ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

A new Common Subexpression Elimination Technique for reducing logic operators and logic depth in FIR filter design

S.Sundar, V.Mohanraj, S.D.Shandeep, E.Subhasri, P.Deepika,
Department of ECE, Bannari Amman Institute of Tech., Sathyamangalam.

Manuscript Info

Manuscript History:

Received: 11 October 2013
Final Accepted: 19 October 2013
Published Online: November 2013

Abstract

The common subexpression elimination techniques minimize the two major metrics namely logic operators (LO) and logic depths (LD) in realizing finite impulse response (FIR) filters. Two classes of common subexpressions occur in the Canonical Signed Digit (CSD) representation of filter coefficients. Common ways of implementing constant multiplication is by a series of shift and add operations. As is well known, if the multiplier is represented in CSD form, then the number of additions (or subtractions) used will be a minimum. In this paper, we analyze the impact of the horizontal and the vertical common subexpression elimination techniques in FIR filters. Further, we analyze an algorithm to optimize the common subexpression elimination that produces FIR filters with fewer numbers of logic operators (LOs). The design examples show that the average reduction of LO achieved using these method over the weight-2 HCSE method which produced the best trade-off between LO and LD. This reduction of logic operators is achieved without any increase in the logic depth. When compared with multiple adder graph (MAG) algorithm, the average reduction of logic operators obtained using our method is 6% and the reduction of logic depth is 25%. All the techniques are designed and simulated using Xilinx ISE Design Suite 12.1.

Introduction

Filters are a basic component of all signal processing and telecommunication systems. Filters are widely employed in signal processing and communication systems in applications such as channel equalization, noise reduction, radar, audio processing, video processing, biomedical signal processing, and analysis of economic and financial data. A digital filter takes a digital input, gives a digital output, and consists of digital components [1].

Multiplier block consists of additions, subtractions and shift operations. The Multiplier Block is used to implement a parallel multiplication of a variable x with a set of fixed coefficients b_k. Generation of such Multiplier Block is known as the multiple constant multiplications (MCM) [2]. Figure.1 shows the basic block diagram for an FIR filter of length 6. The input signal is denoted as x(n). The b_k values are the coefficients used for multiplication, so that the output of multiplication at time n is given as $x(n)*b_k$, the summation of all the delayed samples multiplied by the appropriate coefficients to obtain the filtered output y(n).

Assuming coefficients b_k are known constants, and $x[n]$ is the input data, equation (1) can be rewritten as follows:

$$H(z) = \sum_{k=0}^{N-1} b_k Z^{-1} \quad \text{---------- 1.1}$$

$$y(z) = H(z).x(z) \quad \text{---------- 1.2}$$

$$y(n) = b_0.x(n) + b_1.x(n-1) + b_2.x(n-2) + ... + b_{N-1}.x(n-N+1)$$ ---------- 1.3

The paper is organized as follows. In section 2, we provide filter architecture with a brief review of the CSD approach. In section 3, we illustrate the HCSE technique, VCSE technique and the CSE optimization method and its comparisons are presented in section 4. In section 5, the simulation result for the above techniques and their comparison are given. Section 6 provides our conclusions.

II. Filter Architecture

2.1 Canonical Sign Digit Algorithm (CSD)

The CSD representation is radix-2 signed digit system with the digit set {1, 0,-1}. Given a constant coefficient, the CSD representation is that two nonzero digits are not adjacent [3]-[6]. Encoding a binary number such that it contains the fewest number of non-zero bits is called Canonical Sign Digit. A CSD representation is a kind of sum of signed power of two representations. In binary number representation the value is expressed using only 0 and 1, but in CSD representation we use 0, 1 and -1 [4].

An Example for CSD representation is

$71*X = 1000111_2*X = X \ll 6 + X \ll 2 + X \ll 1 + X$ (shift/add operation) ---- (2.1)

$1000111_2*X = 100100\text{-}1*X = X \ll 6 + X \ll 3 - X$ (CSD) ---------- (2.2)

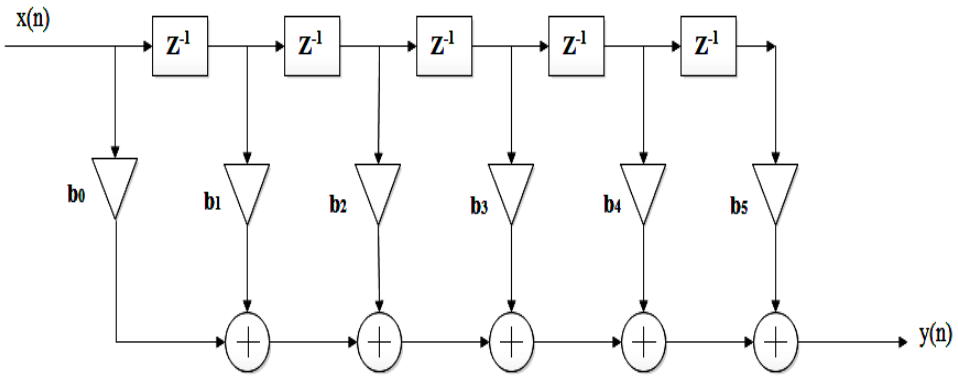

Figure.1.6-tap Direct Form FIR Filter

It is signed digit number system that minimizes the number of non-zero digits. It can reduce the number of partial product additions in a hardware multiplier. They are successful in implementing multipliers with less complexity. Now the multipliers in the digital filters are realized with shifters, adders and subtractions. Figure 2 shows the multiplier block designed using shift and add operation. The use of CSD expression can reduce the number of adders and sub tractors for example, the normal binary representation would need 3 adders, as 15 is represented as 1111_2. The total number of adders and sub tractors is less than the number of non-zero digits by 1. This results in the area reduction of multiplier of the digital filters. The Complexity of a digital filter design depends on the number of non-zero value in the filter Coefficients. So that by using CSD representation for fixed coefficients in FIR filter design will reduces the number of partial products as well as the area and the power consumption[5]-[6]. The Coefficient of FIR Filter is represented in signed and unsigned numbers. Two's complement arithmetic efficiently handles the addition and multiplications of signed numbers. In DSP filtering applications, coefficients are made up of both positive and negative numbers. Depending on the applications data is either positive or negative. Two's complement arithmetic efficiently handles the addition and multiplications of signed numbers. The advantages of two's complement are that we can use the same hardware to add negative numbers and positive numbers and the carry out is discarded. So first the Coefficient firstly convert into two's Complement then CSD Algorithm is applied which convert the representation into maximum number of non-zero terms. The CSD

representation of numbers, which further reduces the computational load when used with Horner's method for multiplication.

III. Common Sub expression Elimination (CSE) Methods

In general, the CSE methods utilize two types of Common Subexpressions (CSs) [7]-[9], the horizontal CSs (HCSs) that exist within each coefficient and his techniques is called the horizontal common subexpression elimination (HCSE) and the vertical CSs (VCSs) that exist across the adjacent coefficients and this technique is called as the vertical common subexpression elimination (VCSE). The CSE method can be explained by taking a filter example. A 6-tap LPFIR filter designed using Parks-McClellan algorithm is used. Figure 3 shows the 6-tap FIR filter coefficients. The pass-band and stop-band edges of the filter are 0.2π and 0.25π respectively. The numbers in the first row (-1,-2…) represent the number of bitwise right shifts. In this paper, we analyze the impact of HCSE and VCSE in exploiting the symmetry of FIR filter coefficients. Further, we present an optimization algorithm to reduce the number of LOs and LD in FIR filters.

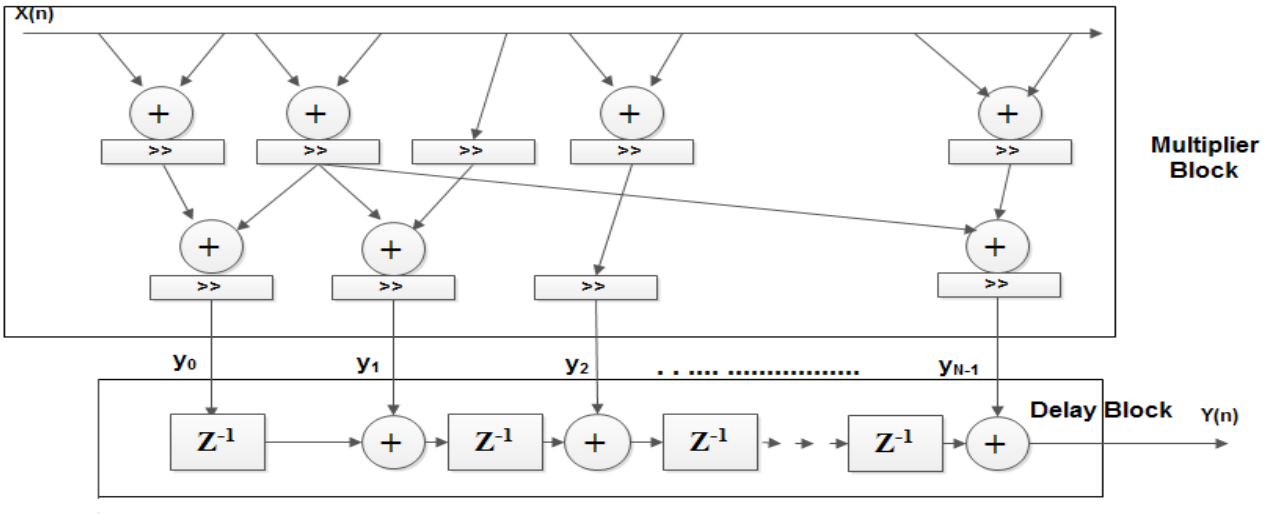

Figure.2.Multiplier Block Using Add and Shift

Bit shift / h(n)	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15	-16
h(0)	0	1	0	0	0	1	0	-1	0	1	0	1	0	1	0	-1
h(1)	0	1	0	-1	0	0	0	1	0	1	0	1	0	-1	0	-1
h(2)	0	1	0	0	-1	0	0	0	1	0	0	0	0	0	-1	0
h(3)	0	1	0	0	-1	0	0	0	1	0	0	0	0	0	-1	0
h(4)	0	1	0	-1	0	0	0	1	0	1	0	1	0	-1	0	-1
h(5)	0	1	0	0	0	1	0	-1	0	1	0	1	0	1	0	-1

Figure 3.CSD representation of 6-tap FIR Filter coefficients

3.1. THE HCSE TECHNIQUE

The HCSE technique utilizes the common horizontal subexpressions that occur within each coefficient to eliminate redundant computations. In general, these methods use Hartley's [10] two most common HCS, i.e., [1 0 1]

and [1 0 –1] and their negated versions [-1 0 -1] and [-1 0 1]. If x_1 is the input signal and 2^{-j} represents shift right by j, the HCS ([1 0 1] and [1 0 -1]) shown in rectangles in figure 4 are given by

$$x_2 = x_1 + 2^{-2} x_1 \quad \text{and} \quad x_2 = x_1 - 2^{-2} x_1 \qquad \text{---------- (3.1)}$$

Using HCSE technique, the output of the filter can be represented as

$$\begin{aligned} &2^{-2}x_1 + 2^{-6}x_3 + 2^{-10}x_2 + 2^{-14}x_3 + 2^{-2}x_3[-1] + 2^{-8}x_2[-1] + 2^{-12}x_3[-1] - 2^{-16}x_1[-1] \\ &+ 2^{-2}x_1[-2] - 2^{-5}x_1[-2] + 2^{-9}x_1[-2] - 2^{-15}x_1[-2] + 2^{-2}x_1[-3] - 2^{-5}x_1[-3] + 2^{-9}x_1[-3] \\ &- 2^{-15}x_1[-3] + 2^{-2}x_3[-4] \end{aligned} \qquad \text{---------- (3.2)}$$

where [-*k*] represents a delay of *k*. Fig. 5 shows the filter implementation using the HCSE method. The numerals adjacent to the data paths in Fig. 5 represent the number of right shifts. The function of the Multiplier Block (MB) shown in Fig. 5 is to compute the sum of partial products obtained when the input signal is convolved by the filter coefficient h_k.

Bit shift	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15	-16
h(n)																
h(0)	0	1	0	0	0	1	0	-1	0	1	0	1	0	1	0	-1
h(1)	0	1	0	-1	0	0	0	1	0	1	0	1	0	-1	0	-1
h(2)	0	1	0	0	-1	0	0	0	1	0	0	0	0	0	-1	0
h(3)	0	1	0	0	-1	0	0	0	1	0	0	0	0	0	-1	0
h(4)	0	1	0	-1	0	0	0	1	0	1	0	1	0	-1	0	-1
h(5)	0	1	0	0	0	1	0	-1	0	1	0	1	0	1	0	-1

Figure 4. Horizontal common subexpression in filter coefficient

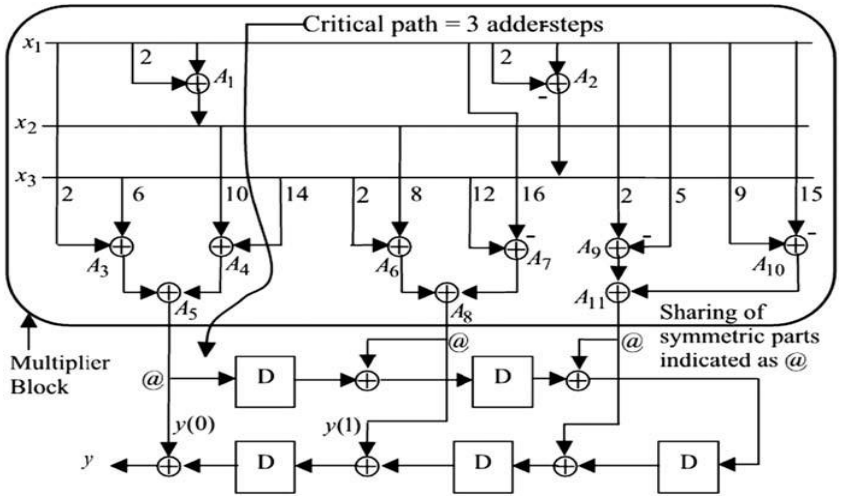

Fig. 5. FIR filter implementation using HCSE method.

There are two types of adders in the filter structure, structural adders (SAs) that compute the sum of convolved signals and MB adders (MBAs) which compute the sum of partial products formed in coefficient multiplication. For a given filter length N, the number of SAs is fixed. The main focus of CSE is to reduce the number of MBAs since they dominate the hardware cost. If N_b represents the number of nonzero bits in the symmetric half coefficient set of an FIR filter of length N, the total number of MBAs, T_{mba}, needed to realize the filter using direct method(direct method is the implementation using shifts and adds and without using CSE techniques)is

$$T_{mba} = N_b - [N/2]$$

Fig. 1, shows that the N_b is 18 and N is 6. Thus15 MBAs are required to realize the filter using direct method. In the HCSE method, since all the nonzero bits forming an HCS exist within the coefficient, it symmetric counterpart can be easily implemented using delays and SAs, i.e., no additional MBAs are required for the symmetric part. Note that the coefficients h(3)–h(5) are symmetric with respect to h(0)–h(2) and hence their outputs can be shared as shown in Fig. 2 using the symbol '@'. Thus, only 11 MBAs (A1–A11) are needed for the HCSE implementation in Fig. 5, which is a reduction of 26% over the direct method. The LDs of the filter circuit are identical (3adder- steps) in both direct method and CSE technique.

3.2. The VCSE ALGORITHM

The VCSE methods [11-13] utilize the VCSs that occur across the adjacent coefficients to tackle the MCM. The VCSs [1 1]and [1 -1], that exist across the coefficients, shown inside the rectangles in Fig. 6 by x4 and x5, respectively:

$$x_4 = x_1 + x_1[-1] \text{ and } x_5 = x_1 - x_1[-1] \text{ ---------(3.3)}$$

Where $x_1[-k]$ represents x_1 delayed by k units. With these VCSs, the filter output using VCSE is

$$2^{-2}x_4 + 2^{-6}x_1 - 2^{-8}x_5 + 2^{-10}x_4 + 2^{-1}x_4 + 2^{-1}x_4 + 2^{-4}x_1[-1] + 2^{-2}x_4[-2] - 2^{-5}x_4[-2]$$
$$+ 2^{-9}x_4[-2] - 2^{-15}x_4[-2] + 2^{-2}x_4[-4] - 2^{-4}x_1[-4] + 2^{-8}x_5[-4] + 2^{-10}x_4[-4] + \quad \text{---- 3.4}$$
$$2^{-12}x_4[-4] + 2^{-14}x_5[-4] - 2^{-16}x_4[-4] + 2^{-6}x_1[-5]$$

Bit shift / h(n)	-1	-2	-3	-4	-5	-6	-7	-8	-9	-10	-11	-12	-13	-14	-15	-16
h(0)	0	1	0	0	0	1	0	-1	0	1	0	1	0	1	0	-1
h(1)	0	1	0	-1	0	0	0	1	0	1	0	1	0	-1	0	-1
h(2)	0	1	0	0	-1	0	0	0	1	0	0	0	0	0	-1	0
h(3)	0	1	0	0	-1	0	0	0	1	0	0	0	0	0	-1	0
h(4)	0	1	0	-1	0	0	0	1	0	1	0	1	0	-1	0	-1
h(5)	0	1	0	0	0	1	0	-1	0	1	0	1	0	1	0	-1

Figure 6. Vertical common subexpression in filter coefficient

Fig. 7 shows the VCSE realization of the filter. Since the bits that form VCSs occur across the coefficients, the symmetry of VCSs cannot be utilized when the bits are of opposite signs. Hence in VCSE, additional MBAs are required to obtain the symmetric part of the coefficients when more than one VCSs with bits of opposite signs exist. Consider the VCSs across the coefficients h(0) and h(1) in Fig. 6.

$$2^{-2}x_4 + 2^{-6}x_1 - 2^{-8}x_5 + 2^{-10}x_4 + 2^{-12}x_4 + 2^{-14}x_5 - 2^{-16}x_4 + 2^{-4}x_1[-1] \quad \text{-------- (3.5)}$$

Its symmetric VCS part across the coefficients h(4) and h(5) is

$$2^{-2}x_4[-4] - 2^{-4}x_1[-4] + 2^{-8}x_5[-4] + 2^{-10}x_4[-4] +$$
$$2^{-12}x_4[-4] + 2^{-14}x_5[-4] - 2^{-16}x_4[-4] + 2^{-6}x_1[-5] \quad \text{---------- (3.6)}$$

Note that (3.6) cannot be directly obtained from (3.5) by simple delay operation since the signs and delays of certain terms of (3.6) are different from that of (3.5). Therefore, (3.6) needs to be obtained from (3.5) using (3.7) and (3.8) as given below

$$2^{-2}x_4 + 2^{-10}x_4 + 2^{-12}x_4 - 2^{-16}x_4 \xrightarrow{[4]} 2^{-2}x_4[-4] + 2^{-10}x_4[-4] + 2^{-12}x_4[-4] - 2^{-16}x_4[-4] \text{ -------- (3.7)}$$

$$-2^{-8}x_5 + 2^{-14}x_5 \xrightarrow{[4]} \xrightarrow{-} 2^{-8}x_5[-4] - 2^{-14}x_5[-4] \text{ ---------- (3.8)}$$

3.3 CSE optimization method

The main aim of the algorithm is to extract the maximum number of most frequently occurring common subexpressions. The HCSs, [1 0 1], [1 0 -1], [1 0 0 1], [1 0 0 -1] and their negated versions are used in our method since they are the most commonly occurring subexpressions. Among all the possible VCSs, we only use [11], [101] and their negated versions, since the signs of nonzero bits in these VCSs are identical (we designate these two VCSs as 'compatible VCSs'). Therefore, the use of these compatible VCSs facilitates better utilization of coefficient symmetry. Note that other HCSs such as [10 0 01] and [10 0 0 01] and VCSs such as [10 01] and [10 01] also exist in the CSD representation of coefficients. However, their frequency of occurrence is relatively smaller when compared to the HCSs and VCSs we have chosen.

For any coefficient, the CSs (HCSs or VCSs) with highest frequency are selected with priority given to HCSs first. If two or more HCSs occur common to different coefficients and they are having identical shifts between them which are known as identical-shift HCSs (IS-HCSs). Each coefficient is compared with all the other coefficients for IS-HCSs. If more than one common IS-HCSs occur between a coefficient pair, the IS-HCSs can be grouped together to further eliminate redundant computations. Our optimization procedure is given in detail in [14].

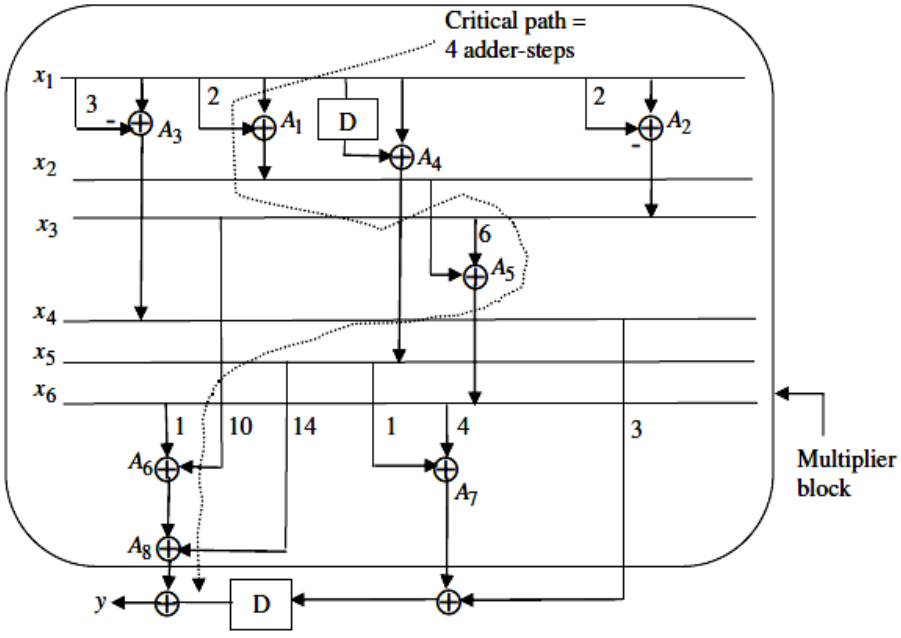

Fig. 7. FIR filter implementation using VCSE method

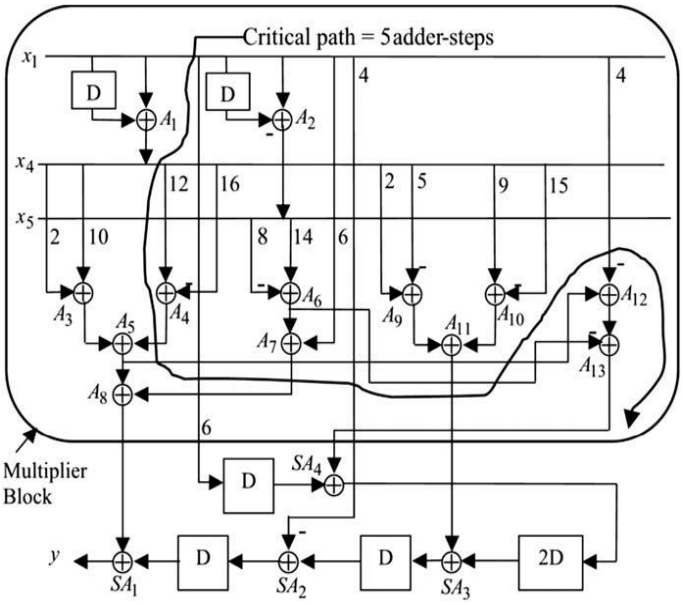

Fig. 8. FIR filter implementation using our CSE optimization.

IV. Simulation Result

Among the CSE techniques discussed the VCSE implementation requires more MBAs (13 MBAs in this case) than the HCSE despite the fact that the number of VCSs (16 VCSs as in Fig. 6) is more than the number of HCSs (12 HCSs as in Fig. 4). Furthermore, the LD in VCSE implementation (5 adder-steps) is larger than the HCSE (3 adder- steps). Hence the VCSE method results in increased LOs and LDs when compared with HCSE. The optimization algorithm produces the best reduction of LOs when compared to the other CSE algorithms in literature without increasing the LD of the coefficient multiplier. Further, the above discussed techniques are simulated using Xilinx ISE Design Suite 12.1 and its results are compared with each other. The Figure 9 shows the comparison table where number of registers, delay and number of IOs utilized by optimized CSE method is less than other methods, where the real time completion of HCSE is less than the optimized CSE but due to the reduction of 3 LO when compared to HCSE make this algorithm the best for subexpression elimination in the filter design.

V. Conclusions

It has been noted that the CSE technique employing HCS offer better reductions in the number of LO as well as LD than their VCS counterpart in FIR filter implementations. By using both HCS and VCS we designed an optimized CSE procedure to produced FIR filters with fewer numbers of LO and shorter LD when compared with other CSE algorithms which offered an average reduction of 15%. The average reduction of logic operators obtained using our method is 5% and the reduction of logic depth is 25%. The optimized CSE method which is having less number of IOs and registers when compared to other CSE techniques. The delay and offset time is also comparatively low when compared with other techniques which makes over conclusion that optimized CSE method is the best technique for FIR filter design.

TECHNIQUE	FILTER TYPE	N	W	LO	LD	IOs	NO OF REGISTERS	DELAY TIME (ns)	OFFSET IN TIME (ns)	OFFSET OUT TIME (ns)	REALTIME COMPLETION TIME (sec)
DIRECT METHOD	**FIR**	**6**	**16**	**15**	**3**	**49**	**0**	**2.865**	**9.07**	**4.626**	**6**
HCSE	**FIR**	**6**	**16**	**11**	**3**	**66**	**314**	**10.626**	**12.240**	**16.191**	**13**
VCSE	**FIR**	**6**	**16**	**13**	**5**	**34**	**337**	**9.253**	**10.815**	**17.419**	**15**
OPTIMIZED CSE	**FIR**	**6**	**16**	**8**	**4**	**34**	**281**	**9.220**	**10.42**	**24.07**	**15**

N-Number of Filter Tap
W-Input Word Length (Number of bits)
LO-Logic Operator
LD-Logic Depth
Figure 9, Comparison of simulation results

REFERENCES

[1] B. Mamatha1, V.V.S.V.S. Ramachandram Design And Implementation Of 120 Order Fir FilterBased On Fpga International Journal of Engineering Sciences & Emerging Technologies, August 2012.

[2] M. Potkonjak, M. B. Shrivasta and P. A. Chandrakasan, "Multiple constant multiplications: Efficient and versatile framework and algorithms for exploring common subexpression elimination, IEEE Trans. Computer-Aided Design, vol. 15, no. 2, pp. 151-161, Feb. 1996.

[3] Hunsoo Choo, Khurram Muhammad, and Kaushik Roy, Fellow, IEEE Complexity Reduction of Digital Filters Using Shift Inclusive Differential Coefficients IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 52, NO. 6, JUNE 2004

[4] M. Mehendale, S. D. Sherlekar, and G. Venkatesh, "Synthesis of multiplierless FIR filters with minimum number of additions," in Proceedings of the 1995 IEEE/ACM International Conference on Computer-Aided Design, Los Alamitos, CA: IEEE Computer Society Press, 1995, pp. 668-671.

[5] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," IEEE Trans. Ckts. Syst. II, vol. 43, pp. 677-688, Oct. 1996.

[6] M. M. Peiro, E. I. Boemo, and L. Wanhammar, "Design of high-speed multiplierless filters using a nonrecursive signed common subexpression algorithm," IEEE Trans. Ckts. Syst. II, vol. 49, no. 3, pp. 196-203, March 2002.

[7] Marcos Martínez-Peiró, Eduardo I. Boemo, and Lars Wanhammar, Member, IEEEDesign of High-Speed Multiplierless Filters Using a Nonrecursive Signed Common Subexpression AlgorithmIEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: ANALOG AND DIGITAL SIGNAL PROCESSING, VOL. 49, NO. 3, MARCH 2002

[8] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova, "A new algorithm for elimination of common subexpressions," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Syst., vol. 18, no. 1, pp.58-68, January 1999.

[9] N.C.Senthilkumar, Dr.E.Logashanmugam Design and Implementation of Low Power and Low area FIR Filter Using CSM ArchitectureInternational Journal of Innovative Research & Studies, June, 2013

[10] R.I. Hartley, Subexpression sharing in filters using canonic signed digit multipliers, IEEE Trans. Circuits Syst. II 43 (1996) 677–688 (October).

[11] Y. Jang, S. Yang, Low-power CSD linear phase FIR filter structure using vertical common sub-expression, Electron. Lett. 38 (15) (2002) 777–779 (July 2002).

[12] A.P. Vinod, E.M.-K. Lai, A.B. Premkumar, C.T. Lau, FIR filter implementation by efficient sharing of horizontal and vertical common subexpressions, Electron. Lett. 39 (2) (2003) 251–253 (January).

[13] Y. Takahashi, M. Yokoyama, New cost-effective VLSI implementation of multiplierless FIR filter using common subexpression elimination, in: Proceedings of International Symposium on Circuits and Systems, vol. 2, Kobe, Japan, May 2005, pp. 1445–1448.

[14] A.P.Vinod a, , EdmundLai b, DouglasL.Maskell a, P.K.Meher cAn improved common subexpression elimination method for reducing logic operators in FIR filter implementations without increasing logic depth INTEGRATION, the VLSI journal 43 (2010) 124–13