Speedy Deconvolution using Vedic Mathematics

Rashmi K. Lomte (Mrs.Rashmi R. Kulkarni), Prof.Bhaskar P.C

Abstract— Deconvolution is a computationally intensive digital signal processing (DSP) function widely used in applications such as imaging, wireless communication, and seismology. In this paper deconvolution of two finite length sequences (NXM)**,** is implemented using direct method to reduce deconvolution processing time. Vedic multiplier is used to achieve high speed. Urdhava Triyakbhyam algorithm of ancient Indian Vedic Mathematics is utilized to improve its efficiency. For division operation non-restoring algorithm is modified and used. The efficiency of the proposed convolution circuit is tested by embedding it on Spartan 3E FPGA. Simulation shows that ,the circuit has a delay of 79.595 ns from input to output using 90nm process library. It also provides the necessary modularity, expandability, and regularity to form different deconvolutions for any number of bits.

Index Terms— Deconvolution, Non-Restoring algorithm, Urdhva Tiryagbhyam

1 INTRODUCTION

THE concept of deconvolution is widely used in the techniques of signal processing and image processing. The concept of deconvolution has applications in reflection seismology, in reversing the optical distortion, to sharpen images etc. Faster additions, multiplications and divisions are of extreme importance in DSP for deconvolution. Speeding up deconvolution using a Hardware Description Language for design entry not only increases (improves) the level of abstraction, but also opens new possibilities for using programmable devices.

In this paper, a novel method for computing the linear deconvolution of two finite length sequences is used. Method is explained in detail in [1]. This method is similar to computing long-hand division and polynomial division.

As a need of project, all required possible adders are studied. All these adders are synthesized using Xilinx9.2i. There delays and areas are compared. Adders which have highest speed and comparatively less area occupied, are selected for implementing deconvolution. Since 4×4 bit multiplier is need of this project, different 4×4 bit multipliers are studied and Urdhava Triyakbhyam algorithm which gives lowest delay among remaining all multipliers is used here. For division, different division algorithms are studied, by comparing drawbacks and advantages of each algorithm, Non restoring algorithm is modified according to need and then used.

This paper can be considered as extension of [2]. where discrete linear convolution of two finite length sequences(4 ×4) is implemented. That convolved output of [2]. is input to this proposed design, impulse response of system is known is another input, this paper proposes design that carry out high speed deconvolution and extracts input samples.

Paper is organized as follows: section 2 gives brief introduction of novel method for deconvolution. Section 3 describes division algorithm. Section 4 discusses the Vedic mathematics and Urdhva Tiryagbhyam algorithm for multiplication. Section 5 presents selection of speedy adder. In section 6 design verification is given. Finally, the conclusion is obtained.

2 NOVEL METHOD FOR CALCULATING DECONVOLUTION

In general, the object of deconvolution is to find the solution of a convolution equation of the form:

$$f * g = h \qquad (1)$$

Usually, *h* is some recorded signal, and *f* is some signal that wish to recover, but has been convolved with some other signal *g* before get recorded. The function *g* might represent the transfer function of an instrument or a driving force that was applied to a physical system.If one know g or at least form of g,then one can perform deterministic deconvolution.

If the two sequences f(n) and g (n) are causal, then the convolution sum is:

$$h(n) = \sum_{k=0}^{n} f(k)\, g(n-k), \qquad n \geq 0 \quad (2)$$

Therefore, solving for *f(n)* given *g(n)* and y(n) results in

$$f(n) = \frac{h(n) - \sum_{k=0}^{n-1} f(k)\, g(n-k)}{g(0)}, \qquad n \geq 1 \quad (3)$$

where

$$f(0) = \frac{h(0)}{g(0)} \qquad (4)$$

where solution requires that $g(0) \neq 0$

This recursion can be carried out in a manner similar to long division. Lets take example ,let h[n] = [16 36 56 17 28 12] and g[n] = [4 4 3 2] , solving for *f(n)* given *g(n)* and h(n). The sequences are set up in a fashion similar to long division, as shown below, but where no carries are performed out of a column.

```
                 4     5      6    => f(n)
4 4 3 2 |   16   36   56   47   28   12
            16   16   12    8
                 20   44   39   28
                 20   20   15   10
                      24   24   18   12
                      24   24   18   12
                                     0
```

fig.1. Deconvolution by novel method

By observing figure 1. one can easily predict, for implementing speedy deconvolution by above method, divisor , multiplier and adder(to achieve subtraction in form of addition) used in design must be speedy.

High speed division can be carried out by selecting proper division algorithm.Vedic multiplier is used to get speedy multiplication.

3 ALGORITHM SELECTION FOR DIVISION

Division is most complex and time consuming operation for DSP.Basically division algorithm is classified as multiplicative and subtractive approaches. Multiplicative division algorithms do not compute the quotient directly, but use successive approximations to converge to the quotient. Normally, such algorithms only yield a quotient, but with an additional step the final remainder can be computed, if needed. Computations include several multiplications each iteration. The Goldschmidt division algorithm, binary search division algorithm, Newton-Raphson algorithm all these algorithms work by calculating estimates. Drawback of these algorithms is one has to compromise with exact result. Different errors need to be considered to calculate result. Though required number of iterations are less, steps required per iteration are more. The fastest implementation of a division function is a table lookup. A lookup-table implementation introduces a latency of just one clock cycle, but requires a table-size that grows exponentially with the input data width. another method is shift-and-subtract division algorithm.The algorithm is based on the well-known paper-and-pencil method of shifting the divisor from left to right of the dividend, and subtracting it when it is smaller then the remainder. Such an implementation scales linearly with the input data width, but requires a number of iterations equal to the input data width.

Hence here we concentrate ourselves to the subtractive approach. In the subtractive idea the algorithm is the same as the division methods that we were taught in the elementary school. Suppose that there are two n-digit numbers, X and D, which represent the dividend and divisor respectively. By the division operation we can find a n-digit quotient and a n-digit remainder denoted as Q and R respectively. The mathematical representations of X, D, Q, and R are as following .[3].

$$R^{(j+1)} = r \times R^{j} - q_{j+1} \times D \qquad (5)$$

Where $j = 0, 1, 2, \ldots, n-1$ is the iteration number.

R j is the partial remainder at iteration j.

r is the radix number.

q_{j+1} is the $j+1$th digit of the quotient

The final quotient is represented as

$$Q = q_1 q_2 q_3 \ldots\ldots q_n$$

we use radix-2, i.e., r=2, for design. Therefore,equation (5) can be rewritten and represented in equation (6) as follows [3].

$$R^{(j+1)} = 2 \times R^{j} - q_{j+1} \times D \qquad (6)$$

In the hardware design we have to check the subtraction at each step to decide the quotient in that digit. There are two ways to find the quotient of the current digit. One is the restoring method, and the other is the non-restoring method.In the restoring approach, when the current partial remainder, R^{j+1} , is positive, the current quotient bit is equal to 1. On the other hand, if the current partial remainder is less than 0, then the current quotient bit is set to 0, and then the partial remainder should be added with the divisor and restore back to the previous partial remainder, R^{j} and it is so called the “restoring” method. In the non-restoring method,if the current partial remainder, $R^{(j+1)}$, is positive,the current quotient bit is equal to 1. On the other hand, if the current partial remainder is less than 0, then the current quotient bit is set to -1, and at the next step we have to add the divisor to the current partial remainder, $R^{(j+1)}$, to form the next partial remainder, $R^{(j+2)}$. By this method, there is no need to add divisor to restore the previous partial remainder.However, the quotient in the non-restoring scheme is represented in the signed bit (digit) format.Therefore, after we finish the division process, the non-restoring method needs an additional step to convert the signed bit format to the binary number representation. Since we do not need to check the polarity of the partial remainder to do the restoring of the partial remainder. Therefore, the speed of the non-restoring division algorithm is faster than the speed of the restoring division algorithm [3],[4].

Here asper need of task,non restoring division algorithm is modified.Here dividend is 8 bit long number and divisor is number represented by 4 bits .Hence quotient would have maximum length of 4bit.

Table 1. Show delay and area utilized by divisor prepared by using non restoring division algorithm.Xilinx 9.2i is used for synthesis and simulation.Family Spartan 3E(90 nm process technology) is selected,with speed grade -5.

TABLE 1.
SIMULATION OF 8BIT BY 4BIT DIVISION USING NON RESTORING ALGORITHM

Area			Delay
Slices	LUTs	IOBs	ns
39	70	16	17.911

4 VEDIC MULTIPLIER

A multiplier is one of the key hardware blocks in convolution system. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following- high speed, low power consumption, regularity of layout and hence less area or even combination of them in multiplier. . However area and speed are two conflicting constraints. So improving speed results always in larger areas. So here in this paper attempt is to find out the best trade off solution among the both of them.

Depending upon the arrangement of the components, there are different types of multipliers available. A Particular multiplier architecture is chosen based on the application. Two most common multiplication algorithms followed in the digital hardware are array multiplication algorithm and Booth multiplication algorithm. The computation time taken by the array multiplier is comparatively less because the partial products are calculated independently in parallel. The delay associated with the array multiplier is the time taken by the signals to propagate through the gates that form the multiplication array. After comparison of 4 bit array multiplier and 4 bit Booth multiplier it is found that array multiplier is superior than Booth. [2].

Among all these multipliers, this paper proposes to use the multiplier based on an algorithm Urdhva Tiryagbhyam (Vertical & Crosswise) of ancient Indian Vedic Mathematics. Vedic mathematics is part of four Vedas (books of wisdom). It is part of Sthapatya- Veda (book on civil engineering and architecture), which is an upa-veda (supplement) of Atharva veda. His Holiness Jagadguru Shankaracharya Bharati Krishna Teerthaji Maharaja (1884-1960) comprised all this work together and gave its mathematical explanation while discussing it for various applications. Swamiji constructed 16 sutras (formulae) and 16 Upa sutras (sub formulae) after extensive research in Atharva Veda. Urdhva Tiryagbhyam Sutra is a general multiplication formula applicable to all cases of multiplication. It literally means “Vertically and crosswise”. It is based on a novel concept through which the generation of all partial products can be done and then, concurrent addition of these partial products can be done. Thus parallelism in generation of partial products and their summation is obtained using Urdhava Tiryagbhyam. 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.

This design need 4×4 bit multilplier.As per.[1], 4×4 bit Array multiplier is fastest among Booth, Wallance tree etc. Here we proposed to use Vedic multiplier based on Urdhava Triyakbhyam algorithm as we found it is fastest of all multipliers. Comparison between Array and Vedic is given in table2

TABLE 2.
SIMULATION OF 4×4 MULTIPLIERS FOR DELAY COMPARISION

Multiplier	Vedic	Array
Spartan 3E	11.676 ns	13.773 ns
Virtex 5	6.036 ns	6.2 ns

.

5 ADDER SLECTION

Choice of speedy and area saving adder is done by implementing and comparing 8 bit carry look ahead adder performance with 8 bit ripple carry adder, below in table 3.It shows delay and area utilized by these adders, for family Spartan 3E(90 nm process technology) and with speed grade -5.

TABLE 3.
SIMULATION OF DIFFERENT ADDERS FOR ADDITION OF TWO NUMBERS, EACH ONE IS 8 BIT LONG

Adder name	Area			Delay
	Slices	LUTs	IOBs	ns
Carry look ahead	9	15	25	12.025
Ripple carry	9	15	25	12.675

After observing results of comparisons, for two 8bit numbers addition carry look ahead adder is selected.

6 DESIGN VERIFICATION OF DECONVOLVER

Verification is carried out by ISE simulator. (Simulator results are shown in figure 2.RTL results are shown in figure 3, of another attached file named ‘Results’.)We used these two arrays with samples:
First input array g[n]: [F_h A_h B_h 9_h]
Second input array h[n]: [$B4_h$ $F0_h$ $13D_h$ $1A0_h$ $F9_h$ AD_h $5A_h$]
Deconvolved output : [C 8 7 A]

Area utilization summary is given below.

Cell Usage :

# BELS	: 605
# LUT2	: 27
# LUT3	: 142
# LUT4	: 397
# MUXF5	: 39
# IO Buffers	: 52
# IBUF	: 36
# OBUF	: 16

7 CONCLUSION

In this paper, we presented an optimized implementation of deconvolution. This particular model has the advantage of being fine tuned for signal processing. To accurately analyze our proposed system, we have coded our design using the VHDL hardware description language and have synthesized it for FPGA products using ISE. The proposed circuit uses less area and less power, and it takes 79.595 ns to complete on 90 nm process technology devices.Thus aim of high speed deconvolver implementation is achieved.

REFERENCES

[1] John W. Pierre, “A Novel Method for Calculating the Convolution Sum of Two Finite Length Sequences”, IEEE transaction on education, VOL.39, NO. 1, 1996.

[2] 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.

[3] Koren,I. Computer Arithemetic Algorithms,Prentice-Hall Inc(1993)

[4] Hwang,K. Computer Arithmethmetic Principles,Arichitecture,and design,John Wiley & Sons.

[5] Ramesh Pushpangadan, Vineeth Sukumaran, Rino Innocent, Dinesh Sasikumar, Vaisak Sundarv, “High Speed Vedic Multiplier for Digital Signal Processors” IETE, VOL.55,page 282-286