MULTISCIA JOURNALS PUBLISHERS

FRONTIERS IN MECHANICAL, MINING AND MATERIAL ENGINEERING

ISSN: (3065- 4025)

https://multisciajournals.com/journals/index.php/fmmme

editor.fmmme@gmail.com

Frontiers in Mechanical, Mining and Material Engineering
Volume 1 Issue 1 2025

Effective Graph-Based Algorithm Design Using a Modified Carry-Save Adder

Thulasi, Karthik Varma, Rithesh Kumar
Department of MECH

Article Info

Received: 29-01-2025 ***Revised:05-03-2025*** ***Accepted:16-03-2025*** ***Published:26-03-2025***

Abstract— The complexity of many digital signal processing systems is eliminated by the introduction of several well-structured algorithms and architectures for the construction of low complexity bit parallel multiple constant multiplication (MCM) operations. The digit serial MCM purpose has not received much consideration. By altering the carry save adder and applying it to the graph based (GB) algorithm, we discuss in this work the challenge of improving the gate level delay in digit serial MCM architectures. Here, power and latency may be greatly decreased, albeit at the expense of more space. Experimental results demonstrate the ability of the suggested optimization algorithms and digit-serial MCM designs to develop finite impulse response filters and digit-serial MCM operations. Compared to the current way, the suggested method achieves greater speed by reducing the delay.

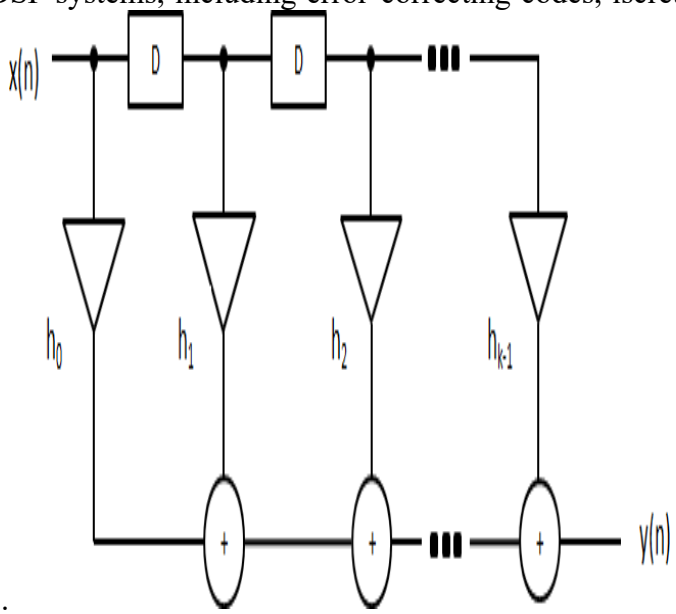
Index Terms— Graph based (GB) algorithm, modified carry save adder (MCSA), finite impulse response (FIR) filters, gate-level delay optimization, multiple constant multiplications.

I. INTRODUCTION

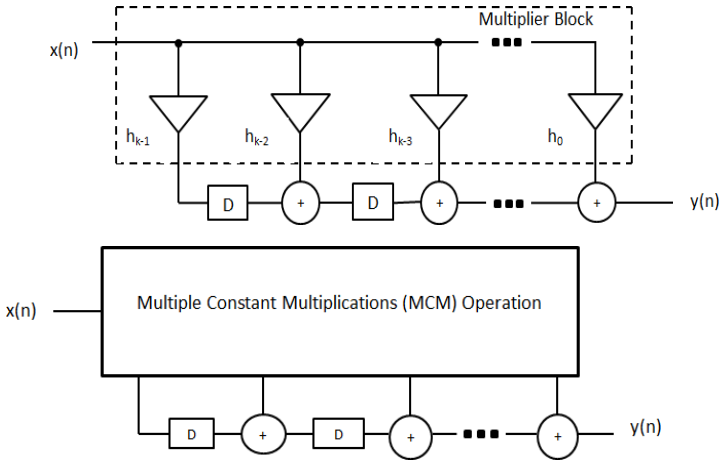
Consuming a shift and add arrangement is the general method for conveying multiplication by a constant multiplication. This approach might be modified by allowing subtraction as well. This technique enhances operation and decreases shift, which are necessary for designing finite impulse response (FIR) filters. These are digital filters with a finite interval response to a unit impulse (unit sample function). Compared to infinite impulse response (IIR) filters, this is different. In digital signal processing systems, finite impulse response (FIR) filters are crucial.

Figure 1 shows the implementations of the direct and transposed-form finite impulse response (FIR) filters. (a) and (b) in that order. Despite having comparable hardware challenges, the transposed version is often chosen because to its improved performance and power economy. Since many constant multiplications are needed, it is acknowledged that multiplying filter coefficients by filter inputs has a significant impact on the design's complexity and performance. This is well known as the operation and performance blockage of

numerous constant multiplications in many different DSP systems, including error-correcting codes, iscrete cosine transforms (DCTs), and rapid Fourier ransforms.



(a)



(b)

(c)

Fig.1.FIR filter implementations. (a) Direct form. (b) Transposed form with generic multipliers. (c) Transposed form with MCM block

Even though area, delay and power efficient multiplier architectures have been proposed, the full flexibility of a mul-tiplier is not necessary for the constant multiplications, since filter coefficients are fixed and determined beforehand by the DSP algorithm. Hence filter coefficients multiplication with input data is generally implemented under a shift-adds archit-ecture where each constant multiplication is realized using a addition/subtraction and shift operation in an MCM operation.

For shift-adds implementation of constant multiplications, a straight forward method and is generally known as digit based recoding, initially defines the constants in binary. Then for each “1” in the binary representation of the constant, according to its bit position, it shifts variables and adds up the shifted variables to obtain the result. As a simple example, consider the constant multiplication 29x and 43x. Their decomposition in binary are listed as F ollows:

$29x = (11101)_{bin}\ x = x << 4 + x << 3 + x << 2 + x$

$43x = (101011)_{bin}\ x = x << 5 + x << 3 + x << 1 + x$

Which requires six addition operations as illustrated in Fig.2.(a)

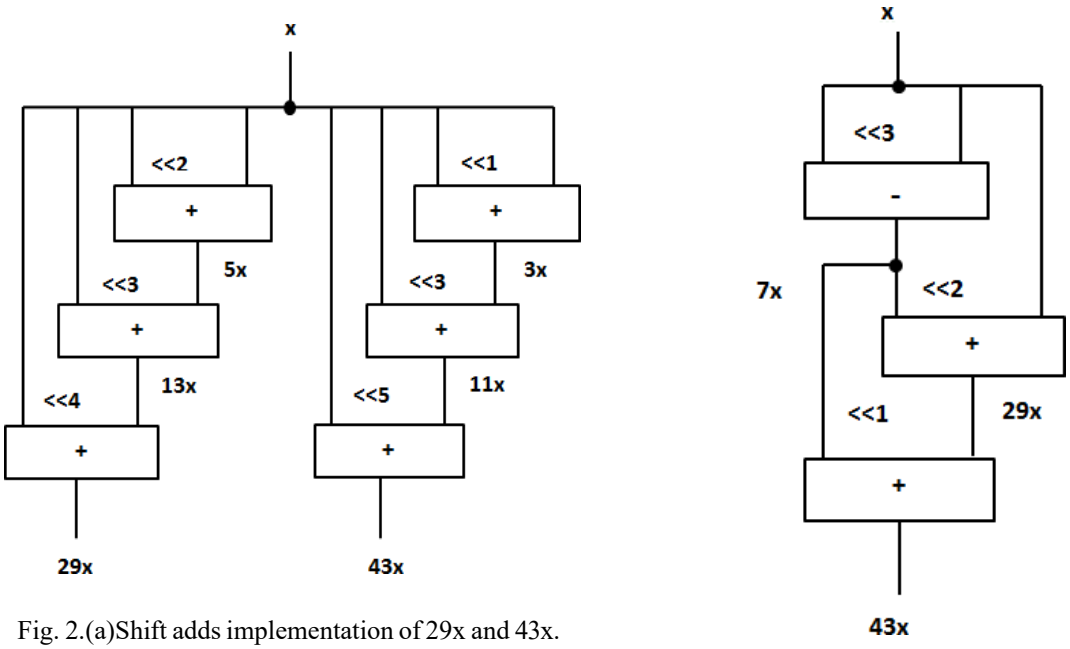


Fig. 2.(a)Shift adds implementation of 29x and 43x.

From the above figure, the shift-adds implementations of 29x and 43x which requires six addition operations for the constant multiplications. The graph based algorithm which is introduced to reduce the addition operations in the shift adds implementations of constant multiplications. These methods are not limited to any particular number representation and consider a larger number of alternative implementations of a constant, yielding better solutions than the shift adds implementation of constant multiplication. These name implies that the operation is minimized by the reduction of graph according to the shift adds implementations of constant multipli-cations in the FIR filters. From this technique the number of repeated sub-expressions can be reduced maximum and produce the efficient result as compared with shift and adds techniques. To ensure this condition Fig.2(b) is compared with Fig.2(a). The graph based algorithm finds a better solution with minimum number of operations by sharing the common partial product 7x in both multiplications. The graph based algorithm for the implementation of 29x and 43x is given in following Fig. 2(b).

Fig. 2.(b)Graph based algorithm of 29x and 43x.

From the above figure addition operation is reduced by introducing subtraction operation and sharing the sub-expression of constant multiplication. Note that the partial product 7x cannot be extracted from the binary representation of 43x in the shift and adds implementation of constant multiplication. In this project we propose the modified carry save adder in the graph based algorithm to increase the gate level speed of constant multiplications of FIR filters. Previously adder is used in the graph based algorithm for the operation of constant multiplications in FIR filters. The proposed method comparatively shows the best result in the gate level delay of constant multiplications. Then for the implementaion of carry save adder in the graph based algorithm is made efficient compared with the existing method of graph based algorithm.

Experimental results on a comprehensive set of instances shows that the solutions of algorithms introduced in this project lead to significant improvements of speed in the graph based algorithm designs, compared to those obtained using the algorithms designed for the multiple constant multiplication (MCM) problem. A large number of constant multiplications are required when filter coefficients are multiplied with the filter inputs, these large number of constant multiplications are refered as multiple constant multiplication (MCM). The backround concepts explains about the implementation of modified carry save adder in the graph based algorithm and discusses about the result compared with the implementation of adder in the graph based algorithm. Finally the result gives the better solution.

The rest of this paper proceeds as follows. Section II gives the background concepts. The experimental

results are presented in Section III and conclusions in Section IV.

II. BACKGROUND

(A) Number Representation

The binary representation decomposes a number in a set of additions of powers of 2. The representation of numbers using a signed digit system makes use of positive and negative digits, {1, 0, −1}. The canonical signed digit (CSD) representation is a signed digit system that has a unique representation for each number and verifies the following main properties:

1) two nonzero digits are not adjacent; and 2) the number of nonzero digits is minimum. Any n digit number in CSD has at most [(n+1)/2] nonzero digit and, on average, the number of nonzero digits is reduced by 33% when compared to binary. The minimal signed digit (MSD) representation is obtained by dropping the first property of the CSD representations under MSD, including its CSD representation, but all with a minimum number of nonzero digits.

Consider the constant 23 defined in six bits. Its binary representation 010111 includes four nonzero digits it is represented as 101001 in CSD, and both 101001 and 011001 denote 23 in MSD using three nonzero digits (where 1 stands for -1).

(B) Modified Carry Save Adder

The 16-bit conventional CSA is shown in Fig 3. It has 17-half adders and 15-full adders. Since the ripple carry adder (RCA) is used in the final stage, this structure yields large carry propagation delay. To reduce this delay, the final stage of CSA is divided into 5 groups as shown in Fig 4.

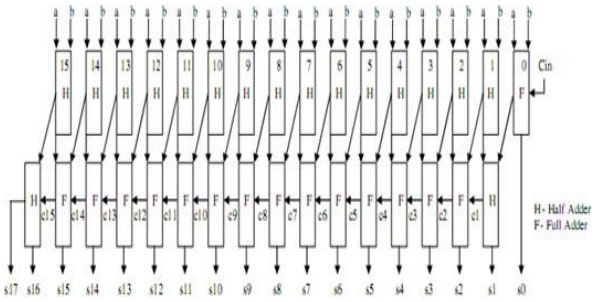


Fig. 3. Conventional carry save adder

The first group includes n 2log 1+ -bit value and other groups includes n 2 log –bit value, where n is the bit size of the adder. The first group of output s[4:0] are directly assigned as the final output; the second group {c7,x[7:5]} manipulates the partial result by considering c4 is 0; the third group {c10,x[10:8]} manipulates the partial result by considering c7 is 0; the fourth group {c13,x[13:11]} manipulates the partial result by considering c10 is 0 and the fifth group x[17:14] manipulates the partial result by considering c13 is 0. By this method carry save adder is modified for GB algorithm.

The divided groups are listed below:

i). {c4,s[4:0]}
ii). {c7,x[7:5]}
iii). {c10,x[10:8]}
iv). {c13,x[13:11]}
v). x[17:14]

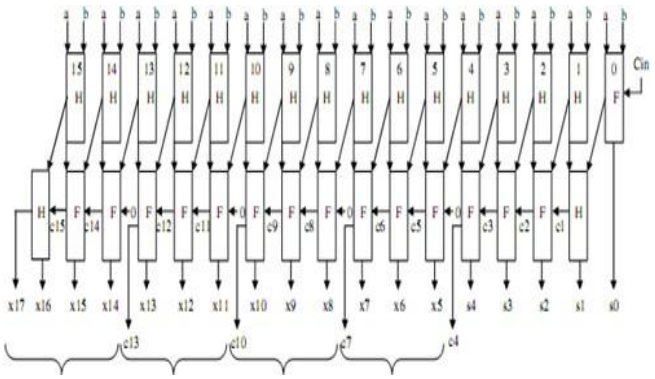


Fig. 4. Modified carry save adder Depending on c4 of the first group, the second group mux gives the final result without the carry propagation delay from c4 to c7; depending on c7 of the second

group final result, the third group mux gives the final result without the carry propagation delay from c7 to c10; depending on c10 of the third group final result, the fourth group mux gives the final result without the carry propagation delay from c10 to c13 and depending on c13 of the fourth group final result, the fifth group mux gives the final result without the carry propagation delay from c13 to s17.

The main advantage of this logic is that each group computes the partial results in parallel and the multiplexers are ready to give the final result "immediately" with the minimum delay of the mux. When the C_{in} of each group arrives, the final result will be determined "immediately". Thus the maximum delay is reduced in the carry propagation path. This same logic has used for 32 and 64-bit adder structures to achieve higher speeds. The area indicates the total cell area of the design and the total power is sum of leakage power, internal power, net power and dynamic power. The proposed result shows that the CLA and CSA have reduced area than MCSA. But the speed of the MCSA architecture has significantly improved when compared to the conventional CSA and CLA. In graph based algorithm MCSA is used for the substitution of addition operation to increase the speed and power, because MCSA is more advance than CSA and CLA.

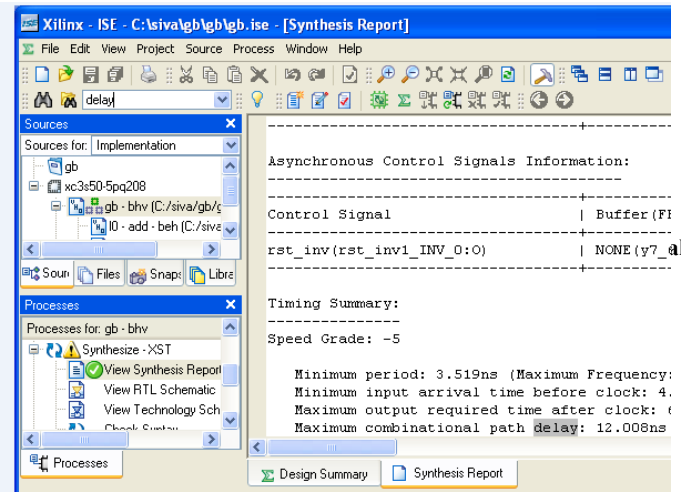
III. EXPERIMENTAL RESULT

The experimental result is given for the implementation of modified carry save adder in the graph based algorithm. Comparision of modified carry save adder with the normal adder is shown in table 1. The synthesis result is given below for the implementation of normal adder in the graph based algorithm. These results which gives the clear idea about the graph based algorithm.

SYNTHESIS RESULT

The graph based algorithm is given for the efficient carry save adder and by the reduction of delay we can improve the speed of the FIR Filter from the implementation of graph based algorithm. The proposed result shows that the CLA and CSA have reduced area than MCSA in graph based (GB) algorithm. But the speed of the MCSA architecture has been significantly improved and has the least value of power-delay product compared to the conventional CSA and CLA.

Flow chart.1



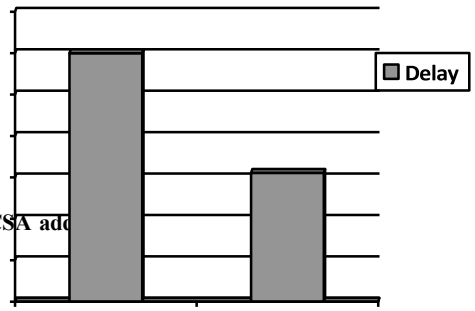


Fig. 6. Flow chart of comparision for delay

The synthesis result for delay is given in the above fig. 5.
It shows the required delay for the implementation of normal adder in graph based algorithm. From the above result, to increase the speed by reducing delay, the modified carry save adder is used in graph based algorithm. The power can also be redused in MCSA.The flow chart and the table is given for the comparision of experimental results for normal adder with modified carry save adder in the implementation of graph based (GB) algorithm is shown as follows.

Table. 1

Work	GB algorithm using normal adder	GB algorithm using Modified carry save adder
power	0.222	0.215
Delay	12.008ns	6.216ns

From table.1 the comparision of an implementation of modified carry save adder in the graph based algorithm with the normal adder is shown. Here we can reduce the delay and increase the speed and we can also reduce the power as shown in above table. From the flow chart the reduction of power and delay can be achieved greatly as shown.

Fig.7. Flow chart of comparision for power

From the above given flow chart .1 and .2. Delay and power can be reduced as much as possible compared with the implemenation of normal adder in the graph based algorithm.

IV CONCLUSION

Therefore, in our project, the time is minimized and the power is minimized as well by implementing a modified carry save adder in the graph-based method. The graph-based (GB) approach is provided to enhance shift and include constant multiplications. The graph-based (GB) method adds operations and decreases the amount of shifts. By using the modified carry save adder (MCSA) in the graph-based algorithm, as shown in the flow chart, the suggested method reduces the power from 0.222 to 0.215 compared to the conventional adder as stated in the table and the maximum delay from 12.008 ns to 6.216 ns to reach faster speeds. 1. Xilinx 10.1 has been used to generate and implement the suggested method. The MCSA architecture's speed has greatly increased, and the experimental outcome also reuses electricity.

REFERENCES

[1] A. Dempster and M. Macleod, "Minimum-adder multiplier blocks in FIR digital filters," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 42, no. 9, pp. 569–577, September 1995.
In February 1964, C. Wallace published "A proposal for a fast multiplier" in IEEE Trans. Electron. Compute., vol. 13, no. 1, pp. 14–17.
IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 8, no. 4, pp. 419–424, Aug. 2000; H. Nguyen and A. Chatterjee, "Number-splitting with shift-and-add decomposition for power and hardware optimization in linear DSP synthesis."

[4] "Digital filter synthesis based on minimal signed digit representation," I.-C. Park and H.-J. Kang, Proc. DAC, 2001.
A methodical approach to designing digit-serial signal processing architectures, by K. Parhi, IEEE Trans. Circuits Syst., vol. 38, no. 4, pp. 358–375, April 1991.
[6] L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Gate-level metrics for optimizing area in digital FIR filters," in Proc. DAC, 2007, pp. 420–423.
IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 6, pp. 1013–1026, June 2008; L. Aksoy, E. Costa, P. Flores, and J. Monteiro, "Exact and approximate algorithms for the optimization of area and delay in multiple constant multiplications."
[8] L. Aksoy, E. Gunes, and P. Flores, "Search algorithms for the multiple constant multiplications problem: Exact and approximate," published in August 2010 in J. Microprocess. Microsyst., vol. 34, no. 5, pp. 151–162.
[9] "Efficient shift-adds design of digit-serial multiple constant multiplications," L. Aksoy, C. Lazzari, E. Costa, P. Flores, and J. Monteiro, Proceedings of the Great Lakes Symp. VLSI, 2011, pp. 61–66.
[10] Digital Arithmetic, M. Ercegovac and T. Lang. Morgan Kaufmann, San Mateo, CA, 2003.
[11] P. Barth, "An enumeration algorithm for linear pseudo-Boolean optimization based on Davis-Putnam," Tech. Rep. MPI-I-95-2-003, Max-Planck-Institut für Informatik, Saarbrät, Germany, 1995.

[12] "An exact algorithm for the maximal sharing of partial terms in multiple constant multiplications," in Proc. Int. Conf. Comput.-Aided Design, Nov. 2005, pp. 13–16, by P. Flores, J. Monteiro, and E. Costa.
[13]R. Hartley, "Canonic signed digit multipliers for subexpression sharing in filters," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 43, no. 10, pp. 677–688, Oct. 1996.
[14] T. Larrabee, "Generating test patterns with Boolean satisfiability," Vol. 11, no. 1, pp. 4–15, IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., January 1992.
Proc. ASPDAC, 2008, pp. 119–124; Y.-H. Ho, C.-U. Lei, H.-K. Kwan, and N. Wong, "Global optimization of common subexpressions for multiplierless synthesis of multiple constant multiplications."
"Design of digit-serial FIR filters: Algorithm, Architectures, and a CAD tool," Levent Aksoy, Cristiano Lazari, IEEE Trans. On very large scale integration (VLSI) systems, vol. 21. No.