

Design of a Multiplier Architecture Based On VHBCSE Algorithm for Fir Filter

Dr. J. Jasper Gnana Chandra¹, K. Mathew²

Professor and Head¹, Assistant Professor²

Department of EEE

St. Peter's College of Engineering & Technology, Chennai¹,

St. Peter's Institute of Higher Education & Research, Chennai²

Corresponding Author's Email: *jaspergnanachandran@gmail.com¹*

Abstract

The aim of the intended work is to design an efficient multiplier architecture based Upon Vertical Horizontal Binary Common Subexpression Elimination (VHBCSE) algorithm for FIR filter synthesis. In any FIR filter, the multiplier is the major constraint which defines the performance of desired filter. Binary Common Subexpression Elimination (BCSE) algorithm introduces the concept of eliminating the Common Subexpression (CS) in binary form for designing an efficient multiplier with low complexity. In VHBCSE, horizontal BCSE algorithm utilizes CSs occurring within each coefficient to get rid of redundant computations, while vertical BCSE uses CSs found across adjacent coefficients to eliminate redundant computations. A 2-bit vertical BCSE has been applied first on the adjacent coefficient, followed by 4 bit and 8 bit horizontal BCSEs to reduce the number of adders required in the multiplier of FIR filters. As a result, the proposed multiplier offers good reduction in the number of adders in realizing FIR filters. The proposed scheme is coded in Verilog language and simulation is performed using Xilinx ISE 13.4.

Keywords: *- BCSE algorithm, Common Subexpression Elimination (CSE), Finite Impulse Response (FIR) filter, low complexity.*

INTRODUCTION

As the scale of integration keeps growing, more and more sophisticated signal processing systems are being implemented on a VLSI chip. These signal processing applications not only demand great computation capacity but also consume considerable amount of energy [9].

The Finite Impulse Response (FIR) Filter is the important component for designing an efficient digital signal processing system. So, in this paper, a FIR filter is constructed, which is efficient not only in terms of power and area but also in terms of delay. When considered the elementary structure of an FIR filter, it is found that it is a combination of multipliers and delays, which in turn are the combination of adders [9]. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the system [8]. Furthermore, it is generally the most area consuming. Hence, optimizing the speed and area of the multiplier is a major design issue. Thus, goal of the research is to design an efficient multiplier which could then be used to build FIR filters.

MULTIPLIER

The majority of real life applications, such as microprocessors and digital processing

implementations, require the computation of multiplier operation. Specifically, speed, area and power efficient implementation of a multiplier is a very challenging problem in FIR filter. Some existing techniques for low complexity multiplier that can be implemented in FIR filter are discussed one by one in the following paragraphs. In general, there are sequential and combinational multiplier implementations. In this, multiplication consists of Partial Products Generation (PPG), Partial Product Reduction (PPR), and finally Carry Propagate Addition (CPA). Here a combinational case is considered because the scale of integration now is large enough to accept parallel multiplier implementations in digital VLSI systems. Also, the analysis of multiplier is carried out to optimize the performance of complex circuit topologies.

Many methods have been proposed to reduce the hardware complexity of digital filters for implementation using VLSI technologies. The digital FIR filters can be divided into two categories: multiplier less based and memory based technique. Multiplierless based designs recognize MCM with shift-and add operations and share the common sub-operations using canonical signed digit (CSD) recoding and common subexpression elimination (CSE)

to minimize the added cost of MCM [13]. In [32], more area savings are achieved by jointly considering the optimization of coefficient quantization and CSE. Most multiplierless MCM-based FIR filter designs use the transposed structure to allow for cross-coefficient sharing and tend to be more rapid, if order of the filter is large. However, the area occupied delay elements is larger compared with that of the direct form due to the range expansion of the constant multiplications and the subsequent additions in the SAs. Memory-based FIR designs [14] consist of two types of approaches: lookup table (LUT) methods and distributed arithmetic (DA) methods. The LUT-based design stores in ROMs odd multiples of the input signal to realize the constant multiplications in MCM [14]. The DA-based approaches recursively accumulate the bit-level partial results for the inner product computation in FIR filtering [13]. An important design issue of FIR filter implementation is the optimization of the bit widths for filter coefficients, which has direct impact on the area cost of arithmetic units and registers. Moreover, since the bit widths after multiplications grow, many DSP applications do not need full-precision outputs. Instead, it is desirable to generate faithfully rounded outputs where the total error introduced in quantization and

rounding is no more than one unit of the last place defined as the weighting of the least significant bit (LSB) of the outputs. In this, different types of algorithms, area, delay and power analyzed.

SUBEXPRESSION ELIMINATION

Constant multiplication can be implemented efficiently by using dedicated shift-and-add multipliers [9]. Subexpression elimination is a numerical transformation of the constant multiplication that can lead to efficient hardware implementations in terms of area, power and speed. Subexpression elimination is applied to a set of constant multiplications with the same constant multiplicand. Subexpression elimination can be performed on constant multiplications that operate on a common variable. Essentially, subexpression elimination is the process of examining the shift-and-add implementations of the constant multiplications and finding redundant operations. Once the redundancies are found, these operations can be performed once and shared among the constant multiplications.

A. Multiple Constant Multiplication (MCM)

Subexpression elimination can be applied to a set of constant multipliers that multiply a common variable. The multiple constant multiplication problem determines how subexpression elimination can be applied to the set of constant multipliers so that the number of shifts and additions required for implementation is minimized. One of the attractive is its versatility and adaptability to various forms of problem [9].

B. Linear Transformation

Subexpression elimination can also be applied to linear transformations. When subexpression elimination is applied to constant multiplication, the constants. The nonzero bits in the binary representation of a constant determine the number of partial products that need to be generated and summed in order to realize the constant multiplier[9]. Consider the general form of the linear transformation: $y = T * x$, where T is m by n matrix, y is a length- m vector, and x is a length- n vector. Writing this in an equivalent form, we have: considering linear transformations, the subexpression elimination problem consists of 3 basic steps. The 1st step is to minimize the number of shifts and additions required to compute the products $tx_i, i = 1, 2, \dots$

VHBCSE ALGORITHM

Vertical and horizontal BCSEs are the two types of BCSE used for eliminating the BCSs present across the adjacent coefficients and within the coefficients respectively in any BCSE method. Vertical BCSE produces more effective BCS elimination than the horizontal BCSE. However, this paper proposes one new BCSE algorithm which is a combination of vertical and horizontal BCSE for designing an efficient FIR filter. The proposed Vertical Horizontal Binary Common Subexpression Elimination (VHBCSE) algorithm is made of the following steps:

- Get the input of 16-bits ($x[15:0]$).
- Store coefficients of 16-bits ($h[15:0]$) in LUT.
- Partition coefficient ($h[15:0]$) into fixed groups of 2 bits each and use these groups as the select lines to the corresponding multiplexer (M7-M0) at layer1 as shown in figure 1.
- Partition the coefficient into groups of 4 bits each ($h[15:12]$, $h[11:8]$, $h[7:4]$, $h[3:0]$).
- Compare $h[15:12]$ with $h[11:8]$ in layer 2. If match is found then skip the output of the adder A2. Otherwise, take the output of the adder A2.

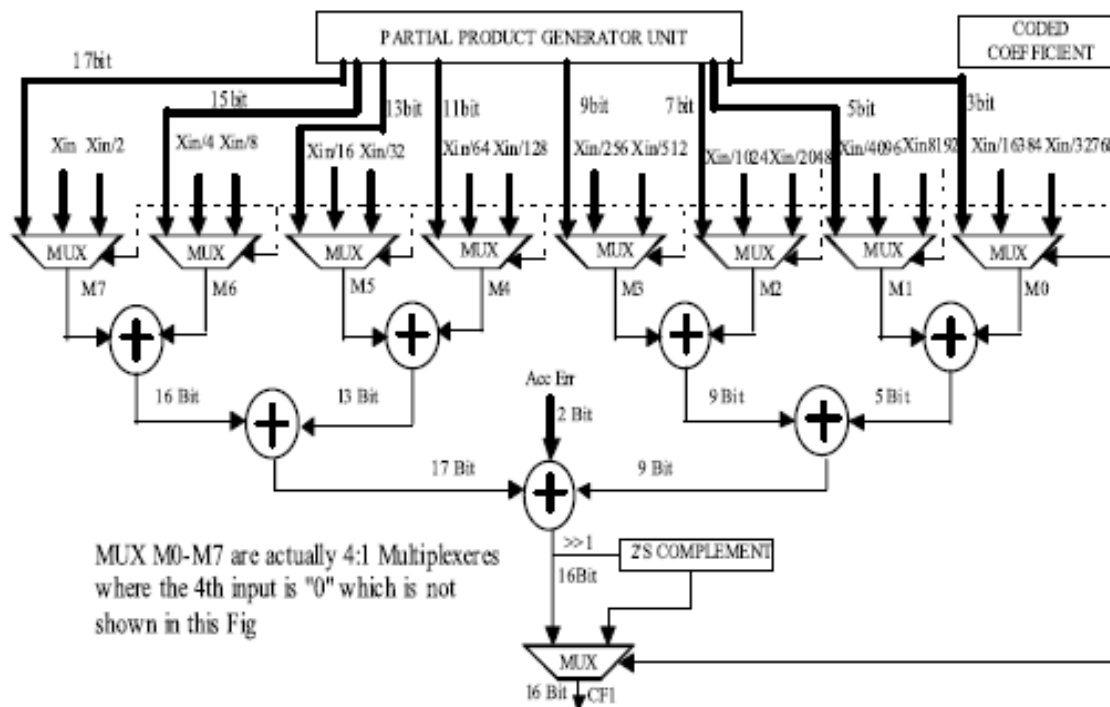


Figure 1: Block diagram of multiplier and final addition unit

- Compare $h[15:12]$ with $h[7:4]$ in the layer-2. If match is found then skip the output of the adder A3.
 - Otherwise, compare $h[11:8]$ with $h[7:4]$ in the layer 2. If match is found then skip the output of the adder (A3).
 - Otherwise, use the output of the adder (A3). Compare $h[15:12]$ with $h[3:0]$ in the layer-2. If match is found then skip the output of the adder A4.
 - Otherwise compare $h[11:8]$ with $h[3:0]$ in the layer-2. If match is found then skip the output of the adder
- $$y = \sum_{i=0}^{n-1} t_i x_i$$
- 2.
- If match is found then skip the output of the adder (A4). Else use the output of the adder (A4).
- Partition the coefficient into fixed group of 8-bit ($h[15:8]$, $h[7:0]$). Compare $h[15:8]$ with $h[7:0]$ in the layer 3.
 - If match is found then skip the output of the adder A6. Else take the output of the adder A6.
 - Obtain the final addition result by performing 1-bit right shift on the

output of the adder A7.

- Multiplication is completed. Store this result, $h \times x$, in the register.

A. Hardware Architecture Ofvhbcse Algorithm

The designed multiplier considers the length of the input (X_{in}) and coefficient (H) as 16-bit and 16-bit respectively while the output is assumed to be 16-bit long. Herein, the sampled inputs are stored in the register first and then the coefficients are stored directly in the LUTs. The data flow diagram of the proposed vertical-horizontal algorithm based constant multiplier (CM) design consists of partial product generator block, multiplexer unit at layer-1, controlled addition at layer-2, controlled addition at layer-3, final addition at layer-4 and control logic

generator. The data flow diagram of the proposed vertical-horizontal algorithm based Constant Multiplier (CM) design is shown in figure 2.

Functionality along with hardware architecture of different blocks of the designed VHBCSE based multiplier is explained in details.

B. Partial Product Generator (PPG)

In BCSE method, shift and add based technique has been used to generate the partial product which will be summed up for producing the final multiplication result. Choice of the size of the BCS defines the number of partial products. This technique helps in reducing the multiplexer's size which is used next to select the proper partial product depending on the coefficient's binary value.

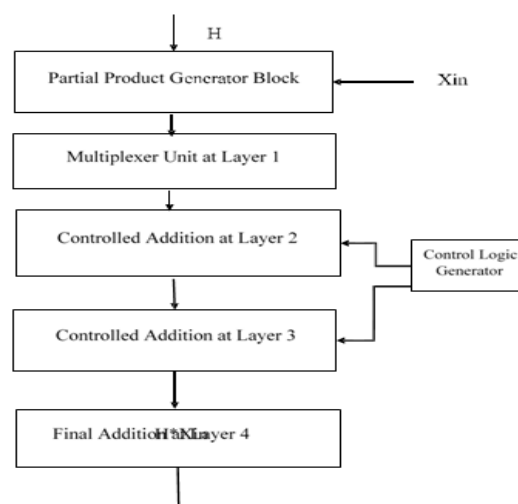


Figure 2: Flow diagram of the CM using VHBCSE algorithm

C. Control Logic (CL) Generator

Control logic generator block takes

the coefficient ($H[15:0]$) as its input and groups it into one of 4-bit each ($H[15:12]$, $H[11:8]$, $H[7:4]$, and $H[3:0]$). Another of 8-bit each ($H[15:8]$, $H[7:0]$).

D. Multiplexer Unit

The multiplexer unit is used to select the appropriate data generated from the PPG unit depending on the coefficient's binary value which would reduce the hardware and power consumption.

E. Controlled Addition at Layer-2

The partial products (PP) generated from eight groups of 2-bit BCSs are added up for the final multiplication results which have been performed in three layers.

F. Controlled Addition at Layer-3

The four multiplexed sums ($AS1$, $AS2$, $AS3$ and $AS4$) generated from layer-2 are now summed up in layer-3. In our algorithm, controlled additions are performed, instead of direct addition of these four sums as shown in figure 3. This addition ($A6$) is controlled by the control signal ($C7$) which has been generated based on 8-bit BCSE from the CS generator block.

G. Final Addition at Layer-4

This block performs the addition operation between the two sums ($AS5$ - $AS6$) produced by layer-3 to finally produce the multiplication result between the input and the coefficient.

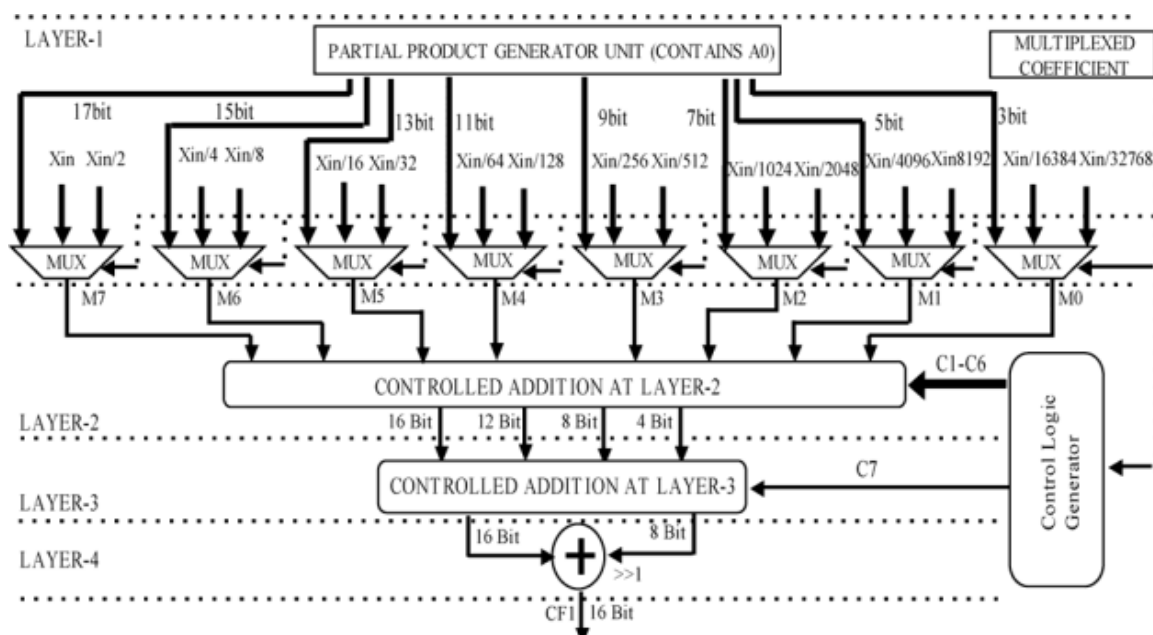


Figure 3: Multiplier Architecture

RESULTS AND DISCUSSION

The VHBCSE algorithm based multiplier

is coded in Verilog language. The simulation results are obtained using Xilinx

ISE 13.4. Using VHBCSE algorithm simulation is performed for 16 bit multiplication, x as its multiplicand and h as its multiplier which eliminates the common subexpression and produce the

result. The figure 4 shows the simulation result of vertical horizontal binary common subexpression algorithm based multiplier for FIR filter.

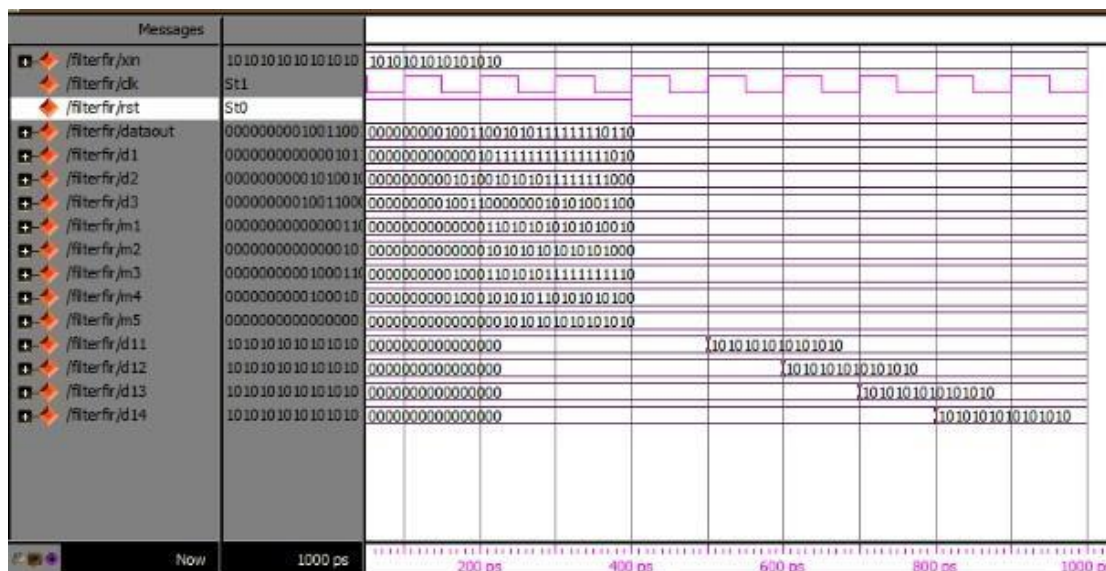


Figure 4: Waveform of VHBCSE based multiplier for FIR filter

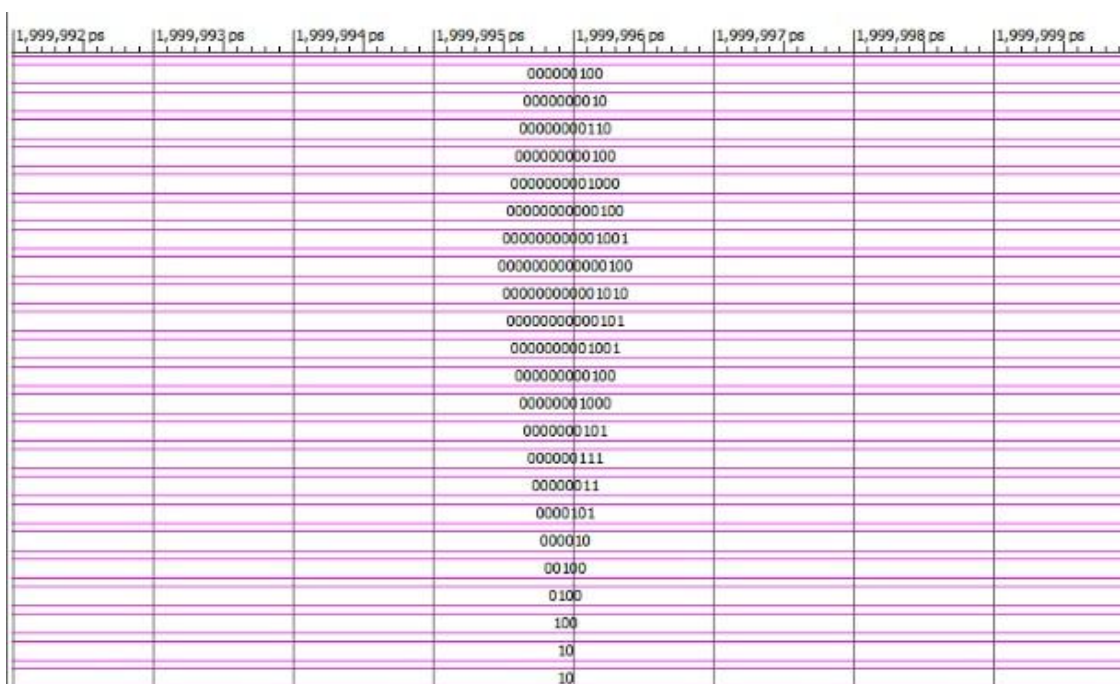


Figure 5: Waveform of common subexpression

Device Utilization Summary (estimated values)				[-]
Logic Utilization	Used	Available	Utilization	
Number of Slices	439	4656	9%	
Number of 4 input LUTs	783	9312	8%	
Number of bonded IOBs	64	232	27%	

Figure 6: Design utilization summary of VHBCSE algorithm

The figure 5 shows the common subexpression found during multiplication operation. An efficient multiplier should have following characteristics:-

- Accuracy: A good multiplier should give correct result.
- Speed: Multiplier should perform operation at high speed.
- Area: A multiplier should occupy less number of slices and LUTs.
- Power: Multiplier should consume less power.

CONCLUSION

The VHBCSE algorithm based multiplier architecture for FIR filter presents one new Vertical Horizontal BCSE algorithm which removes the initial common sub-expressions (CSs) by applying 8 bit and 4 bit common expressions. It has been shown that proposed algorithm reduces the number of slice usage.

REFERENCES

- I. Indranil Hatai, Indrajit Chakrabarti and Banarjee S., 'An efficient VLSI architecture of a reconfigurable pulse-shaping FIR interpolation filter for multi-standard DUC', IEEE Transactions On Very Large Scale Integration (VLSI) Systems, Vol. 23, no. 6, pp 1150-1154, 2015.
- II. Baugh C R. and Wooley B A., 'A two's complement parallel array multiplication algorithm', IEEE Transactions on Computer, vol.c-22, pp.1045-1047, 1973.
- III. Meher P K., 'New approach to look-up-table design and memory-based realization of FIR digital filter', IEEE Transactions on Circuits Systems-I, Regular Papers, vol. 57, no. 3, pp. 592–603, 2010.
- IV. Wallace C., 'A suggestion for a fast multiplier', IEEE Transactions on

- Electronics Computer, vol. EC-13, no. 1, pp. 14–17, 1965.
- V. Chia-Yu Yao, Hsin-Horng Chen, Tsuan-Fan Lin, Chiang-Ju Chien, and Chun-Te Hsu., ‘A Novel Common-Subexpression-Elimination Method for Synthesizing Fixed-Point FIR Filters’, IEEE Transactions on Circuits and Systems-I: Regular papers, Vol. 51, no.11, pp. 2215-2221, 2004.
- VI. Ko H J. and Hsiao S F., ‘Design and application of faithfully rounded and truncated multipliers with combined deletion, reduction, truncation, and rounding’, IEEE Transactions on Circuits Systems-II, Analog Digital Signal Processing, vol. 58, no. 5, pp. 304-308, 2011.
- VII. Indranil Hatai, Indrajit Chakrabarti, and Swapna Banerjee, ‘An efficient constant multiplier architecture based on vertical horizontal common subexpression algorithm for reconfigurable FIR filter synthesis’, IEEE Transactions on Circuits and Systems-I: Regular Papers, Vol. 62, no. 4, pp.1071-1080, 2015.
- VIII. Dandapat A., Ghosal S., Sarkar P and Mukhopadhyay P., ‘A 1.2 ns16×16-Bit Binary Multiplier Using High Speed Compressors’, International Journal of Electrical, Computer, and Systems Engineering, pp. 234-239, 2009. [9] Keshab K Parhi., ‘VLSI Signal Processing Systems’, John Wiley & sons, (2008).
- IX. Morris Mano M., ‘Digital Design’, vol.3rd Edition, Prentice Hall of India private limited, 2006.
- X. G.Rajakumar, S. Jeslin,’ Ultra Low Power Design Using Average 8T Differential Sensing Sub-Threshold SRAM’, International Journal of Applied Engineering Research, Vol. 10, No.4 , Page No.11353- 11359, April 2015, ISSN 0973-4562. (Anna University Listed Annexure- II Journal)
- XI. Nazieh Botros, ‘HDL Programming fundamentals VHDL & Verilog’, DaVinci engineering press, 2000.
- XII. Wayne Wolf, ‘Modern VLSI Design System-on-Chip Design’, vol. 3rd Edition, prentice Hall, 2000.

- XIII. Peiro M M. , Boemo E I., and Wanhammar I., ‘Design of high-speed multiplierless filters using a nonrecursive signed common subexpression algorithm’, IEEE Transactions Circuits Systems-II, Analog Digital Signal Processing, vol. 49, no. 3, pp. 196–203, 2002.

- XIV. Meher P K., Chandrasekaran S., and Amira A., ‘FPGA realization of FIR filters by efficient and flexible systolization using distributed arithmetic’ IEEE Transactions on Signal Processing, vol. 56, no. 7, pp. 3009–3017, 2008.

- XV. Yu Y J. and Lim Y C., ‘Design of linear phase FIR filters in subexpression space using mixed integer linear programming’, IEEE Transactions Circuits Systems I, Regular Papers, vol. 54, no. 10, pp. 2330, 2007.