Circuits, Systems, and Signal Processing (2021) 40:4407–4427
https://doi.org/10.1007/s00034-021-01671-3

Design of Low-Power Wallace Tree Multiplier Architecture Using Modular Approach

Vaibhavi Solanki[1] · A. D. Darji[1] · Harikrishna Singapuri[1]

Received: 26 December 2019 / Revised: 7 January 2021 / Accepted: 30 January 2021 /
Published online: 18 March 2021

Abstract

With the advancement in technology, various designs of multipliers offering low power consumption, high speed and less area have been proposed by many researchers. The main concern of electronic system designers is the energy minimization with the minimum penalty in speed and area for designing portable devices. Recent development focuses on the design of low-power multiplier for applications like biomedical signal processing requiring the least power consumption and delay-tolerant multiplier. This paper proposes a power-efficient design of the Wallace tree multiplier using a power-efficient 7:3 counter consisting of multiplexer and ex-or gates. The maximum power of the multiplier is consumed in the partial product tree reduction, and hence, in the proposed counter-based modular Wallace tree (CBMW) multiplier partial products are reduced using sequential 7:3 counter and the multi-bit addition in a single column reduces the complexity of the multiplier due to improvement in the locality. These proposed changes make design low power and scalable. The hardware utilization is minimum when a single 7:3 counter is used in partial product tree reduction per stage. The proposed multiplier is implemented in the Xilinx ISE design suite 14.7 using Verilog language on Spartan 3E FPGA. The design is also synthesized in Synopsys Design Compiler using 180 nm CMOS technology cell library. The corner analysis of the proposed design is performed in Synopsys PrimeTime, and the design meets all timing specifications. Also, the multiply and accumulate (MAC) unit is designed using the proposed multiplier to demonstrate an application of it. The detailed comparison is

This work performed under Special Manpower Development Program for Chips to System Design (SMDP-C2SD) Project.

✉ Harikrishna Singapuri
hari4_krishna@yahoo.co.in

Vaibhavi Solanki
solanki.vaibhavi1994@gmail.com

A. D. Darji
add@eced.svnit.ac.in

[1] Electronics Engineering Department, Sardar Vallabhbhai National Institute of Technology, Surat 395 007, India

performed for 8-bit as well as 16-bit operands, and it shows that the CBMW multiplier gives better delay performance and consumes the least power compared to existing multipliers. It is proved to be an efficient multiplier in terms of Power-Delay Product (PDP). The power consumption of the proposed 16×16 multiplier is 88.16 mW and 87.7 μW for Spartan 3E FPGA and ASIC design for 180 nm technology, respectively. It provides a promising performance for FPGA as well as for the ASIC platform.

Keywords FPGA · Higher-order counter · Low-power multiplier · MAC unit · Power delay product · Wallace tree multiplier

1 Introduction

Nowadays, the demand for low-power portable devices is increased which leads to the requirement of developing low-power and accurate system. In such applications, power versus speed trade-off is of great concern affecting the complexity of design in order to achieve a long battery life, reasonable weight and low packaging cost. The performance of digital computing heavily depends on the design of multipliers which is an essential hardware block used in multiply and accumulate (MAC) to perform various real-time applications such as computer vision, microprocessors, FIR filters, image processing and digital signal processing. Almost every real-time application uses an integer multiplier [12]. These multipliers not only consume a significant amount of power and area, but they are slowest elements compared to other arithmetic computing components. Many research papers have been published to optimize the design of an integer multiplier for improving the performance of the overall system. The efficiency of an integer multiplier is enhanced by reducing the partial products and method to add them. Existing integer multipliers provide a wide range of trade-off in terms of power, speed and complexity.

Among a large number of integer multipliers proposed in the literature, tree-based multipliers were proven to be one of the fastest and also had an efficient power-delay product [12]. The three main stages in the operation of a tree-based multiplier are: generation of partial product, tree reduction and final addition. N^2 AND gates are required for generating partial product tree of the N row in N-bit multiplier. The next operation is the reduction of the tree which uses half adders and full adders to add columns in a parallel fashion, and this tree of partial products is reduced until only two rows are left. The last step includes the addition of two rows using any fast adder to produce the final product. A tree-based multiplier was first proposed by Wallace [26] achieves high speed by reducing the number of sequential stages required for addition.

One of the drawbacks of Wallace multiplier is that the number of half adders used in the initial stage of partial product tree reduction does not reduce the number of partial products. Further, the complexity of the Wallace multiplier can be reduced by minimizing the number of half adders. Various researches have been carried out for improving the performance of the Wallace tree multiplier. The architecture discussed in [27] is similar to the technique of Wallace tree multiplier named as reduced complexity Wallace (RCW) multiplier. In RCW multiplier, the group of two bits is not processed at the initial stage, but passed to the next stage and processed at the last stage. RCW

has the same delay and slight reduction in power delay product (PDP) compared to the traditional Wallace tree multiplier. The number of 1's in each column is computed using ripple carry adder (RCA) and full adders. It achieves twice the speed of the earlier scheme but with increased complexity [23].

To further reduce the delay in the partial product tree reduction, different architectures using high-speed counters were proposed. In counter-based Wallace (CBW) tree multipliers, the partial product tree is readjusted in a reverse pyramid form. Afterward, the partial product tree reduction is carried out using 4:3, 5:3, 6:3 and 7:3 counters or compressors along with full adders and half adders. In single column, more than three bits are added at a time using higher-order counters, which results in the reduction of delay. The generation and propagation of intermediate carry increase the complexity and speed but at the same time degrade the power performance of the multipliers. Compared to the traditional Wallace tree multiplier, it is possible to reduce the partial product tree in fewer stages due to the use of high-speed counters.

The CBW multiplier consumes a significant amount of power. Taking this constraint into consideration, low-power Wallace tree multiplier design with lower delay is of great interest. There are various designs that utilize the fact that the optimized adder design can improve the multiplier's performance. The design discussed in [25] uses a multiplexer based 5:2 and 4:2 compressors to perform addition and achieves a reduction in latency and power consumption. A full-custom design of 4:2 and 5:2 compressors is discussed in [15]. It achieves low power consumption and low latency because of the pass transistor logic is used in the design. But this design fails to operate at an ultra-low voltage and unable to drive the next state due to low driving capability. A similar design of 4:2 and 5:2 compressor implemented by CMOS logic is described in [5]. The design is capable of performing at a low voltage of 0.6V and overcomes the drawback of a weak internal node of the pass transistor but requires more number of transistors due to the complementary CMOS structure. Delay-efficient CBW multiplier architectures are proposed in [4,6,20]. In [4], a generic strategy proposed to design Wallace tree multiplier using a multi-bit counter. The generation and propagation of the carry bit in an intermediate stage increase the complexity of the design and degrade the speed–power performance of this CBW multipliers. In [6], a new counter design is proposed which uses a stacking circuit and combines the smaller stack to form larger stacks. It achieves high speed and also consumes less power. In [20], the design of the 7:3 counter is modified such that it achieves a reduction in spurious transition. The hardware is optimized and balanced for all input and output and eliminates the redundant carry generators. In [14], a carry-save adder is replaced by a modified carry-save adder and that adder is implemented using a multiplexer. In this design, the speed depends on the delay of NOT gate and two multiplexers. In the literature [8] and [19], the modified design of adder based on multiplexers has been explained to reduce the power consumption in adders as well as delay. Various techniques [2, 7, 16, 18, 21] like Booth encoding are combined with the Wallace tree multiplier in order to increase speed and to reduce the power and area. The further improvement in performance can be obtained by truncating the operation [1, 10, 11, 17, 22]. Different architectures are proposed using counters [3].

Using different counters at a different stage, the power, delay and area of CBW multiplier can be optimized. In this paper, a modular approach of the Wallace tree

multiplier is proposed in which the main concern is to achieve a power-efficient multiplier. The circuit is designed to reduce power dissipation by applying the partial products in each column to the single 7:3 counter. The complexity of the multiplier is reduced due to a multi-bit addition in a single column and thus power efficiency is achieved. The reduction of the partial product tree is performed until two rows are left and they can be added using any fast adder.

The next section describes the design and architecture of 16 × 16-bit conventional Wallace tree multiplier, RCW multiplier, CBW multiplier and proposed counter-based modular Wallace (CBMW) tree multiplier along with the design of a 7:3 counter using a conventional full adder module and 7:3 counter using a multiplexer and ex-or gates based adder module. Section 3 includes the simulations and results comparison of different 7:3 counters and the proposed multiplier architecture with other existing architectures. It also includes the PVT analysis of the proposed design for different corner cases, and Sect. 4 describes the implementation of MAC incorporated with different multipliers. Finally, the concluding remarks are discussed in Sect. 5.

2 Architecture of Various Wallace Tree Multiplier

2.1 Conventional Wallace Tree Multiplier

In the conventional Wallace tree multiplier [26], the partial product tree of N^2 bits is formed and adjacent rows are grouped into either three-row group or two-row group. The full adder or half adder is applied according to the number of bits in the group. And if there is only one bit, then it is not processed at the present stage but passed to the next stage. This reduction process is repeated until the number of rows is reduced to two. Any carry-propagating adder is used for the addition of the remaining two rows to generate the product. Figure 1 represents 16 × 16 Wallace tree multiplier in which a single-bit partial product is represented by a color-filled circle. Different color-filled circles are used to represent different stages. The half adder is represented by a green rectangle and the full adder by an orange rectangle. The reduction process starts from the rightmost column. The reduction is carried out in seven stages, and the last stage will need a 23-bit carry-propagating adder represented by purple rectangle to produce the final multiplication product. A large number of full adders are needed and lead to complex interconnects. The delay of the conventional Wallace tree multiplier is proportional to the logarithm of multiplier operand's word-length.

2.2 Reduced Complexity Wallace (RCW) Tree Multiplier

In the RCW multiplier [27], the first phase is inverted pyramid array generation obtained by shifting the partial product array's left half in an upward direction. The method used for the generation of partial products tree is the same as in a Wallace tree multiplier only the vertical positions of partial products are shifted. The next stage of RCW is similar to a Wallace approach in which it uses the full adders as much as possible. The difference between the RCW and conventional Wallace approach is that

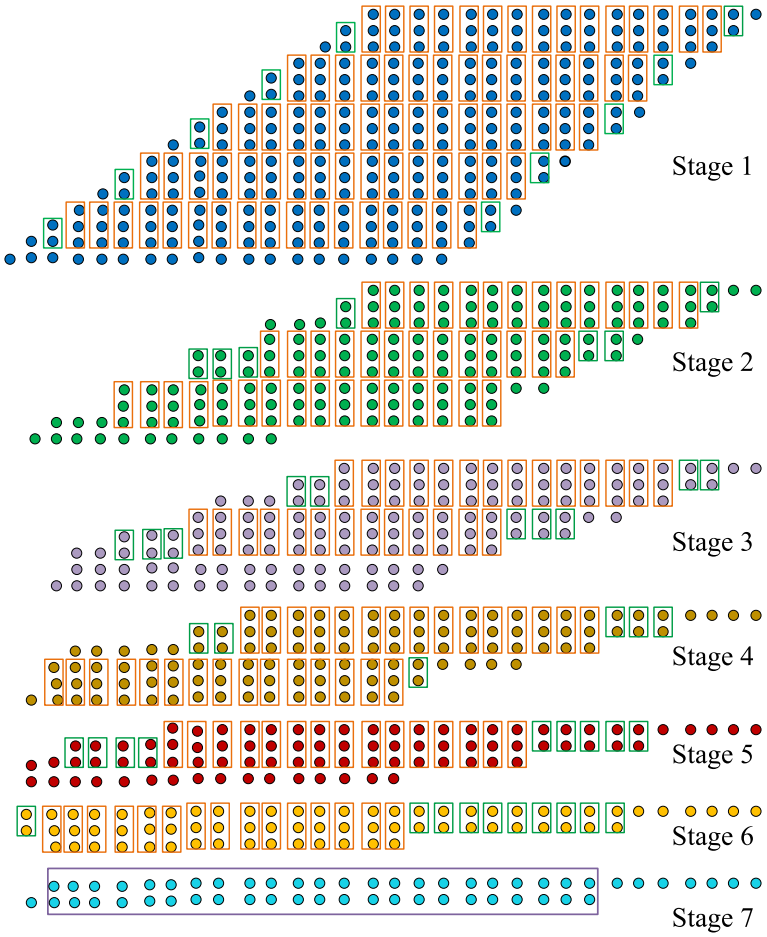

Fig. 1 Architecture of conventional Wallace tree multiplier [26]

RCW uses half adder only by ensuring the number of reduction stages is the same as the conventional Wallace tree multiplier. In this, single bits are passed to the next stage similar to the Wallace tree multiplier. The architecture of the 16 × 16 RCW multiplier [2] is shown in Fig. 2. In 16 × 16-bit multiplier, the 7 stages are required for the partial product tree reduction and the last stage requires 23-bit carry-propagating adder to produce the final result of the multiplier. The RCW has the same delay as the Wallace tree multiplier with insignificant increment in the number of full adders with the reduction in the number of half adders. The RCW multiplier achieves a reduction in PDP compared to the Wallace tree multiplier.

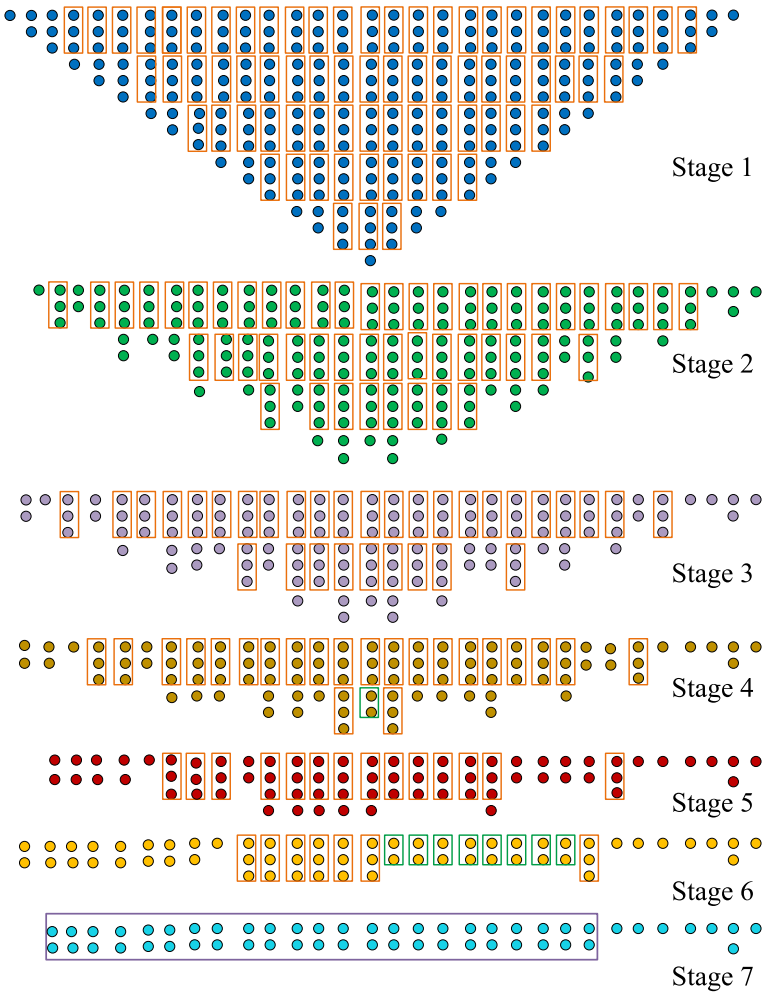

Fig. 2 Architecture of RCW multiplier [2]

2.3 Counter-Based Wallace (CBW) Tree Multiplier [20]

An elegant way to reduce the complexity of partial product tree reduction in the multiplier is to add multi-bit in a single column using higher-order counter or compressors. In the CBW multiplier, each column uses the counter according to the number of rows. For example, if a particular column has 8 rows, then it will use 7:3 counters and the remaining row is passed to the next stage. The result of the counter is produced in terms of sum, carry_1 and carry_2, whereas carry_1 is passed to the next column and carry_2 to the next-to-next column. This reduction process is carried out in a repetitive manner until the final result of the multiplication is obtained. Figure 3 represents the architecture of CBW multiplier. In Fig. 3, a half adder is represented by a green rectangle, full adder by an orange rectangle, 4:3 counter by a pink rectangle, 5:3 counter

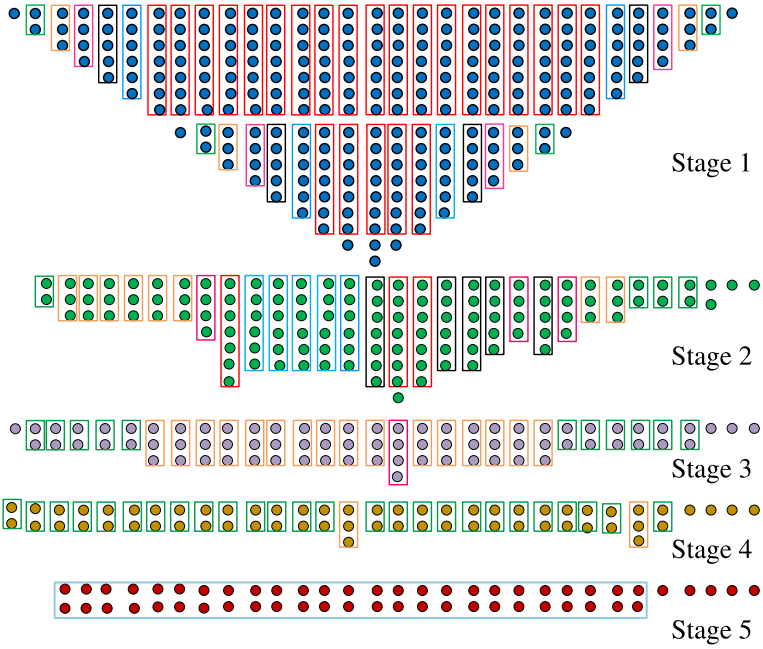

Fig. 3 Architecture of CBW multiplier [20]

by a black rectangle, 6:3 counter by a sky blue rectangle, 7:3 counter by a red rectangle and last carry-propagating adder by a light blue rectangle. The reduction of the partial product tree to two rows is done in four stages. The number of stages in the CBW multiplier is less compared to the conventional Wallace tree multiplier as it uses higher-order counters. Simultaneously, it has more power consumption and less delay due to improved locality.

2.4 Proposed Counter Based Modular Wallace (CBMW) Tree Multiplier

Usually in multiplier design, the parallelism increases the amounts of shifts between partial products and the addition of intermediate sum results into better speed, higher silicon area due to irregular structure and also higher power consumption due to the complex routing of interconnects. Serial multipliers achieve better area and power performance at the expense of speed. So, the selection of the serial or parallel multiplier depends on the nature of the application.

The main objective of the proposed CBMW architecture is to achieve a power-efficient multiplier with less delay. CBMW multiplier utilizes a single 7:3 counter per stage of partial product reduction as inputs are applied serially. Since only one 7:3 counter is used per stage the hardware utilization as well as power is minimized. The modular architecture enables to construct the multiplier of any required size. In 7:3 counter, the circuit takes equal weight 7 bits as input and produces a 3-bit output equal

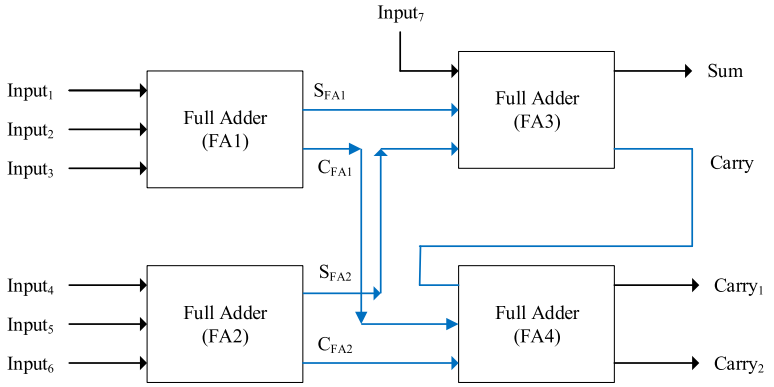

Fig. 4 Architecture of 7:3 counter using 4 conventional full adders

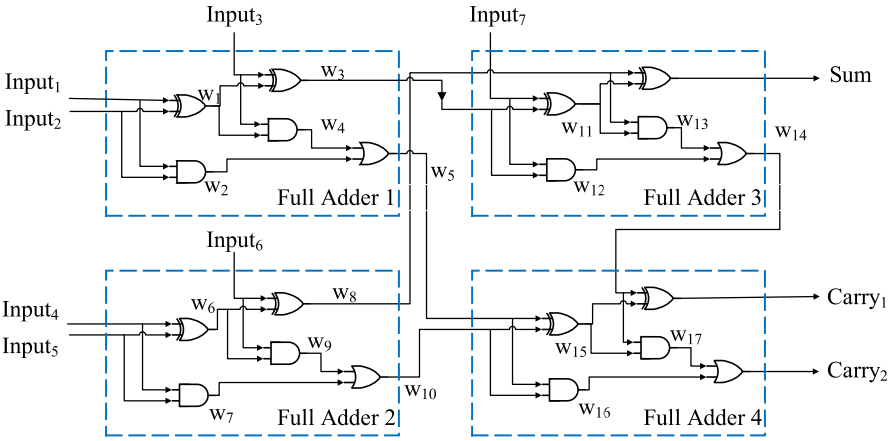

Fig. 5 7:3 counter using ex-or and AND gate-based adder

to the number of ones. The design of the 7:3 counter is widely studied in the past, and a large number of different architectures were proposed.

7:3 counter circuit is constructed using four conventional full adders as shown in Fig. 4. Input_1 to Input_7 represent seven input bits that are partial product terms and Sum, Carry_1 and Carry_2 are output bits. The S_{FA1}, S_{FA2}, C_{FA1}, C_{FA2} and Carry (or w_{14}) are the intermediate sum and carry bits. The first three partial product bits, i.e., Input_1, Input_2 and Input_3 are applied to FA1, whereas next three partial product bits Input_4, Input_5 and Input_6 are applied to FA2 and Input_7 is applied to FA3 along with the sum of FA1 and FA2, i.e., S_{FA1} and S_{FA2} and in the same way carry bits of FA1, FA2 and FA3, i.e., C_{FA1}, C_{FA2} and Carry to the FA4. The sum is provided by the FA3 and two carry bits, i.e., Carry_1 and Carry_2 are provided by the FA4. The counters greater than 7:3 are not used as it increases the complexity, delay as well as power consumption. Figure 5 shows the detailed architecture of the 7:3 counter using conventional full adder. The Sum, Carry_1 and Carry_2 of 7:3 counter using ex-or and AND gate based adder are given by (1), (2) and (3), respectively. The *Sum* is obtained

by ex-oring w_{11} and w_8, $Carry_1$ by ex-oring w_{14} and w_{15} and $Carry_2$ by oring w_{16} and w_{17}.

$$\text{Sum} = w_8 \oplus w_{11} \tag{1}$$
$$\text{Carry}_1 = w_{14} \oplus w_{15} \tag{2}$$
$$\text{Carry}_2 = w_{16} + w_{17} \tag{3}$$

$$\begin{aligned} where, w_1 &= \text{Input}_1 \oplus \text{Input}_2, w_2 = \text{Input}_1 \cdot \text{Input}_2, w_3 = \text{Input}_3 \oplus w_1, \\ w_4 &= \text{Input}_3 \cdot w_1, w_5 = w_2 + w_4, w_6 = \text{Input}_4 \oplus \text{Input}_5, \\ w_7 &= \text{Input}_4 \cdot \text{Input}_5, w_8 = \text{Input}_6 \oplus w_6, w_9 = \text{Input}_6 \cdot w_8, w_{10} = w_7 + w_9, \\ w_{11} &= \text{Input}_7 \oplus w_3, w_{12} = \text{Input}_7 \cdot w_3, w_{13} = w_8 \cdot w_{11}, w_{14} = w_{12} + w_{13}, \\ w_{15} &= w_5 \oplus w_{10}, w_{16} = w_5 \cdot w_{10}, w_{17} = w_{14} \cdot w_{15} \end{aligned}$$

The bottleneck of counter design using conventional full adder is high power consumption and delay as it consists of a chain of ex-or gates on the critical paths. Hence, the conventional counters are not suitable for low-power applications. The design of the 7:3 counter proposed in [13] implemented using simple logic gates has less delay and power consumption compared to conventional 7:3 counter but still occupies a large silicon area, large delay and also consumes a significant amount of power. Multiplexers are employed in counter-based designs to reduce the number of ex-or gates used on the critical path. To reduce the delay and power consumption, the full adder implemented using 4:1 multiplexers is discussed in [1]. It achieves a reduction in power but the further reduction in delay is achieved by reducing the critical path delay in which full adder was implemented using six 2:1 multiplexer [9]. But this approach achieves a reduction in power at the expense of an increase in area. 7:3 counter implemented in [24] also uses multiplexer in the design of full adder but has more critical path delay. Another stack-based 7:3 counter proposed in [6] achieves high speed but when used in the multiplier consumes a significant amount of power and delay. The 7:3 counter proposed in [20] achieves a reduction in delay as redundant carry generators are eliminated but at the same time have more leakage power and power dissipation.

7:3 counter using 2:1 multiplexer and ex-or gate as proposed in [8] is the most power and delay efficient design. One input and its complemented form are applied to the multiplexer as inputs. The other two inputs are applied to the ex-or gate and the output of that ex-or gate is applied to the select line of both multiplexers. This results in reduced switching activity and hence achieves a reduction in power as well as increase the speed. Hence, it is used in the proposed CBMW multiplier. Figure 6 shows the architecture of the 7:3 counter using adder made of multiplexers and ex-or gates. It consists of 8 multiplexers and four ex-or gates which form four adders shown by a blue dotted line. In Fig. 6, the selection line of each Mux pair Mux 1 and Mux 2, Mux 3 and Mux 4, Mux 5 and Mux 6 and Mux 7 and Mux 8 is common and represented as s_1, s_2, s_3 and s_4, respectively. The intermediate sum of adder 1 is p_1, and that of adder 2 is p_3. The intermediate carry of adder 1 is p_2, adder 2 is p_4, and the carry of adder

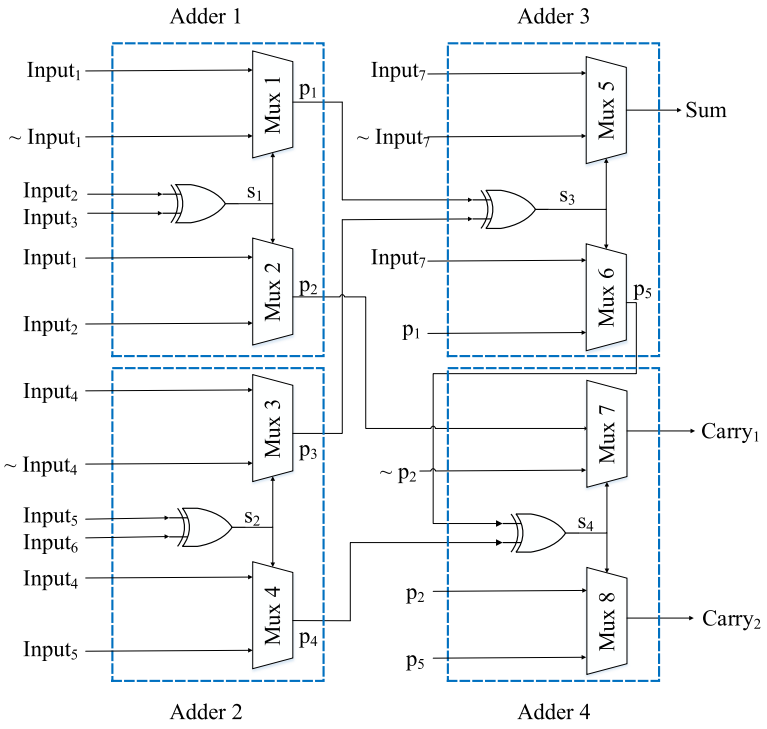

Fig. 6 7:3 counter using Mux and ex-or-based adder [8]

3 is p_5. The equations of Sum, Carry_1, Carry_2 of 7:3 counter using multiplexer and ex-or-based adder are given by (4), (5) and (6), respectively.

$$\text{Sum} = \overline{s_3} \cdot \text{Input}_7 + s_3 \cdot \overline{\text{Input}_7} = \overline{(p_1 \oplus p_3)} \cdot \text{Input}_7 + (p_1 \oplus p_3) \cdot \overline{Input_7} \quad (4)$$

$$\text{Carry}_1 = \overline{s_4} \cdot p_2 + s_4 \cdot \overline{p_2} = \overline{(p_5 \oplus p_4) \cdot p_2} + (p_5 \oplus p_4) \cdot \overline{p_2} \quad (5)$$

$$\text{Carry}_2 = \overline{s_4} \cdot p_5 + s_4 \cdot p_1 = \overline{(p_5 \oplus p_4) \cdot p_5} + (p_5 \oplus p_4) \cdot p_1 \quad (6)$$

$$\begin{aligned} where, s_1 &= \text{Input}_2 \oplus \text{Input}_3, s_2 = \text{Input}_5 \oplus \text{Input}_6, s_3 \\ &= p_1 \oplus p_3, s_4 = p_5 \oplus p_4 \\ p_1 &= \overline{s_1} \cdot \text{Input}_1 + s_1 \cdot \overline{Input_1} = \overline{(Input_2 \oplus Input_3)} \cdot Input_1 \\ &\quad + (Input_2 \oplus Input_3) \cdot \overline{Input_1} \\ p_2 &= \overline{s_1} \cdot Input_2 + s_1 \cdot Input_1 = \overline{(Input_2 \oplus Input_3)} \cdot Input_2 \end{aligned}$$

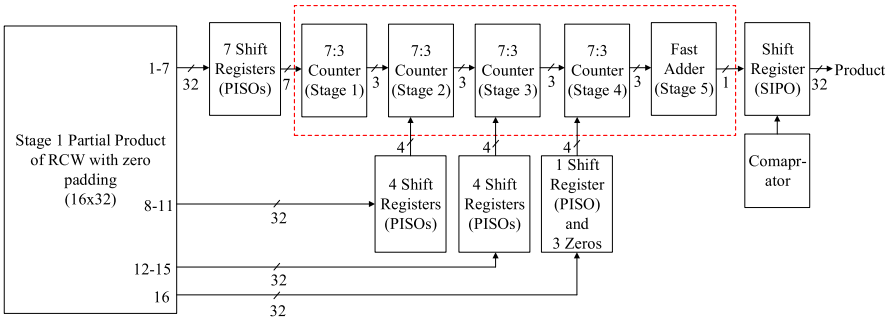

Fig. 7 Architecture of counter-based modular Wallace tree multiplier

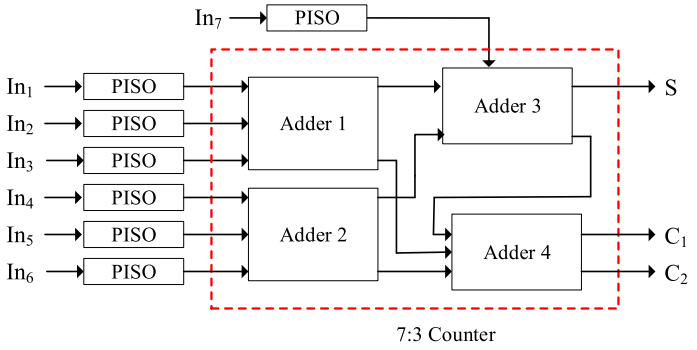

Fig. 8 Architecture of stage 1 in counter-based modular Wallace tree multiplier

$$
\begin{aligned}
&+ (Input_2 \oplus Input_3) \cdot Input_1 \\
p_3 = \overline{s_2} \cdot Input_4 + s_2 \cdot \overline{Input_4} &= \overline{(Input_5 \oplus Input_6)} \cdot Input_4 \\
&+ (Input_5 \oplus Input_6) \cdot \overline{Input_4} \\
p_4 = \overline{s_2} \cdot Input_5 + s_2 \cdot Input_4 &= \overline{(Input_5 \oplus Input_6)} \cdot Input_5 \\
&+ (Input_5 \oplus Input_6) \cdot Input_4 \\
p_5 = \overline{s_3} \cdot p_1 + s_3 \cdot Input_7 &= \overline{(p_1 \oplus p_3)} \cdot p_1 + (p_1 \oplus p_3) \cdot Input_7
\end{aligned}
$$

Figures 7 and 8 show the architecture of proposed CBMW multiplier and stage 1, respectively. In Fig. 7, the red dashed line shows the stages required to achieve the product, and in Fig. 8, the red dashed line highlights the 7:3 counter consisting of multiplexer and ex-or gates. In Fig. 8, In_1, In_2, In_3, In_4, In_5, In_6, In_7 represent the partial product and S, C_1, C_2 represent the output of 7:3 counter for first stage.

Figure 9 shows the algorithm of the CBMW multiplier. In the proposed multiplier, the process is carried out in a repeated manner to perform the multiplication after the formation of partial products to obtain the product of the multiplier and the multiplicand. The first stage is to produce partial products. The partial product tree is arranged in the inverse pyramid form as shown in Fig. 2. The inverse pyramid is padded with zero where the partial product is vacant hence 16x32-bit matrix is obtained and it is divided into a group of 7 bits. As the partial product tree is parallel, each column of the

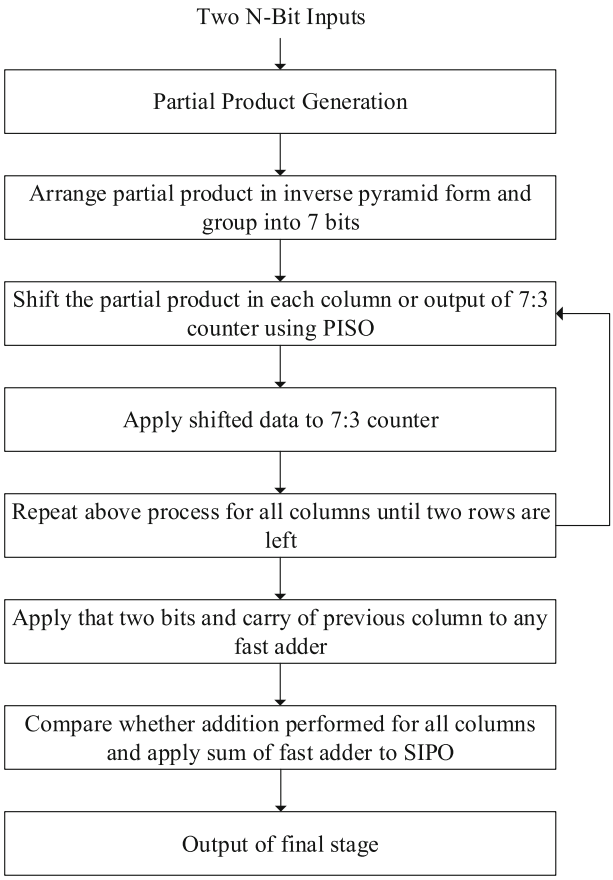

Fig. 9 Algorithm of counter-based modular Wallace tree multiplier

partial product tree is applied to the shifter that is parallel to serial shift register (PISO) that shifts it by one bit and applied to the 7:3 counter module serially. The output S of the 7:3 counter is placed at the same column, C_1 is passed to the next column that is shifted by one, and C_2 is passed to the next-to-next column that is shifted by two as the weight of S is 2^0, C_1 is 2^1 and C_2 is 2^2, respectively. The output of first stage 7:3 counter and the remaining partial product is applied to the next stage 7:3 counter. The same process is repeated for the next stage and it is repeated until two rows are remaining. In the last stage, the result of the previous stage is added along with the carry of the previous column using carry-save adder and it is available serially at the output hence applied to the shifter that is serial to parallel shift register (SIPO) to obtain the final product of multiplication. The comparator is used at the final stage to obtain final output by comparing whether the reduction is performed for each column and stage or not. For a 16×16 multiplier, five stages are required and the output is available after 32 clock cycles.

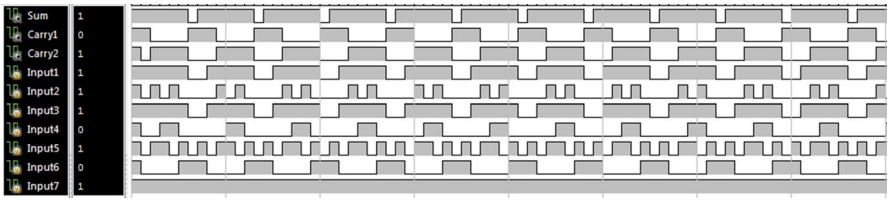

Fig. 10 Output of 7:3 counter using Mux and ex-or-based adder

3 Simulation Result and Comparison

The proposed architecture has been synthesized for FPGA as well as for ASIC implementation. The performance difference between FPGA and ASIC implementation is due to the architecture difference. In FPGAs, logic circuits are implemented using LUTs, whereas, in ASIC, logic cells are used. Usually, FPGA implementation can give an estimate of area, power and delay for further ASIC implementation. The proposed CBMW multiplier achieves efficient performance on both FPGA and ASIC as it consists of simple shift registers with D-flipflops, one 7:3 counter with a simple multiplexer and ex-or gates per stage and one comparator. This structure can be implemented efficiently using LUTs on FPGA and logic cells on ASIC. The early research work in this domain was more focused on the parallel implementation of multi-bit counters as per the design requirement to reduce delay but it results in more switching activity and hence power consumption goes higher. The proposed structure uses a single 7:3 counter in the input stage irrespective the number of partial products to provide direct input to the internal nodes. This arrangement reduces switching activity without compromising delay.

3.1 7:3 Counter

Figure 10 shows the response of the 7:3 counter used in the proposed design. Input 1, input 2, input 3, input 4, input 5, input 6 and input 7 represent the input of 7:3 counter and output bits are Sum, Carry_1 and Carry_2. The weight of the Sum is 2^0, whereas the weight of Carry_1 and Carry_2 are 2^1 and 2^2, respectively.

3.2 Comparison of Various Designs for 7:3 Counter

Tables 1 and 2 show the comparison of the 7:3 counter used in the proposed multiplier design with other most recent designs of 7:3 counter which are synthesized for FPGA using Xilinx ISE design suite 14.7 and ASIC using Synopsys Design Compiler, respectively. The 7:3 counter using a multiplexer and ex-or-based adder has the least power consumption of 85.65 mW for FPGA and 4.71 μW for ASIC using 180 nm technology compared to other existing designs of 7:3 counter. It has less delay of 4.60 ns and 0.92 ns for FPGA and ASIC, respectively. Also, it has the least power-delay product. Hence, it is employed in multiplier design to optimize the power delay product parameter.

Table 1 Comparison of different 7:3 counters (FPGA Spartan 3E)

Multiplier	Power (mW)	Dynamic power (mW)	Minimum period (Maxi. Freq.) (ns, MHz)	Minimum I/p arrival time before clock, Maxi. O/p required time after clock (ns, ns)	LUTs	Power delay product (pJ)
[13]	90.67	9.57	4.65	1.94	9	421.61
			214.86	4.04		
[4]	90.69	9.58	4.65	1.94	9	421.70
			214.86	4.28		
[24]	92.50	11.37	4.59	1.94	8	424.57
			217.58	4.28		
[20]	92.63	11.50	4.27	1.94	9	395.53
			234.08	4.28		
[6]	89.66	8.57	5.73	1.94	22	513.75
			174.33	4.28		
[14]	92.34	11.21	4.61	1.94	8	425.68
			216.73	4.36		
[8]	85.65	4.61	4.60	1.94	8	393.99
			212.81	4.36		

Table 2 Comparison of different 7:3 counters using 180 nm technology in Design Compiler

Multiplier	Power (μW)	Delay (ns)	Total cell area	Power delay product (fJ)
[13]	7.75	1.67	306	12.94
[4]	7.78	1.67	306	12.99
[24]	7.82	1.03	319	8.05
[20]	7.83	0.99	324	7.75
[6]	5.11	1.04	375	5.31
[14]	8.37	1.48	364	12.38
[8]	4.71	0.92	275	4.33

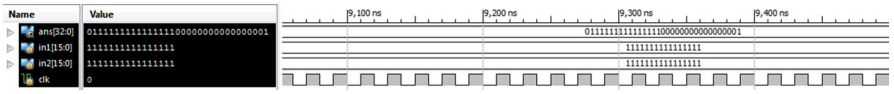

Fig. 11 Output response of 16 × 16 counter-based modular Wallace tree multiplier

3.3 Proposed Counter-Based Modular Wallace Tree Multiplier

The output of 16 × 16 proposed CBMW multiplier is shown in Fig. 11. The in1 and in2 are 16-bit inputs of the multiplier and ans shows the output that is the product of multiplier and multiplicand. In order to prove the efficiency of the proposed CBMW

Table 3 Comparison of 8 × 8 multipliers (FPGA Spartan 3E)

Multiplier	Power (mW)	Dynamic power (mW)	Minimum period (Maxi. Freq.) (ns, MHz)	Minimum I/p arrival time before clock, maxi. O/p required time after clock (ns, ns)	LUTs	Power delay product (pJ)
Wallace tree [26]	70.93	37.06	10.14	1.97	184	719.23
			98.57	4.04		
RCW [27]	117.23	35.80	11.87	1.97	184	1391.63
			84.23	4.28		
CBW [3]	123.72	42.20	12.70	1.97	190	1571.24
			78.69	4.28		
[4]	88.67	7.59	9.57	3.22	164	848.57
			104.42	4.36		
[20]	88.35	7.28	9.47	3.22	162	836.67
			105.52	4.36		
[6]	88.16	7.09	11.83	3.22	169	1042.93
			84.48	4.36		
[14]	88.21	7.14	9.53	3.22	165	840.64
			104.93	4.36		
Proposed	87.12	6.06	9.29	3.22	161	809.34
			107.55	4.28		

multiplier, the comparison is shown for both 8 × 8 and 16 × 16 multiplier. Tables 3 and 4 show the comparison of the 8 × 8 and 16 × 16 CBMW multiplier with other multipliers for Spartan 3E FPGA, respectively. Tables 5 and 6 represent the delay, power and area parameters for 8 × 8 and 16 × 16 multiplier using 180nm technology in Synopsys Design Compiler. The power consumption is found as 87.12 mW for FPGA and 0.0037 mW for ASIC using 180nm technology implementation for the 8 × 8 multiplier design. For 16-bit, the power consumption is 88.16 mW for FPGA and 0.0877 mW for ASIC. It indicates that the CBMW multiplier is power efficient with less delay compared to existing multipliers due to reduced switching activity. CBMW multiplier achieves better PDP performance compared to different versions of the Wallace tree multiplier.

3.4 PVT analysis

The PVT analysis is done to evaluate the performance of the design to verify whether the design meets the specifications or not across various corners. In this paper, the corner analysis is carried out for three cases that are typical, fastest and slowest for the design implemented using 180 nm technology in Synopsys PrimeTime as shown in Table 7. The temperature is varied from -40 $^{\circ}C$ to 125 $^{\circ}C$ and voltage from 1.98 V to 1.62 V. The operational clock frequency is 100 MHz. The maximum timing, minimum

Table 4 Comparison of 16 × 16 multipliers (FPGA Spartan 3E)

Multiplier	Power (mW)	Dynamic power (mW)	Minimum period (Maxi. Freq.) (ns, MHz)	Minimum I/p arrival time before clock, Maxi. O/p required time after clock (ns,ns)	LUTs	Power delay product (pJ)
Wallace tree [26]	127.17	45.61	23.06	1.97	701	2932.54
			43.35	26.47		
RCW [27]	134.70	53.05	16.85	2.05	789	2269.69
			59.34	23.14		
CBW [3]	129.43	47.84	17.05	2.05	821	2206.78
			58.62	24.22		
[4]	90.19	9.09	11.78	3.43	484	1062.43
			84.84	4.36		
[20]	99.20	9.10	10.27	3.34	476	1018.78
			97.34	4.28		
[6]	89.35	8.27	10.27	3.43	500	917.62
			97.34	4.28		
[14]	89.37	8.29	11.40	3.34	482	1018.81
			87.68	4.36		
Proposed	88.16	7.09	10.27	3.34	473	905.40
			97.34	4.28		

Table 5 Comparison of 8 × 8 multipliers using 180 nm technology (Design Compiler)

Multiplier	Power (mW)	Delay (ns)	Total cell area	Power delay product (pJ)
Wallace tree [26]	0.6791	72.47	7474	49.2143
RCW [27]	0.6740	72.37	7311	48.7773
CBW [3]	0.6080	73.38	8180	44.6150
[4]	0.1475	72.21	4493	10.6509
[20]	0.1451	72.45	4428	10.5124
[6]	0.1477	72.40	4753	10.6934
[14]	0.1462	73.38	4553	10.7281
Proposed	0.0037	71.84	2214	0.2678

timing and power for all three cases are reported in Table 8. It can be investigated from Table 8 that it meets timing specifications with low power consumption for all corners.

4 Application of CBMW Multiplier in MAC

Multipliers are widely used in the MAC unit to realize many digital signal processing applications such as ALU (arithmetic logic unit), Fourier transform, convolution, fil-

Table 6 Comparison of 16 × 16 multipliers using 180 nm technology (Design Compiler)

Multiplier	Power (mW)	Delay (ns)	Total cell area	Power delay product (pJ)
Wallace tree [26]	0.2206	74.96	8013	16.5361
RCW [27]	0.2089	74.48	8023	15.5588
CBW [3]	0.2061	78.50	9280	16.1788
[4]	0.1653	72.12	5564	11.9214
[20]	0.1514	72.11	5451	10.9217
[6]	0.1504	73.43	5715	11.0453
[14]	0.1514	73.87	5526	11.1883
Proposed	0.0877	72.05	3605	6.3187

Table 7 PVT analysis cases

Case	Voltage (V)	Temperature ($^{\circ}C$)
I	1.98	−40
II	1.80	25
III	1.62	125

Table 8 Comparison of proposed design for different PVT cases

Design	Case	Max timing slack (ns)	Min timing slack (ns)	Power (mW)
7:3 Counter	Case I	100.41	0.92	0.0067
	Case II	100.60	1.30	0.0054
	Case III	100.88	2.02	0.0050
8-bit proposed multiplier	Case I	117.06	0.15	0.0373
	Case II	115.71	0.22	0.0303
	Case III	112.99	0.39	0.0246
16-bit proposed multiplier	Case I	117.06	0.20	0.0877
	Case II	114.88	0.23	0.0061
	Case III	113.02	0.37	0.0049

tering, etc. As the multiplier lies on the data path of the MAC unit, the multiplier is required to be fast and power efficient. The MAC unit is implemented using CBMW to prove its efficiency. The architecture of MAC unit is shown in Fig. 12. MAC unit is synthesized for the Spartan 3E FPGA and also for 180nm technology using Synopsys Design Compiler for 8-bit and 16-bit and results are reported in Tables 9, 10, 11, 12. The power consumed in the MAC unit with the proposed multiplier when implemented on FPGA is 87.44 mW for 8-bit and 89.79 mW for 16-bit. And when implemented on ASIC using 180nm technology, the power consumed is 0.1445 mW and 0.4099 mW for 8-bit and 16-bit, respectively.

Table 9 Comparison of 8-bit MAC using different multipliers (FPGA Spartan 3E)

Multiplier	Power (mW)	Dynamic power (mW)	Mini. period (Maxi. Freq.) (ns,MHz)	Mini. I/p arrival time Maxi. O/p time (ns)	LUTs	PDP (pJ)
Wallace tree [26]	126.72	45.16	11.73	1.94	236	1486.42
			85.25	4.53		
RCW [27]	126.55	45.00	12.23	1.97	240	1547.70
			81.75	4.53		
CBW [3]	129.63	48.04	12.60	1.94	238	1633.33
			79.35	4.53		
[4]	93.91	12.77	14.82	3.22	242	1391.74
			67.47	4.83		
[20]	93.81	12.66	14.82	3.22	242	1390.26
			67.47	4.83		
[6]	90.97	9.86	14.83	3.22	230	1349.08
			84.48	4.83		
[14]	93.39	12.25	14.82	3.22	242	1384.03
			67.47	4.83		
Proposed	87.44	6.38	14.82	3.22	242	1295.86
			66.71	4.74		

Table 10 Comparison of 16-bit MAC using different multipliers (FPGA Spartan 3E)

Multiplier	Power (mW)	Dynamic power (mW)	Mini. period (Maxi. Freq.) (ns,MHz)	Mini. I/p arrival time maxi. O/p time (ns)	LUTs	PDP (pJ)
Wallace tree [26]	143.13	61.37	22.59	2.05	838	3233.30
			44.25	4.49		
RCW [27]	138.83	57.13	15.97	2.05	931	2208.78
			62.59	4.53		
CBW [3]	140.74	59.01	19.49	1.94	957	2743.02
			51.29	4.53		
[4]	93.41	12.27	11.19	3.34	629	1045.25
			89.35	4.62		
[20]	93.07	11.94	11.19	3.34	631	1041.45
			89.35	4.62		
[6]	94.13	12.99	13.77	3.34	648	1296.17
			72.61	4.53		
[14]	91.77	10.65	13.77	3.34	658	1263.67
			72.61	4.62		
Proposed	89.79	8.70	13.77	3.34	654	1236.40
			72.61	4.62		

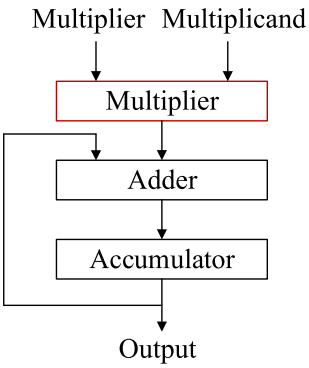

Fig. 12 Architecture of MAC unit

Table 11 Comparison of 8-bit MAC using 180 nm technology in Design Compiler

Multiplier	Power (mW)	Delay (ns)	Total cell area	Power delay product (pJ)
Wallace tree [26]	0.6548	90.44	9190	59.2201
RCW [27]	0.6740	72.47	9192	48.8455
CBW [3]	0.6035	74.69	9027	45.0754
[4]	0.1572	89.45	9129	14.0615
[20]	0.1550	90.40	9167	14.0120
[6]	0.1548	90.44	9190	13.9954
[14]	0.1563	90.41	9518	14.1310
Proposed	0.1445	71.96	8213	10.3982

Table 12 Comparison of 16-bit MAC using 180 nm technology in Design Compiler

Multiplier	Power (mW)	Delay (ns)	Total cell area	Power delay product (pJ)
Wallace tree [26]	0.8956	75.18	10874	67.3312
RCW [27]	0.8830	74.76	10877	66.0130
CBW [3]	0.8632	76.56	10885	66.0865
[4]	0.5222	74.12	9844	38.7054
[20]	0.5324	72.90	9829	38.8119
[6]	0.5229	90.45	9801	47.2963
[14]	0.5285	74.11	9958	39.1671
Proposed	0.4099	72.70	8345	29.7997

5 Conclusion

In this paper, a low-power and scalable CBMW multiplier is proposed. It allows the easy and efficient implementation of the Wallace tree multiplier on FPGA as well as on the ASIC platform. The proposed multiplier uses a 7:3 counter consisting of the multiplexer and ex-or gate-based adders which are more efficient compared to

the other existing designs of 7:3 counter. The last stage includes single fast adder in which previous stage output and carry from the previous column are added to produce the product. As only one 7:3 counter is used per stage, the hardware requirement is less as well as there is a significant reduction in power consumption due to increased locality. The modular multiplier architecture is synthesized for FPGA and AISC, and results are compared with similar multipliers reported in literature. The performance is evaluated for the MAC unit designed using the proposed CBMW. Since the proposed CBMW has better PDP performance, it can be used in low-power applications.

Acknowledgements This work was carried out under the support of the Special Manpower Development Program for Chips to System Design (SMDP-C2SD) project sponsored by Ministry of Electronics & Information Technology (MeitY), Government of India (New Delhi).

Data Availability The data that support the findings of this study are available from the corresponding author on request.

References

1. M.S. Ansari, H. Jiang, B.F. Cockburn, J. Han, Low-power approximate multipliers using encoded partial products and approximate compressors. IEEE J. Emerg. Select. Top. Circ. Syst. **8**(3), 404–416 (2018). https://doi.org/10.1109/JETCAS.2018.2832204
2. S. Asif, Y. Kong, Performance analysis of wallace and radix-4 booth-wallace multipliers. In Proceedings of the 2015 Electronic System Level Synthesis Conference (ESLsyn), pp 17–22 (2015)
3. S. Asif, Y. Kong, Analysis of different architectures of counter based wallace multipliers. In Proceedings of the 2015 Tenth International Conference on Computer Engineering Systems (ICCES), pp 139–144, (2015). https://doi.org/10.1109/ICCES.2015.7393034
4. S. Asif, Y. Kong, Design of an algorithmic wallace multiplier using high speed counters. In Proceedings of the 2015 Tenth International Conference on Computer Engineering Systems (ICCES), pp 133–138 (2015). https://doi.org/10.1109/ICCES.2015.7393033
5. C. Chang, J. Gu, M. Zhang, Ultra low-voltage low-power cmos 4–2 and 5–2 compressors for fast arithmetic circuits. IEEE Trans. Circ. Syst. I Reg. Papers **51**(10), 1985–1997 (2004). https://doi.org/10.1109/TCSI.2004.835683
6. C. Fritz, A. T. Fam. Fast binary counters based on symmetric stacking. IEEE Trans. Very Large Scale Integ. (VLSI) Syst. 25(10):2971–2975 (2017). https://doi.org/10.1109/TVLSI.2017.2723475
7. N. Jagadeeshkumar, D. Meganathan, A novel design of low power and high speed hybrid multiplier. In Proceedings of the 2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN), pp 1–6 (2017). https://doi.org/10.1109/ICSCN.2017.8085724
8. K. B. Jaiswal, V. Nithish Kumar, P. Seshadri, G. Lakshminarayanan, Low power wallace tree multiplier using modified full adder. In Proceedings of the 2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN), pp 1–4, (2015). https://doi.org/10.1109/ICSCN.2015.7219880
9. Y. Jiang, A. Al-Sheraidah, Y. Wang, E. Sha, J. Chung, A novel multiplexer-based low-power full adder. IEEE Trans. Circ. Syst. II Exp. Briefs **51**(7), 345–348 (2004). https://doi.org/10.1109/TCSII.2004.831429
10. W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, F. Lombardi, Design of approximate radix-4 booth multipliers for error-tolerant computing. IEEE Trans. Comput. **66**(8), 1435–1441 (2017). https://doi.org/10.1109/TC.2017.2672976
11. S. Malek, S. Abdallah, A. Chehab, I. Elhajj, A. Kayssi, Low-power and high-speed shift-based multiplier for error tolerant applications. Microprocess. Microsyst. **52**, 566–574 (2017). https://doi.org/10.1016/j.micpro.2017.07.002
12. P. Mavuri, B. Velan, Design and performance analysis of a high speed mac using different multipliers. In Proceedings of the 2015 Fifth International Conference on Advances in Computing and Communications (ICACC), pp 151–154 (2015). https://doi.org/10.1109/ICACC.2015.95

13. M. Mehta, V. Parmar, E. Swartzlander, High-speed multiplier design using multi-input counter and compressor circuits. In [1991] Proceedings 10th IEEE Symposium on Computer Arithmetic, pp 43–50, (1991). https://doi.org/10.1109/ARITH.1991.145532
14. S. Murugeswari, S.K. Mohideen, Design of area efficient and low power multipliers using multiplexer based full adder. Second Int. Conf. Curr. Trends Eng. Technol. ICCTET **2014**, 388–392 (2014). https://doi.org/10.1109/ICCTET.2014.6966322
15. K. Prasad, K. K. Parhi, Low-power 4–2 and 5–2 compressors. In Conference Record of Thirty-Fifth Asilomar Conference on Signals, Systems and Computers (Cat.No.01CH37256), vol 1, pp 129–133 (2001). https://doi.org/10.1109/ACSSC.2001.986892
16. R. Prathiba, P. Sandhya, R. Varun, Design of high performance and low power multiplier using modified booth encoder. In Proceedings of the 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), pp 794–798 (2016). https://doi.org/10.1109/ICEEOT.2016.7754795
17. L. Qian, C. Wang, W. Liu, F. Lombardi, J. Han, Design and evaluation of an approximate wallace-booth multiplier. In Proceedings of the 2016 IEEE International Symposium on Circuits and Systems (ISCAS), pp 1974–1977, (2016). https://doi.org/10.1109/ISCAS.2016.7538962
18. N. Ravi, T. Rao, B. Rao, T. Prasad, A new reduced multiplication structure for low power and low area modified booth encoding multiplier. Int. Conf. Model. Optim. Comput. Proced. Eng. **38**, 2767–2771 (2012). https://doi.org/10.1016/j.proeng.2012.06.324
19. S. Ravi, A. Patel, M. Shabaz, P. Chaniyara, H. Kittur. Design of low-power multiplier using ucsla technique. In Artificial Intelligence and Evolutionary Algorithms in Engineering Systems, pp 119–126, New Delhi. Springer, New York (2015)
20. A. Saha, R. Pal, A. Naik, D. Pal, Novel cmos multi-bit counter for speed-power optimization in multiplier design. AEU Int. J. Electron. Commun. **95**, 189–198 (2018). https://doi.org/10.1016/j.aeue.2018.08.015
21. A. K. Sahu, L. Kumre, Low-power less-area bypassing-based multiplier design. In Proceedings of the 2017 International Conference on Inventive Computing and Informatics (ICICI), pp 522–526 (2017). https://doi.org/10.1109/ICICI.2017.8365186
22. J. Selvakumar, V. C. Bhaskar, Low power and area optimized truncated multiplier architecture. In IET Chennai 3rd International on Sustainable Energy and Intelligent Systems (SEISCON 2012), pp 1–6, (2012). https://doi.org/10.1049/cp.2012.2209
23. E.E. Swartzlander, Parallel counters. IEEE Trans. Comput. **22**(11), 1021–1024 (1973). https://doi.org/10.1109/T-C.1973.223639
24. S. Veeramachaneni, A. Lingamneni, M. Krishna, M. Srinivas, Novel architectures for efficient (m, n) parallel counters. Proceedings of the 17th ACM Great Lakes symposium on VLSI, pp 188–191, (2007). https://doi.org/10.1145/1228784.1228833
25. C. Vinoth, V. S. K. Bhaaskaran, B. Brindha, S. Sakthikumaran, V. Kavinilavu, B. Bhaskar, M. Kanagasabapathy, B. Sharath, A novel low power and high speed wallace tree multiplier for risc processor. In Proceedings of the 2011 3rd International Conference on Electronics Computer Technology, vol 1, pp 330–334, (2011). https://doi.org/10.1109/ICECTECH.2011.5941617
26. C.S. Wallace, A suggestion for a fast multiplier. IEEE Trans. Elect. Comput. **13**(1), 14–17 (1964). https://doi.org/10.1109/PGEC.1964.263830
27. R.S. Waters, E.E. Swartzlander, A reduced complexity wallace multiplier reduction. IEEE Trans. Comput. **59**(8), 1134–1137 (2010). https://doi.org/10.1109/TC.2010.103