MANTECH Publications

Journal of Power Electronics and Drives
Volume 3, Issue 1, January-April 2018

Efficient Lut Based Filter Design Using Approximate Algorithm

N P Raju Mandapaka[1], Venkata Satish Dhulipudi[2]

Design Engineer in Space Systems[1], Assistant Professor[2]

Department of ECE[2]

Ananth Technologies Limited, Hyderabad[1], Prasiddha

College Of Engineering and Technology[2]

***Corresponding Authors' email id**: mnprawinraj@gmail.com[1], vsatish.dhulipudi@gmail.com[2]*

Abstract

The filter design optimization (FDO) problem is defined as finding a set of filter coefficients that yields a filter design with minimum complexity, satisfying the filter constraints. It has received a tremendous interest due to the widespread application of filters. Assuming that the coefficient multiplications in the filter design are realized under a shift-adds architecture, the complexity is generally defined in terms of the total number of adders and subtractors. In this paper, i present an exact FDO is enhanced by using Look UP Table approach. APC and OMS binary properties are used for implementing LUT based FIR filter by using APC And OMC reduced the number of storage elements almost 50%. This method has better performance and it consumes very less power and occupy less space compare to FDO method

Keywords: *Anti Symmetric Product Coding (APC), Odd multiple Storage (OMS), Look up Table (LUT), Finite Impulse Response (FIR).*

INTRODUCTION

Filter is a frequency selective network. It passes a band of frequencies while attenuating the others. Filters are classified as analog and digital depending on nature of inputs and outputs. Filters are further classified as finite impulse response and infinite impulse response filters depending on impulse response. This chapter gives a brief about the types of filters.

Digital filters are used extensively in all areas of electronic industry. This is

because digital filters have the potential to attain much better signal to noise ratios than analog filters and at each intermediate stage the analog filter adds more noise to the signal, the digital filter performs noiseless mathematical operations at each intermediate step in the transform. The digital filters have emerged as a strong option for removing noise, shaping spectrum, and minimizing inter-symbol interference in communication architectures. These filters have become popular because their precise reproducibility allows design engineers to achieve performance levels that are difficult to obtain with analog filters

Digital Filters can be constructed from 3 fundamental mathematical operations.

- Addition (or subtraction)
- Multiplication (normally of a signal by a constant)
- Time Delay i.e. delaying a digital signal by one or more sample periods

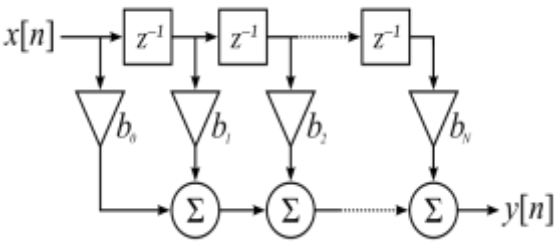

Figure 1: Block diagram of a Simple Digital Filter

Figure 1 shows a graphical means of describing a digital filter whereby the behavior of the filter is described by using the mathematical operations mentioned above.

The Impulse Response of a digital filter, h(n) is the response of the filter to an input consisting of the unit impulse function, δ(n). If the impulse response of a system is known, it is possible to calculate the system response for any input sequence x (n). By definition, the unit impulse is applied to a system at sample index n=0. So, the impulse response is non-zero only for values of n greater than or equal to zero i.e. h (n) is zero for n<0. This impulse response is said to be causal otherwise the system would be producing a response before an input has been applied.

It is known from the time-invariance property of a Linear Time Invariant System that the response of a system to a delayed unit impulse δ (n-k) will be a delayed version of the unit impulse, i.e. h (n-k). It is also known from the linearity property that the response of a system to a weighted sum of inputs will be a weighted sum of responses of the system to each of the individual inputs. Therefore, the response of a system to an arbitrary input x (n) can be written as follows:

$$y(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k)$$

Fast multipliers are essential parts of digital signal processing systems. The speed of multiply operation is of great importance in digital signal processing as well as in the general purpose processors today, especially since the media processing took off. In the past multiplication was generally implemented via a sequence of addition, Subtraction, and shift operations. Multiplication can be considered as a series of repeated additions. The number to be added is the multiplicand, the number of times that it is added is the multiplier, and the result is the product. Each step of addition generates a partial product. In most computers, the operand usually contains the same number of bits. When the operands are interpreted as integers, the product is generally twice the length of operands in order to preserve the information content. This repeated addition method that is suggested by the arithmetic definition is slow that it is almost always replaced by an algorithm that makes use of positional representation. It is possible to decompose multipliers into two parts. The first part is dedicated to the generation of partial products, and the second one collects and adds them.

The basic multiplication principle is twofold, i.e. evaluation of partial products and accumulation of the shifted partial products. It is performed by the successive Addition's of the columns of the shifted partial product matrix. The 'multiplier' is successfully shifted and gates the appropriate bit of the 'multiplicand'. The delayed, gated instance of the multiplicand must all be in the same column of the shifted partial product matrix. They are then added to form the product bit for the particular form. Multiplication is therefore a multi operand operation. To extend the multiplication to both signed and unsigned numbers, a convenient number system would be the representation of numbers in two's complement format. Furthermore, it is generally the most area consuming. Hence, optimizing the speed and area of the multiplier is a major design issue. However, area and speed are usually conflicting constraints so that improving speed results mostly in larger areas. As a result, whole spectrums of multipliers with different area-speed constraints are designed with fully parallel processing. In between are digit serial multipliers where single digits consisting of several bits are operated on. These multipliers have moderate performance in both speed and area. However, existing digit serial multipliers have been plagued by

complicated switching systems and/or irregularities in design. Radix 2^n multipliers which operate on digits in a parallel fashion instead of bits bring the pipelining to the digit level and avoid most of the above problems. They were introduced by M. K. Ibrahim in 1993. These structures are iterative and modular. The pipelining done at the digit level brings the benefit of constant operation speed irrespective of the size of' the multiplier. The clock speed is only determined by the digit size which is already fixed before the design is implemented.

Y[n]=b0x[n]+b1x[n-1]+b2x[n-2]+b3x[n-3]

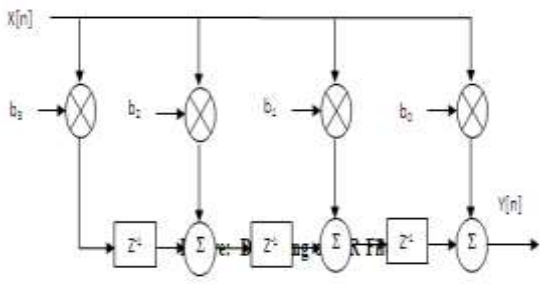

Figure 2: FIR multiplier

MULTIPLIERLESS DESIGN:

ECHO-A AND ECHO-D:

To realize the MCM block of the transposed form with the minimum number of adder-steps, in SIREN and NAIAD, we respectively used the modified versions of the approximate algorithms of [9] and [3] that can handle the delay constraint. Whenever a set of fixed point filter coefficients is determined in SIREN and NAIAD, the minimum adder-steps of coefficients is computed as given in Section II-B-1 and it is given to the algorithms of and as a delay constraint. In order to target the direct form of the FIR filter, in SIREN and NAIAD, ECHO-A is used to compute the smallest number of operations in the CAVM block and ECHOD is used for the design of the CAVM block with a small number of adder-steps. Note that in direct form filters, the total number of operations in the filter, i.e., TA, is determined by the solution of ECHO-A or ECHO-D on the set of filter coefficients. The proposed methods can target different filter constraints. For example, when the lower and upper bounds of , and , in (4) are set to 1, the filter constraints of are aimed. Setting and respectively to 0.7 and 1.4 corresponds to the 3 Db gain tolerance in the filter design.

The proposed algorithms can also target asymmetric filters taking into account the related filter constraints. The proposed algorithms can target the optimization of the gate-level area of the filter design. In this case, whenever a set of coefficients is

found, an algorithm, that can find the shift-adds design of the multiplier block of the filter occupying minimum area, should be used. In the transposed form filter, the size of registers and adders in the register add block should also be considered.

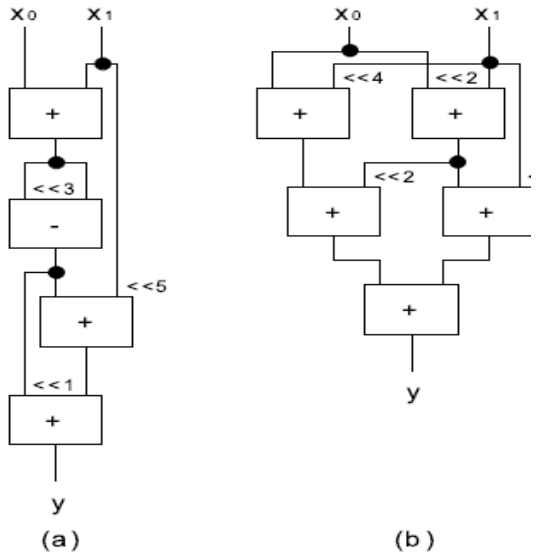

Figure 3: Multiplier less realization of constant multiplications using the DBR technique

LUT BASED FILTERS DESIGN:

A new approach to LUT design is presented, where only the odd multiples of the fixed coefficient are required to be stored, which is referred to as the OMS. In addition, by the APC approach, the LUT size can also be reduced to half, where the product words are recoded as anti-symmetric pairs.

The APC approach, although providing a reduction in LUT size by a factor of two, incorporates substantial overhead of area and time to perform the two's complement operation of LUT output for sign modification and that of the input operand for input mapping. However, it is found that when the APC approach is combined with the OMS technique the two's complement operations could be very much simplified since the input address and LUT output could always be transformed into odd integers.1

However, the OMS technique cannot be combined with the APC scheme, since the APC words generated according to are odd numbers. Moreover, the OMS scheme in does not provide an efficient implementation when combined with the APC technique. In this brief, a different form of APC is presented combined with a modified form of the OMS scheme for efficient memory based multiplication.

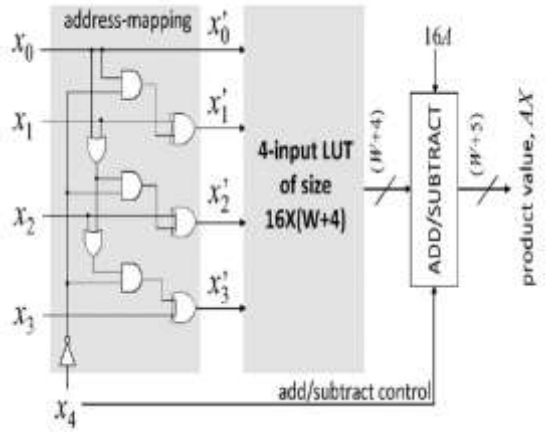

Figure 4: Proposed APC Part

APC WORDS FOR DIFFERENT INPUT VALUES FOR $L = 5$

Input, X	product values	Input, X	product values	address $x'_3x'_2x'_1x'_0$	APC words
0 0 0 0 1	A	1 1 1 1 1	$31A$	1 1 1 1	$15A$
0 0 0 1 0	$2A$	1 1 1 1 0	$30A$	1 1 1 0	$14A$
0 0 0 1 1	$3A$	1 1 1 0 1	$29A$	1 1 0 1	$13A$
0 0 1 0 0	$4A$	1 1 1 0 0	$28A$	1 1 0 0	$12A$
0 0 1 0 1	$5A$	1 1 0 1 1	$27A$	1 0 1 1	$11A$
0 0 1 1 0	$6A$	1 1 0 1 0	$26A$	1 0 1 0	$10A$
0 0 1 1 1	$7A$	1 1 0 0 1	$25A$	1 0 0 1	$9A$
0 1 0 0 0	$8A$	1 1 0 0 0	$24A$	1 0 0 0	$8A$
0 1 0 0 1	$9A$	1 0 1 1 1	$23A$	0 1 1 1	$7A$
0 1 0 1 0	$10A$	1 0 1 1 0	$22A$	0 1 1 0	$6A$
0 1 0 1 1	$11A$	1 0 1 0 1	$21A$	0 1 0 1	$5A$
0 1 1 0 0	$12A$	1 0 1 0 0	$20A$	0 1 0 0	$4A$
0 1 1 0 1	$13A$	1 0 0 1 1	$19A$	0 0 1 1	$3A$
0 1 1 1 0	$14A$	1 0 0 1 0	$18A$	0 0 1 0	$2A$
0 1 1 1 1	$15A$	1 0 0 0 1	$17A$	0 0 0 1	A
1 0 0 0 0	$16A$	1 0 0 0 0	$16A$	0 0 0 0	0

For $X = (0\ 0\ 0\ 0\ 0)$, the encoded word to be stored is $16A$.

Figure 5: Stored APC Words

16 x (W+4) → 16 Locations and each location having (W+4) bits

Let the product values on the second and fourth columns of a row be u and v, respectively. Since one can write

u = [(u + v)/2 − (v − u)/2] and

v = [(u + v)/2 + (v − u)/2], for (u + v) = 32A,

U=16A+ [(V-U)/2]

V=16A-[(V-U)/2]

The product values on the second and fourth columns of above figure therefore have a negative mirror symmetry.

This behavior of the product words can be used to reduce the LUT size, where, instead of storing u and v, only [(v − u)/2] is stored for a pair of input on a given row. The 4-bit LUT addresses and corresponding coded words are listed on the fifth and sixth columns of the table, respectively. Since the representation of the product is derived from the antisymmetric behavior of the products, we can name it as antisymmetric product code. The 4-bit address X_ = (x_3,x_2,x_1,x_0) of the APC word is given by

$$X' \begin{cases} X_i, \text{if } x4=1 \\ X'_i, \text{if } x4=0 \end{cases}$$

RESULTS

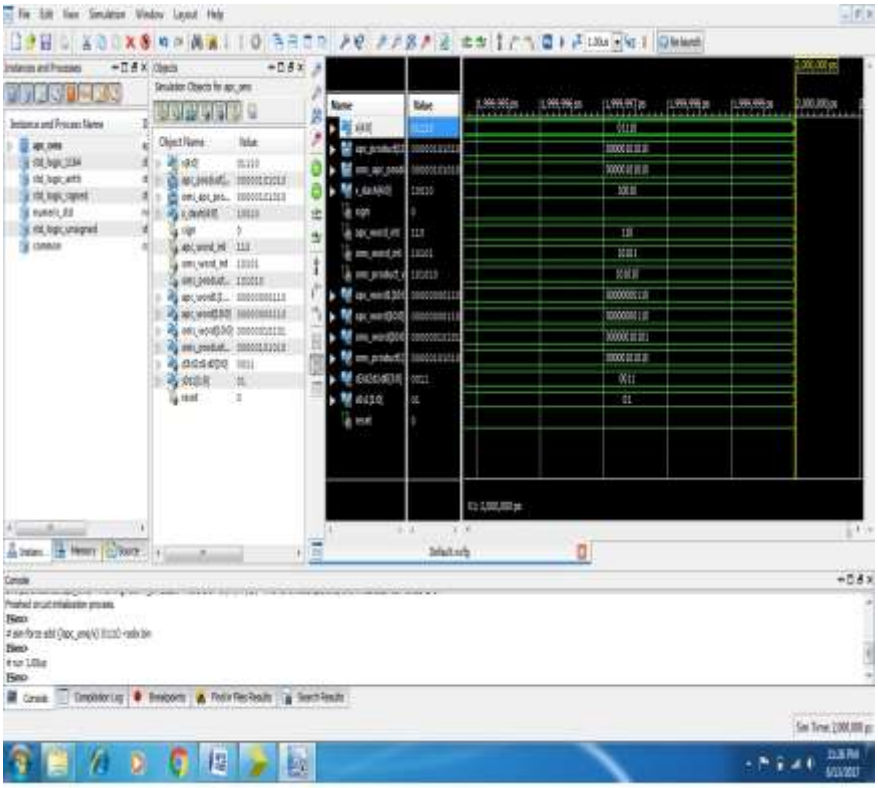

Figure 6: Experimental Result

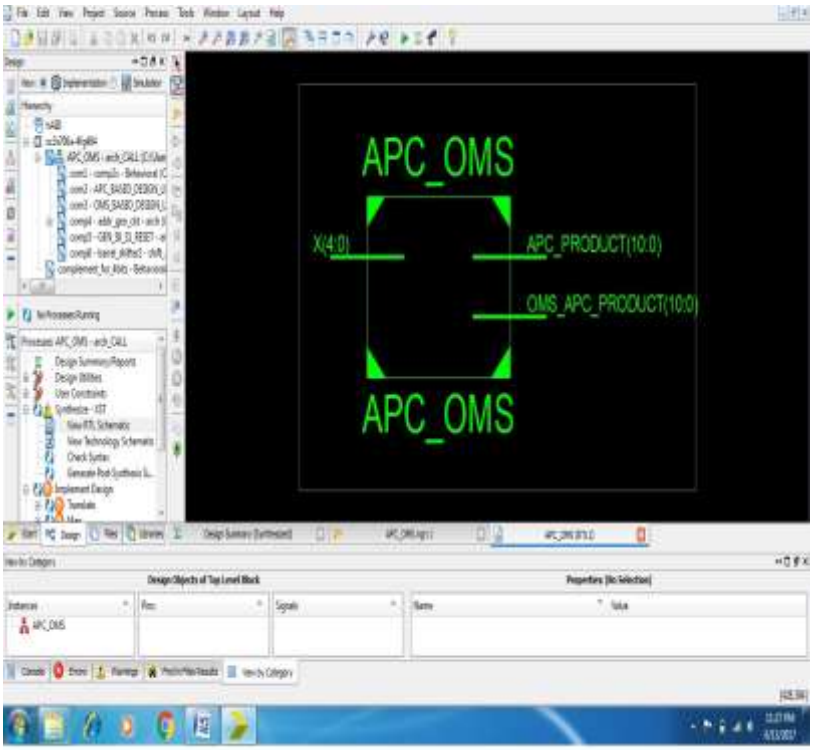

Figure 7: RTL Schematic

CONCLUSION

This article addressed the problem of optimizing the number of operations in the FIR filter design while satisfying the filter constraints, generally known as the FDO problem. It presented exact and approximate FDO algorithms, all of which are equipped with efficient methods to find the fewest operations in the shift-adds design of the coefficient multiplications. Moreover, it showed how these algorithms can be modified to target different filter constraints and filter forms and to handle a delay constraint in the multiplier blocks of filters. It was observed that the exact FDO method can handle filters with a small number of coefficients, on which approximate FDO methods can find solutions very close to the minimum. It was also shown that heuristic methods are indispensable for filters with a large number of coefficients, on which the proposed approximate method can find better solutions in terms of the number of operations than prominent FDO algorithms. It was indicated that the total number of operations, EWL value, filter length, quantization value, and filter form have a significant impact on the gate-level area, delay, and power dissipation results of filter designs. Finally, area reduction using LUT approach is proposed for further improvement.

REFERENCES

I. L.Wanhammar, DSP Integrated Circuits. New York, NY, USA: Academic, 1999.

II. H.Nguyen andA. Chatterjee, "Number-splittingwith shift-and-add decomposition for power and hardware optimization in linear DSP synthesis," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 8, no. 4, pp. 419–424, 2000.

III. Y. Voronenko and M. Püschel, "Multiplierless multiple constant multiplication," ACM Trans. Algorithms, vol. 3, no. 2, 2007, doi: 10.1145/ 1240233.1240234.

IV. P. Cappello and K. Steiglitz, "Some complexity issues in digital signal Processing," IEEE Trans. Acoust., Speech, Signal Process., vol. 32, no. 5, pp. 1037–1041, 1984.

V. R. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," IEEE Trans. Circuits Syst. II, vol. 43, no. 10, pp. 677–688, 1996.

VI. I.-C. Park and H.-J. Kang, "Digital filter synthesis based on minimal signed digit representation," in Proc. Design Autom. Conf., 2001, pp. 468–473.

VII. L. Aksoy,E. Costa, P. Flores, and J.Monteiro, "Exact and approximate algorithms for the optimization of area and delay in multiple constant Multiplications," IEEE Trans. Comput.-Aided Design Intrgr. CircuitsSyst., vol. 27, no. 6, pp. 1013–1026, 2008.

VIII. A. Dempster and M. Macleod, "Use of minimumadder multiplierblocks in FIR digital filters," IEEE Trans. Circuits Syst. II, vol. 42, no.9, pp. 569–577, 1995.

IX. L. Aksoy, E. Gunes, and P. Flores, "Search algorithms for the multipleconstant multiplications problem: Exact and approximate," Elsevier J.Microprocessors Microsyst., vol. 34, no. 5, pp. 151–162, 2010.

X. K. Johansson, O. Gustafsson, and L. Wanhammar, "A detailed complexity model for multiple constant multiplication and an algorithm to minimize the complexity," in Proc. IEEE Eur. Conf. Circuit Theory Design, 2005, pp. 465–468.

XI. H.-J. Kang and I.-C. Park, "FIR Filter Synthesis Algorithms for Minimizing the Delay and the Number of Adders," IEEE Tran. on Circuits and Systems II, vol. 48, no. 8, pp. 770–777, 2001.

XII. L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Finding the Optimal Tradeoff Between Area and Delay in Multiple Constant Multiplications," Elsevier Journal on Microprocessors and Microsystems, vol. 35, no. 8, pp. 729–741, 2011.