International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 11
July 2016

Design of Digit-Serial FIR Filters using CSD Adder Graph Multiplier

* N.MANASA ** V.Ramya

*PG Scholar, Dept of ECE ,Vaagdevi College Of Engineering, Warangal, Telangana.

** Associate Professor, Dept of ECE, Vaagdevi College Of Engineering, Warangal, Telangana.

Abstract:

Finite Impulse Response (FIR) filters are widely applied in multi-standard wireless communications. A novel efficient algorithms and architectures have been introduced for the design of low complexity bit-parallel multiple constant multiplications (MCM) operation which dominates the complexity of many digital signal processing systems. In digit-serial MCM design that offers low complexity MCM operations that offers a low delay. In this previous design a MCM operations performed by CSE algorithm. A new greedy CSD adder graph multiplier based algorithm based on Canonic Signed Digit (CSD) representation of coefficients multipliers for implementing low complexity higher order FIR filters.

Keywords: FIR-Finite Impulse Response Filters, CSD – Canonic Signed-Digit Multiplier, CSE- Common Sub-expression Elimination Algorithms, MSD – Minimum Signed-Digit Multiplier, MAG-Minimum Added Graph Multiplier.

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 11
July 2016

I. INTRODUCTION

Finite impulse response (FIR) filters are of great importance in digital signal processing (DSP) systems since their characteristics in linear-phase and feed-forward implementations make them very useful for building stablehigh-performance filters. The direct and transposed-form FIRfilter implementations are illustrated in Fig. 1(a) and (b), respectively. Although both architectures have similar complexity in hardware, the transposed form is generally preferred because of its higher performance and power efficiency [1]. The multiplier block of the digital FIR filter in its transposed form [Fig. 1(b)], where the multiplication of filter coefficients with the filter input is realized, has significant impact on the complexity and performance of the design because a large number of constant multiplications are required. This is generally known as the multiple constant multiplications(MCM) operation and is also a central operation and performance bottleneck in many other DSP systems such as fast Fourier transforms, discrete cosine transforms (DCTs), and error-correcting codes. Although area-, delay-, and power-efficient multiplier architectures, such as Wallace [2] and modified Booth [3] multipliers, have been proposed, the full flexibility of multiplier is not necessary for the constant multiplications, since filter coefficients are fixed and determined beforehand by the DSP algorithms [4]. Hence, the multiplication of filter coefficients with the input data is generally implemented under a shift-adds architecture [5], where each constant multiplication is realized using addition/subtraction and shift operations in an MCM operation [Fig. 1(c)].

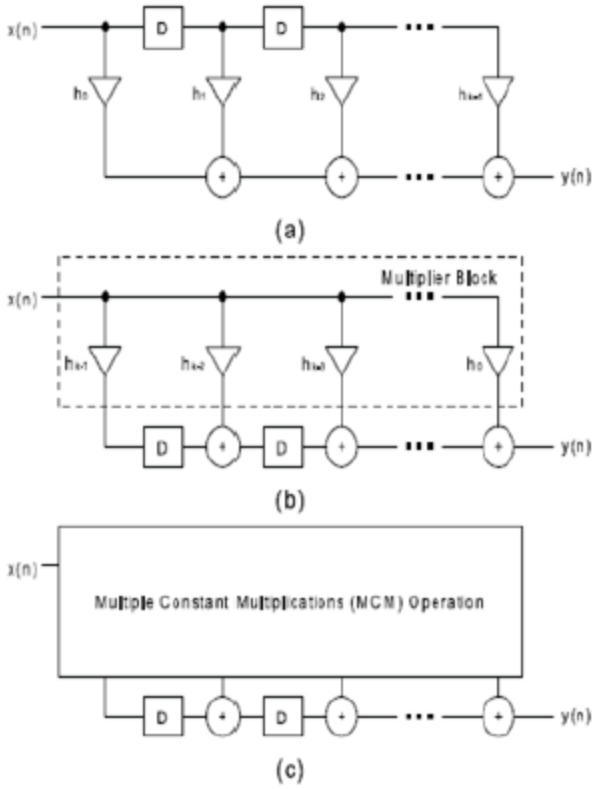

Fig.1. FIR Filter implementations (a)Direct form. (b)Transposed form. (c)Transposed form with an MCM block.

For the shift-adds implementation of constant multiplicat-ions, a straightforward method, generally known as digit based recoding [6], initially defines the constants in binary. Then, for each "1" in the binary representation of the constant, according to its bit position, it shifts the variable and add sup the shifted variables to obtain the result. As a simple example, consider the constant multiplications 29x and 43x.Their decompositions in binary are listed as follows:

$29x = (11101)\ bin_x = x <<4 + x <<3 + x <<2 + x$
$43x = (101011)\ bin_x = x <<5 + x <<3 + x <<1 + x$

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

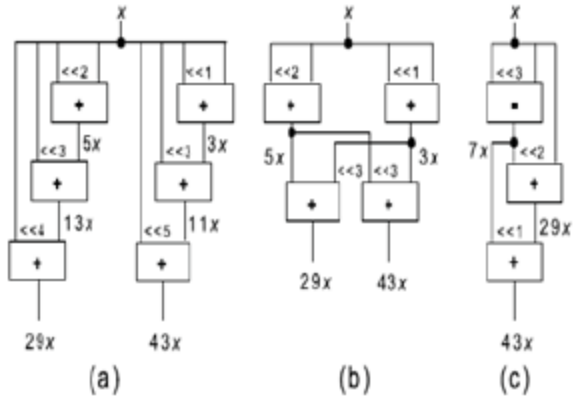

Fig.2.Shift-adds implementations of 29x and 43x (a) without partial product sharing [6] and with partial product sharing. (b) Without CSE algorithm [9]. (c) Exact GB algorithm.

Exact GB algorithm.

However, the digit-based recoding technique does not exploit the sharing of common partial products, which allows great reductions in the number of operations and, consequently, in area and power dissipation of the MCM design at the gate level. Hence, the fundamental optimization problem, called the MCM problem, is defined as finding the minimum number of addition and subtraction operations that implement the constant multiplications. Note that, in bit-parallel design of constant multiplications, shifts can be realized using only wires in hardware without representing any area cost. The algorithms designed for the MCM problem can be categorized in two classes: common sub expression elimination (CSE) algorithms [7]–[9] and graph-based (GB) techniques[10]–[12]. The CSE algorithms initially extract all possible sub expressions from the representations of the constants when they are defined under

binary, canonical signed digit (CSD) [7], or minimal signed digit (MSD) [8]. Then, they find the "best" sub expression, generally the most common, to be shared among the constant multiplications. The GB 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 CSE algorithms, as shown in [11] and [12].

Returning to our example in Fig. 2, the exact CSE algorithm of [9] gives a solution with four operations by finding the most common partial products 3x = (11)binx and 5x = (101) binx when constants are defined under binary, as illustrated in Fig.2(b). On the other hand, the exact GB algorithm [12] finds a solution with the minimum number of operations by sharing the common partial product 7x in both multiplications, as shown in Fig. 2(c). Note that the partial product 7x =(111) binx cannot be extracted from the binary representation of 43x in the exact CSE algorithm [9].However, all these algorithms assume that the input data x is processed in parallel. On the other hand, in digit-serial arithmetic, the data words are divided into digit sets, consisting of d bits that are processed one at a time [13]. Since digit serial operators occupy less area and are independent of the data word length, digit-serial architectures offer alternative low complexity designs when compared to bit-parallel architectures. However, the shifts require the use of D flip-flops, as opposed to the bit-parallel MCM design where they are free in terms of hardware. Hence, the high-level algorithms should take into account the sharing of shift operations as well as the sharing of addition/subtraction operations in digit-serial MCM design. Furthermore, finding the minimum number of operations realizing an MCM operation does not always yield an MCM design with optimal area at the gate level [14]. Hence, the high-level algorithms should consider the implementation cost of each digit-serial operation at the gate level.

In this paper, we initially determine the gate-level implementation costs of digit-serial addition, subtraction, and left shift operations used in the shift-adds design of digit-serial MCM operations. Then, we introduce the exact CSE algorithm [15] that formalizes the gate-level area optimization problem as a 0–1 integer linear programming (ILP) problem when constants are defined under a particular number

representation. We also present a new optimization model that reduces the 0–1 ILP problem size significantly and, consequently, the runtime of a generic 0–1 ILP solver. Since there are still instances which the exact CSE algorithm cannot handle, we describe the approximate GB algorithm [16] that iteratively finds the "best" partial product which leads to the optimal area in digit-serial MCM design at the gate level. This paper also introduces a computer-aided design (CAD) tool called SAFIR which generates the hardware descriptions of digit-serial MCM operations and FIR filters based on design architecture and implements these circuits using a commercial logic synthesis tool. In SAFIR, the digit-serial

constant multiplications can be implemented under the shift adds architecture, and also can be designed using generic digit serial constant multipliers [17].

In this proposed different graph based multipliers types i.e. CSD – canonic signed-digit multiplier, MSD – minimum signed-digit multiplier MAG – minimum adder graph multiplier, CSDAG – CSD adder graph multiplier. They are used for low complexity, low power and low areaapplications. The section II explains the complexity of serial constant multipliers. Section III explains graph based multipliers. Section IV explains results and analysis.

II. COMPLEXITYOF SERIALCONSTANT MULTIPLIERS

In this chapter, the possibilities to minimize the complexity of bit-serial single-constant multipliers are investigated [57]. This is done in terms of the required number of building blocks, which includes adders and shifts. The multipliers are described using a graph representation. It is shown that a minimum set of graphs, required to obtain optimal results given certain restrictions, can be found. In the case of single-constant multipliers, the number of possible solutions can be limited because of the finite number of graph topologies. However, if a shift-and-add network realizing several coefficients is required, a multiple-constant multiplication (MCM) problem is obtained. Different heuristic algorithms can then be used to reduce the complexity, by utilizing the redundancy between the coefficients. Two algorithms suitable to achieve efficient realization of MCM using serial arithmetic are presented [56],[62],[66]. It is shown that the new algorithms reduce the total

complexity significantly. Furthermore, we study the trade-offs in implementations of FIR filters using MCM and digit-serial arithmetic. Comparisons considering area, speed, and energy consumption, with respect to the digit-size, are performed[61],[67].

II. GRAPH MULTIPLIERS

In this section, different types of single-constant graph multipliers will be defined, with respect to constraints on adder cost and throughput. Furthermore, the possibilities to exclude some graphs from the search space are examined. The investigation covers all coefficients up to 4095 and all types of graph multipliers containing up to four adders. All possible graphs, using the representation discussed in Section 3.1, for adder costs from 1 to 4are presented in Fig.3 [24].Note that although bit-serial arithmetic will be assumed for the multipliers, results considering adder and flip-flop costs are generally also valid for any digit-serial implementation. However, the numbers of registers that are required to perform pipelining depend on the digit-size. Furthermore, the cost difference between adders and shifts becomes higher for larger digit-sizes, since the number of full adders increases linearly while the number of flip-flops is constant. Hence, such trade-offs are mainly of interest for small digit-sizes.

A. Multiplier Types

Different multiplier types can be defined based on the requirements considering adder cost, flip-flop cost, and pipelining. The types that will be discussed here are described in the following.

CSD – Canonic Signed-Digit multiplier: Multiplier based on the CSD representation, as discussed in Section 4.1, withan added cost equal to one less than the number of nonzero digits.

MSD – Minimum Signed-Digit multiplier: Similar to the CSD multiplier and requires the same number of adders, but can in some cases decrease the flip-flop cost by using other MSD representations, which were discussed in Section 3.1.

MAG – Minimum Adder Graph multiplier: Graph multiplier that is based on any of the topologies in Fig. 4.1 and, for any given coefficient, has the lowest possible adder cost.

CSDAG – CSD Adder Graph multiplier: Similar to the MAG multiplier, but may use the same number of adders as the corresponding CSD/MSD multiplier, and can by that reduce the flip-flop cost

.

B. Example

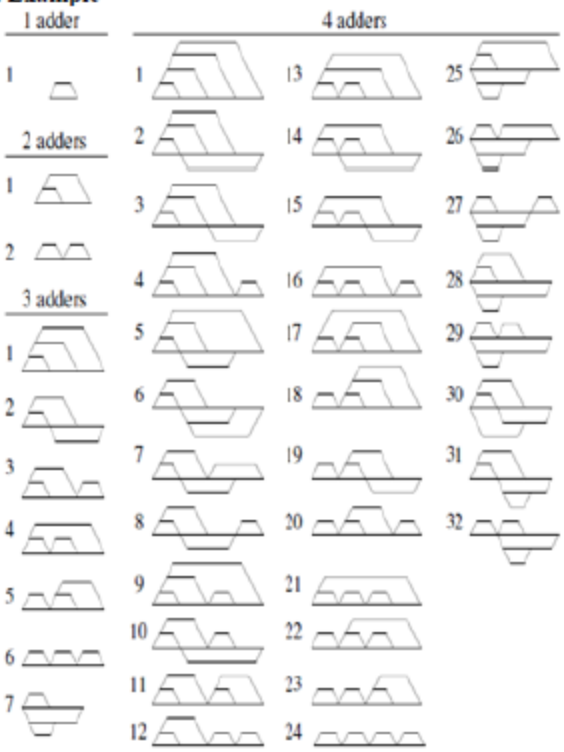

Fig.3. Possible graph topologies for an adder cost up to four.

To describe the difference between the defined multiplier types, corresponding realizations of the coefficient 2813, which has the CSD representation 1010100000101, are shown in Fig.4. There are other possible solutions for all types except the CSD multiplier. However, note that the values corresponding to the nonzero digits in the CSD representation can be added in different orders, resulting in other structures. Since this may eliminate the pipeline feature, the basic structure used in Fig.4 (a) will be assumed for CSD multipliers. The adder costs for the multipliers in figs. 4. (a), (b), (c), and (d) are 4, 4, 3, and 4, respectively. The flip-flop costs are 12, 11, 11 and 10. This implies that it is possible to save either two shifts, or one adder and one Shift compared to the CSD multiplier. Note that shifts may be shared as discussed in Section 1.5.2, for example, the two 27-edges in Fig. 4.(d). Pipelined CSDAG and MAG can be obtained from the multipliers. 4.2 (b) and (c) with an extra cost of 0 and 1 register,

respectively. Note that the flip-flop cost will include both shifts and pipelining registers, since both correspond to a single flip-flop in bit-serial arithmetic.

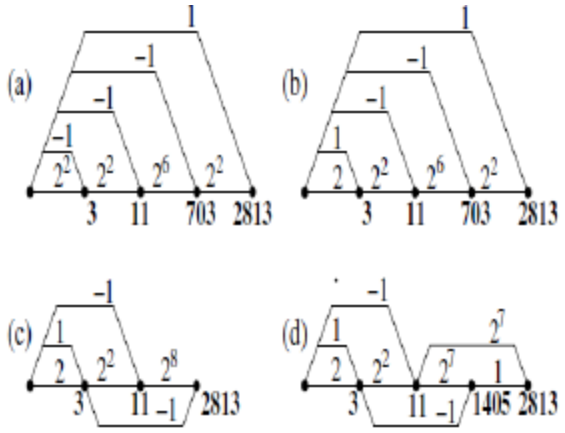

Fig.4.Differentrealizations of the coefficient 2813. (a) CSD, (b) MSD, (c) MA G, and (d) CSDAG.

IV. RESULTS AND ANALYSIS

Results and analysis of this paper is shown in bellow Figs.5 and 6.

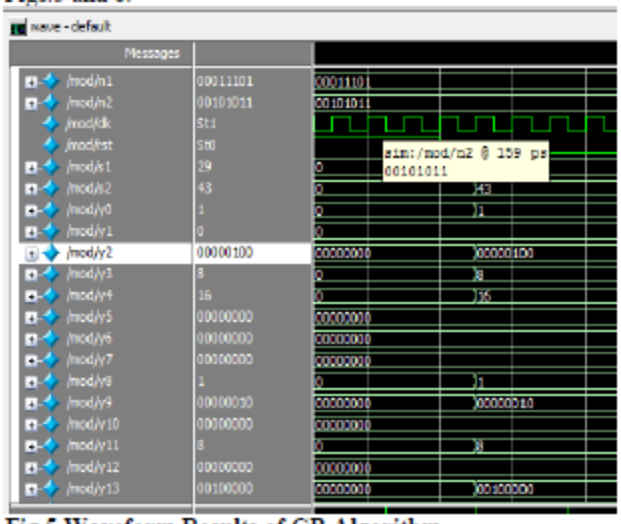

Fig.5.Waveform Results of GB Algorithm.

TABLE I: The comparison Result of CSE and BE Algorithm

Algorithm name	Delay(ns)	Area(%)
CSE	2.780	33
BE	2.58	30

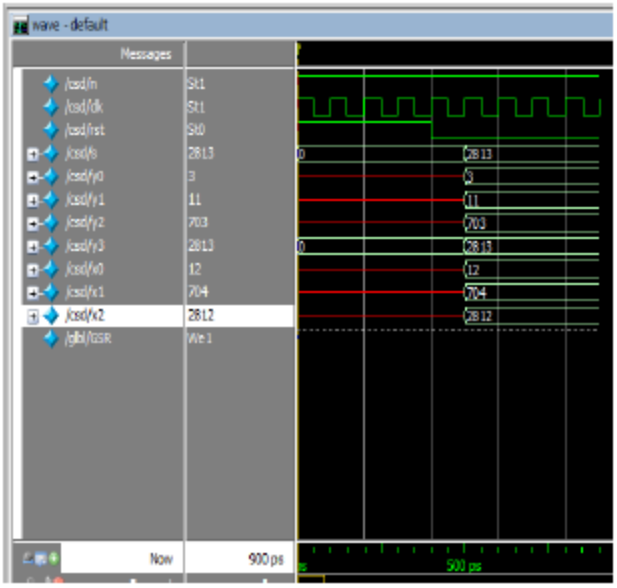

Fig.6. Waveform Results of CSDAG Algorithm.

TABLE II: The Comparison Result of MAG and CSDAG

Graph based Multipliers	Delay(ns)	Area (%)
MAG	2.58	24
CSDAG	2.58	19

The comparison tables and waveforms show the analysis of different algorithms and different graph based multipliers. In this BE algorithm shows efficient results compare to CSE algorithms. Again analyses in graph based multipliers i.e. CSD, MSD, and MAG, CSDAG. The CSDAG shows good area and delay in table 2.

V.CONCLUSION

The proposed new approach is CSDAG for implementing reconfigurable higher order filters with low complexity. The proposed CSDAG method make use of architecture with fixed number of multiplexers and the reduction in complexity is achieved by applying the greedy BE algorithm. The CSDAG architecture results in high speed filters and low area and thus low power filter implementations. The CSDAG also provides the flexibility of changing the filter coefficient word lengths dynamically. The proposed reconfigurable architectures can be easily modified to employ any graph based (BE) method, which results in architectures that offers good area and power reductions

and speed improvement reconfigurable FIR filter implementations.

VI. REFERENCES

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

[2] C. Wallace, "A suggestion for a fast multiplier," IEEE Trans. Electron.Comput., vol. 13, no. 1, pp. 14–17, Feb. 1964.

[3] W. Gallagher and E. Swartz lander, "High radix booth multipliers usingreduced area adder trees," in Proc. Asilomar Conf. Signals, Syst. Comput.,vol. 1. Pacific Grove, CA, Oct.–Nov. 1994, pp. 545–549.

[4] J. McClellan, T. Parks, and L. Rabiner, "A computer program fordesigning optimum FIR linear phase digital filters," IEEE Trans. AudioElectroacoust., vol. 21, no. 6, pp. 506–526, Dec. 1973.

AUTHOR 1:-

*N.MANASA completed her B.Tech SUMATHI REDDY INSTITUTE OF TECHNOLOGOGY FOR WOMEN in 2013 and completed M.Tech in VAAGDEVI COLLEGE OF ENGINEERING in 2015

AUTHOR 2:-

**V.RAMYA working as associate professor in vaagdevi college of engineering