Iranian Journal of Electrical and Electronic Engineering

Journal Homepage: ijeee.iust.ac.ir

Research Paper

Modified 32-Bit Shift-Add Multiplier Design for Low Power Application

R. Pinto*(C.A.)

Abstract: Multiplication is a basic operation in any signal processing application. Multiplication is the most important one among the four arithmetic operations like addition, subtraction, and division. Multipliers are usually hardware intensive, and the main parameters of concern are high speed, low cost, and less VLSI area. The propagation time and power consumption in the multiplier are always high. The multiplier speed usually determines the speed of the processor. Hence in this work, a design of a 32-bit multiplier is proposed by modifying the conventional shift-add multiplier. The proposed structure reduces the power consumed by the technique of minimizing the switching activities in the design. A 32-bit parallel prefix adder based on the modified Ling equation is also proposed to speed up the addition of the partial products in the multiplier. The design is modeled in VHDL and implementation is carried out in CADENCE software with 90 nm and 180 nm CMOS technology.

Keywords: Shift-Add Multiplier, Parallel Prefix Adder, Low-Power, VLSI Implementation.

1 Introduction

COMPUTERS usually spend a large amount of time in multiplication and division operations. The size of the hardware required for such operations is considerably more. Multipliers are an important part of today's signal processing applications because these applications need to support extensive number of arithmetic operations [1]. Multipliers play a pivotal role in the system performance and power consumption. Other than DSP application, multipliers are largely used in ALU and MAC unit [2]. Latest advancement in technology has led to many researchers working to design a fast, area, and power-efficient multiplier. The multiplier design always has a trade-off between speed and power. In this work, a power-efficient multiplier design is proposed by modifying the conventional multiplier. This structure reduces the power consumption because of the reduction of the switching activities in the multiplier design. The partial product addition is increased in the multiplier by developing a 32-bit parallel prefix adder. The adder structure is designed and developed to make it more efficient in terms of the gate count. The parallel prefix adder is designed using Ling adder which saves a logic element in each carry bit.

The paper is organized as follows. Section 2 gives a brief survey of the previous multipliers and adders. Conventional and modified shift-add multiplier is explained in detail in Section 3. Parallel prefix adder structure is presented in Section 4. Section 5 evaluates the performance of the multiplier and gives the comparison results. Finally, the paper is concluded in Section 6.

2 Previous Works

Many efficient multiplication algorithms and hardware designs are developed and presented in the literature. Shen and Chen [3] developed a 2's complement multipliers using Booth algorithm. For power reduction, this multiplier increased the probability of partial product becoming zero. Theoretical equations are derived to prove the reduction

Iranian Journal of Electrical and Electronic Engineering, 2020.
Paper first received 18 January 2020, revised 12 March 2020, and accepted 16 March 2020.
* The author is with the Department of Electronics and Communication Engineering, St. Joseph Engineering College, Vamanjoor, Mangalore, India.
E-mail: rohanp@sjec.ac.in.
Corresponding Author: R. Pinto.

of the switching activities in the multiplier. Chen *et al*. [4] designed a 16-bit 2's complement multiplier. The switching activities in the multiplier are reduced for power efficiency. Booth algorithm is used for partial product reduction. The power dissipation is further reduced by realizing the multipliers based on row-based and hybrid-based adder trees. Wang *et al*. [5] developed a fixed-width array-multiplier for low-power application. Left-to-right algorithm is used for partial product reduction. The proposed multiplier has high-speed features along with low power and area efficiency. Zhijun and Ercegovac [6] presented a high-performance power-efficient design of a linear array multiplier. The multiplier is a combination of three techniques. Signal flow optimization technique is used for partial product reduction. Left-to-right leapfrog and reduction array splitting techniques are used for better performance. To increase the speed and reduce the power consumption of the multiplier the authors Chen and Chu [7] have applied spurious power suppression technique (SPST) on the multiplier. The switching power in the multiplier is further reduced by designing the control signal asserting circuit. In the work proposed by Krad and Taie [8], the authors have compared the performance of a 32-bit multiplier which uses carry-look-ahead adder for partial product addition and a 32-bit multiplier which uses ripple carry adder for addition. Here, speed is considered as a factor for comparison. In terms of gate delay the multiplier with carry-look-ahead adder has outperformed the multiplier with ripple carry adder. A modified shift-add multiplier namely bypass zero feed A directly (BZ-FAD) is developed by Dastjerdi *et al*. [9]. The switching activities in the multiplier are reduced to minimize the dynamic power. The multiplier is used for power-efficient application where speed is not a matter of concern. For high-performance DSP application Kalaiyarasi and Saraswathi [10] proposed a high-speed Booth multiplier. This multiplier is fast and power-efficient. The multiplier reduces the partial products, in turn increasing the computation time. Radix-4 array multiplier and Booth multiplier is proposed for multiplication of both signed and unsigned numbers. Mohapatra *et al*. [11] presented a parallel multiplier for high-speed application. The multiplier is based on shift-add algorithm, Vedic design 1 algorithm and Vedic design 2 algorithm. This novel algorithm reduces the combinational path delay. The results of the modified shift-add multiplier and Vedic multiplier are then compared with the conventional shift-add and Vedic multipliers and found to be efficient. Sangeetha and Khan [12] proposed high speed, power-efficient multiplier using shift-add approach. In this paper, the author have presented the implementation of Braun and Wallace multiplier. These multipliers are simulated by building the schematic circuit of the logic gates. With these circuits the multiplier is developed. Area and power comparison is done to prove the efficiency of the proposed design. Shao and Li [13] developed a fixed-width multiplier of 16-bit and a squarer. To reduce the processing errors, a general model of array-based approximate arithmetic computing is built to guide the multiplier. A high-speed Booth multiplier is designed by He *et al*. [1]. To increase the speed and reduce the complexity of the circuit, dynamic error-compensated circuit is designed. This circuit can be implemented for 32-bit or 64-bit word length. A 4-bit multiplier is developed by Shabbir *et al*. [2] using Dadda algorithm. The multiplier is fast and power-efficient. An adder is also developed for addition in the multiplier. To reduce the glitches in the output, flip-flops are designed which are used in the pipeline registers.

Several designs of parallel prefix adder are presented in the literature. A hybrid prefix adder structure is developed by Efstathiou *et al.* [14] that combined the conventional carry and Ling carry [15]. Three different adder structures are developed based on the CLA and Ling equations. Dimitrakopoulos and Nikolos [16] modified the Ling equation and developed a novel parallel prefix adder based on the modified equation. The new adder structure is fast and efficient. The fan-out in the structure is minimum. Poornima and Kanchana [17] developed a hybrid parallel prefix adder structure which is fast and area-efficient. This hybrid structure is a combination of two structures proposed by Ladner and Fisher [18] and Kogge and Stone [19]. Most of the prior work done on the parallel prefix adders focused on the speed of the structure. Rather, less importance is given to the area occupied and power consumed by the adder structure. So, in this work a fast and area-efficient 32-bit parallel prefix Ling adder is proposed which can be used for the summation of the partial products in the design of modified shift-add multiplier.

3 Modified Shift-Add Multiplier

The main concern in the conventional shift-add multiplier is often realized by *K* cycles of shifting and adding. The process of shit-add multiplication involves partial product generation and partial product addition. During the process of multiplication around six switching activities are involved [9]. Consider if two numbers are multiplied, the switching activities taking place in the multiplier are shifting of the multiplier bits, activity in the adder, activity in the counter, shifting of the partial product bits, switching between zero and multiplicand in the multiplexer and activity in the multiplexer select line controlled by 0^{th} bit of the multiplier. All these switching activities are the reason for the major part of power dissipation. The aim of this work is to reduce a few switching activities to minimize the power. Hence, a modified shift-add multiplier is derived from the conventional shift-add multiplier [20] and compared with the other multiplier designs presented in the literature.

An area power-efficient multiplier structure is designed by modifying the conventional shift-add multiplier. The structure in Fig. 1 shows the proposed modified shift-add multiplier. Here in this structure few of the switching activities are reduced to increase the power efficiency. The shifting of the multiplier bits to right by one bit leads to switching activity. This activity of bit shifting can be eliminated by ANDing the multiplier bit by one. The controller selects the multiplier bit which is to be checked for zero or one. The switching activity of addition of zero when the multiplier bit is zero can be eliminated by inserting a feeder register to store the intermediate partial product. When the multiplier bit is one the partial product stored in the feeder register is summed up with the multiplicand and the 0^{th} bit of the result is stored in the product register. The remaining bits from 1 to *n*, where *n* is the total number of product register bits are shifted right by one bit position and the resultant is stored in the feeder register. Now, if the multiplier bit is zero, the contents of the feeder register and multiplicand are added and result is stored back in the feeder register with the most significant bit stored in the product register. In each cycle one bit of the product is finalized. The bit positions in the product register are selected by the controller. In the last cycle when the multiplier bit is checked and processed the final product lower half is stored in the product register and upper half is stored in the feeder register. Here in this process the switching activity of shifting of the entire partial product bits is eliminated.

4 Parallel Prefix Ling Adder

The performance of the multiplier can be improved by increasing the summation speed of the partial products.

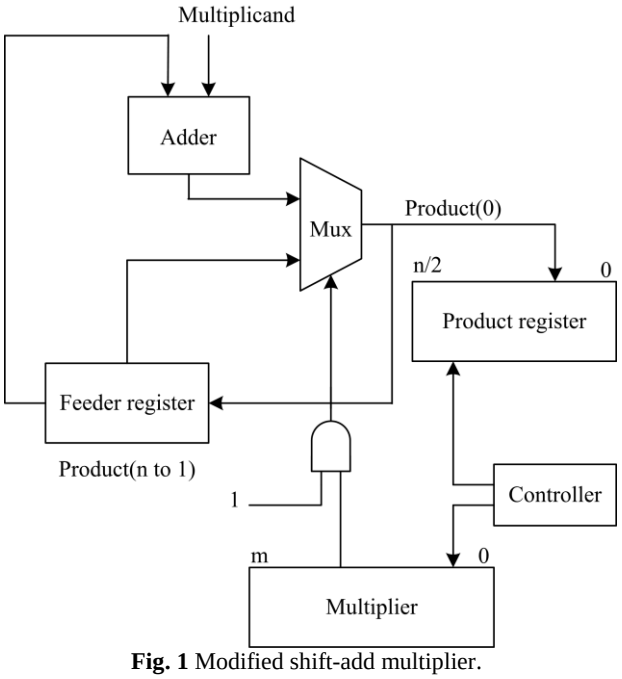

Fig. 1 Modified shift-add multiplier.

To intensify the summation process an efficient and fast parallel prefix adder is designed based on the concept of modified Ling equation for the 32-bit multiplier. This adder achieves a significant hardware saving.

Here in the Ling adder, pseudo carry (H_i) is calculated instead of the real carry (c_i) which is obtained in the conventional adder. The pseudo carry equation for the 5^{th} bit position is given as

$$H_5 = g_5 + g_4 + p_4.g_3 + p_4.p_3.g_2 + p_4.p_3.p_2.g_1 + p_4.p_3.p_2.g_0 \quad (1)$$

where g_i and p_i are generate and propagate bits.

In general, the pseudo carry equation is given as

$$H_i = g_i + g_{i-1} + p_{i-1}.g_{i-2} + p_{i-1}.p_{i-1}.g_{i-3} + \ldots + p_{i-1}.p_{i-2}\cdots p_1.g_0 \quad (2)$$

Even though the pseudo carry calculation reduces the area but the sum calculation using the pseudo carry is complicated. As a result, the real carry (c_i) is calculated from the pseudo carry without changing the area requirement to make the sum calculation easy, which is given as

$$S_i = d_i \; XOR \; c_{i-1} \quad (3)$$

In the proposed method the real carry is calculated for the lower half bits from 0 to 15 by ANDing the pseudo carry with the respective propagate bits, given by the equation

$$c_i = H_i . p_i \quad (4)$$

and for the higher half bits from 16 to 31 the carry is computed by rearranging the pseudo carry of the higher half bits. The pseudo carry is given in terms of intermediate generate bit and intermediate propagate bit. The final rearranged pseudo carry is ANDed with the respective propagate bit.

Consider for example the pseudo carry for the 5^{th} bit. G_f^* and P_f^* are the intermediate generate and intermediate propagate bits respectively and *f* is the bit location.

$$H_f = (g_f + g_{f-1}) + p_{f-1}.p_{f-2}.(g_{f-3} + g_{f-4}) + p_{f-1}.p_{f-2}.p_{f-3}.p_{f-4}.(g_{f-4} + g_{f-f}) \quad (5)$$

$$= G_{f:f-1} + P_{f-1:f-2}.G_{f-3:f-4} + P_{f-1:f-2}.P_{f-3:f-4}.G_{f-4:f-f} \quad (6)$$

Equation (6) can be represented by an associative operator $\odot$ which associates two pairs of inputs (g,p) and (g',p') as (g+$p.g'$,$p.p'$)

$$H_f = (G_{f:f-1}, P_{f-1:f-2}) \odot (G_{f-2:f-3}, P_{f-3:f-4}) \odot (G_{f-4:f-f}, P_{f-f:f-6}) \quad (7)$$

$$=\left(G_f^* ,P_{f-1}^*\right)\odot\left(G_{f-2}^*,P_{f-3}^*\right)\odot\left(G_{f-4}^*,P_{f-f}^*\right) \quad (8)$$

$$=\left(G_{f:f-1},P_{f-1:f-2}\right)\odot\left(G_{f-2:f-f},P_{f-3:f-6}\right) \quad (9)$$

$$=\left(\left(G_{f:f-1}+P_{f-1:f-2}.G_{f-2:f-f}\right),\left(P_{f-1:f-2}.P_{f-3:f-6}\right)\right) \quad (10)$$

Multiplying the first term of (10) with the propagate bit gives the real carry. In general, the carry for the higher half bits is given as

$$c_i=\left(G_{i:k}+P_{i-1:k-1}.G_{k-1:j+1}\right).p_i \quad (11)$$

The structure of the proposed 32-bit parallel prefix adder is shown in Fig. 2. The proposed adder is based on the modified Ling equation. This adder computes the real carry (c_i) from the pseudo carry (h_i). As a result, the final sum computation is a simple XOR operation. This method saves one logic element in the higher half carry position from 16 to 31. The proposed grey circle saves one logic element each saving a considerable amount of area in the adder cell.

The white square node [16] in Fig. 2 computes generate bit (g_i), propagate bit (p_i) and half sum bit (d_i) bits given by

$$g_i=x_i\ AND\ y_i \quad (12)$$

$$p_i=x_i\ OR\ y_i \quad (13)$$

$$d_i=x_i\ XOR\ y_i \quad (14)$$

Intermediate generate and intermediate propagate bits are computed by the black square node [16] in Fig. 2 given in (15) and (16), respectively.

$$G_i^*=g_i+g_{i+1} \quad (15)$$

$$p_i^*=p_i.p_{i-1} \quad (16)$$

Group generate (G) and group propagate (P) bits are calculated by the black circle node [16] in Fig. 2 and the grey circle node in the figure computes only the group generate bits given by the following equations

$$G_{i:j}=G_{i:k}+G_{m:j}.P_{i:k} \quad (17)$$

$$P_{i:j}=P_{i:k}.P_{m:j} \quad (18)$$

The white hexagon node in Fig. 3 computes the lower half-real carry as given in (4). It is a simple AND operation of pseudo carry and propagate bit. The black hexagon node in Fig. 3 computes the real carry for the upper half as given in (11). The proposed structure saves 30 logic elements when compared with the conventional parallel prefix adder.

The advantages of the proposed parallel prefix adder structure are, it has a simple and regular structure for any bit structure. The number of gates required to build the structure is reduced as compared to the conventional parallel prefix adder. It has a standard logic depth of $\log_2 n$, where n is the number of bits. The fan-out of the structure is four, due to which the propagation delay and loading of the gates is reduced.

6 Results and Discussion

The entire multiplier design was modeled in VHDL. This modeled system was implemented using 90 nm and 180 nm CMOS technology. Cadence NC-Simulator software was used for formal verification. 10,000 random test vectors were generated in MATLAB and saved in a text file. This text file was read as a test bench file in Cadence tool and fed as an input to the multiplier. The multiplier product correctness of the design was confirmed. Cadence RTL Compiler tool was used for synthesis and thereby area and power results were noted. Cadence Encounter tool was used to obtain the layout by feeding the tool with design constrains and derived netlist file obtained from the compiler tool. The layout was checked for DRC, placed and routed and finally register capacitor (RC) parasitic information was drawn. The proposed multiplier simulation was observed after mapping onto the Cadence tool with 90 nm technology file. Fig. 3 shows the multiplier output window in Cadence Simvision tool.

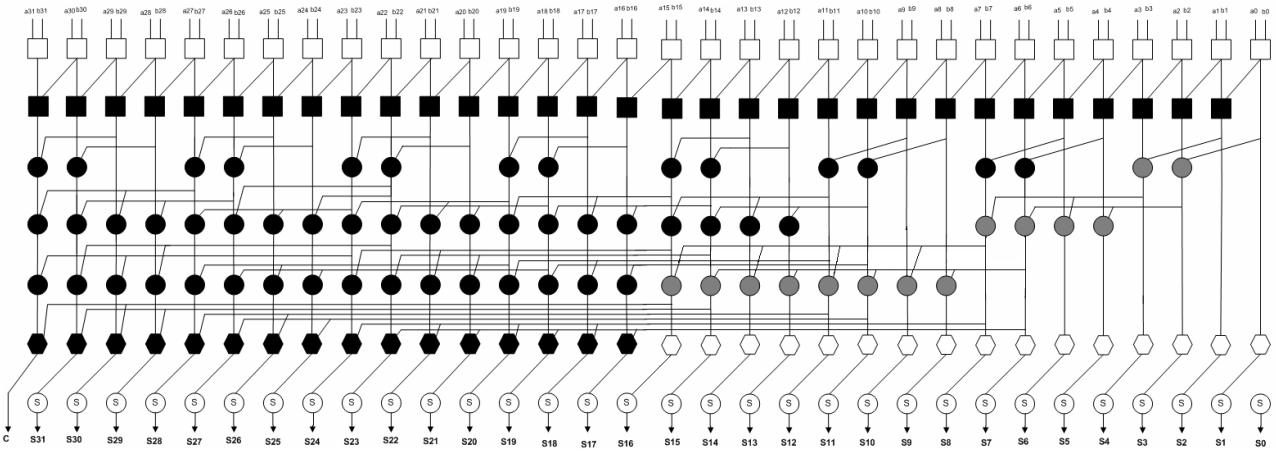

Fig. 2 Proposed 32-bit parallel prefix ling adder.

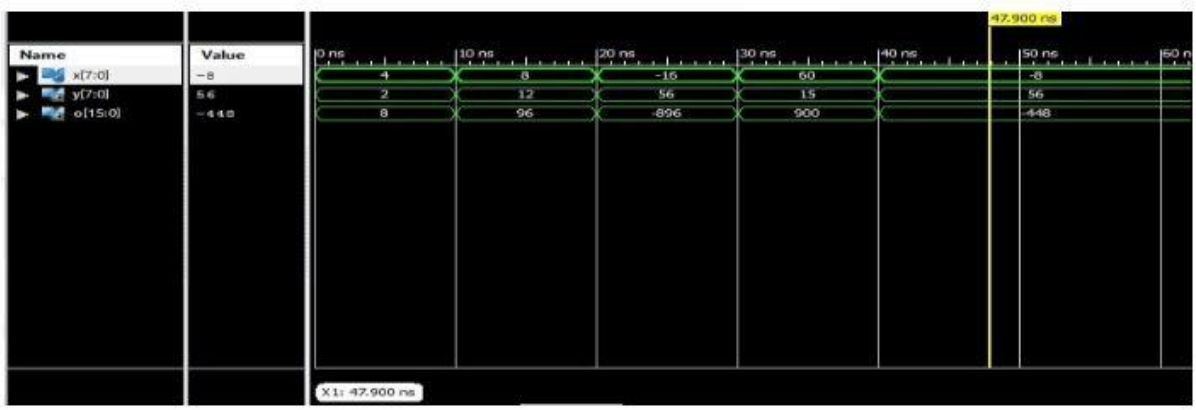

Fig. 3 Simulation results.

Table 1 Power consumption of 32-bit multipliers.

Technology	Conventional multiplier [20]	Proposed multiplier
90 nm	7.15 mW	5.5 mW
180 nm	49.5 mW	47.1 mW

Table 2 Comparison of area and power of various 32-bit multipliers (90 nm technology).

Structure	Power [mW]	Area [μm^2]
Proposed Structure	5.5	63,958
Basiri [21]	5.8	2,50,000
Mulkalapally [22]	26.6	50,407
Wang [23]	12.1	85,983
Själander [24]	7.3	99,000

In Table 1 the proposed multiplier power consumption result is compared with the conventional shift-add multiplier of 32-bit. The conventional multiplier uses carry look-ahead adder (CLA) for partial product addition. From the table, it can be observed that the proposed structure is power efficient. This improvement can largely be attributed to the design of the multiplier in reducing the switching activities in the multiplier.

The proposed structure is also compared with other 32-bit multipliers implemented in 90 nm and 180 nm CMOS technology. The comparison results in Table 2 shows that the proposed multiplier implemented in 90 nm is area as well as power efficient. The use of 32-bit parallel prefix adder for partial product addition in the multiplier has led to the reduction of the gate count. As a result, the area is reduced compared to all the other multiplier structures. Few of the switching activities in the proposed multiplier are eliminated as a result the dynamic power consumption has reduced. The Wallace tree multiplier developed by Basiri *et al.* [21] is a power-efficient structure but the area required is very high. The Vedic multiplier proposed by Mulkalapally [22] has the minimum area among all the multiplier designs but this is at the cost of high power consumption. The multipliers proposed by Wang *et al.* [23] and Själander *et al.* [24] have a high area and power compared to the proposed modified shift-add multiplier. The LR and RL structure of Wang *et al.* [5] has a high area implemented in 180 nm technology given in Table 3.

Table 3 Comparison of area and power of various 32-bit multipliers (180 nm technology).

Structure	Power [mW]	Area [μm^2]
Proposed Structure	47.1	2,50,285
Mulkalapally [22]	161.9	1,93,489
Wang [5] (LR structure)	79.8	10,30,225
Wang [5] (RL structure)	93.3	11,83,308
Zhijun [6]	40.65	74,598

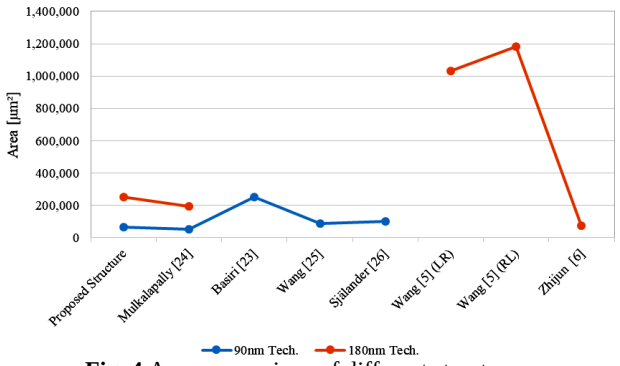

Fig. 4 Area comparison of different structures.

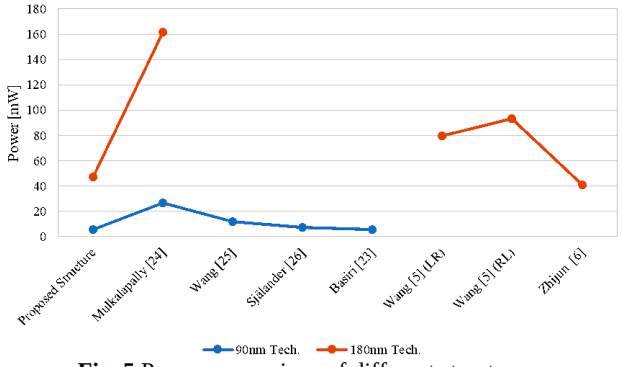

Fig. 5 Power comparison of different structures.

The three parameters of the circuit namely area, delay, and power of the 32-bit multiplier implemented in 90 nm and 180 nm technologies are measured and delay is found to be 40.4 ns and 77.1 ns. The plot showing the area and power comparison of the proposed work with

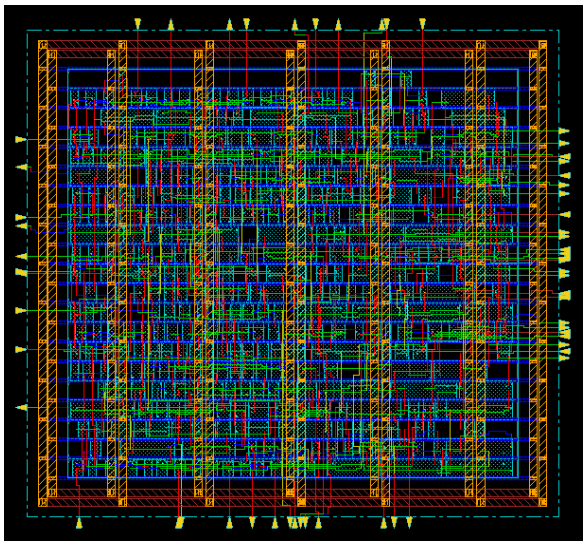

Fig. 6 Physical layout of the proposed 32-bit modified shift-add multiplier.

the other work is shown in Figs. 4 and 5, respectively.

The proposed structure is fast and area-power efficient. The computation of pseudo carry from the real carry has saved one logic element from each carry path. This area saving is not possible if conventional carry is computed. Long interconnects in the carry path are avoided to form a simple structure and minimize the complexity in the adder structure. The physical layout of the proposed multiplier is shown in Fig. 6. The layout is made up of a single polysilicon layer (Poly1) and six different metal layers from (M1 to M6). Static timing analysis and power analysis is done after routing for constant Process-Voltage-Temperature (PVT) variation. The analysis shows positive worst negative slack (WNS) and zero total negative slack (TNS).

7 Conclusion

In this work, an area-power efficient modified shift-add multiplier for 32-bit is proposed. The speed in the structure is compromised to gain a high power and area reduction. The power dissipation results from the table show that a minimum of 40% power saving can be gained for the proposed 32-bit multiplier. The proposed multiplier utilizes parallel prefix adder for summation of the partial products. Here a 32-bit adder design is proposed based on the modified Ling equation. This adder has a reduced gate count leading to considerable area saving. It has a regular structure and simple interconnection. The proposed multiplier is area and power efficient which finds its advantage in any digital signal processor.

References

[1] W. Q. He, Y. H. Chen, and S. J. Jou, "Dynamic error compensated fixed-width Booth multiplier based on conditional-probability of input series," *Circuits, Systems, and Signal Processing*, Vol. 35, No. 8, pp. 2972–2991, Oct. 2016.

[2] Z. Shabbir, A. R. Ghumman, and S. M. Chaudhry, "A reduced-sp-D3L$_{Sum}$ adder based high frequency 4×4 Bit multiplier using Dadda algorithm," *Circuits, Systems, and Signal Processing*, Vol. 35, No. 9, pp. 3113–3134, Nov. 2016.

[3] N. Y. Shen and O. C. Chen, "Low-power multipliers by minimizing switching activities of partial products," In *IEEE International Symposium on Circuits and Systems (Cat. No. 02CH37353)*, Vol. 4, pp. 93–96, 2002.

[4] O. T. C. Chen, S. Wang, and Y. W. Wu, "Minimization of switching activities of partial products for designing low-power multipliers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 11, No. 3, pp. 418–433, 2003.

[5] J. S. Wang, C. N. Kuo, and T. H. Yang, "Low-power fixed-width array multipliers," in *Proceedings of International Symposium on Low Power Electronics and Design*, pp. 307–312, 2004.

[6] Z. Huang and M. D. Ercegovac, "High-performance low-power left-to-right array multiplier design," *IEEE Transactions on Computers*, Vol. 54, No. 3, pp. 272–283, Mar. 2005.

[7] K. H. Chen and Y. S. Chu, "A low-power multiplier with the spurious power suppression technique," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 15, No. 7, pp. 846–850, Jul. 2007.

[8] H. Krad and A. Y. Al-Taie, "Performance analysis of a 32-Bit multiplier with a carry look ahead adder and a 32-bit multiplier with a ripple adder using VHDL," *Journal of Computer Science*, Vol. 4, No. 4, pp. 305–308, 2008.

[9] M. Mottaghi-Dastjerdi, A. Afzali-Kusha, and M. Pedram, "BZ-FAD: A low-power low-area multiplier based on shift-and-add architecture," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 17, No. 2, pp. 302–306, Feb. 2009.

[10] D. Kalaiyarasi and M. Saraswathi, "Design of an efficient high speed radix-4 Booth multiplier for both signed and unsigned numbers," in *4th International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*, Chennai, India, pp. 1–6, Oct. 2018.

[11] A. Mohapatra, A. Bisoyi, and A. Tripathy, "Design of novel multipliers-Vedic and shift-add for IEEE 754-2008 single precision floating-point unit in high speed applications," in *IEEE International Symposium on Smart Electronic Systems (iSES)(Formerly iNiS)*, Rourkela, India, pp. 160–163, Feb. 2020.

[12] P. Sangeetha and A. Ali Khan, "Modified PEB formulation for hardware efficient fixed-width Booth multiplier," in *4th International Conference on Devices, Circuits and Systems (ICDCS)*, Coimbatore, India, Mar. 2018.

[13] B. Shao and P. Li, "Array-based approximate arithmetic computing: A general model and applications to multiplier and squarer design," *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol. 62, pp. 1081–1090, Apr. 2015.

[14] C. Efstathiou, H. T. Vergos, and D. Nikolos, "Ling adders in standard CMOS technologies," in *9th International Conference on Electronics, Circuits and Systems*, pp. 485–48, 2002.

[15] H. Ling, "High-speed binary adder," *IBM Journal of Research and Development*, Vol. 25, No. 3, pp. 156–166, 1981.

[16] D. Giorgos and D. Nikolos, "High-speed parallel-prefix VLSI ling adders," *IEEE Transactions on Computers*, Vol. 54, No. 2, pp. 225–231, Feb. 2005.

[17] N. Poornima and V. S. Kanchana Bhaaskaran, "Area efficient hybrid parallel prefix adders," *Procedia Materials Science*, Vol. 10, pp. 371–380, 2015.

[18] R. E. Ladner and M. J. Fisher, "Parallel prefix computation," *Journal of the ACM (JACM)*, Vol. 27, No. 4, pp. 831–838, 1980.

[19] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Transactions on Computers*, Vol. 22, No. 8, pp. 786–793, 1973.

[20] B. Parhami, *Computer arithmetic*. Oxford University Press, New York, 2000.

[21] M. M. A. Basiri, S. C. Nayak, and N. M. Sk, "Multiplication acceleration through quarter precision Wallace tree multiplier," in *International Conference on Signal Processing and Integrated Networks (SPIN)*, pp. 502–505, 2014.

[22] M. Mulkalapally, J. Manning, P. Gatewood, and T. Nikoubin, "High speed, area and power efficient 32-bit Vedic multipliers," in *Proceedings of the 7th International Conference on Computing Communication and Networking Technologies*, ACM New York, NY, USA, pp. 1–7, 2016.

[23] L. R. Wang, M. H. Tu, S. J. Jou, and C. L. Lee, "Well-structured modified Booth multiplier and its application to reconfigurable MAC design," *IEICE Transactions on Electronics*, Vol. 94, No. 6, pp. 1112–1119, Dec. 2011.

[24] Magnus Själander and Per Larsson-Edefors, "Multiplication acceleration through twin precision," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 17, No. 9, pp. 1233–1246, 2009.

R. Pinto did his B.E. from Vivesvaraya Technological University, Belgaum, India in 2004, M.Tech from Vivesvaraya Technological University, Belgaum, India in 2006 and Ph.D. from Manipal Academy of Higher Education, Manipal, India in 2019. He has 8 years of teaching experience and currently working as an Associate Professor in the Department of Electronics and Communication Engineering at Saint Joseph Engineering College, Mangaluru, India. His areas of research interest are signal processing, digital system design, and VLSI architecture design.