Efficacious Convolution and Deconvolution VLSI Architecture for Productiveness DSP Applications

[1]Dr.R.Kalaivani [2]Thamizharasan .V, [3] Renugadevi.K.S
[1] Associate Professor [2,3] Assistant Professor
[1,2,3] Department of Electronics and Communication Engineering
[1,2,3] Erode Sengunthar Engineering College, Perundurai, Erode.

Abstract— Current scenario in mobile Communication and multimedia applications require high-performance and low-power VLSI signal processing (DSP) systems. Most broadly used operations in DSP are FIR and IIR filter. The convolution and deconvolution with a very lengthy sequence is everywhere in wide area of application in Digital Signal Processing. Convolution used to compute the output of a system with arbitrary input, with information of impulse response of the system. Primary constriction of any application to work fast is that boost the speed of their basic building block. Both operations consume much of time. More number of techniques is developed to improve the speed of the Multiplier and Divider, The Vedic Multiplier and Divider technique is more considerable and satisfied all the constraints. Because, of faster working and low power consumption. The most considerable aspect of the proposed method is the growth of a multiplier and divider architecture based on Ancient Indian Vedic Mathematics sutras Urdhvatriyagbhyam and Nikhilam algorithm. In this paper the time delay of Convolution and Deconvolution is improved using Vedic multiplier and Divider. We proposed to design of linear convolution and circular convolution using Vedic mathematics is functionally verified and simulated using Modelsim software and implemented on Spartan 3E FPGA kit using Xilinx software, parameter like area, speed and power will be compared to their implementation using conventional multiplier & divider architectures.

Index Terms—VLSI digital signal processing ,Vedic Mathematics sutras Urdhvatriyagbhyam and Nikhilam algorithm, FIR filter, IIR filter, Spartan 3E FPGA, Xilinx software

I. Introduction

The three most extensive traditional metrics for computing the performance of a circuit parameters are power, delay and area. Minimizing the area,power and delay has at all times have been considered an important for improving the circuit performance. With the growing intensity of device integration and the enlargement in the complexity of micro-elctronic circuits, power efficiency has come to fore as a primary design goal while power efficiency has always been desirable in electronic circuits.

Advancement in now a days mobile communication and multimedia applications require high performance and low-power VSP systems. Multiplication is plays a vital role in digital and commercial processor. Most of the Mobile application in communication field need for a filter. In that Finite Impulse Response (FIR) is most suitable for mobile application because high speed . Convolution filter is most commonly called as FIR filter. since convolution is a fundamental element of FIR filter.

II. Convolution by direct method

Convolution is the most essential concept in a analysis of signal processing. By using convolution, we can build the output of system for any arbitrary input signal, if we know the impulse response of system. Convolution is a numerical way of value merging the two signals to produce a third signal. It is an integral that evaluate the value of overlapping one function with another shifted function.It therefore "mingles" one function with another. i.e., synthesis the imaging, the considered polluted map is a convolution of the "true" CLEAN map with the dirty beam (The Fourier transform of the sampling distribution). The convolution is a also called as, faltung ("folding").

The mathematical way of convolution in discrete time domain is

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k]$$

Where x[n] is a input signal for a DT system, h[n] is a impulse response for a DT system, and y[n] is output response of a system. * indicated by convolution. In that we multiply the terms of x[k] by the terms of a time-shifted version of h[n] and add each set of value.

In order to understand the meaning of convolution in signal processing, we are going to start from the concept of signal decomposition (impulse decomposition). Systems are described by a signal called the impulse response. The input signal is decomposed into simple preservative components, and the system response of the input signal results in by adding the output of these components passed through the system. For example, in Fourier series, any periodic signal (even rectangular pulse signal) can be represented by a sum of sine and cosine functions. But here, we use impulse (delta) functions for the basis signals, instead of sine and cosine.

Observe the subsequent example (Fig.1) how a signal is decomposed into a set of impulse (delta) functions.

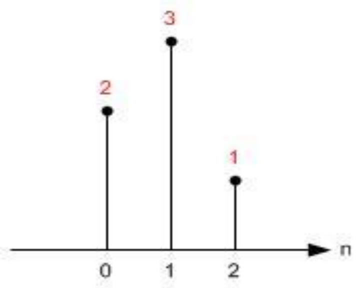


Fig.1.Input signal x[n]

Since the impulse function, δ[n] is 1 at n is 0, and zeros at n not equal to 0. *x*[0] can be written as 2·δ[n]. And, *x*[1] will be 3·δ[n-1], because

δ[n-1] is 1 at n=1 and zeros at others. In same way, we can write *x*[2] by shifting δ[n] by 2, *x*[2]

= 1·δ[n-2]. Therefore, the signal, *x*[n] can be represented by adding 3 shifted and scaled impulse functions.

In general, a signal can be written as sum of scale and shifted delta functions

$$x[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot \delta[n-k]$$

$$x[n] = x[0] \cdot \delta[n-0] + x[1] \cdot \delta[n-1] + x[2] \cdot \delta[n-2]$$

III. Deconvolution by direct method

The process of Deconvolution is the filtering of a signal to balance for an unpreserved convolution. The objective of deconvolution is to reconstruct the signal as it existed earlier than the convolution take place. This is frequently needed the characteristics of the convolution (i.e., the frequency or impulse response) to be recognized. This can be distinguished from partially sighted deconvolution, where the characteristics of the sponging convolution are *not* recognized. Partially sighted deconvolution is a greatly more difficult problem that does not have general solution, and the approach must be modified to the meticulous application. Deconvolution is extensively used in the field of signal and image processing. Since these techniques are in used in lot of scientific and engineering disciplines, & finds its many applications. In general, the purpose of deconvolution is to generate the solution of a convolution from the following equationRS= SS*BRS

Usually, RS is some recorded signal, and SS is some signal that we wish to recover, but has been convolved with some other signal BRS before we recorded it. The function BRS might represent the transfer function of an instrument or a driving force that was applied to a physical system. If we know BRS, or at least know the form of BRS, then we can carry out the procedure of deconvolution. However, if we do not know BRS in advance, then we want to approximate it. This is most often done using methods of statistical estimation.

IV. Multiplication by direct method

Multiplication of binary numbers is performed in the same way as with normal decimal numbers multiplication. The multiplicand is multiplied by each bit of the multiplier, starting from the LSB. The result of each such multiplication forms a partial product. Successive partial products are shifted one bit to the left. The product is obtained by adding these shifted partial products.

Consider an example(Fig.2.) of multiplication of two numbers, say A and B (2 bits each), C = A * B. Initially partial product is generated by multiplying the B_1B_0 by A_0. The multiplication of two bits such that A0 and B_0 generates a 1 if both bits are set to 1; otherwise it produces a 0 i.e. AND operation. So theV. partial products can be implementing with an AND gates. The next partial product is created by multiplying the B_0B_1 by A_1 and is moved by one bit position to the left.

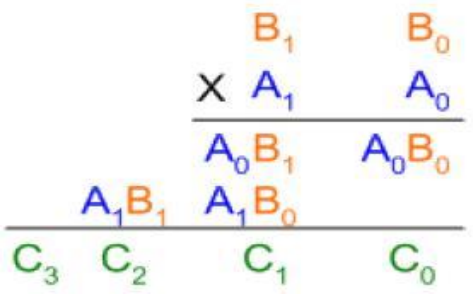


Fig.2.Basic Multiplication with partial products

The Fig.3. shows that two partial products are added with two half adders (HA). Usually there are more bits in the partial products, and then it will be necessary to use Full Adders.

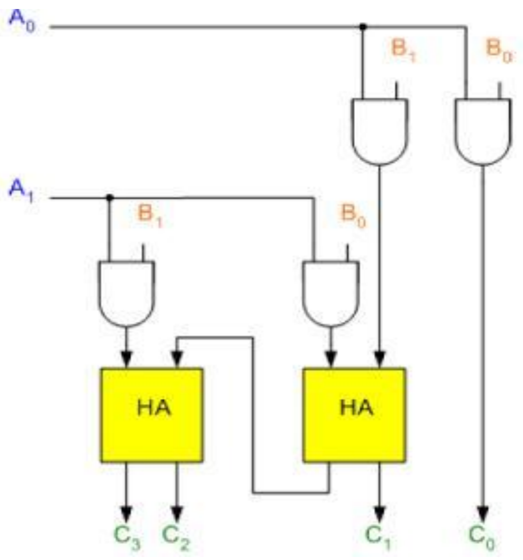


Fig.3. Architecture Array Multiplication

The least significant bit of the product does not have to go through an adder, since it is formed by the output of the first AND gate as shown in the Figure. Similarly A binary multiplier can be constructed with more no. of bits Multiplication using Vedic Multiplier

Multiplication is an essential role in arithmetic operations. Multiplication operations for example multiply and Accumulation (MAC), inner products are regularly used in the Arithmetic Functions & employed in several DSP application likely convolution, Fast Fourier Transform and filters. For any DSP applications latency and throughput are the most important parameter for speed of operation.

A new way to multiply the binary numbers are the vertically and crosswise technique Nikhilam Sutra and it is much more proficient in the multiplication of large numbers and it reduces the multiplication number of multiplication stage. This Sutra also illustrate that the efficiency of to reduce the N*N multiplier structure into a proficient 4*4 multiplier structures. The Architecture of Multiplier is Vertical and Crosswise algorithm of ancient Indian Vedic Mathematics. The vertically and crosswise multiplication technique was admired by Swami Bharati Krishna Tirthaji's *Vedic Mathematics*, published posthumously in 1965. Tirthaji declared to have restructured the mathematical content of sixteen *sutras* (concise rules) from an appendix to the Atharvaveda, one of the four Vedas, texts held sacred in Hinduism.the *Atharvaveda* dates to as early as 1000BCE. While his claim of the Vedic origins of Tirthaji's methods is questionable, the mathematics itself is quite cool.

What way to work this method: At every stage, you are calculating those partial products whose place value matches the place value of the digit you are computing.

Why the method is interesting: There are moderately few intermediary results to keep track .With a little practice, one can do all of the arithmetic in one's head, and when presented with a multiplication problem of any complexity proceed directly to writing down the digits of the answer.

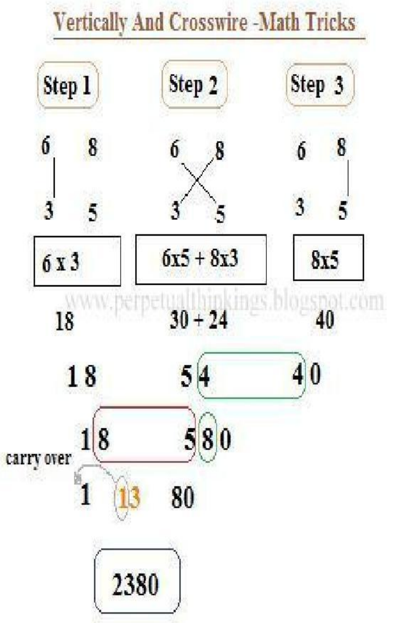
vedic multiplier is faster than array multiplier and Booth multiplier. As the number of bits increases from 8x8 bits to 16x16 bits, the timing delay is greatly reduced for Vedic multiplier as compared to other multipliers. Vedic multiplier has the greatest advantage as compared to other multipliers over gate delays and regularity of structures. Delay in Vedic multiplier for 16 x 16 bit number is 32ns while the delay in Booth and Array multiplier are 37 ns.

Advantages for the Architecture of Vedic multiplier based on speed specification is designed here for following criteria

Increase the Speed of the system

- To acquire good efficiency of the system
- Reduce the time delay as well as path delay in the multiplier
- The combinational path delay of Vedic multiplier obtained after compared with normal multipliers and found that the proposed Vedic multiplier

Example for vedic multiplication:


Vertically And Crosswire -Math Tricks
Step 1
Step 2
Step 3
6x3
6x5 + 8x3
8x5
www.perpetualthinkings.blogspot.com
18
30 + 24
40
carry over
2380

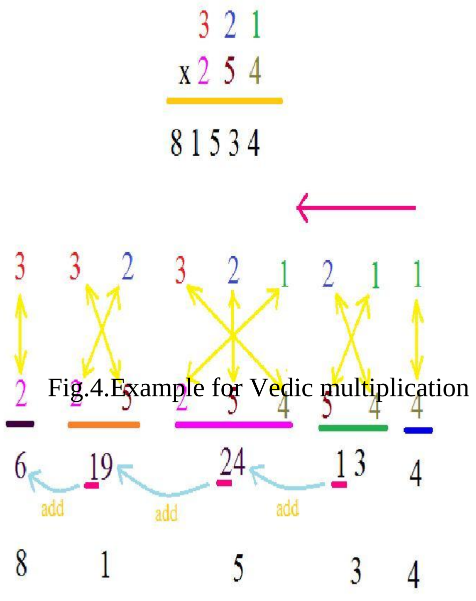


Fig.4.Example for Vedic multiplication

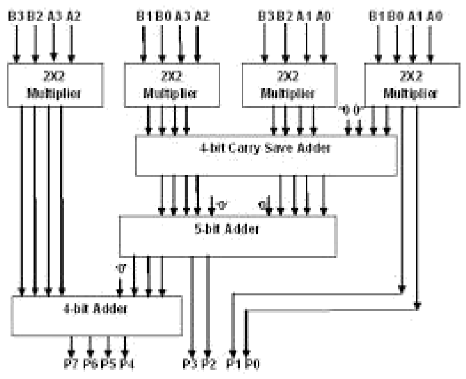
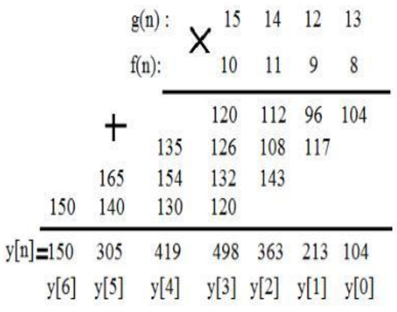


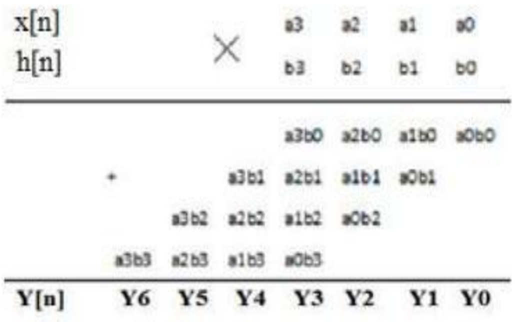
Fig.5.Architecture Vedic multiplication

VI. Convolution by Vedic method

The calculation procedure of the convolution is similar to decimal multiplication calculation, except carries are not passed to next column. This first example shows the simplicity of this method and how easily the calculation can be performed. As shown below, this method can be used to check intermediate values the convolution sum is computed using graphical convolution. Multiplication and summation for a given value of n, the summation is a product of the sequence f(k) and the folded and trans lated sequence g(n -k).



g(n):			×	15	14	12	13
f(n):				10	11	9	8
+				120	112	96	104
			135	126	108	117	
		165	154	132	143		
	150	140	130	120			
y[n]=	150	305	419	498	363	213	104
	y[6]	y[5]	y[4]	y[3]	y[2]	y[1]	y[0]



x[n]			×	a3	a2	a1	a0
h[n]				b3	b2	b1	b0
				a3b0	a2b0	a1b0	a0b0
	+		a3b1	a2b1	a1b1	a0b1	
		a3b2	a2b2	a1b2	a0b2		
	a3b3	a2b3	a1b3	a0b3			
Y[n]	Y6	Y5	Y4	Y3	Y2	Y1	Y0

Fig.6.Convolution partial products of Vedic multiplication

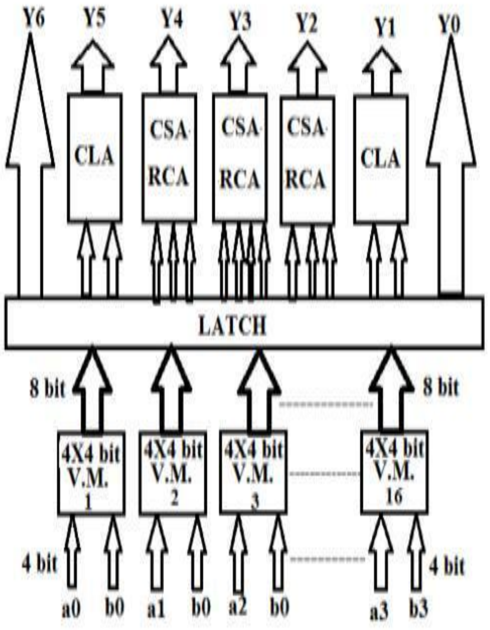


Fig.7.Architecture of convolution by Vedic Multiplier

VII. Deconvolution by Vedic method

Division operation is realized by using Nikhilam algorithm based on Vedic mathematics while to acquire partial products Vedic multiplier is used. For occurrence, the first example in subsection A may be reworked,solving for f(n) given g(n) and y(n).

```
               4  2  3  => f(n)
4 5 3 4 | 16 28 34 37 17 12
        | 16 20 12 16
        |--------------------
        |     8 22 21 17 12
        |     8 10  6  8
        |    ----------------
        |       12 15  9 12
        |       12 15  9 12
        |       -------------
        |        0  0  0  0
```

Fig.8.Deconvolution using Vedic Multiplication

The sequences are set up in a manner similar to long division but where no carries are performed out of a column.

```
8      2
1 2 |  3
0 2 |  8
----------
1 4 | 11
1 5 |  3
```

The Nikhilam sutra goes as follows: Nikhilam Navatascaramam Dasatah, literally meaning all from 9 and the last from 10. For example, Let us work out 123/8. The first line consists of the denominator followed by its 10's complement (2 is 8's 10's complement). The numerator has been divided by a "j" such that there are as many digits to the right of the "j" as there are digits in the denominator. We then put a zero under the first digit of the numerator. Now add up the digits in that column of the numerator to get a sum of 1 (1 + 0 = 1).

Multiply it by the 10's complement to get 2 (1 x

2 = 2). Put that under the second digit of the numerator. The sum of the digits under the second column is 4. Multiplying this by the 10's complement gives us 8. Put the 8 under the third digit of the numerator, right of the "j". Now we add up the numbers under the columns. Note that there is no carry over from the right of the "j" to the left of it. Following the rules on how to deal with a remainder greater than the denominator, we divide the remainder by the denominator and add the new quotient to the original quotient and retain the new remainder as the final remainder. This method is extended for other numbers. Nikhilam division algorithm just involves the addition of numbers which is very much different from the traditional division technique including multiplication of big numbers by the trial digit of the quotient at each step and subtracts that result from dividend at each step

VIII. Simulation Results

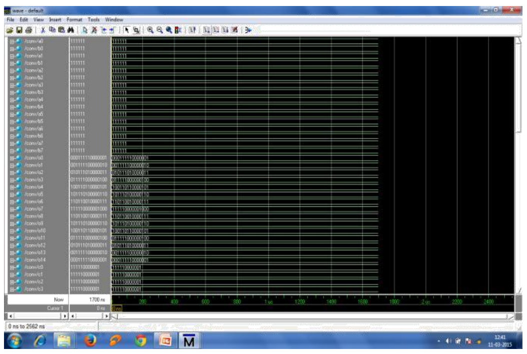
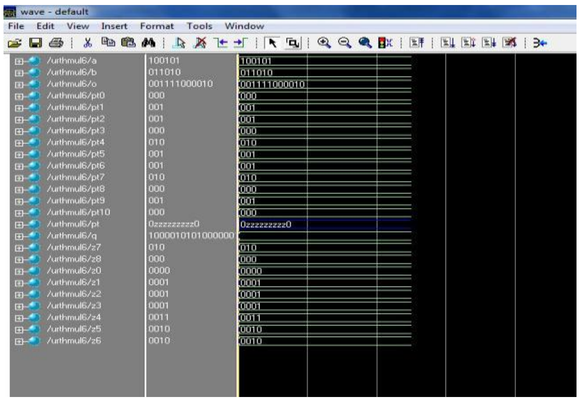


Fig. 9. Output Waveform of (6*6) Urdhva Triyagbhyam Algorithm

Fig.9 shows the simulation of Urdhva Triyagbhyam algorithm output waveform. When the input values are forced, the signal level corresponding to that values and their output are generated in their corresponding region



```
/urthmul6/a      100101
/urthmul6/b      011010
/urthmul6/o      001111000010
/urthmul6/pt0    000
/urthmul6/pt1    001
/urthmul6/pt2    001
/urthmul6/pt3    000
/urthmul6/pt4    010
/urthmul6/pt5    001
/urthmul6/pt6    001
/urthmul6/pt7    010
/urthmul6/pt8    000
/urthmul6/pt9    001
/urthmul6/pt10   000
/urthmul6/pt     0zzzzzzzzz0
/urthmul6/q      1000010101000000
/urthmul6/z7     010
/urthmul6/z8     000
/urthmul6/z0     0000
/urthmul6/z1     0001
/urthmul6/z2     0001
/urthmul6/z3     0001
/urthmul6/z4     0011
/urthmul6/z5     0010
/urthmul6/z6     0010
```

Fig.10. Output Waveform of Convolution using Urdhva Triyagbhyam

Fig.10 shows the simulation of convolution using Urdhva triyagbhyam algorithm. When the input values are forced, the signal level corresponding to that values and their output are generated in their corresponding regions. The following table shows the values forced in the above simulation.

IX. Implementation

The device utilization summary & timing report of the various Multipliers for shown in below table. The implementation is done by Spartan-3E FPGA Kit using Xilinx software.

S. No.	Methods	No. of Slices	No. of input LUTs	Speed
1	Normal Convolution	3564	6300	43.485ns
2	Urdhva Triyaghbhyam	3515	6214	37.561ns

Conclusion

In this paper we have proposed convolution and deconvolution using Vedic mathematics(Urdhava Tiryagbhyam) implemented in Spartan-3E FPGA kit. It is shown that the Urdhava Tiryagbhyam method is faster than the conventional method. The above synthesis report and timing report were generated using Xilinx software. Observation of the above synthesis report,the Vedic multiplier achieves higher speed by Reducing gate delay. Vedic Multiplier can be further optimized by implementing Carry Select Adder (CSA) so as to increase the speed. Most of the important DSP algorithms, such as convolution, discrete Fourier transforms, fast Fourier transforms, digital filters, etc, incorporate multiply-accumulate computations. Since the multiplication time is generally far greater than the addition time, the total processing time for any DSP algorithm primarily depends upon the number of multiplications. Hence, this multiplier can be used to implement the above DSP algorithms.

REFERENCE

1. S. S. Kerur, 1Prakash Narchi, 1Jayashree C N, 2Harish M Kittur and 3Girish V A "Implementation of Vedic Multiplier for Digital Signal Processing" International Conference on VLSI, Communication & Instrumentation (ICVCI) 2011 .
2. Shamim Akhter,"VHDL Implementation Of Fast NXN Multiplier Based On Vedic Mathematics", Jay pee Institute of Information Technology University, Noida, 201307 UP, INDIA, 2007 IEEE.
3. Charles. Roth Jr. "Digital Systems Design using VHDL," Thomson Brooks/Cole, 7th reprint, 2005
4. Rudagi, J. M., Vishwanath Ambli, Vishwanath Munavalli, Ravindra Patil, and Vinaykumar Sajjan. "Design and implementation of efficient Multiplier using Vedic mathematics." (2011): 162-166.
5. Vaidya, Summit, and Deepak Dandekar. "Delay-Power Performance Comparison of multipliers in VLSI circuit design." International Journal of Computer Networks & Communications (IJCNC) 2.4(2010): 47-56
6. Lomte, Rashmi K., and P.C. Bhaskar. "High Speed Convolution and Deconvolution Using Urdhva Triyagbhyam." VLSI (ISVLSI), 2011 IEEE Computer Society Annual Symposium on.
7. Itawadiya, Akhalesh K., et al."Design a DSP operations using Vedic Mathematics." Communications and Signal Processing (ICCSP), 2013 International Conference on. IEEE, 2013.
8. Khader Mohammad, Sos Agaian, "Efficient FPGA implementation of convolution", Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics,San Antonio, TX, USA - October 2009.
9. Hanumantharaju, M. C., et al. "A High Speed Block Convolution using Ancient Indian Vedic Mathematics." Conference onComputational Intelligence and Multimedia Applications, 2007. International Conference on. Vol. 2. IEEE, 2007.
10. Akhter, Shamim. "VHDL implementation of fast NxN multiplier based on vedic mathematic." In Circuit Theory and Design, 2007. ECCTD2007. 18th European Conference on, pp. 472-475. IEEE, 2007.
11. Ramesh Pushpangadan, Vineeth Sukumaran, Rino Innocent,Dinesh Sasikumar, Vaisak Sundarv, "High Speed Vedic Multiplier for Digital Signal Processors" IETE, VOL.55,page 282-286.