ACSIJ Advances in Computer Science: an International Journal, Vol. 2, Issue 3, No.4, July 2013
ISSN : 2322-5157
www.ACSIJ.org

ACSIJ
WWW.ACSIJ.ORG

Efficient Area and Speed Optimized Multiplication Technique Using Vedic and Tree Addition Structure

Namrata Mishra[1] and Utsav Malviya[2]

[1] Department of Electronics and Communication
Rajiv Gandhi Technical University
Gyan Ganga Institute of Technology and Sciences
Jabalpur, M.P., India
Email-numrataamishrra06@gmail.com

[2] Departments of Electronics and Communication
Rajiv Gandhi Technical University
Gyan Ganga Institute of Technology and Sciences
Jabalpur, M.P., India
Email-utsavmalviya@yahoo.com

Abstract

Now days we are living in digital world, where all the operations get performed more reliably and with highest accuracy by digital signal processor. The multiplier is the key element of all these processor like Microprocessor, Microcontroller, DSP processor etc. After through study and deep analysis work we have seen that the existing Vedic multiplication hardware has some limitation in terms of area. To overcome these limitations a novel approach has been proposed to design the Vedic multiplier with unique addition structure, which is used to add partially generated products. To meet our major concern 'Speed' we need particular high speed multiplier, the speed of multiplier greatly depends upon the type of multiplication technique used in it. We have come up with the idea to merge two different multiplication techniques Vedic and Tree addition structure and these gives us a fast and area efficient multiplication approach.

Keywords- Digital Signal Processor (DSP), Arithmetic and Logical Unit (ALU), Multiply and Accumulate (MAC)

1. Introduction

ALU is the key element of Microprocessors, Microcontrollers and embedded systems. Multiplication is one of the prime requirements of any ALU designing & multiplication is known as MAC in modern Microprocessors & Microcontroller, the method which we choose for designing MAC will affect Microprocessor or Microcontroller performance.

1.1 Conventional Multiplication

- Multiply 32 with first RHS digit 4 = 128. And store it in register.
- Then multiply second RHS digit 4 with 32 = 128.
- Provide shifting of 1 digit = 1280.
- Add first partial product with second. And get result.

```
   32
 x 44
------
  128
 1280
------
 1408
```

1.2 Vedic Multiplication (Urdhva Triyambakam sutra)

The multiplier is based on an algorithm **Urdhva Triyambakam** (Vertical & Crosswise) of ancient Indian Vedic Mathematics. **Urdhva Triyambakam** Sutra is a general multiplication formula applicable to all cases of multiplication. It literally means "Vertically and crosswise".

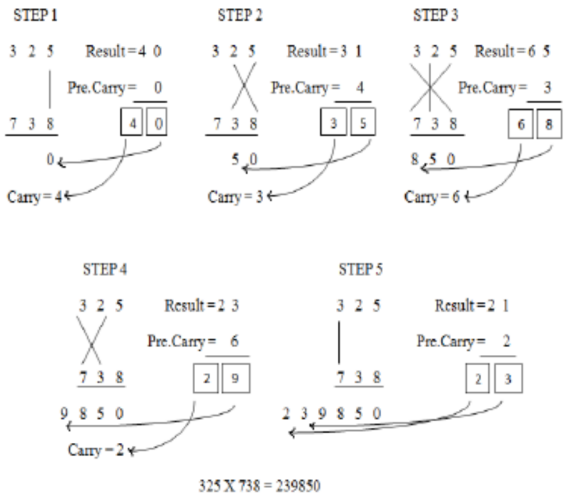

Fig.1- The method of Urdhva Triyambakam

Fig.1 shows multiplication of two decimal numbers- 325*738 by Urdhva Triyambakam method; to illustrate this multiplication scheme let us consider the multiplication of two decimal numbers (325 * 738). 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. The net advantage is that it reduces the need of microprocessors to operate at increasingly high clock frequencies.

2. Methodology

2.1 Structure of Design

To design Vedic multiplier we need to design major blocks of the complete application, we have design a 4 bit Vedic MAC which is actually our proposed research design module and then we have design 16 bit Vedic MAC using 4 bit Vedic MAC, after designing 16 bit MAC we have design 16 bit logical module to perform all logical operations. At the end we did integration of 16 bit MAC with 16 bit logical module.

2.1.1 The top module

Figure 2 shown below is the top module of our design structure tree; it has two components 16 bit MAC and 16 bit Logical block.

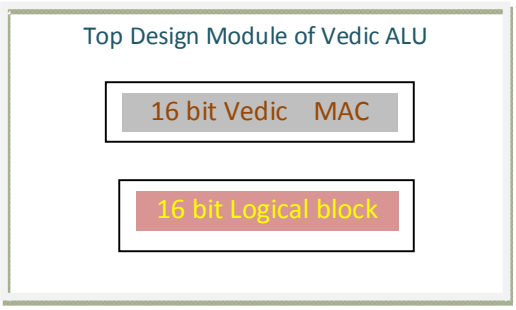

Fig. 2- Top Design Module of Vedic ALU

2.2 The 16 bit MAC

Figure 3 shown below is the design of our proposed 16 bit MAC it has used sixteen 4 bit Vedic MAC to perform computation on 16 bit input numbers.

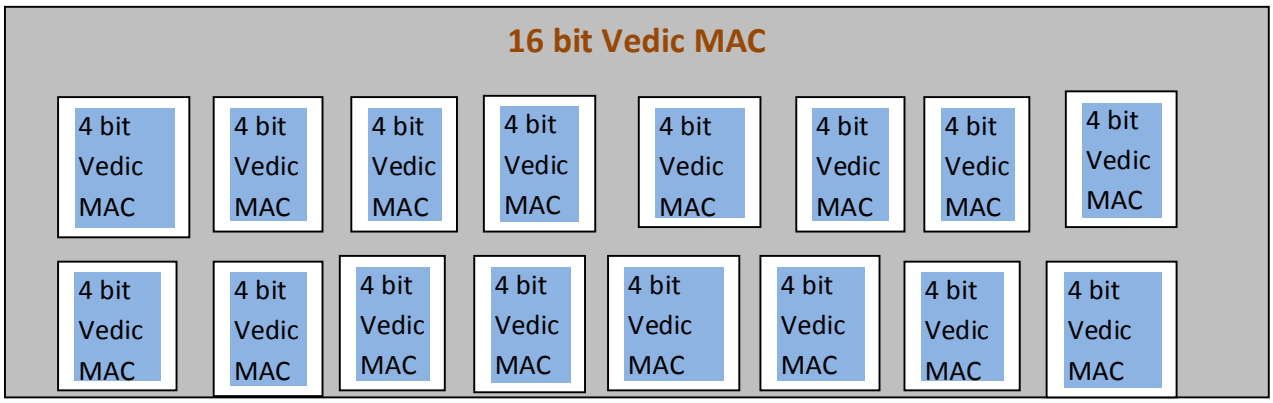

Fig. 3- 16 bit Vedic MAC

2.3 The 4 bit MAC

Figure 4 shown below is the process when we have 4 bit binary numbers (In1 & In2) and we need to perform multiplication of them.

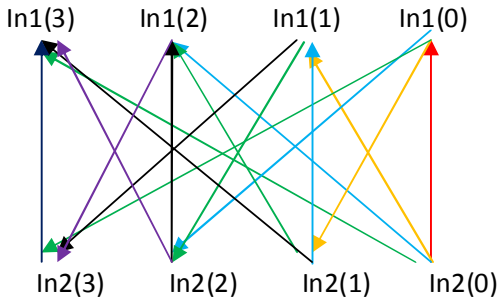

Fig.4- 4 bit Vedic MAC

In the program entity "vedicmau" is the 4 bit MAC (Multiply and accumulate), In1 & In2 are the 4 bit numbers, below is the explanation of 4 bit Vedic MAC.
In figure above every arrow is the logical 'AND' operation
The red produce "t1"
The yellow produce t2 & t3
The light blue produce t4 & t5 & t6
The green produce t7 & t8 & t9 & t10
The black produce t11 & t12 & t13
The purple produce t14 and t15
The dark blue produce t16
Figure 5 shown below is the approach which we have prepare for 4 bit Vedic MAC operation it has unique our proposed Vedic cum Tree addition structure.

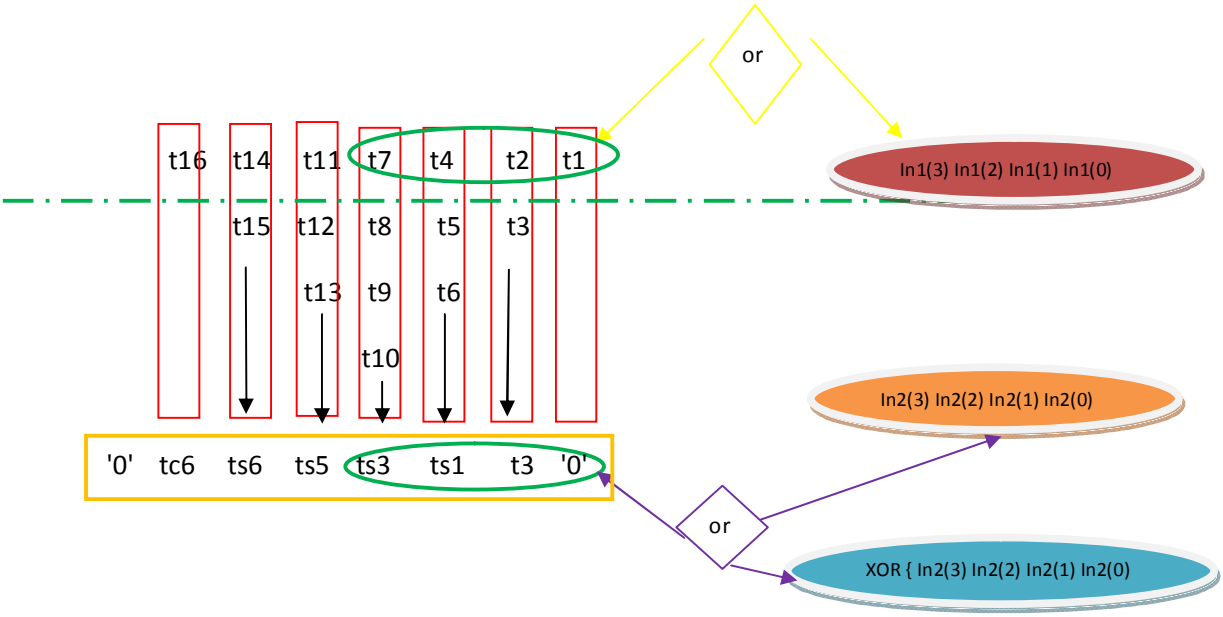

Fig.5- Vedic cum Tree Addition Structure

The addition of the above shown in figure above by red box produce result of Vedic multiplication. We also suppose to perform 4 bit addition and subtraction, so first we did addition of only which has shown below the green dashed line in above figure and produce result shown in orange box. Now we made choice between green circle or red circle above the green dashed line, we select red circle for addition & subtraction, we continue with green circle for multiplication.

2.4 The 16 bit Vedic multiplication using 4 bit Vedic

Now for 16 bit multiplication, we have 16 bit inputs 'x' & 'y' and ans. will be produce in z which will be of 32 bit. We perform "4 bit multiplication "on each arrow as shown in figure below.

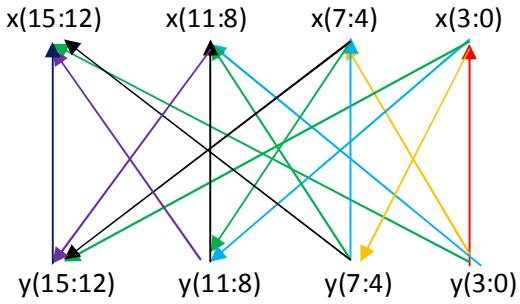

Fig.6- The 16 bit Vedic multiplication using 4 bit Vedic

In figure above:-

- Red arrow multiplication produces 8 bit ans. [om11:ol11]
- Yellow arrow multiplication produces 8 bit ans. [om21:ol21], [om22:ol22],
- Blue arrow multiplication produces 8 bit ans. [om31:ol31], [om32:ol32], [om33:ol33]
- Green arrow multiplication produce 8 bit ans. [om41:ol41], [om42:ol42], [om43:ol43] ,[om44:ol44]
- Black arrow multiplication produce 8 bit ans. [om51:ol51], [om52:ol52], [om53:ol53]
- Purple arrow multiplication produces 8 bit ans. [om61:ol61], [om62:ol62],
- Dark blue arrow multiplication produces 8 bit ans. [om71:ol71]

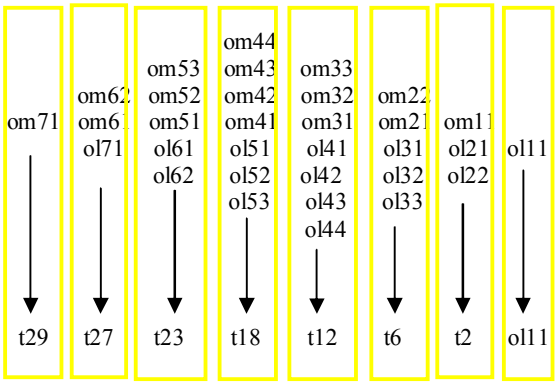

Fig.7- Tree Addition Approach

Figure 7 above is the tree addition approach. We have used this approach to produce result of multiplication. Each of the variables shown above is of 4 bit, the addition of yellow boxes shown above results the multiplication of 16 bit numbers and result is of total 32 bit.

3. Results

Table 1
Results of Vedic 16 bit ALU (the complete design)

No of Slices	501
No of 4 input LUT	874
No of bounded IOBs	68
Logical Delay	9.448 ns
Power	25 mw

Table 2
Results of proposed 4 bit Vedic multiplier (our proposed Vedic design)

No of Slices	19
No of 4 input LUT	33
No of bounded IOBs	16
Logical Delay	5.623 ns

Table 3
Comparative results of with 4 bit multiplier

	Logical Delay	Slice
Proposed Vedic 4 bit multiplier	5. 623 ns	19
[7]	6.216 ns	27
[8]	-	25
[3]	10.95 ns	20

Table 4
Results of proposed 8 bit Vedic multiplication

No of Slices	98
No of 4 input LUT	171
No of bounded IOBs	32
Logical Delay	6.655 ns

Table 5
Comparative results of with other 8 bit multipliers

	Logical Delay	Slice
Proposed Vedic 8 bit bit multiplier	6.655 ns	98
Ref[6]	36.563 ns	-
Ref[5]	15.685 ns	-
Ref[4]	15.484 ns	-
Ref[8]	15.518 ns	-

Table 6
Comparative results with base papers

Design	Sub design s(If any)	Logical Delay	Slice
Proposed Vedic 4 bit MAC		5.623 ns	19
[3] (4x4 MAC)		10.95 ns	20
Design	Sub design s(If any)	Logical Delay	Slice
Proposed Vedic 8 bit bit MAC		6.655 ns	98
[1] (8x8 bit multiplier)	Kartsuba	31.029 ns	-
	Vedic kartsuba	18.695 ns	-
	Optimised Vedic	15.418 ns	-
[2] (8x8 MAC)		48.80 ns	-
[3](8x8 MAC)		15.52 ns	92

Above we can observe that we are better in aspect of area (i.e. less number of Slice) and speed (i.e. logical delay) in 4 bit MAC/multiplication as compare to base 3 paper. Above we can observe that we are better in aspect of speed (i.e. logical delay) in 8 bit as compare to base1, base2 & base3 papers. We have design 4 bit Vedic multiplier (our actual research work) and use it to design 16 bit multiplier. So we should make comparison with 4 bit Vedic only.

References

[1]- M. Ramalatha, K. Deena Dayalan,, P. Dharani, S. Deborah Priya," ***High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques*** " ACTEA , *July 15-17, 2009.*

[2]- Anvesh Kumar,Ashish Raman,"***Low Power ALU Design by Ancient Mathematics***", IEEE, Volume 5, 2010.

[3]- Devika Jaina, Kabiraj Sethi, Rutuparna Panda, "***Vedic Mathematics Based Multiply Accumulate Unit***" International Conference On Computational Intelligence And Communication System, IEEE, 2011.

[4]- M. Ramalatha,, K. Deena Dayalan,, P. Dharani,,S. Deborah Priya, "***High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques***" ACTEA 2009.

[5]- Ramesh Pushpangana, Vineeth Sukumaran, Rino Innocent, Dinesh Salikumar, Vaisak Sundar,"***High Speed Vedic Multiplier for Digital Signal Processors***", IETE Journal for Research, Volume 55, ISSUE 6, Nov-Dec 2009.

[6]- S. S. Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur and Girish V A," ***Implementation of Vedic Multiplier for Digital Signal Processing***", International Conference on VLSI, Communication & Instrumentation (ICVCI) 2011 Proceedings published by International Journal of Computer Applications® (IJCA)

[7]- Umesh Akare, T. V. More, R. S. Lonkar," ***Performance Evaluation and Synthesis of Vedic Multiplier***", National Conference on Innovative Paradigms in Engineering & Technology (NCIPET-2012) Proceedings published by International Journal of Computer Applications® (IJCA).

[8]- G.Ganesh Kumar,V.Charishma,"**Design of High Speed Vedic Multiplier using Vedic Mathematics Techniques***", International Journal of Scientific and Research Publications, Volume2, Issue 3, March 2012.*