.92
	Kang *et al.* [32]	0.660	4.10	-	-
16	Proposed	2.80	5.1	8.4	8.9
	Kuang *et al.* [30]	3.06	5.6	12.8	0
	Yeh *et al.* [31]	3.51	5.6	-0.86	1.7
	Kang *et al.* [32]	3.48	5.7	-	-
32	Proposed	15.46	6.12	7.97	7.96
	Kuang *et al.* [30]	16.80	6.65	0.77	0.598
	Yeh *et al.* [31]	16.93	6.69	9.23	1.32
	Kang *et al.* [32]	18.65	6.78	-	-

Table III: % improvement in Power Delay Product

n	Multiplier Name	PDP (10^{-12})	% PDP Change
8	Proposed	1.57	15.6
	Kuang *et al.* [30]	1.86	24.1
	Yeh *et al.* [31]	2.45	9.5
	Kang *et al.* [32]	2.71	-
16	Proposed	14.3	18.1
	Kuang *et al.* [30]	17.1	13.2
	Yeh *et al.* [31]	19.7	0.5
	Kang *et al.* [32]	19.8	-
32	Proposed	094.60	15.3
	Kuang *et al.* [30]	111.72	1.36
	Yeh *et al.* [31]	113.26	10.4
	Kang *et al.* [32]	126.45	-

CONCLUSION

Table II above presents the comparison between the existing multipliers and the proposed multiplier. The improvement is due to a number of factors.

Firstly flip flops used are explicit pulsed and only one powerful pulse generator drives all the flip flops in a register. The flip flop [33] consists of two inverters and a transmission gate in between. This results in considerable saving in power consumption in the partial product generator. Secondly, addition process is achieved with the help of two, three and four input NAND gates, inverter. When a transmission gate is used as in XOR gate it is ensured that a transmission gate is preceded and followed by a CMOS inverter or a CMOS NAND gate. This improves the signal speed and also eliminates static and dynamic power loss. Thirdly, a large number of glitches or spurious transitions occurring in the partial product generation [9] and addition process are eliminated in this design.

In this partial product generation and addition process simple and less hardware results in a reduction in power consumption. The 1st and 2nd partial product rows are added in a unique way as shown in Fig. 7. This results in huge saving of gates. By using 3 bit adder a 17 bit adder is avoided. By avoiding complex structure and interconnections, eliminating a large number of unbalanced paths that cause substantial glitch generation and propagation the reduction in power has been achieved.

REFERENCES

1. Swee, K.L.S. and L.H. Hiung, 2012. Performance comparison review of 32 bit multiplier designs, 4th International Conf. on Intelligent & Advanced Systems (ICIAS), 2: 836-841.
2. Arunachalan, T. and S. Kirubaveni, 2013. Analysis of high speed multipliers, International Conf. on Communications and Signal Processing (ICCSP), pp: 211-214.
3. Choi, J., J. Jeon and K. Choi, 2000. Power minimization of functional units by partially guarded computation, Proc. IEEE Int. Symp. Low Power Electron. Des., pp: 131-136.
4. Chen, O., R. Sheen and S. Wang, 2002. A Low Power Adder operating on effective dynamic data ranges, IEEE trans. VLSI Syst., 10(4): 435-453.
5. Chen, O., S. Wang and Y.W. Wu, 2003. Minimization of switching activities of partial products for designing low power multipliers, IEEE trans. VLSI Syst., 11(3): 418-433.

6. Benini, L., G.D. Micheli, A. Macaii, E. Macaii, M. Poncino and R. Scarsi, 2000. GliticHing Power minimization by selective gate freezing, IEEE trans. VLSI Syst., 8(3): 287-297.
7. Henzler, S., G. Georgakos, J. Berthold and D. Schmidtt-Landsiedel, 2004. Fast-Power efficient circuit-block switch-off scheme, Electronic Lett., 40(2): 103-104.
8. Huang, Z. and M.D. Ercegovac, 2001. On signal gating schemes for low power adders, Proc. 35th Asilomar Conf. Signal, Syst., Comput., pp: 867-871.
9. Huang, Z. and M.D. Ercegovac, 2005. High performance low-power left-to-right array multiplier design, IEEE Trans. Comput., 54(3): 272-283.
10. Oklobdzija, V.G. and P.F. Stelling, 1997. Implementing multiply-accumulate operation in multiplication time, Proc. 13th IEEE Symp.on Comput. arithmetic, 1997.
11. Wen, M.C., S.J. Wang and Y.N. Lin, 2005. Low power parallel multiplier with column bypassing, Electron. Lett., 41(12): 581-583.
12. Zhang, X., F. Boussaid and A. Bermak, 2014. 32 bit × 32 bit Multiprecision Razor based Dynamic Voltage scaling Multiplier with operands Scheduler, IEEE trans. VLSI Syst., 22(4): 759-770.
13. Yano, K., T. Yamanaka, T. Nishida, M. Saito, K. Shimohigashi and A. Shimizu, 1990. A 3.8ns CMOS 16 × 16 b Multiplier using Complementary Pass-Transistor Logic, IEEE J. Solid State ckts., 25(2).
14. Abu-Khater, I.S., A. Bellaoar and M.I. Elmasry, 1996. "Circuit Techniques for CMOS low power high performance Multipliers, IEEE J. Solid State Ckts, 31(10).
15. Chen, K.H. and Y.S. Chu, 2007. A low power Multiplier with the spurious power suppression techniques", IEEE Trans. VLSI systems, 15(7).
16. Dasterji, M.M., A. A. Kusha and M. Pedram, 2009. Z-FAD: A low-power low-area Multiplier based on shift and add architecture, IEEE Trans. VLSI systems, 17(2).
17. Kuang, S.R., J.P. Wang and C.Y. Gua, 2009. Modified Booth Multipliers with a regular partial product array, IEEE Trans. Ckts. and Sys. - II, 56(5).
18. Gee, B.H., J.S. Chang, Y. Shi, C.C. Chua and K.S. Chong, 2009. A low voltage micro power asynchronous multiplier with shift-add multiplication approach, IEEE Trans. Ckts. and Sys. - I, 56(7).
19. Namin, A.H., H. Wu and M. Ahmadi, 2009. A high speed word level finite field Multiplier in F_2^m using redundant representation, IEEE Trans. VLSI Systems, 17(10).
20. Paul, B.C., S. Fujita and M. Okajima, 2009. ROM based logic (RBL) Design: A low power 16 bit multiplier, IEEE J. Solid State Ckts, 44(11).
21. Seo, Y.H. and D.W. Kim, 2010. A new VLSI architecture of parallel Multiplier accumulator based on radix-2 Modified Booth Algorithm, IEEE Trans. VLSI Systems, 18(2).
22. Moshanvaga, V.G. and K. Tamaru, 1995. A comparative study of switching activity reduction techniques for design of low power multipliers, Proc. IEEE Int. Symp. Ckts. Syst., Apr. pp: 1560-1563.
23. Angel, E. and E.E. Swatzlander, Jr., 1996. Low power parallel multipliers in Proc. IEEE Workshop VLSI Signal Processing, 1996, ppL 199 - 208.
24. Yu, Z., L. Wasserman and A. Willson, Jr., 2000. A painless way to reduce power dissipation by over 18% in Booth-encoded carry-save array multipliers for DSP, Proc. IEEE Workshop Signal Processing Syst., pp: 571-580.
25. Goldovsky, A., B. Patel, M. Schutle and R. Kolgotla, 2000. Design and implementation of a 16 by 16 bit low power two's complement multiplier, Proc. Int. Symp. Ckts and Syst., 5: 345-348.
26. Mahant- Shetti, S., P. Balasara and C. Lemonds, 1999. "High performance low power array multiplier using temporal tilting, IEEE Trans. VLSI Syst., 7: 121-124.
27. Lee, H., 2004. A power aware scalable pipelined booth multiplier, in Proc. IEEE Int. SOC Conf., pp: 123-126.
28. Ercegovac, M.D. and T. lang, 1992. On the fly Rounding, IEEE Trans. on Computers, 41(12).
29. Joshi, M., D.S. Chauhan and B.K. Kaushik, 2015. A New high speed full adder cell, World Applied Sciences Journal, 33(4): 599-603.
30. Kuang, S.R., J.P. Wang and C.Y. Guo, 2009. Modified Booth Multipliers with a regular partial product array, IEEE Trans. on Ckts. and Syst. - II, 56(5).
31. Yeh, W.C. and C.W. Jen, 2009. High speed booth encoded parallel multiplier design, IEEE Trans. on Ckts. and Syst. - II, 56(5).
32. Kang, J.Y. and J.L. Gaudiot, 2006. A simple high speed multiplier design, IEEE Trans. on Computers, 55(10): 1253-1258.
33. Joshi, M., D.S. Chauhan and B.K. Kaushik, 2014. A Novel Explicit Pulsed Dual Edge Triggered D Flip Flop, World Applied Sciences Journal, 31(10): 1837-1842, 2014.