Implementation of A High Speed Multiplier for High-Performance and Low Power Applications

G Ganesh Kumar, Subhendu K Sahoo
Department of Electrical and Electronics Engineering
BITS-Pilani, Hyderabad Campus
India, 500078.
Email: ggkumar1988@gmail.com, sahoo@hyderabad.bits-pilani.ac.in

***Abstract*—The performance of multiplication in terms of speed and power is crucial for most of the Digital Signal Processing (DSP) applications. Many researchers have come up with various multipliers such as array, Booth, carry save, Wallace tree and modified Booth multipliers. However, for the present day applications Vedic multipliers based on Vedic Mathematics are presently under focus due to their high speed and low power consumption. In this paper, we propose a design of 8 and 16-bit multipliers using fast adders (carry save adder, Brent-Kung adder and carry-select adder) to minimize the power-delay product of multipliers intended for high-performance and low-power applications. Implementation results demonstrate that the proposed Vedic multipliers with fast adders really achieve significant improvement in delay, and power-delay product when compared with the conventional multipliers.**

Keywords*—Vedic multiplier, carry save array, power-delay product, carry-select adder.

I. Introduction

Multipliers play an important role in many DSP applications [1] such as convolution, Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT) and filtering. The speed of the DSPs largely depends on the multiplier block. This in turn increases the demand for high speed multipliers. Over the past few years, many researchers have developed various multipliers using several algorithms such as array, Booth, carry save, Wallace tree and modified Booth algorithms. A number of multiplier architectures also have been proposed based on these algorithms that include parallel, serial and serial-parallel multipliers.

In an array multiplier, multiplication is based on shift and add. The partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added with carry propagate adder. Array multiplier is easy to design due to its regular structure [2]. However, the main disadvantage of the array multiplier is that as the width of the multiplier increase the delay becomes more. This is because the worst-case delay of the multiplier proportional to the width of the multiplier.

The delay of the multiplier is further reduced by the arrangement of adders using Carry Save Array (CSA) method and a Wallace tree adder method. In CSA method [3], every carry and sum signal is passed to the adders of the next stage. The final product is obtained in a final adder by using fast adder (Carry Lookahead Adder). The CSA multiplier has regular structure to design but, there is some difficulty in achieving high speed. In the Wallace tree method [4], [5], the partial product bits are summed up in parallel by means of a tree of carry save adders. The main advantage of this method is that, it achieves high speed operation. However, in Wallace tree method the circuit layout is complex and has irregular wires.

The major improvement in the multipliers is by reducing the number of partial products generated. The Booth multiplier and modified Booth encoded Wallace tree (MBW) [6]–[8] are such multiplier that, reduces the number of adders. However, the multiplication process involves various intermediate operations that include comparisons, additions and subtractions which reduces the speed as the width of multiplier and multiplicand increases.

In order to overcome the disadvantage with respect to speed of the aforesaid algorithms, [9] and [10] presented a new multiplier design approaches based on Vedic Mathematics. The partial products are calculated in advance and then added based on the Vedic Math approach to obtain the final product. Furthermore to enhance the speed of multiplier, in [11]–[15] authors have presented new designs, where the partial products are added by using ripple carry adder, carry save adder and carry lookahead adder . In [16], a compressor based Vedic multiplier (CVM) has been designed, to further enhance the speed by replacing the full adders and half adders with compressors. However, in today's applications one of the major challenges for high-performance DSP applications is the power dissipation, both static and dynamic. Therefore, there is a need to find an optimum between speed and power, instead of targeting them independently. This is represented by the average energy dissipated for one switching event and it is known as power-delay product.

In this paper, the partial products of the multiplier are added by using fast adders (carry save adder, Brent-Kung adder and carry-select adder) to achieve delay and power-efficiency. Experimental results show that the proposed multiplier with fast adders can achieve significant improvement in delay, and power-delay product when compared to ripple carry based Vedic (RCV) multiplier, CVM, CSA and MBW.

In Section II, we discuss the Urdhva Tiryakbhyam (vertical and crosswise) method of multiplication using Vedic maths in detail. Section III describes the proposed Vedic multiplier that adds the partial products using carry save adder, Brent-

Kung adder and carry-select adder. The implementation results for different multipliers are presented in Section IV. Finally, Section V presents a conclusion.

II. Vedic Mathematics-Urdhva Tiryakbhyam Sutra

Vedic Mathematics [17] is an ancient and eminent approach which is mainly based on 16 Sutras dealing with various branches of mathematics like arithmetic, algebra, trigonometry, analytical geometry etc. In order to perform multiplication, one of the most preferred algorithm among these 16 Sutras is the Urdhva Tiryakbhyam Sutra. The words Urdhva and Tiryakbhyam are derived from Sanskrit which mean verticality and crosswise respectively. The main advantage of utilizing this algorithm in comparison with the existing multiplication techniques is that, the partial products required for multiplication are generated in parallel [9]. These partial products are added in such a way that saves a lot of processing time. This algorithm is applicable for the multiplication of binary numbers, so we have chosen this Sutra for implementation of Vedic multiplier.

Let us consider two inputs X_1X_0 and Y_1Y_0, each of 2 bits. $P_3P_2P_1P_0$ represents each bit of the final computed product. The result is obtained after generating partial products and adding them as per the basic method of multiplication shown below.

$$\begin{array}{cccc} & & X_1 & X_0 \\ & \times & Y_1 & Y_0 \\ \hline & & X_1Y_0 & X_0Y_0 \\ & X_1Y_1 & X_0Y_1 & \\ \hline P_3 & P_2 & P_1 & P_0 \end{array}$$

In Vedic mathematics, the vertical multiplication of bits X_0 and Y_0 gives product P_0. The product term P_1 is obtained by the addition of crosswise bit multiplication i.e. X_1 & Y_0 and X_0 & Y_1. P_2 is addition of the vertical product of bits X_1 & Y_1 along with the carry generated from the previous addition during P_1. P_3 output is the carry generated from the previous calculation of P_2. Fig. 1 shows the module that generate 4 outputs from the two 2-bit inputs by using AND gates and half adder and it is known as 2×2 multiplier block. This is similar to shift and add technique however, to further improve the performance of the multiplier, divide and conquer strategy is used for higher bits.

Now, let us analyze for a 4×4 multiplication that gives output product as $P_7P_6P_5P_4P_3P_2P_1P_0$. The multiplicand and multiplier are decomposed equally as X_H=X_3X_2 and X_L=X_1X_0 for X and Y_H=Y_3Y_2 and Y_L=Y_1Y_0 for Y, where H and L represents higher and lower order bits of X and Y. Fig. 2 shows the 4 bit multiplication by taking two bit at a time and using 2×2 multiplier block.

According to Vedic mathematics, initially vertical multiplication of X_L & Y_L is carried out, which gives partial product outputs $p_0[3:0]$. Then, the middle one shows crosswise multiplication of two, 2×2 multiplier with inputs X_H & Y_L and X_L & Y_H, generates $p_1[3:0]$ and $p_2[3:0]$ partial product outputs respectively. Finally, the last block inputs X_H

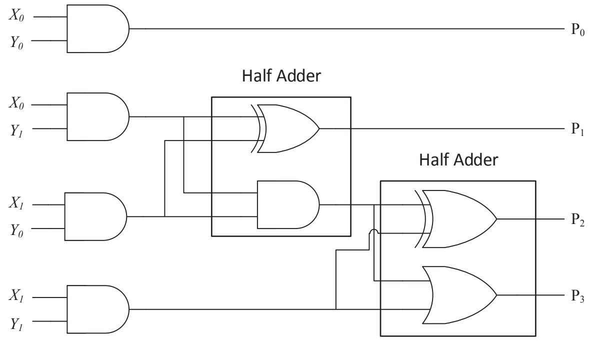
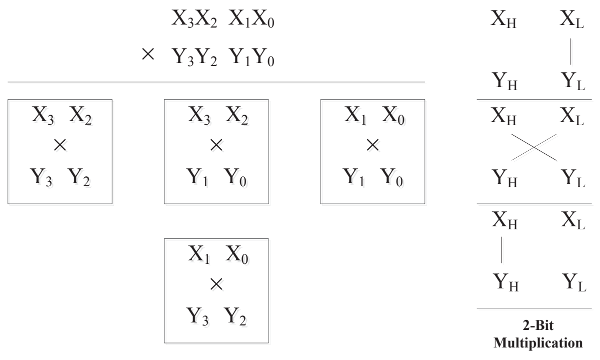


Fig. 1. 2×2 binary multiplier



X_3X_2 X_1X_0
$\times$ Y_3Y_2 Y_1Y_0
X_3 X_2 $\times$ Y_3 Y_2
X_3 X_2 $\times$ Y_1 Y_0
X_1 X_0 $\times$ Y_1 Y_0
X_1 X_0 $\times$ Y_3 Y_2
X_H X_L
Y_H Y_L
2-Bit Multiplication

Fig. 2. Block diagram representation of 4×4 multiplier and 2×2 multiplier

& Y_H multiplies vertically and generates the partial product outputs as $p_3[3:0]$. The first two final product outputs P_0 and P_1 are same as that of the partial products $p_0[0]$ and $p_0[1]$. The remaining product terms are obtained by using three conventional adders as shown in Fig. 3. The inputs for adder1 are $\{p_3[3:0], 00\}$ and $\{00, p_2[3:0]\}$ and for adder2 are $p_1[3:0]$ and $\{00, p_0[3:2]\}$. The outputs of adder1 and adder2 are given inputs to the adder3, that generates the final product terms $P[7:2]$. Thus, the final product terms are obtained by parallel multiplication using sub multiplier blocks and addition as $P_7P_6P_5P_4P_3P_2P_1P_0$, that can reduces the delay of the multiplier.

In the above design the partial products are added using parallel addition of full adders and half adders. In [16], to further improve the speed of the design a novel method have been presented by replacing the adders with compressors. However, the power-delay product of this design is more. To minimize the power-delay product, we proposed multiplier where the partial products are added using fast adders, that is discussed in the next section.

III. Design of the Proposed 8 and 16-bit Vedic Multipliers

In this section we extend the Vedic algorithm to design a 16-bit multiplier using fast adders, which can add the partial products with a high speed and reduces the power-delay product. At first 2-bit, 4-bit and 8-bit multiplier blocks are to be designed, in order to obtain a 16×16 Vedic Multiplier. The design of a 2×2 multiplier block is similar to shift and

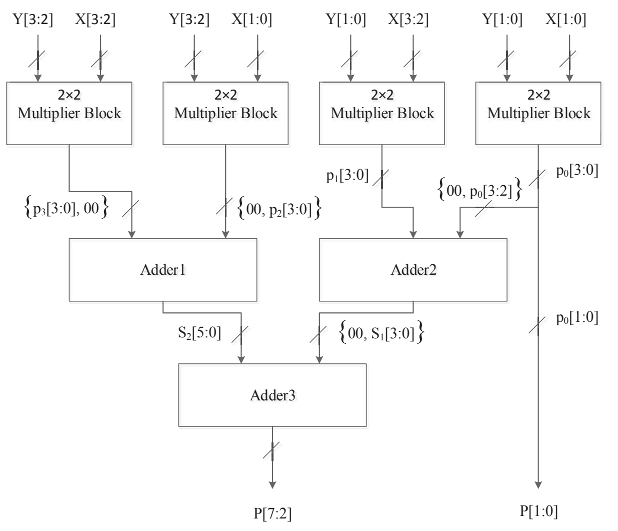


Fig. 3. Block diagram of 4×4 Vedic multiplier

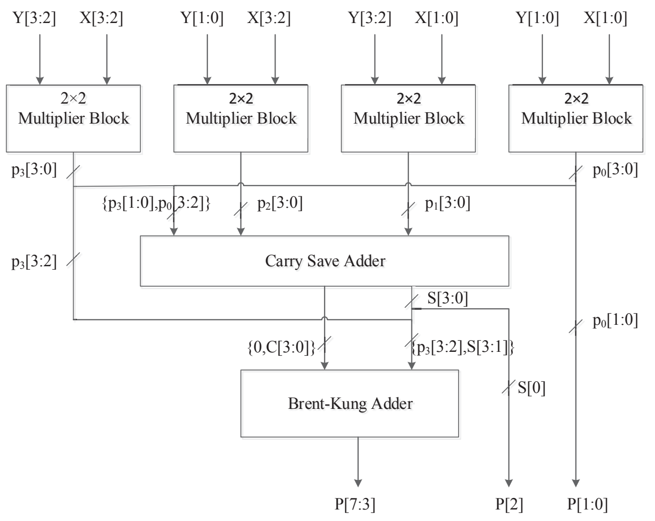


Fig. 4. Proposed 4-bit Vedic Multiplier

add method as shown in Fig. 1.

For 4×4 multiplier block, the multiplicand and multiplier bits are decomposed equally as X_H=X_3X_2 and X_L=X_1X_0 for X and Y_H=Y_3Y_2 and Y_L=Y_1Y_0 for Y. According to the Vedic algorithm, the decomposed bits are given as inputs to the individual 2-bit multiplier block as shown in Fig. 4. The delay of the design reduces, as all the partial products are obtained in parallel, and it is further reduced by using the carry save adder and Brent-Kung adder in order to obtain the final product.

To obtain 8×8 multiplier block, the multiplicand and multiplier bits are decomposed equally as X_H=$X_7X_6X_5X_4$ and X_L=$X_3X_2X_1X_0$ for X and Y_H=$Y_7Y_6Y_5Y_4$ and Y_L=$Y_3Y_2Y_1Y_0$ for Y as shown in Fig. 5. The first four final product outputs $P[3:0]$ are same as that of the partial products $p_0[3:0]$. The other partial products which includes $p_1[7:0]$, $p_2[7:0]$ and $\{p_3[3:0], p_0[7:4]\}$ are added using carry save adder which can generates the sum and carry output bits as $S[7:0]$ and $C[7:0]$ respectively. The product output $P[4]$ is

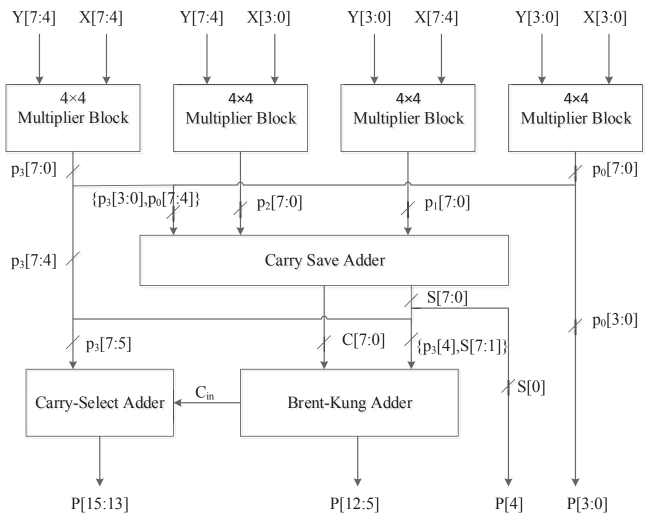


Fig. 5. Proposed 8-bit Vedic Multiplier

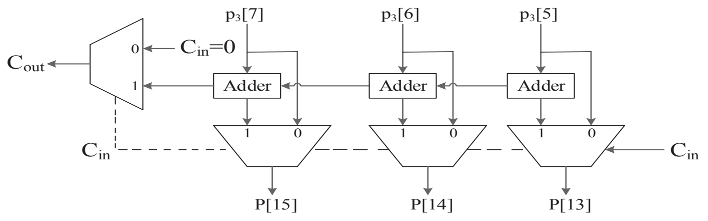


Fig. 6. Carry-select adder for 8-bit Vedic Multiplier

the output of sum $S[0]$. Then, the sum and carry bits of carry save adder are added by using Brent-Kung adder along with the partial product bit $p_3[4]$ that generates the final product term $P[12:5]$ and a carry bit C_{in}.

The final stage of addition is done by using carry-select adder that contains multiplexers and half adders as shown in Fig. 6. If the input of the carry-select adder is $C_{in} = 0$, the multiplexer gives the input of the partial product output $p_3[15:9]$ to the final product terms $P[31:25]$ directly without any computation, that can reduces the switching power. If the input carry bit $C_{in} = 1$ then, the output of the multiplexer is the addition of the partial product output $p_3[15:9]$ with the carry bit using half adders. The final carry out bit (C_{out}) is discarded in the final product.

In the similar way of design, a 16×16 Vedic Multiplier is obtained by using 8×8 multiplier blocks, carry save adder, Brent-Kung adder, and carry select adder as shown in Fig. 7. The usage of the fast adder in the proposed design improves the performance in terms of delay, and power-delay product. Hence, the signal processing can be made faster using the proposed multiplier in Multiplier-Accumulator (MAC) unit, Fast Fourier Transform (FFT), convolution, and filtering. However, the major drawback of the proposed multiplier is that, it requires larger area for computation.

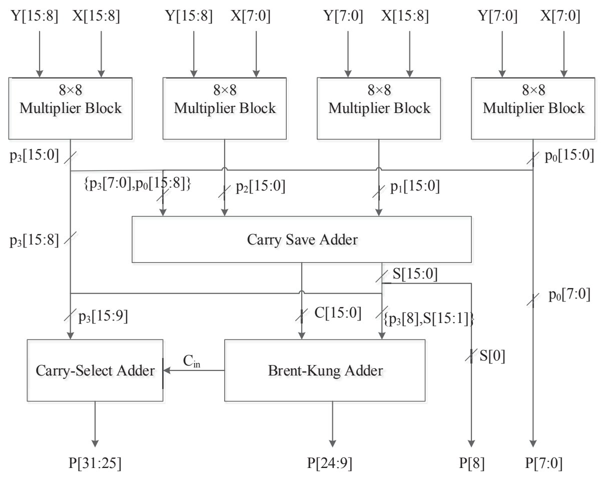


Fig. 7. Proposed 16-bit Vedic Multiplier

IV. Experimental results

For comparison, we have implemented various multipliers whose partial product bits are added by using different approaches. These multipliers were modeled in Verilog HDL and synthesized by using Synopsys Design Compiler with UMC $65nm, 1.32-V$ Standard Cell Library. The synthesized netlists were then fed into Cadence Encounter Digital Implementation System to perform placement and routing. Power dissipation was estimated from the same netlists by feeding them into Synopsys Prime Time to perform full transistor-level power simulation with 1000 random input patterns.

The implementation results, including the hardware area, critical path delay, and power consumption for these multipliers with $N = 8$ and 16, are listed in Table I, where RCV [13], CSA [3], CVM [16] and MBW [8] denote the ripple carry based Vedic, carry save array, compressor based Vedic and modified Booth encoded Wallace tree multipliers. As can be seen in Table I, the ripple carry adder based Vedic multiplier typically has the largest delay due to its ripple carry adder structure. Though the area and power consumed by the proposed 16-bit Vedic multiplier is slightly more, but on average it improves the total delay by 28.30% and power-delay product by 12.97% compared to RCV, CSA, CVM and MBW. The results show that the proposed approach can efficiently reduce the delay and power-delay product of multipliers.

TABLE I. Experimental results of different multipliers

N	Design	Delay (ns)	Area (μm^2)	Power (mW)	Power-delay Product (pJ)
8	RCV [13]	1.39	723.83	0.350	0.486
	CSA [3]	1.31	576.63	0.346	0.453
	CVM [16]	1.25	728.92	0.352	0.440
	MBW [8]	1.17	581.11	0.348	0.407
	Proposed	1.04	749.12	0.358	0.372
	Percentage Improvement (%)	18.42	-16.27	-2.57	16.34
16	RCV [13]	2.87	3007.35	1.420	4.075
	CSA [3]	2.68	1776.95	1.297	3.475
	CVM [16]	1.82	3104.54	1.832	3.334
	MBW [8]	1.80	2003.11	1.821	3.227
	Proposed	1.57	3331.52	1.948	3.058
	Percentage Improvement (%)	28.30	-42.96	-25.16	12.97

V. Conclusion

In this paper, we presented a Vedic multiplier to minimize the power-delay product of 8 and 16-bit multipliers for high performance and low-power applications. In the proposed multipliers, the summation of partial products of Vedic multiplier was slightly modified using carry save adder, Brent-Kung adder, and carry-select adder in order to improve the efficiency of the multipliers. Experimental results have demonstrated that the proposed multiplier with fast adders can achieve significant improvement in delay and power-delay product when compared with the conventional multiplier architectures.

References

[1] J. Pihl and E. J. Aas, "A multiplier and squarer generator for high performance DSP applications," in *Circuits and Systems, 1996., IEEE 39th Midwest symposium on*, vol. 1. IEEE, 1996, pp. 109–112.

[2] H. Guilt, "Fully iterative fast array for binary multiplication," *Electronics Letters*, vol. 5, p. 263, 1969.

[3] T. G. Noll, "Carry-save architectures for high-speed digital signal processing," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 3, no. 1-2, pp. 121–140, 1991.

[4] G. Goto, T. Sato, M. Nakajima, and T. Sukemura, "A 54× 54-b regularly structured tree multiplier," *Solid-State Circuits, IEEE Journal of*, vol. 27, no. 9, pp. 1229–1236, 1992.

[5] G. Goto, "High speed digital parallel multiplier," Nov. 7 1995, uS Patent 5,465,226.

[6] A. D. Booth, "A signed binary multiplication technique," *The Quarterly Journal of Mechanics and Applied Mathematics*, vol. 4, no. 2, pp. 236–240, 1951.

[7] A. Cooper, "Parallel architecture modified Booth multiplier," *Electronic Circuits and Systems, IEE Proceedings G*, vol. 135, no. 3, pp. 125–128, 1988.

[8] J. Fadavi-Ardekani, "M*N Booth encoded multiplier generator using optimized Wallace trees," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 1, no. 2, pp. 120–125, 1993.

[9] H. Thapliyal, M. Srinivas, and H. R. Arabnia, "Design and Analysis of a VLSI Based High Performance Low Power Parallel Square Architecture." in *AMCS*, 2005, pp. 72–76.

[10] S. Akhter, "VHDL implementation of fast NxN multiplier based on Vedic mathematic," in *Circuit Theory and Design, 2007. ECCTD 2007. 18th European Conference on*. IEEE, 2007, pp. 472–475.

[11] H. D. Tiwari, G. Gankhuyag, C. M. Kim, and Y. B. Cho, "Multiplier design based on ancient Indian Vedic Mathematics," in *SoC Design Conference, 2008. ISOCC'08. International*, vol. 2. IEEE, 2008, pp. II–65.

[12] M. Ramalatha, K. D. Dayalan, P. Dharani, and S. D. Priya, "High speed energy efficient ALU design using Vedic multiplication techniques," in *Advances in Computational Tools for Engineering Applications, 2009. ACTEA'09. International Conference on*. IEEE, 2009, pp. 600–603.

[13] D. Jaina, K. Sethi, and R. Panda, "Vedic mathematics based multiply accumulate unit," in *Computational Intelligence and Communication Networks (CICN), 2011 International Conference on*. IEEE, 2011, pp. 754–757.

[14] V. Kunchigi, L. Kulkarni, and S. Kulkarni, "High speed and area efficient Vedic multiplier," in *Devices, Circuits and Systems (ICDCS), 2012 International Conference on*. IEEE, 2012, pp. 360–364.

[15] Y. Bansal, C. Madhu, and P. Kaur, "High speed vedic multiplier designs-a review," in *Engineering and Computational Sciences (RAECS), 2014 Recent Advances in*. IEEE, 2014, pp. 1–6.

[16] S. R. Huddar, S. R. Rupanagudi, M. Kalpana, and S. Mohan, "Novel high speed vedic mathematics multiplier using compressors," in *Automation, Computing, Communication, Control and Compressed Sensing (iMac4s), 2013 International Multi-Conference on*. IEEE, 2013, pp. 465–469.

[17] S. B. K. Tirtha, *Vedic mathematics*. Motilal Banarsidass Publ., 1992, vol. 10.