ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Implementation of Efficient Algorithms for the Filter Design Optimization Problem

U.Sowmya Latha
M.Tech,
VLSI System Design,
GNIT JNT University, Hyd.
uslatha.369@gmail.com

M.Suman Kumar
Associate Professor,
GNIT JNT University, Hyd.
Sumankumar.gnit@gmail.com

B.Kedarnath
HOD,
Dept of ECE,
GNIT JNT University, Hyd.
bkedarnath@gmail.com

Dr.S.Sreenatha Reddy
Principal,
GNIT JNT University,
Hyd.
Sreenath_sakkamm@yahoo.com

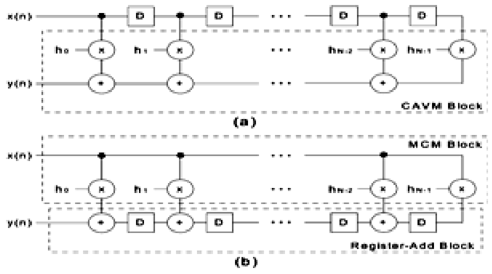
Abstract:
The filter design optimization (FDO) problem is defined as finding a set of filter coefficients that yields a filter design with minimum complexity, satisfying the filter constraints. It has received a tremendous interest due to the widespread application of filters. Assuming that the coefficient 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, we present an exact FDO algorithm that can guarantee the minimum design complexity under the minimum quantization value, but can only be applied to filters with a small number of coefficients. We also introduce an approximate algorithm that can handle filters with a large number of coefficients using less computational resources than the exact FDO algorithm and find better solutions than existing FDO heuristics. We describe how these algorithms can be modified to handle a delay constraint in the shift-adds designs of the multiplier blocks and to target different filter constraints and filter forms. Experimental results show the effectiveness of the proposed algorithms with respect to prominent FDO algorithms and explore the impact of design parameters, such as the filter length, quantization value, and filter form, on the complexity and performance of filter designs.

I. INTRODUCTION:
Digital filtering is a ubiquitous operation in digital signal processing (DSP) applications and is realized using infinite impulse response (IIR) or finite impulse response (FIR) filters.

Although an FIR filter requires a larger number of coefficients than an equivalent IIR filter, it is preferred to the IIR filter due to its stability and phase linearity properties. The computation of the output of an -tap FIR filter is given by

$$y(n) = \sum_{i=0}^{N-1} h_i \cdot x(n-i)$$

Where N is the filter length, h_i is the ith filter coefficient, and x(n-i) is the previous filter input. The straightforward realization of is depicted in Fig. 1 which is known as the direct form. Alternatively, the realization of (1) in the transposed form is shown in Fig.



The complexity of the FIR filter design is dominated by the multiplication of filter coefficients by the time-shifted versions of the filter input, i.e., the constant array-vector multiplication (CAVM) block in the direct form of Fig. 1(a) or by the multiplication of filter coefficients by the filter input, i.e., the multiple constant multiplications (MCM) block in the transposed form of Fig. 1(b). Since filter coefficients are fixed and determined beforehand and the realization of a multiplier in hardware is expensive in terms of area, delay, and power dissipation, these CAVM and MCM operations are generally implemented under a shift-adds architecture using only shifts, adders, and subtractors.

Note that shifts by a constant value can be implemented using only wires which represent no hardware cost. Thus, a well-known optimization problem [3] is defined as: given a set of constants, find the minimum number of adders/subtractors that realize the constant multiplications. Note that this is an NP-complete problem even in the case of a single constant multiplication. In the last two decades, many efficient algorithms were proposed for the multiplierless design of the MCM block, targeting not only the minimization of the number of operations, but also the optimization of gate-level area, delay, throughput, and power dissipation of the MCM design. The algorithm of guarantees the least number of operations in the CAVM design and incorporates efficient techniques to reduce the gate-level area and delay of the CAVM design.

Many efficient FDO algorithms were proposed, considering different filter constraints, targeting different filter forms, using different search methods during the exploration of possible filter coefficients, and applying different techniques to reduce the filter design complexity. However, none of these algorithms can guarantee that their solutions (a set of filter coefficients) lead to a filter design with the minimum number of adders/subtractors. This is due to two main facts: i) they do not explore the whole search space; and/or ii) they are not equipped with the exact techniques that can find the minimum number of operations for the constant multiplications. In this article, we present the exact FDO algorithm, called SIREN, that can find a set of fixed-point filter coefficients, satisfying the filter constraints and leading to a filter design with the minimum number of adders/subtractors under the minimum quantization value. SIREN is equipped with a depth-first search (DFS) method to explore the search space exhaustively, the exact algorithm of to find the minimum number of operations in the MCM block of the transposed form, and efficient search pruning and branching techniques to speed up the search process.
Since the size of the search space. of the FDO problem grows exponentially with the filter length, SIREN can only handle filters with a small number of coefficients. It was observed that it can find solutions to the symmetric filters including less than 40 coefficients in a reasonable time.

II. EXISTING SYSTEM:

Multiple constant multiplication (MCM) constitutes a typical fixed-point arithmetic operation in digital signal Processing. It is the focus of a lot of research on high-speed and low power devices in communication systems and signal processing systems. In multiplier less MCM, multipliers are replaced by simpler components such as adders and hard-wired shifts (adders in our paper include also subtractors as their hardware costs are similar). By using the Negative digits (subractor in circuit) in their signed-digit representations, coefficients may be synthesized with fewer adders; therefore the area and power consumption of the circuit can be reduced. An example of a multiplier-based and a multiplier less based MCM implementations respectively, wherein 4 multiplications are replaced by 5 adders and 5 hard-wired shifts. Such Multiplier less MCMs are utilized, for example, in the design of finite-impulse response Filters.

III. PROPOSED SYSTEM ALGORITHM: 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 ECHO-D 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.

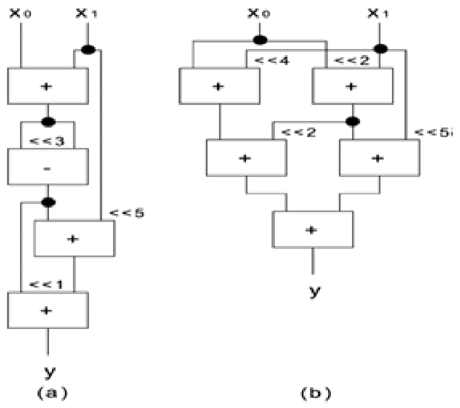


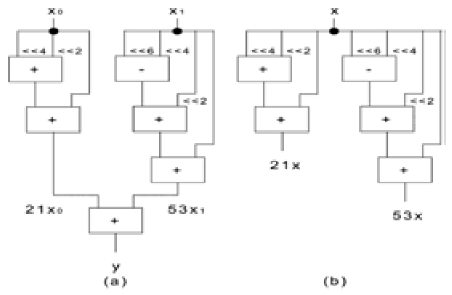
Fig: Multiplier less realization of constant multiplications using the DBR technique

IV. MODULES:

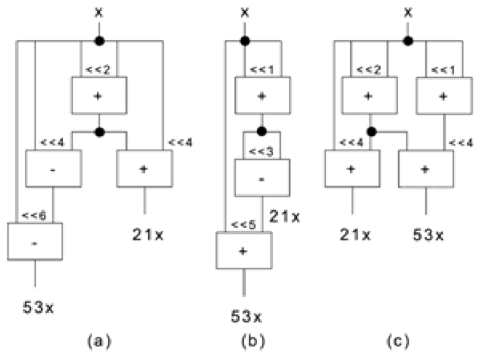
- Constant multiplications using the DBR
- Exact CSE algorithm, exact GB algorithm, approximate GB algorithm
- ECHO-A and ECHO-D
- FIR filter

V. MODULE DESCRIPTION:

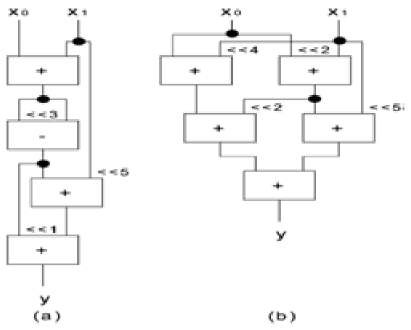
CONSTANT MULTIPLICATIONS USING THE DBR:



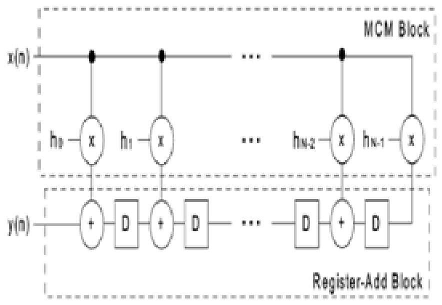
EXACT CSE ALGORITHM, EXACT GB ALGORITHM, APPROXIMATE GB ALGORITHM:



ECHO-A and ECHO-D:



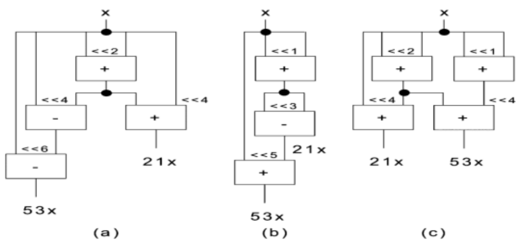
FIR FILTER OPTIMIZATION:



CSE AND GB ALGORITHM:

An algorithm for efficient solution of the multiple constant multiplication problem. Common sub expression elimination (CSE) as a way to tackle the MCM problem was already proposed by various authors primarily as a possible method for the optimization of finite-duration impulse response (FIR) filter area through the reduction of the multiplier block logic a number of other applications in which the MCM transformation can be successfully applied were proposed. In this work, we will introduce an algorithm able to solve the CSE problem in an efficient way.

The idea of CSE can be demonstrated on a FIR filter design. The optimization procedure targets the minimization of the multiplier block area. After expressing the coefficients in a canonical signed digit (CSD) format in order to reduce the total number of nonzero bits (thus also the additions/subtractions necessary), an add shift expansion is performed. The goal of CSE is to identify the bit patterns that are present in the coefficient set more than once. Since it is sufficient to implement the calculation of the multiple identical expressions only once, the resources necessary for these operations can be shared.

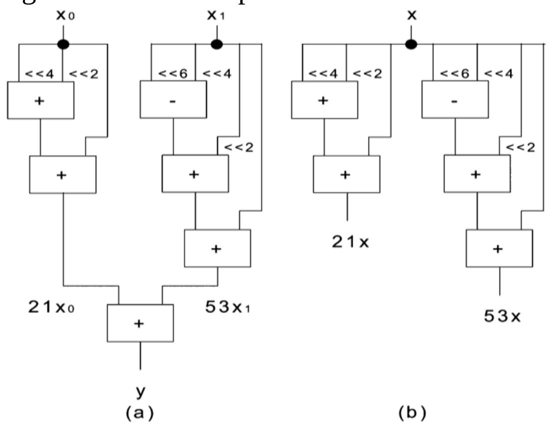


VI. GRAPH BASE(GB) ALGORITHM:

The optimization of gate-level area problem in digit-serial MCM design is an NP complete problem due to the NP-completeness of the MCM problem. Thus, naturally, there will be always 0–1 ILP problems generated by the exact CSE algorithm that current 0–1 ILP solvers find difficult to handle. Hence, the GB heuristic algorithms, which obtain a good solution using less computational resources, are indispensable.

In our approximate algorithm called MINASDS, as done in algorithms designed for the MCM problem given in Definition 1, we find the fewest number of intermediate constants such that all the target and intermediate constants are synthesized using a single operation. However, while selecting an intermediate constant for the implementation of the not yet synthesized target constants in each iteration, we favor the one among the possible intermediate constants that can be synthesized using the least hardware and will enable us to implement the not-yet synthesized target constants in a smaller area with the available constants.

After the set of target and intermediate constants that realizes the MCM operation is found, each constant is synthesized using an A-operation that yields the minimum area in the digit-serial MCM design. The area of the digit-serial MCM operation is determined as the total gate-level implementation cost of each digit-serial addition, subtraction, and shift operation under the digit size parameter d as described in Section II-D. The Preprocessing phase of the MINAS-DS algorithm is the same as that of the exact CSE algorithm, and its main part and routines are given. The right shift of an A-operation is assumed to be zero.

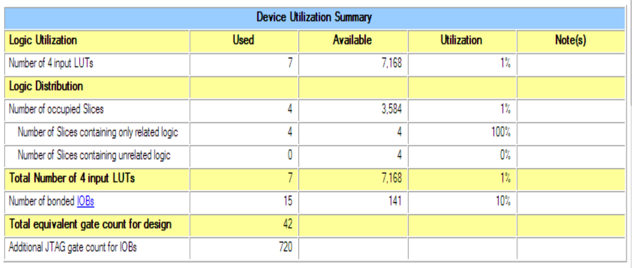


VII. APPLICATIONS:

1. Semiconductor Memory
2. System on- Chips
3. Digital Signal Processing (DSP)

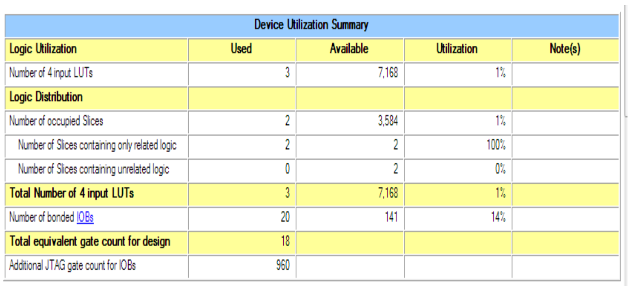
VIII. RESULTS:

DBR1:



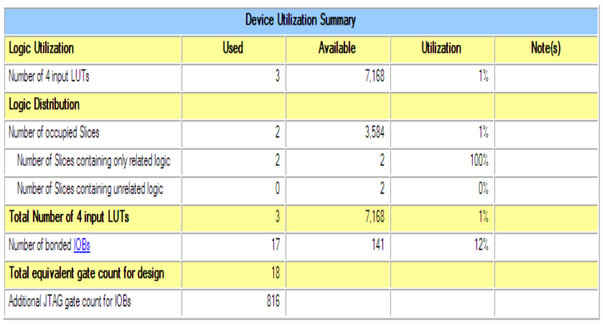
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	7	7,168	1%	
Logic Distribution				
Number of occupied Slices	4	3,584	1%	
Number of Slices containing only related logic	4	4	100%	
Number of Slices containing unrelated logic	0	4	0%	
Total Number of 4 input LUTs	7	7,168	1%	
Number of bonded IOBs	15	141	10%	
Total equivalent gate count for design	42			
Additional JTAG gate count for IOBs	720			

DBR2:



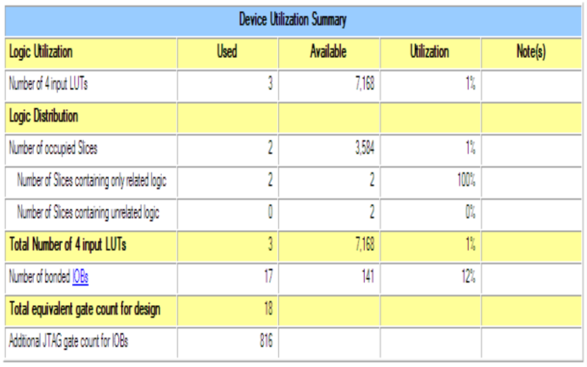
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	3	7,168	1%	
Logic Distribution				
Number of occupied Slices	2	3,584	1%	
Number of Slices containing only related logic	2	2	100%	
Number of Slices containing unrelated logic	0	2	0%	
Total Number of 4 input LUTs	3	7,168	1%	
Number of bonded IOBs	20	141	14%	
Total equivalent gate count for design	18			
Additional JTAG gate count for IOBs	960			

EXACT GB:



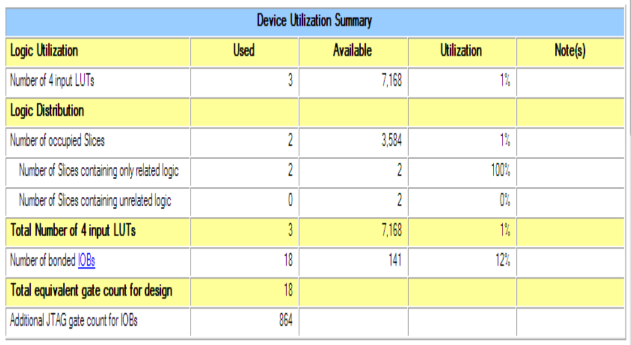
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	3	7,168	1%	
Logic Distribution				
Number of occupied Slices	2	3,584	1%	
Number of Slices containing only related logic	2	2	100%	
Number of Slices containing unrelated logic	0	2	0%	
Total Number of 4 input LUTs	3	7,168	1%	
Number of bonded IOBs	17	141	12%	
Total equivalent gate count for design	18			
Additional JTAG gate count for IOBs	816			

EXACT CSE:



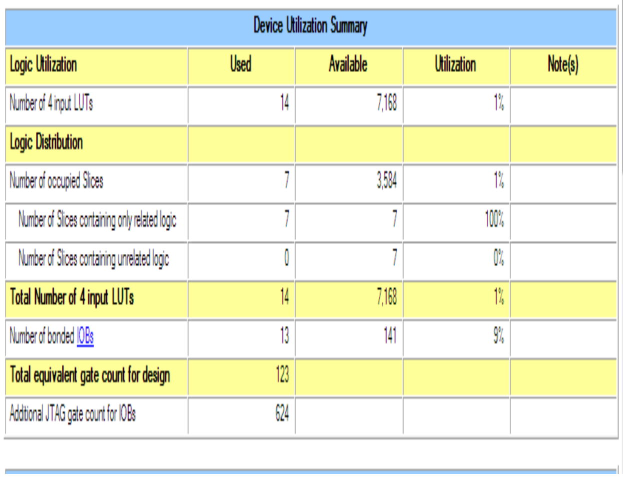
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	3	7,168	1%	
Logic Distribution				
Number of occupied Slices	2	3,584	1%	
Number of Slices containing only related logic	2	2	100%	
Number of Slices containing unrelated logic	0	2	0%	
Total Number of 4 input LUTs	3	7,168	1%	
Number of bonded IOBs	17	141	12%	
Total equivalent gate count for design	18			
Additional JTAG gate count for IOBs	816			

APPROXIMATE GB:



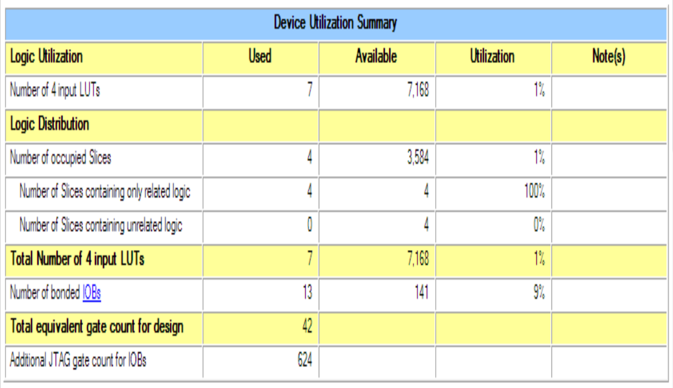
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	3	7,168	1%	
Logic Distribution				
Number of occupied Slices	2	3,584	1%	
Number of Slices containing only related logic	2	2	100%	
Number of Slices containing unrelated logic	0	2	0%	
Total Number of 4 input LUTs	3	7,168	1%	
Number of bonded IOBs	18	141	12%	
Total equivalent gate count for design	18			
Additional JTAG gate count for IOBs	864			

ECHO-A:



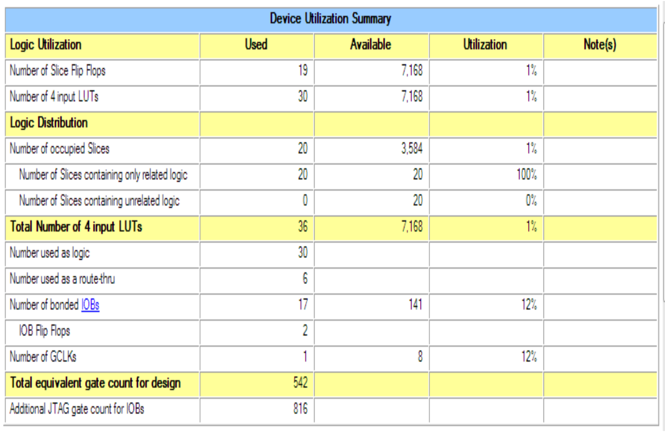
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	14	7,168	1%	
Logic Distribution				
Number of occupied Slices	7	3,584	1%	
Number of Slices containing only related logic	7	7	100%	
Number of Slices containing unrelated logic	0	7	0%	
Total Number of 4 input LUTs	14	7,168	1%	
Number of bonded IOBs	13	141	9%	
Total equivalent gate count for design	123			
Additional JTAG gate count for IOBs	624			

ECHO-D:



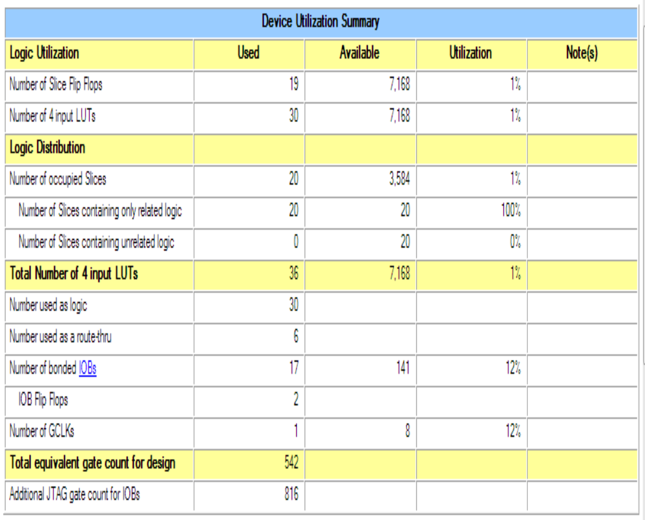
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of 4 input LUTs	7	7,168	1%	
Logic Distribution				
Number of occupied Slices	4	3,584	1%	
Number of Slices containing only related logic	4	4	100%	
Number of Slices containing unrelated logic	0	4	0%	
Total Number of 4 input LUTs	7	7,168	1%	
Number of bonded IOBs	13	141	9%	
Total equivalent gate count for design	42			
Additional JTAG gate count for IOBs	624			

FIR WITH GBR:



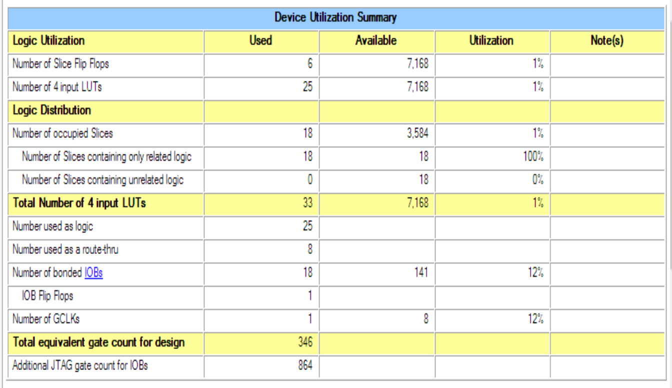
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	19	7,168	1%	
Number of 4 input LUTs	30	7,168	1%	
Logic Distribution				
Number of occupied Slices	20	3,584	1%	
Number of Slices containing only related logic	20	20	100%	
Number of Slices containing unrelated logic	0	20	0%	
Total Number of 4 input LUTs	36	7,168	1%	
Number used as logic	30			
Number used as a route-thru	6			
Number of bonded IOBs	17	141	12%	
IOB Flip Flops	2			
Number of GCLKs	1	8	12%	
Total equivalent gate count for design	542			
Additional JTAG gate count for IOBs	816			

FIR WITH EXACT GB:



Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	19	7,168	1%	
Number of 4 input LUTs	30	7,168	1%	
Logic Distribution				
Number of occupied Slices	20	3,584	1%	
Number of Slices containing only related logic	20	20	100%	
Number of Slices containing unrelated logic	0	20	0%	
Total Number of 4 input LUTs	36	7,168	1%	
Number used as logic	30			
Number used as a route-thru	6			
Number of bonded IOBs	17	141	12%	
IOB Flip Flops	2			
Number of GCLKs	1	8	12%	
Total equivalent gate count for design	542			
Additional JTAG gate count for IOBs	816			

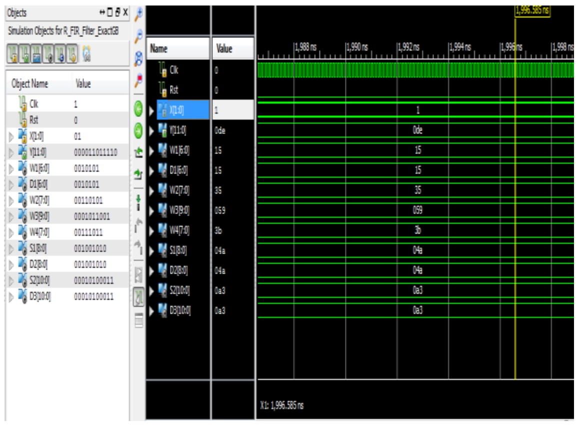
FIR WITH ECHO-D:



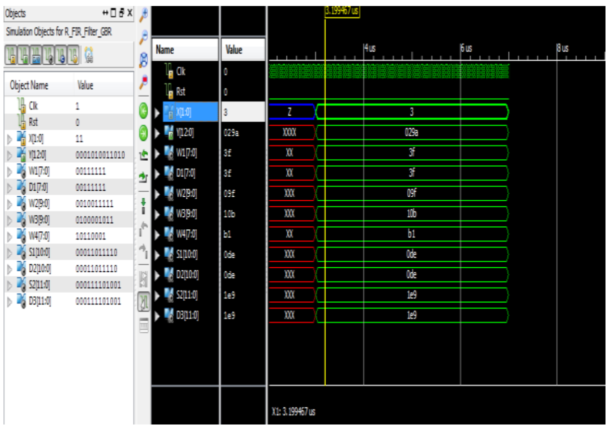
Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	6	7,168	1%	
Number of 4 input LUTs	25	7,168	1%	
Logic Distribution				
Number of occupied Slices	18	3,584	1%	
Number of Slices containing only related logic	18	18	100%	
Number of Slices containing unrelated logic	0	18	0%	
Total Number of 4 input LUTs	33	7,168	1%	
Number used as logic	25			
Number used as a route-thru	8			
Number of bonded IOBs	18	141	12%	
IOB Flip Flops	1			
Number of GCLKs	1	8	12%	
Total equivalent gate count for design	346			
Additional JTAG gate count for IOBs	864			

IX. SCREENSHOTS OF RESULTS:

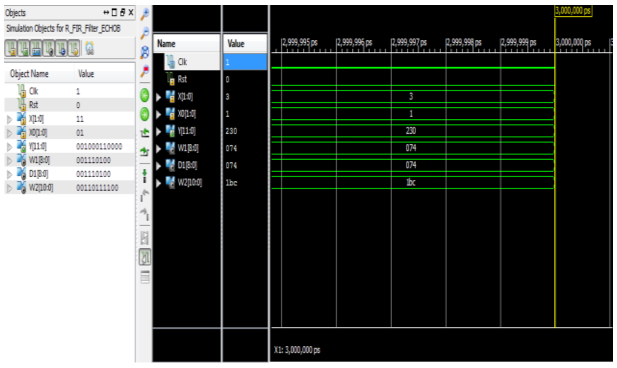
FIR GB MODIFIED:



FIR GB:

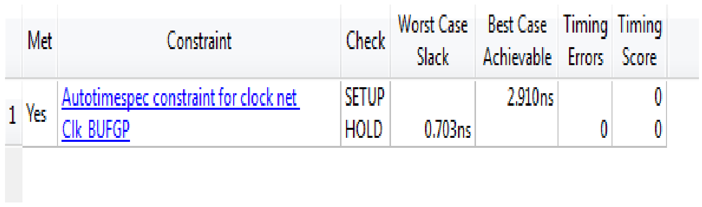


FIR ECHO B:



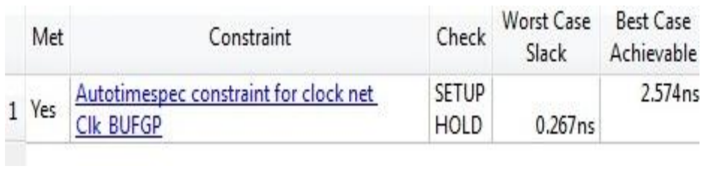
TIMING REPORT:

FIR GB:



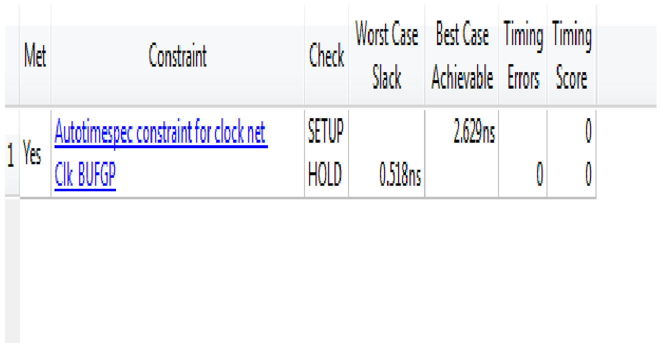
	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	Yes	Autotimespec constraint for clock net Clk_BUFGP	SETUP		2.910ns		0
			HOLD	0.703ns		0	0

FIR ECHO B:



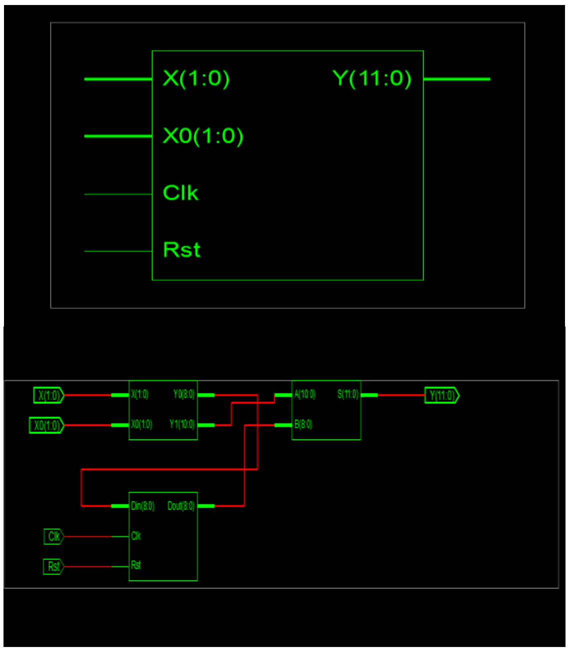
	Met	Constraint	Check	Worst Case Slack	Best Case Achievable
1	Yes	Autotimespec constraint for clock net Clk_BUFGP	SETUP		2.574ns
			HOLD	0.267ns	

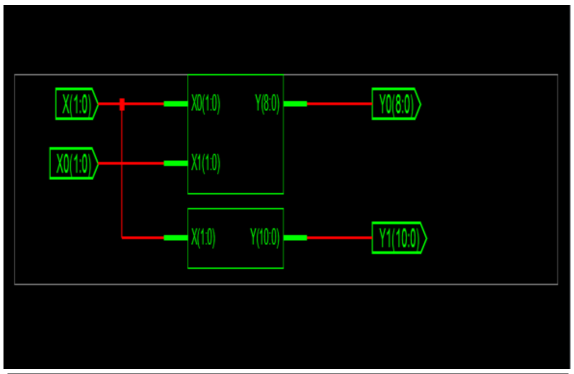
FIR GB MODIFIED:

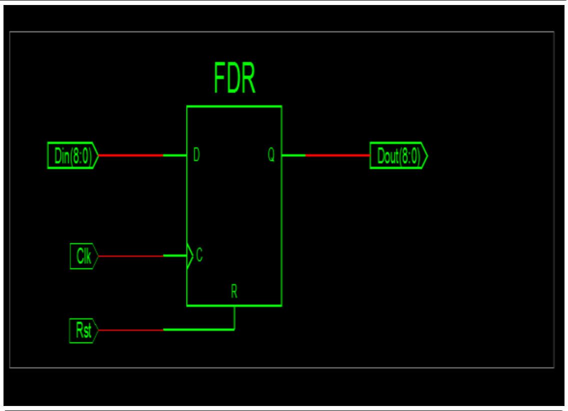


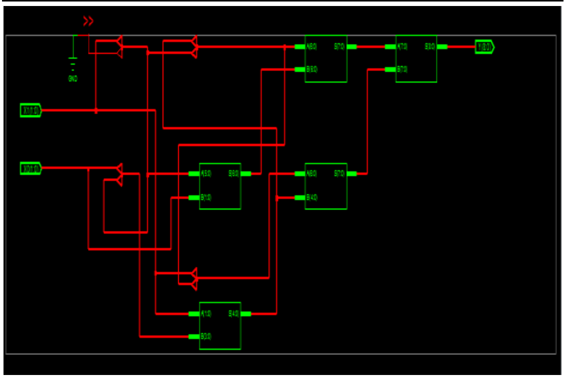
	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	Yes	Autotimespec constraint for clock net Clk_BUFGP	SETUP		2.629ns		0
			HOLD	0.518ns		0	0

RTL SCHEMATIC

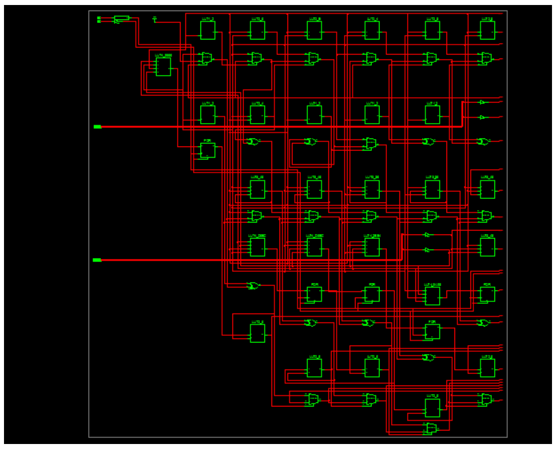








TECHNOLOGY SCHEMATIC



X. 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.

REFERENCES:

[1]L.Wanhammar, DSP Integrated Circuits. New York, NY, USA: Academic, 1999.

[2]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.

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

[4]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.

[5]R. Hartley, “Subexpression sharing in filters using canonic signed digit multipliers,” IEEE Trans. Circuits Syst. II, vol. 43, no. 10, pp. 677–688, 1996.

[6]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.

[7]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.

[8]A. Dempster and M. Macleod, “Use of minimum-adder multiplierblocks in FIR digital filters,” IEEE Trans. Circuits Syst. II, vol. 42, no.9, pp. 569–577, 1995.

[9]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.

[10]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.