A Novel Architecture of Multiple Constant Multiplication for Efficient FIR Filter Implementation by Using Bit Level Optimization of Adder Trees

Ume Salma S.
Department of Electronics and Communication Engineering
Student, Narayana Engineering College, Nellore, Andhra Pradesh, India*

Syed Athika Sultana
Asst. prof . ,Department of Electronics and Communication Engineering
Narayana Engineering College, Nellore, Andhra Pradesh, India

SK Sabiha Begum
Asst. prof., Department of Electronics and Communication Engineering
Narayana Engineering College, Nellore, Andhra Pradesh, India

Abstract- Multiple Constant Multiplication (MCM)[8] scheme is mostly used for the transposed direct-form of FIR filter implementation. This project explains the resource minimization problem by the optimization of adder trees. In order to minimize the total number of add/sub these MCM has more effective optimization of adder-trees by the elimination of common sub expression. MCM is widely used for reducing the computational complexity of fir filter .Fir filter complexity is largely eliminated by the filter coefficients with the multiplication of input samples. and the coefficients are constant for a given filter. In this paper, the resource minimization problem has identified by adder-tree scheduling for the MCM block operations and a mixed integer programming (MIP) is presented for minimizing the hardware resources such as adders.

Keywords- FIR, MCM, pipelining, parallel processing

I. INTRODUCTION

Finite Impulse Response is a commonly used digital filters in many digital signal processing(DSP) ,image and video processing applications. The complexity of a FIR filter is essentially ruled by using the multiplication of filter coefficients by the input samples. But happily, the coefficients are constants for a given filter, in order that multiplications are applied by adders, subtractors and hardwired shifts network, wherein the number of adders and subtractors are minimized with the aid of a steady multiplication scheme. The current most enter pattern at any given clock length is accelerated with all of the filter coefficients of a transposed direct form of FIR filter. A fixed of intermediate consequences are generated. In this situation, the shared across all of the multiplications in order to limit the entire number of additions/subtractions the usage of a couple of multiple constant multiplication(MCM).

To expand powerful algorithms and to become aware of the most reliable set of non-redundant sub expressions to achieve the logic operators of the minimal quantity and the minimum logic intensity of the MCM[1-2] has accomplished a splendid deal of research.

Irrespective of differences in technique and the level of optimality in all these works, after the common sub expression terms are decided for the non-redundant sub expressions (or phrases) the add/SUB network is formed, the product value similar to every adder-tree that sums up the computed coefficient for its applicable phrases. A short characterization observe of the variety of add/SUB in different parts of operators in arbitrary filters using 2-bit recursive sub expression removal for MCMs shows that the range of operators required to form the adder-tree networks could be very huge, from time to time some instances, in comparison to that within the term networks. Even as the principle research attention of MCM is on more powerful common sub expression sharing strategies, optimizations on adder-trees are largely reduced .No matter which common

sub expression identity is used for the formation of an adder-tree algorithm and is commonly treated by way of the equal tree-top minimization set of rules ,that ensures the peak of generated adder-tree is the minimal at the operator level.

In this paper, to limit the adder tree aid, we present a method of derivation of equal adder-trees by suitable scheduling of operations by bit level analysis on the adder the shift ADD/SUB networks can be reduced. We discover that area and power reduction is (up to fifteen% and 11.6% respectively with a mean of 8.46% and 5.96%)can be performed on already optimized MCM blocks.

II. PROPOSED ALGORITHM

Adder Tree Scheduling - An adder-tree scheduling set of rules is to outline an assignment of binary addition and sub- traction operations to sum up the input phrases such that the final output delay is reduced.

$$y(n) = x(n).z(n)$$

$$z(n) = \frac{x(n)}{y(n)} \qquad (1)$$

the above equation shows the direct form of fir filter realization.

where, x(n)-inputs,
y(n)-output and
z(n)-coefficients

A. Greedy scheduling algorithm-

In this system a memory based multiplier using greedy scheduling algorithm[3] for an efficient realization of fir filters.fig (1)shows the greedy scheduling by adder tree algorithm here in this method for example the operator performing bit width of 8-bits.

The operator is first be shifted and then subtracted and these type of operator performing cost up to 11FAs and 7 INVs.

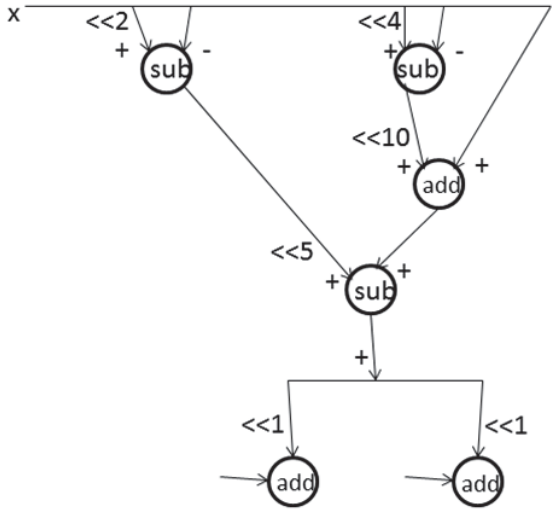


figure 1

And these type of algorithm is to minimize the terms of each coefficients and to reduce the optimal adder tree[4]. The tree minimization is by using an ADD/SUB for reducing the delay.

Either a positive or negative sign is assigned for the input terms. In order to reduce the hardware cost of the adder tree.

Depending on the type of constants the coefficients can be defined as below equation

$$(x \ll 7\text{-}x) - x \ll 5 - x \ll 4 - x \ll 3 \qquad (2)$$

The sign of the output edge should be same as that of the input edge sign and the FIR filter multiple adder trees can be reduced by using a negation to the adder tree.

For example the operator performing the base bit width of 8-bit performs 1≪4 - 1 and these type of operation (2)costs up to 11 FAs and 7 INVs

Note that if the adder tree determines "-" sign it consumes more hardware than an ADD operator require.

B. Bit-Level resource optimal adder-tree

We present a method to minimize number of adder tree resources by using equivalent adder-trees by suitable scheduling of operations by bit level analysis on the adder[8]. The shift ADD/SUB network can be reduced.

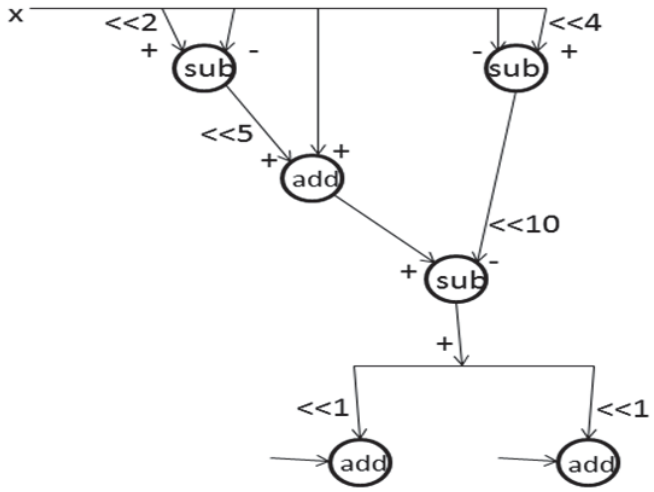


figure 2

fig (2)shows the adder tree by bit-level optimization here in this method the operator is first be subtracted and then it will be shifted and these type of operator (4) performing cost up to 8FAs and 8 INVs.

The bit level can be analyzed by scheduling the adder tree operation

$$x \ll 6 + x \ll 2 + x \ll 1 + x \qquad (3)$$

In this algorithm the scheduling of the operation(3) is changed and here first the input term is subtracted and then it is shifted and by having these type of operation the hardware for the optimal adder tree is reduced and here there is no need of compliments for the negative sign[5]. And the negation is used in the adder tree for reducing the number of hardware resources in the optimal adder tree.

The operator performance for base bit width of 8-bits on fig2 as 1-1<<4 .. .. (4) and its cost up to 8FAs and 8INVs compared to that of the previous algorithm and here the operator performance is same as that of the fig(1) but here the scheduling of the operators is different as compared with another if the signed bits are the subtrahend then there is no need of using inverters and these value takes the inverted signed of the subtrahend. by reducing the number of hardware resources we can reduce area .Therefore by reducing the area delay also reduced .The above scheduling algorithm performs series type of operation which consumes more time.

C. Minimum Resource Adder-tree Scheduling Using MIP-

To minimize the number of hardware resources such as adders a scheme has been designed known as mixed integer programming (MIP) algorithm for scheduling the adder tree

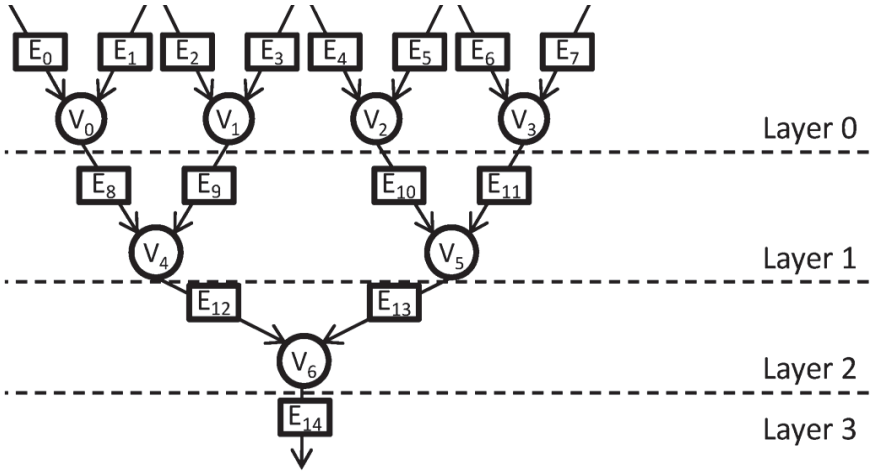


figure 3

The above fig(3) shows the binary adder tree of MIP by the logic depth L=3
For Binary Adder tree the logic depth of L=3 is used here and we are using the depth of three. The scheduling of adder tree problem[6] is for finding an assignment of input terms on the binary tree for the minimization of the adder tree resources.

A set of input terms given as $T=\{T_{0,\ldots\ldots\ldots\ldots\ldots,}T_{N-1}\}$ and the earliest delay time is as D_i for forming a logic depth L to minimize the number of input hardware resources and we are constructing a model of binary tree G (V,E) of logic depth of L for the above problem. The tree node binary operator position is to hold ADD/SUB and |V|= 2l-1 and the input edge position of an input term is $E=2^{L+1}-1$

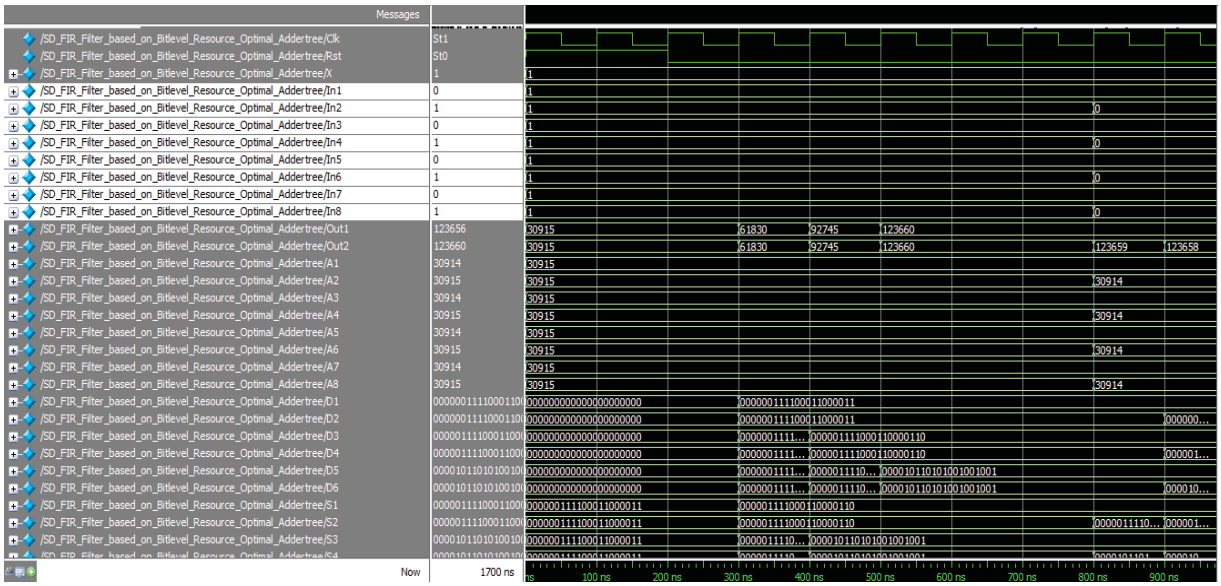
For example if a latency of 1 clock cycle of the k^{th} output is available in $(k+1)^{th}$ clock cycle in a 1-stage pipelined system. The two main drawbacks of the pipelining are increasing the number of latches and in system latency. The k^{th} clock cycle the 3-inputs x(3k), x(3k+1) and x(3k+2) are processed and 3 samples are generated at the output for the parallel processing system depending on the number of input clock cycles

Here the scheduling of the input term is for finding problem for the assignment of the inputs for the binary adder tree [7] so that we can reduce the adder tree resources.

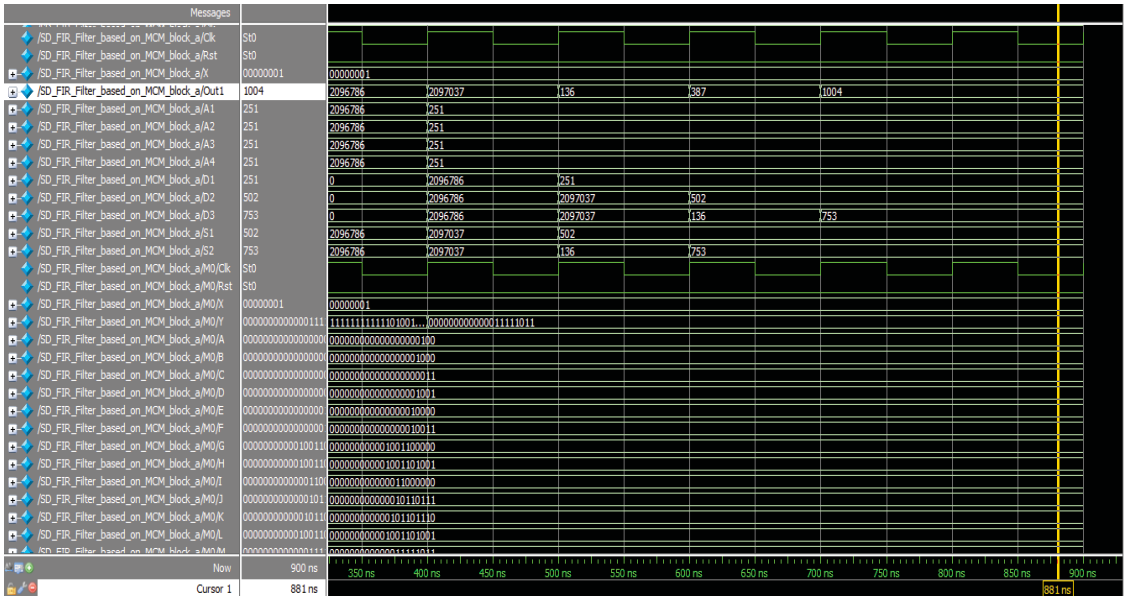
Here in this binary adder tree a direct path is exists for all the input terms and for example both E12 and E8 depends on E0.while E12 and E8 on each pair edge positions depends on the binary adder tree. At the end of the binary tree null operator is assigned as an empty operator with either ADD or SUB at the end of the binary tree.

The coefficients of the FIR filters logic depth is smaller without effecting the filter performance and by scheduling the coefficients in the parallel order it will concurrently operate all the filter coefficients of the adder tree the entire resource of the filter is minimized.

III.SIMULATION RESULT



Simulation result for bit level optimization of adder tree



Simulation results for binary adder tree using MIP modeling

Parameters	Adder tree by greedy scheduling algorithm	Bit level resource optimal adder tree	Binary adder tree by using MIP modeling
Delay	18.207ns	17.210ns	8.042ns
Number of flip flops	3%	2%	1%
Number of 4 input LUTs	4%	4%	1%
Number of IOBs	50%	30%	13%

Table -1 Experiment Result

IV. CONLUSION

In this paper the resource minimization problem has been identified for the MCM block of the direct form of FIR filter in the scheduling of adder tree and here we are representing an algorithm of MIP for the implementation of FIR filter with pipelined and parallel processing for the implementation of any DSP applications for exact bit level resource operation and reducing the number of hardware resources such as adders.

The experimental results of the adder trees shows the reduction of area up to 30% and by reducing the area, delay can also be automatically reduced and it is having the high speed performance operation compared to that of the previously used MCM block of the ADD/SUB networks.

REFERENCES

[1] D. R. Bull and D. H. Horrocks, " Digital filter Primitive operator," *IEEE Proceedings-G*,vol.138, no. 3, pp. 401–412, Jun. 1991.

[2] L. Aksoy, C. Lazzari, E. Costa, P. Flores, and J. Monteiro, " Digit-serial FIR filter designs: Algorithms, architectures, and a CAD tool," *IEEE Trans. Very Large Scale Integration (VLSI) Syst.*, vol. 21, no. 3, pp. 498–511, Mar. 2013.

[3] M. B. Gately, M. B. Yeary, and C. Y. Tang, "Multiple real-constant multiplication, greedy and optimal searches with improved cost model ," in *Proc. IEEE ISCAS*, May 2012, pp. 588–591.

[4] A. G. Dempster and M. D. Macleod, "Use of FIR digital filters in minimum-adder multiplier blocks ," *IEEE Trans. Circuits Syst. II, Digit. Signal Process.*, vol. 42, no. 9, pp. 569–577, 1995.

[5] S. D. S. M. Mehendale and G. Venkatesh, "Synthesis of minimum number of additions with multiplier-less FIR filters ," in *Proc. IEEE ICCAD*, 1995.

[6] R. Hartley and A. Casavant, "Tree- height minimization in pipelined architectures", *in Proc. IEEE ICCAD,Nov.1989*.

[7] M. Kumm, P. Zipf,M. Faust, and C.-H. Chang, "High speed multiple constant multiplication for pipelined adder graphs," in *Proc. IEEE ISCAS*, May 2012, pp. 49–52.

[8] P. K. Meher and Y. Pan, "Mcm- block fir filters based for implementing high speed and low power applications", in Proc. VLSI and System-on-Chip(VLSI-S0C),2011 IEEE/IFIP 19th *Int.Conf.,Oct.2011,pp. 118-121.*