International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161
©2014 INPRESSCO®, All Rights Reserved
Available at http://inpressco.com/category/ijcet

Research Article

Studies and Performance Evaluation of Vedic Multiplier using Fast Adders

Y. Narasimha Rao[Á*], G.Samuel Vara Prasada Raju[Á] and Penmetsa V Krishna Raja[Ć]

[Á]Gitam University, Visakhapatnam, AP, India
[Á]Department of Computer Science, Andhra University, Visakhapatnam, AP, India
[Ć]AIMS College of Engineering, Mummidivaram, Amalapuram , East Godavari AP, India

Accepted 10 May 2014, Available online 01 June 2014, **Vol.4, No.3 (June 2014)**

Abstract

In many generic systems it is challenging to reduce the load resulted by many coprocessors, which are used to provide special functions like arithmetic operations, signal processing and many other applications. The speed of processors or coprocessors mainly depends on its internal arithmetic circuits like multipliers in ALU. Many signal processing and scientific computers are demanding multiple number of high speed multipliers on single chip. But this increasing instruction cycle time which further causes decrease in system performance. Maintaining higher throughput in arithmetic operations is important to achieve the desired performance in many real-time applications. One of the key arithmetic operations in such applications is to achieve faster multiplication. Vedic Mathematics is one of the fast and low power multiplier. In this paper the Vedic Multiplier is designed by Urdhva Tiryagbhyam (UT) technique. In the present work the multiplier design and the Multiplier performance was analyzed with various existing fast adders like Carry Look Ahead Adder (CLA), Carry Select Adder (CSLA), Parallel prefix adder (PPA). Here the Vedic multiplier is constructed with different fast adders and analyzed with FPGA using Xilinx Synthesis Tool (XST).

Keywords*: Vedic Multiplier, Fast Adders, High speed circuits, system performance, ALU*

1. Introduction

Multiplication is an important and basic operation in many arithmetic computations. Some of the Multiplication-based applications are such as Multiply and Accumulate (MAC), Intensive Arithmetic Functions(CIAF), Digital Signal Processing (DSP) applications such as convolution, Fast Fourier Transform(FFT), filtering and in many microprocessors and microcontrollers (Rudagi et al., 2011) (vamsi et al., 2012). But unfortunately multiplier circuit consumes much of the processor time and causes diminishing in processor performance. As the multiplication plays more important role in many applications it is highly essential to select high performance multiplier in terms of low area, low power and less delay. At the same time high speed multipliers are required to maintain high throughput in arithmetic computations beside achieving desired performance.

Vedic Multiplier is one of the fast and low power multiplier used in many computations to reduce the complexity, execution time, power etc (Rudagi et al., 2011). Vedic Mathematics is the ancient practice of Indian mathematics which is calculated based on 16 Formulae. These formulae help to reduce the carry propagation from LSB to MSB in the evaluation of the partial products and sums (karishma et al., 2014). In this paper, an 8x8 bit multiplier was constructed with various fast adders individually to analyze their efficiency by comparing their carry propagation individually (Mehta et al., 2009).

2. Vedic Multiplier

Vedic mathematics is mainly based on 16 principles. Vedic mathematics is mainly based on 16 Sutras (or aphorisms) dealing with various branches of mathematics like arithmetic, algebra, geometry etc. These principles along with their brief meanings are enlisted below alphabetically (Bharati krsna, 1986)(Tiwari et al., 2008) (Kunchigi et al., 2012)(Bansal et al., 2014).

1) (Anurupye) Shunyamanyat - If one is in ratio. The other is zero
2) Chalana-Kalanabyham Differences and Similarities.
3) Ekadhikina Purvena - By one more than the previous one
4) Ekanyunena Purvena - By one less than the previous one
5) Gunakasamuchyah - The factors of the sum is equal to the sum of the factors
6) Gunitasamuchyah - The product of the sum is equal to the sum of the product
7) Nikhilam Navatashcaramam Dashatah - All from 9 and the last from 10
8) Paraavartya Yojayet - Transpose and adjust.
9) Puranapuranabyham - By the completion or noncompletion
10) Sankalana-vyavakalanabhyam - By addition and by subtraction

*Corresponding author: **Y. Narasimha Rao**

11) Shesanyankena Charamena - The remainders by the last digit
12) Shunyam Saamyasamuccaye - When the sum is the same that sum is zero
13) Sopaantyadvayamantyam - The ultimate and twice the penultimate
14) Urdhva-tiryakbyham - Vertically and crosswise
15) Vyashtisamanstih - Part and Whole
16) Yaavadunam - Whatever the extent of its deficiency

3. Urdhva Tiryagbhyam Sutra

In the present paper we have discussed Urdhva Tiryagbhyam principle as it can be applied to all types of multiplication. This multiplier is based on an algorithm Urdhva Tiryagbhyam (UT) , which means Vertically and crosswise. The partial products and their summation of multiplication are obtained parallelly in Urdhva Tiryagbhyam shown in fig 1 (Kunchigi et al., 2012)(Bansal et al., 2014).

The algorithm can be generalized for n x n bit number. Since the partial products and their sums are calculated in parallel, the multiplier is independent of the clock frequency of the processor. Thus the multiplier will require the same amount of time to calculate the product and hence is independent of the clock frequency (Rudagi et al., 2011)(Kerur et al., 2011). So the intervention of microprocessor is less but dissipation is more which causes in increase in device temperature. Most of these problems can be minimized with Vedic multiplier.

Algorithm for 4x4 bit Vedic Multiplier using Urdhva Tiryagbhyam

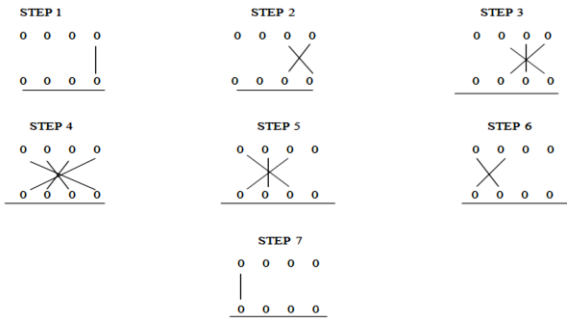

Fig.1 Line diagram of Binary Multiplication in UT method

For the simplicity the UT method can be explained by substituting two 4-bit binary numbers a3 a2 a1 a0 and b3 b2 b1 b0 in the circles of line diagram for binary multiplication. In the result the least significant bit r0 is obtained by multiplying the least significant bits of the multiplicand and the multiplier. The result expressions are shown in the following algorithm are evaluated by following the steps shown in fig.1. Here the efficiency of the carry propagation from one bit position to another bit improves the multiplier performance (Mohammed et al., 2013).

Algorithm

Algorithm for 4x4 bit Vedic Multiplier using Urdhva Tiryagbhyam (Vertically and crosswire) for two binary numbers(sreenath et al., 2012):

```
        a3 a2 a1 a0      Multiplicand
        b3 b2 b1 b0      Multiplier
  ---------------
r7 r6 r5 r4 r3 r2 r1 r0
  ---------------
```

1) r0 = (a0 x b0)
2) r1 = (a1 x b0) + (a0 x b1)
3) r2 = (a2 x b0) + (a0 x b2) + (a1 x b1)
4) r3 = (a3 x b0) + (a0 x b3) + (a2 x b1) + (a1 x b2)
5) r4 = (a3 x b1) + (b1 x a3) + (a2 x b2)
6) r5 = (a3 x b2) + (a2 x b3)
7) r6 = (a3 x b3)
8) r7 = carry propagated from previous bits

Algorithm for 8x8 bit Vedic Multiplier using Urdhva Tiryagbhyam (Vertically and crosswire) for two binary numbers(sreenath et al., 2012)(Kerur et al., 2011):

$A = A_H\ A_L$ (A7 A6 A5 A4 | A3 A2 A1 A0) $B = B_H\ B_L$ (B7 B6 B5 B4 | B3 B2 B1 B0)

$$\begin{array}{r} A_H\ A_L \\ *\ B_H\ B_L \\ \hline CP4\ CP3\ CP2\ CP1 \\ \hline \end{array}$$

$CP1 = (A_L \times B_L)$
$CP2 = (A_H \times B_L) + (A_L \times B_H)$
$CP3 = (A_H \times B_H)$

The 8-bit Binary multiplier circuit is constructed with the help of 4x4 Binary multiplier blocks is shown in figure 2. This

design enables to calculate the cross products (CP) in parallel form, which reduces the latency of product. The adders which are using for addition of cross product must compute the sum in faster way. There are many fast adders present to calculate the sum effectively. But, it is very important to select a proper fast adder by considering all design constraints (Fabio et al., 2009)(jyoshna et al., 2011). In the following sections some frequently used fast adders are analyzed and tested through simulating tools.

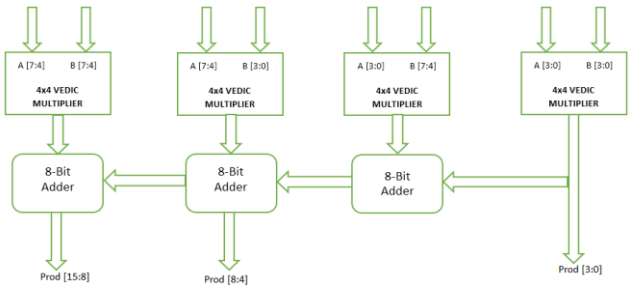

Fig. 2: 8x8 Vedic Multiplier using 4x4 Vedic Multipliers

3.1 Carry Look-Ahead Adder

The carry look-ahead adder solves this problem by reducing carry propagation time by propagating the carry signals in advance. The Carry Look Ahead adder is shown

in figure 3. The Propagate P and generate G in a full-adder, is given as (sreenath et al., 2012) (Lecture notes, Concordia Univ.):

$P_i = A_i \oplus B_i$ Carry propagate
$G_i = A_i B_i$ Carry generate

The new expressions for the output sum and the carryout are given by (Priyanka et al., 2013):
$S_i = P_i \oplus C_{i-1}$
$C_{i+1} = G_i + P_i C_i$

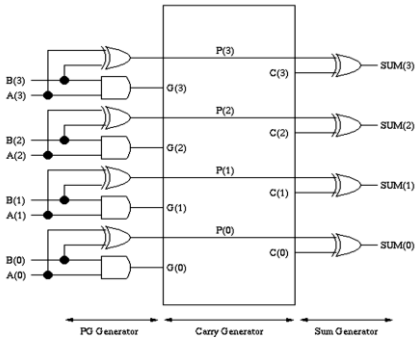

Fig. 3: Carry Look Ahead adder

these equations can be derived for a 4-bit adder as (sreenath et al., 2012),

$C1 = G0 + P0C0$
$C2 = G1 + P1C1 = G1 + P1(G0 + P0C0) = G1 + P1G0 + P1P0C0$
$C3 = G2 + P2C2 = G2 + P2G1 + P2P1G0 + P2P1P0C0$
$C4 = G3 + P3C3 = G3 + P3G2 + P3P2G1 + P3P2P1G0 + P3P2P1P0C0$

These expressions show that C2, C3 and C4 do not depend on its previous carry-in. Therefore C4 does not need to wait for C3 and C4 can reach steady state as soon as C0 arrived. The same is true for C2 and C3 also. The general expression is (sreenath et al., 2012)
$C_{i+1} = G_i + P_iG_{i-1} + P_iP_{i-1}G_{i-2} + \ldots\ldots. P_iP_{i-1}\ldots.P2P1G0 + P_iP_{i-1} \ldots.P1P0C0.$

In real time, for CLA it is not possible to achieve constant delay for the wider-bit adders because of substantial loading capacitance. This causes larger delay and larger power consumption in CLA. The CLA requires fastest growing area and consumes high power with respect to the bit size. So there is a Speed diminution with increase in bit size (Hoe et al., 2011)(Nehru et al., 2012).

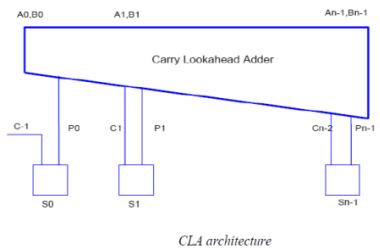

Fig. 4: Synthesized circuit for Carry Look Ahead adder

3.2 Carry Select Adder

Unlike CLA the carry-select adder increases its area requirements to enhance its speed performance. Once the carry-in is propagated, the correct circuit is selected using a MUX to produce the desired output as shown in figure 5. So irrespective of arrival of carry-in, the sum will arrive at output (Lecture notes, Concordia Univ.)(Desiree et al., 2013). So it will take less time to compute the sum when compared with other methods (Lecture notes, Concordia Univ.).

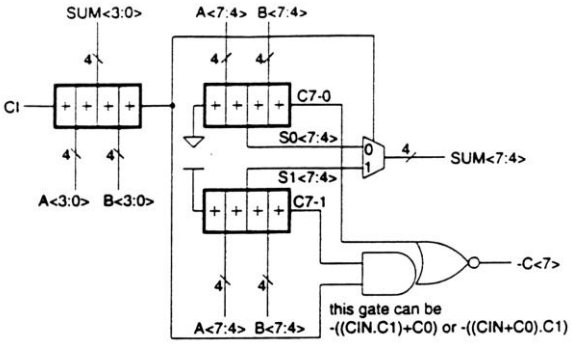

Fig 5: 4-bit Carry – Select Adder

A 16-bit carry-select adder with four individual circuits is look like as shown in Fig. 6. Each circuit consists of two 4-bits ripple-carry adders(Lecture notes, Concordia Univ.) (Desiree et al., 2013).

The delay of n-bit carry select adder based on an m-bit CLA blocks can be given by the following equation when using constant carry number blocks (Srivastava et al., 2014)

$T = t_{seup} + m\, t_{carry} + (n/m)\, t_{mux} + t_{sum}$

And by the following equation when using successively incremented carry number blocks respectively.

$T = t_{seup} + m\, t_{carry} + (2n)1/2\, t_{mux} + t_{sum}$

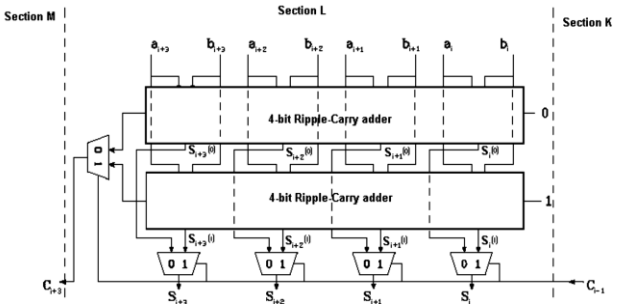

Fig. 6: One section of a larger Carry Select Adder

3.3 Parallel Prefix Adders

In parallel prefix adder the addition will be done faster when compared with other methods. The main disadvantage of prefix adder is the large fan-out and long interconnection wires which causes increase in delay. The large fan-out can be eliminated by increasing the number

of levels of cells and buffers are inserted to minimize the delay generated due to long interconnections (Uma et al., 2012)(Roy et al., 2013) (sreenath et al., 2012)(Lecture notes, Concordia Univ.)(Desiree et al., 2013).

a) Preprocessing - This step involves computation of generate and propagate signals corresponding to each pair of bits in A and B. These signals are given by the logic equation (Desiree et al., 2013)

$$p_i = A_i \, xor \, B_i$$
$$g_i = A_i \, xor \, B_i$$

b) CLA Network- This block differentiates KSA from other adders and is the main force behind its high performance. This step involves computation of carriers corresponding to each bit. It uses group of propagation and generation as intermediate signals which are given by the logic equations below (Desiree et al., 2013).

$$P_{i:j} = P_{i:k+1} \; and \; P_{k:j}$$
$$G_{i:j} = G_{i:k+1} \; or \; (P_{i:k+1} \; and \; G_{k:j})$$

c) Postprocessing- This is the final step and is common to all adders of this family. It involves computation of sum bits, sum bits are computed by the logic given below (Desiree et al., 2013)

$$s_i = p_i \, xor \, c_{i-1}$$

The production of the carries can be designed in many different ways. Some types of parallel prefix adders are Kogge-Stone Adder (KGA), Ladner Fischer Adder (LFA) and Bren-Kung Adder (BKA).

3.3.1Kogge-Stone Adder

It is most widely used design for fast addition and for high performance. Although it takes more area, it has very low fan-out which causes performance improvement. The KSA has minimum logic depth as shown in figure 7 (jyoshna et al., 2011)(sreenath et al., 2012)(Lecture notes, Concordia Univ.).

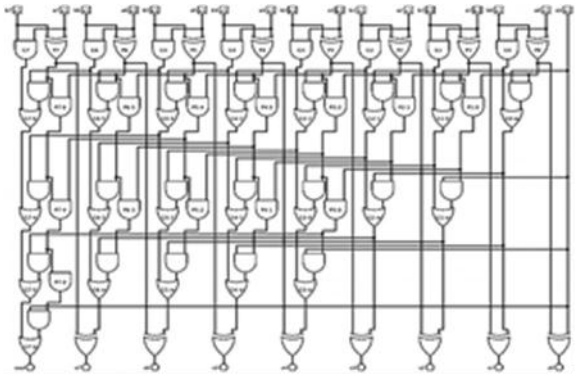

Fig.7: 8-Bit Kogge-Stone Adder

3.3.2Ladner-Fischer Adder

This adder requires less area when compared with KSA, but has large fan-out. The LFA structure is shown in figure 8 (sreenath et al., 2012)(Lecture notes, Concordia Univ.).

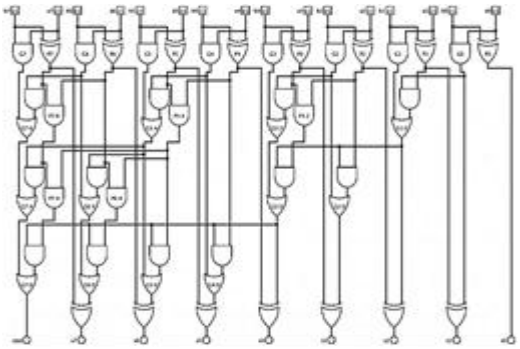

Fig. 8: 8-bit Ladner-Fischer Adder

3.3.3Brent-kung Adder

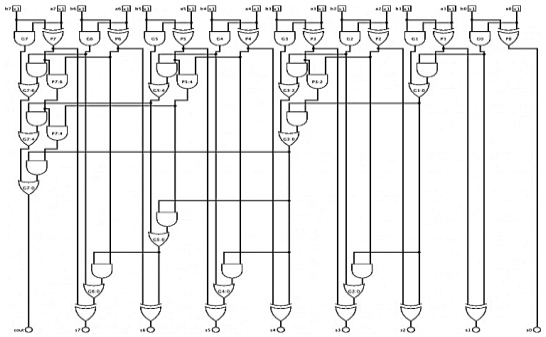

Fig. 9: 8-bit Brent-kung Adder

The Brent-Kung adder requires less area and minimum interconnecting wires than Kogge Stone adder. The structure of BKA is shown in figure 9 (sreenath et al., 2012)(Lecture notes, Concordia Univ.).

Table -1 Area & Delay Reports of 8x8 Multiplier with various Fast Adders

	CLA	CSLA	KGA	LFA	BKA
AREA	89 Slices	93 Slices	105 Slices	89 Slices	91 Slices
DELAY	29.981 n Sec	28.644 n Sec	31.788 n Sec	28.981 n Sec	29.944 n Sec

Conclusion

The 8x8 multiplier design was implemented by using various fast adders and synthesized using Xilinx Synthesis Tool (XST). The XST synthesis tool targeted the SPARTAN-3E FPGA Family. The synthesized parameters are derived in compatible with the targeted SPARTAN-3E FPGA Family. The Table-1 describes the Area & delay analysis reports of the 8x8 bit Vedic Multiplier with various fast adders. This analysis will help to find the suitable design for the desired application like Low Area, Low Power and High Performance designs.

References

Jagadguru Swami Sri Bharati Krsna TilihjiMotilal Maharaja. (1986), *Vedic Mathematics*, Motilal Banarsidas, Varanasi, India,. prof J M Rudagi, Vishwanath Ambli, Vishwanath

Munavalli, Ravindra paril, Vinay kumar sajjan (2011), Design And Implementation Of Efficient Multiplier Using Vedic Mathematics, *proc. Of Int. conf. on Advanced in Recent Technologies in Communication and computing*.

Nehru. K., Shanmugam, A. Vadivel, S. (2012), Design of 64-bit low power parallel prefix VLSI adder for high speed arithmetic circuits, *Int. conf. Computing, Communication and Applications (ICCCA)*.

Uma, R. Ponnian, (2012) Systolic FIR Filter Design with Various Parallel Prefix Adders in FPGA: Performance Analysis, *J.Int.symposium Electronic System Design (ISED).*

Roy. S, Choudhury, M. Puri, R.Pan, D.Z. (2013), Towards optimal performance-area trade-off in adders by synthesis of parallel prefix structures, *Design Automation Conference (DAC), 50th ACM / EDAC / IEEE*.

Hoe, D.H.K., Martinez, C. Vundavalli, S.J. (2011) Design and characterization of parallel prefix adders using FPGAs , *IEEE 43rd Southeastern Symposium on System*

Tiwari, H.D. Gankhuyag, G. Chan Mo Kim, Yong Beom Cho, (2008). Multiplier design based on ancient Indian Vedic Mathematics, *International SoC Design Conference, ISOCC '08.*

Kunchigi, V. Kulkarni, L. Kulkarni, (2012) , High speed and area efficient vedic multiplier, *S. International Conference on Devices, Circuits and Systems (ICDCS)*.

Bansal, Yogita, Madhu, Charu, Kaur, Pardeep (2014), High speed vedic multiplier designs-A review, *Recent Advances in Engineering and Computational Sciences (RAECS)*.

Mehta, P., Gawali, D. (2009) , Conventional versus Vedic Mathematical Method for Hardware Implementation of a Multiplier, *International Conference on Advances in Computing, Control, & Telecommunication Technologies, ACT '09*.

Fabio Frustaci, Marco Lanuzza, Paolo Zicari, Stefania Perri, Member, IEEE, and Pasquale Corsonello, (2009) Designing High-Speed Adders in Power-Constrained Environments, *IEEE Transactions on circuits and systems—ii: express briefs* , vol. 56, NO. 2, Feb.

G.Jyoshna P.Murali Krishna B.Doss, (2011), Parallel-Prefix Adder Architecture With Efficient Timing-Area Characteristic, *UACEE International journal of Advances in Electronics Engineering Vol:1 Issue:1 ISSN 2278 - 215X* .

V.Vamshi Krishna, et al., (2012), High Speed, Power and Area efficient Algorithms for ALU using Vedic Mathematics, *ijsrp*, vol2no7,july.

Karishma P D et al., (2014), Area Efficient Architecture for Convolution Using Vedic Mathematics, *IJSR*, Vol3 No 3, March .

Sreenath K, (2012), Reconfigurable VLSI architecture for FFT computation, *IJSER*, vol 3 no6, june.

Parallel Adder - Concordia University, http://users.encs.concordia.ca/~asim/COEN_6501/Lecture_ Notes /~WRL2849.tmp

Desiree JV,(2013) Fast and Area Efficient RSA Cryptosystem Design Using Modified Montgomery Multiplication for FPGA Applications, *IJSER*, vol4 no7, July.

Kerur et al., (2011), Implementation of Vedic Multiplier for Digital Signal Processing, ICVCI, *IJCA*.

Mohammed et al., (2013) Study, Implementation and Comparison of Different Multipliers based on Array, KCM and Vedic Mathematics Using EDA Tools, *IJSRP*, vol 3 no 6,June.

Priyanka et al., (2013) Design Tradeoff Analysis and Implementation of Digital Binary Adders Using Verilog, *IJSRP*, vol3 no 8, August .

Shrivastava et al., (2014), Design and implementation of low power high speed 32-bit hcsa, *IJVES*, vol 5 april.