

Optimization of Area and Delay Using Improved Signed Digit Representation Approach for Constant Vector Multiplication

K.Sudheer, K Manjunath

Research Student, Department of ECE, Sree Rama Institute of Technical Education, Tirupathi, India

Assistant Professor, Department of ECE, Sree Rama Institute of Technical Education, Tirupathi, India

ABSTRACT: Real-time implementation of many digital signal processing (DSP) algorithms and multimedia applications are limited by the available speed, energy efficiency, and area requirement of multiplication. The multiplier-free implementation of the constant vector multiplication is reexamined. A novel improved signed digit representation technique is proposed to overcome the two main drawbacks of the current multiplier-free techniques: 1) computational redundancy and 2) circuit irregularity. The fundamental difference between the proposed technique and the existing multiplier-free techniques is a novel optimization framework based on vector decomposition. The constant vector is decomposed into two terms: a “public” vector and a “private” matrix which consist of the public operations shared by all the entries and the private operations of each individual entry respectively. In this way, the overall data flow can be divided into two regular steps: multiplied by the “public” vector firstly then by the “private” matrix. The computational complexity reduction task is then achieved by minimizing the length of the “public” vector and the number of operations in the “private” matrix. Experimental results demonstrate that the proposed technique outperforms the existing multiplier-free techniques in less operations and more regular circuit structure. In proposed system using Brent Kung adder (BKA) and Vedic multiplier. The proposed scheme coded in Verilog HDL and simulated using Xilinx 14.3.

KEYWORDS: CSD, vector multiplication, Signed-Digit.

I. INTRODUCTION

Multiplication is a very important operation for digital computing systems, process controllers, and signal processors. Multiplication involves two basic steps: The first one is generation of partial products and the second step is their accumulation. Therefore, the efficiency of multiplication is improved by reducing the number of partial products or accelerating their

accumulation. Besides, the performance of partial products accumulation depends on the numbering system used to represent the operands. Although, Conventional weighted binary number system like two's complement representation is widely used in arithmetic units, the obtained carry propagation chain in the conventional number system increases the delay of multiplication operation. Therefore, the performance is reduced. In n-bit fast adder for conventional binary number system, c_i the input carry to position i , is a function of all inputs from 0 to i , when $(i=0, 1, \dots, n-1)$. The dependency results in increasing the multiplication delay which is reliant on the size of the operands, in the worst case. Binary signed-digit number representation is an appropriate numbering system to restrict or eliminate the carry propagation chain in addition by eliminating the dependency between output and input carries. In digital circuits, numbers are represented as a string of bits, using the logic symbols 0 and 1. Normally the processed data are assumed to take values in the range $[-1, 1]$.

Approximate computing has been motivated by the large and growing class of applications that demonstrate inherent error resilience, such as DSP, multimedia (images/audio/video), graphics, wireless communications, and emerging workloads such as recognition, mining, and synthesis.

The requirement for an exact computation is relaxed due to several factors: (1) the limited perceptual capability of humans (e.g., audio, video, graphics due to the ability of the human brain to “fill in” missing information and filter out high-frequency patterns), (2) a golden result is difficult to define or does not exist (e.g., web search, data analytics), (3) users are willing to accept approximate but good-enough results, and (4) noisy inputs.

The class of methods and techniques referred to as approximate computing usually relax the requirement of exact equivalence between the specification and implementation of a computing system. This relaxation allows trading the accuracy of numerical outputs for reductions in area, delay, and/or power dissipation of the design.

In the last two decades, many design techniques, circuits, and algorithms have been introduced for approximate computing. The reader is referred to, for detailed surveys on approximate, stochastic, and probabilistic computing. The approaches for the design of approximate DSP systems can generally be grouped in three categories: (i) transistor level, (ii) gate level, and (iii) algorithmic level.

Multiplier less approaches, such as distributed arithmetic or shift/add using CSD, can be employed to implement digital filters with constant coefficients. However, for filters with coefficients that are not known a priori, such as adaptive filters, general multipliers are needed, and they are often implemented using shift-and-add operations based on binary representation. Canonical-signed-digit (CSD) representation and minimum signed-digit (MSD) representation, both of which have a minimum number of nonzero digits in their representations, are frequently used to reduce hardware complexity and increase throughput.

The development of fast CSD and MSD conversion algorithms has been the focus of much effort. In 1951, Booth's recoding was presented to efficiently multiply two numbers using recoding multipliers. In 1960, Reitwiesner developed an algorithm to convert two's complement numbers to CSD.

Although all the existing techniques can reduce the computational redundancy significantly, the regularity of the circuit may be sacrificed. For example, in the greedy-based techniques, any vertices of the data flow can be used as an intermediate result for the computation of other vertices. The variant add-paths for different entries affect the routability of the circuit and make the structured layout style intractable.

In this paper, we propose a novel improved signed digit (ISD) representation approach for the constant vector multiplication. The fundamental difference between the proposed technique and the existing multiplier-free techniques is that the circuit regularity is considered jointly with the redundancy reduction process. We propose a novel optimization framework based on vector decomposition. To be specific, in the new framework, the constant vector is decomposed into two terms: a “public” vector and a “private” matrix. In this way, the multiplication of the input signal and the constant vector is achieved by a successive multiplication of the “public” vector and “private” matrix. The “public” vector generates a set of regular intermediate vertices of the data flow which can be reused for each entry. The overall computational redundancy can be reduced by minimizing the length of the “public” vector and the number of operations of the “private” matrix. Then we propose a greedy strategy to solve the vector decomposition problem.

II. RELATED WORK

i) Number Representation

The binary representation decomposes a number in a set of additions of powers of two. The representation of numbers using a signed digit system makes the use of positive and negative digits. Thus, an integer number represented in the binary signed digit (BSD) system using n digits can be written as $\sum_{i=1}^{n-1} d_i 2^i$, where $d_i \in \{1, 0, \bar{1}\}$ and $\bar{1}$ denotes -1 . The BSD system is a redundant number representation system, e.g., both 0101 and $10\bar{1}\bar{1}$ correspond to 5.

The canonical signed digit (CSD) representation, a subset of BSD, has a unique representation for each number and verifies the following two main properties: i) two nonzero digits are not adjacent; ii) the number of nonzero digits is minimal. Any n digit number in CSD has at most $\lceil (n+1)/2 \rceil$ nonzero digits, and on average, the number of nonzero digits is reduced by 33% when compared to binary. This representation is widely used in the multiplierless design of constant multiplications since it reduces the hardware requirements due to the minimum number of nonzero digits.

As an example, consider the constant 23 defined in six bits. The representation of 23 in binary, 010111, includes 4 nonzero digits. The constant is represented as $10\bar{1}00\bar{1}$ in CSD using 3 nonzero digits.

ii) Problem Definitions

Here, we initially present the problem of optimizing the number of operations in a CMVM design and then, introduce the problem of optimizing the number of operations under a delay constraint.

Optimization of the Number of Operations Given an $m \times n$ constant matrix C with $c_{jk} \in \mathbb{Z}$ and an $n \times 1$ variable vector X with $x_k \in \mathbb{Z}$, the multiplication of C by X is a linear transformation from $\mathbb{Z}^n$ to $\mathbb{Z}^m$ and each linear transform can be computed as

$$y_j = \sum_{k=1}^n c_{jk} x_k$$

where j and k range from 1 to m and n , respectively. Thus, the problem of optimizing the number of operations in linear transforms can be defined as follows:

Definition 1. cmvm problem. Given $Y = \{y_1, \dots, y_m\}$, a set of linear transforms, find the minimum number of addition and subtraction operations that generate the linear transforms.

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 3 & 11 \\ 5 & 13 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$y_1 = 3x_1 + 11x_2$$

$$y_2 = 5x_1 + 13x_2$$

(a)

$$\begin{aligned} y_1 &= 3x_1 + 11x_2 = (11)_{bin}x_1 + (1011)_{bin}x_2 \\ &= x_1 + x_1 \ll 1 + x_2 + x_2 \ll 1 + x_2 \ll 3 \\ &= x_1 + 2x_1 + x_2 + 2x_2 + 8x_2 \\ y_2 &= 5x_1 + 13x_2 = (101)_{bin}x_1 + (1101)_{bin}x_2 \\ &= x_1 + x_1 \ll 2 + x_2 + x_2 \ll 2 + x_2 \ll 3 \\ &= x_1 + 4x_1 + x_2 + 4x_2 + 8x_2 \end{aligned}$$

(b)

Figure. 1. (a) A CMVM realizing $y_1 = 3x_1 + 11x_2$ and $y_2 = 5x_1 + 13x_2$; (b) Decomposition of the constants in the linear transforms in binary.

A straightforward way for the multiplierless realization of linear transforms, generally known as the digit-based recoding method [25], is to define the constants in binary, and for each 1 in the binary representation of the constant, is to shift the variable and add up the shifted variables. As a simple example, consider the multiplication of a constant matrix by a variable vector given in Figure 1(a). The decomposed forms of linear transforms are given in Figure 1(b).

IV. PERFORMANCE EVALUATION OF THE PROPOSED APPROACH

We use three experiments to evaluate the performance of the proposed ISD technique: a Gaussian filter, two reported filters and random filters with variant orders.

We choose a Gaussian filter with 11 orders as an example to show the circuit design procedure with the proposed ISD technique.

The filter coefficients are shown in below

$$[1 \ 4 \ 19 \ 57 \ 108 \ 134 \ 108 \ 57 \ 19 \ 4 \ 1]$$

Considering the symmetry of the filter coefficients, we implement as shown in Fig.2.

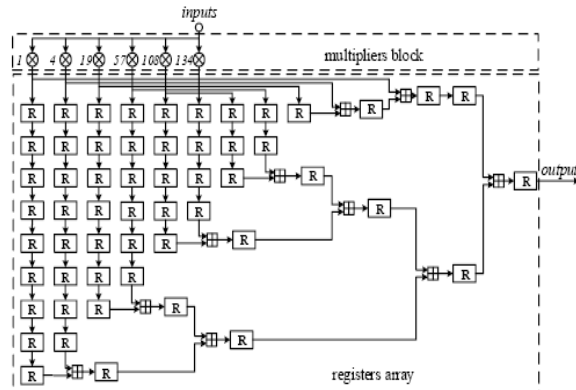


Figure .2 Gaussian filter implementation

In Figure.2, this implementation consists of two parts: the multipliers block and the registers array. The multipliers block accomplishes the multiplications of every input and the filter coefficients. The registers array gets the filtering results by adding the multiplication results after different delays. It is obvious that the most complex operation in this implementation is to achieve the multiplications of the inputs and six filter coefficients.

In order to get the CSD form multiplier-free implementation, the filter coefficients are represented as follows.

$$\begin{bmatrix} 1 \\ 4 \\ 19 \\ 57 \\ 108 \\ 134 \end{bmatrix} = \begin{bmatrix} 2^0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2^0 & 0 & 0 & 0 & 0 & 0 \\ 2^0 & 2^0 & 0 & 0 & 2^0 & 0 & 0 & 0 \\ 2^0 & 0 & 0 & -2^0 & 0 & 0 & 2^0 & 0 \\ 0 & 0 & 2^0 & 2^0 & 0 & 2^0 & 2^0 & 0 \\ 0 & 2^0 & 2^0 & 0 & 0 & 0 & 0 & 2^0 \end{bmatrix} \times \begin{bmatrix} 2^0 \\ 2^1 \\ 2^2 \\ 2^3 \\ 2^4 \\ 2^5 \\ 2^6 \\ 2^7 \end{bmatrix}$$

According to above matrix, we can get the multipliers block hardware in CSD form as shown in Figure. 3.

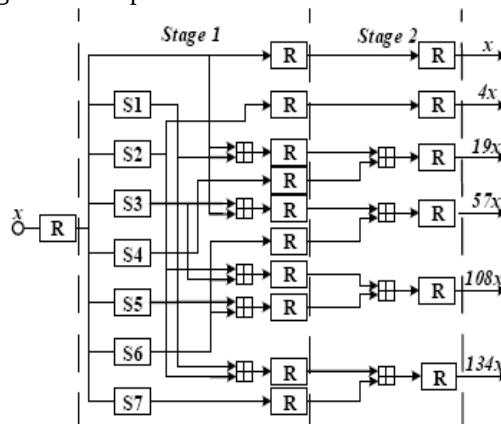


Figure.3 Multipliers block implementation in CSD form

In Fig.4, the implementation of the multipliers block needs 9 adders, and the maximum delay between every two registers is one adder.

With the proposed ISD approach, we can get below matrix and the implementation in Figure.4.

$$\begin{bmatrix} 1 \\ 4 \\ 19 \\ 57 \\ 108 \\ 134 \end{bmatrix} = \begin{bmatrix} 2^0 & 0 & 0 \\ 2^2 & 0 & 0 \\ 2^1 & 0 & 2^0 \\ 0 & 2^3 & 2^0 \\ 0 & 2^3 & 2^2 \\ -2^1 & 0 & 2^3 \end{bmatrix} \times \begin{bmatrix} 2^0 \\ 2^2 + 2^0 \\ 2^4 + 2^0 \end{bmatrix}$$

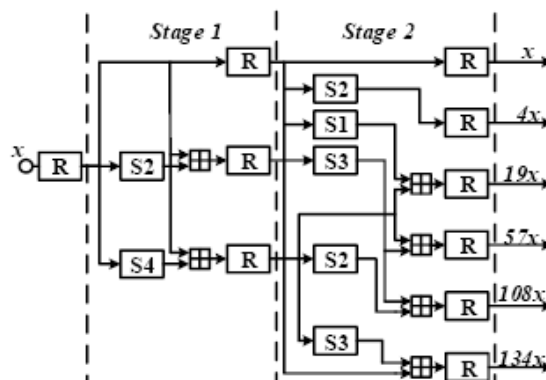


Figure.4 Multipliers block implementation with the ISD technique

In Fig.4, this multipliers block implementation consists of two stages. The first stage accomplishes the same sub-operators, 2^0x , $(2^0 + 2^2)x$ and $(2^0 + 2^4)x$. In the second stage, the intermediate results of Stage 1 are shifted and added with several binary adder trees. The shift bits in the second stage are decided by the coefficients of **PM**. It is obvious that the maximum delay between every two registers in this implementation is also one adder. However, the number of adders is decreased to 6.

The detailed numbers of operations in the two kinds of implementations are listed in Table 1.

Table 1: comparison of CSD and ISD operations

Implementation	CSD form	Proposed ISD
Number of adders	9	6
Number of registers	17	10
Maximum delay between every two registers	One adder	One adder

A. VEDIC MULTIPLIER

In the proposed technique multiplier block using Vedic multiplier. Vedic mathematics for the multiplier design will enhance the speed of multiplication operation. The multiplier architecture is based on UrdhvaTiryagbhyam (vertical and cross-wise algorithm) sutra. An illustration of Urdhva Tiryagbhyam sutra is shown in Figure 5.

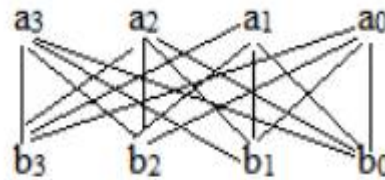


Figure 5. Illustration of Urdhva Tiryagbhyam sutra

The 2 x 2 Vedic multiplier operations are shown in Figure- 6

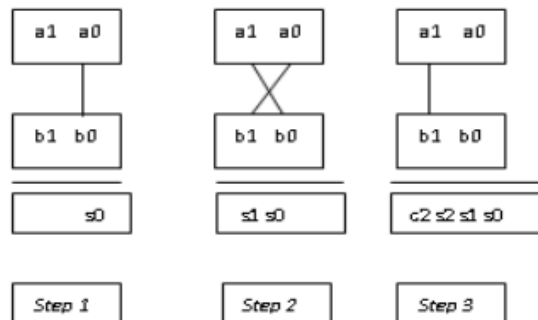


Figure-6. 2x2 Vedic multiplier operation

Step-1: Vertical multiplication First step to multiply $a_0 \times b_0$ which gives s_0 .

Step-2: The Cross multiplication Second step to multiply crosswise.

Step-3: Vertical multiplication the third step again vertical multiplication $a_1 \times b_1$ which gives s_2 and c_2 .

B. PARALLEL PREFIX ADDER

The PPA is like a Carry Look Ahead Adder. The production of the carriers the prefix adders can be designed in many different ways based on the different requirements. We use tree structure form to increase the speed of arithmetic operation. Parallel prefix adders are faster adders and these are faster adders and used for high performance arithmetic structures in industries.

Parallel Prefix Adders are classified into

1. Kogge- Stone Adder
2. Brent-Kung Adder
3. Ladner-Fischer Adder

Brent Kung Adder

Compared to previous adders the BrentKung adder consists of less delay, so figure.7 Brent Kung adder is used in the proposed system.

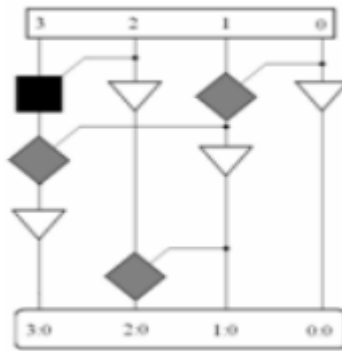


Figure.7.Brent Kung 4 bit adder

The Brent kung adder consists of three cells black cell , buffer cell, grey cell.

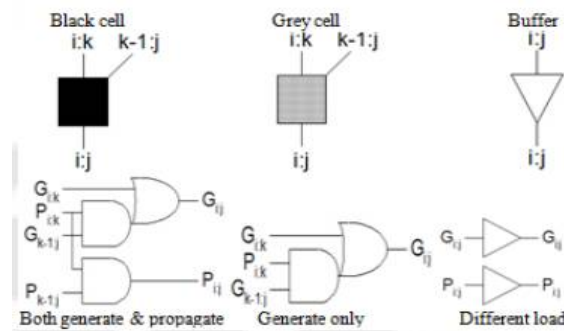
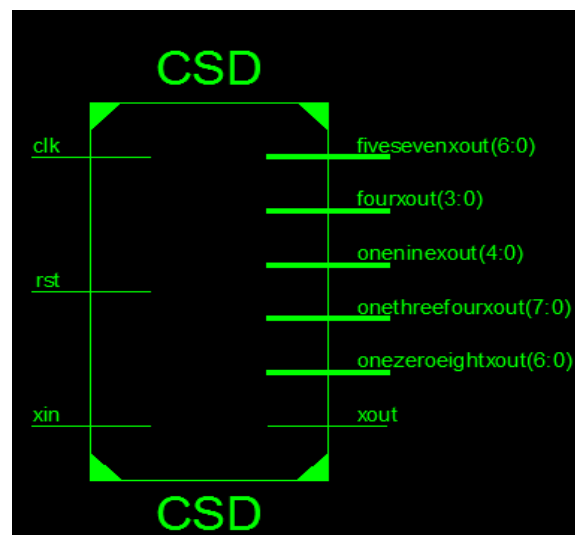


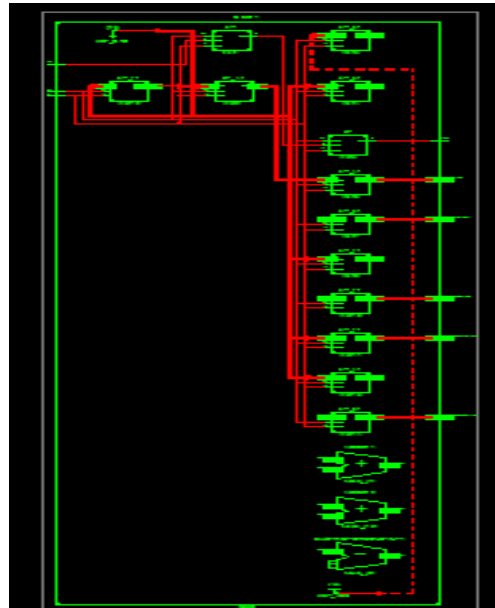
Figure 8: Black, Grey cells, Buffer and there Schematics.

V. RESULTS

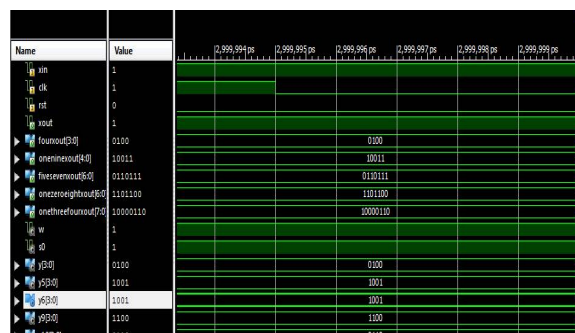
Block diagram



RTL schematic



Simulation results



VI. CONCLUSION

We propose an improved signed digit (ISD) representation approach for constant vector multiplication. In the proposed ISD technique, the constant vector is decomposed into two terms: a “private” matrix and a “public” vector: the “private” matrix contains the private sub-operators for each entry and the “public” vector contains the same sub-operators for all the entries. Based on the decomposition results, the “public” vector generates a set of regular intermediate vertices of the data flow which can be reused for each entry. Therefore using Vedic and Brent Kung adder the area and delay will be reduced compared to existing systems. The propose ISD technique, which can improve the routability in the implementation and make it more suitable for the circuit design than the existing techniques.

REFERENCES

- [1] L. I. Bluestein, "A linear filtering approach to the computation of discrete Fourier transform," *IEEE Transactions on Audio & Electroacoustics*, vol. 18, pp. 451-455, 1971.
- [2] D. P. M. Netravali and A. N., "Reconstruction Filters In Computer-Graphics," *Siggraph Proceedings of Annual Conference on Computer Graphics & Interactive Techniqu*, vol. 22, pp. 221-228, 1988.

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization

Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

- [3] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biological Cybernetics*, vol. 61, pp. 103-113, 1989.
- [4] W. A. Gardner, "Cyclic Wiener filtering: theory and method," *Communications, IEEE Transactions on*, vol. 41, pp. 151-163, 1993.
- [5] M. Unser, P. Thevenaz, and L. Yaroslavsky, "Convolution-based interpolation for fast, high-quality rotation of images," *IEEE Transactions on Image Processing*, vol. 4, pp. 1371-1381, 1995.
- [6] G. Dimitrakopoulos and D. Nikolos, "High-speed parallel-prefix VLSI ring adders", *IEEE Trans. Comput.*, vol. 54, no. 2, Feb.2005, Pages. 225–231.
- [7] Lagae, A. Lefebvre, S. Drettakis, G. Dutré, and P. &Dutré, "Procedural noise using sparse Gabor convolution," *Acm Transactions on Graphics*, vol. 28, pp. 341-352, 2009.
- [8] K. A. Syed, "The Discrete Cosine Transform (DCT): Theory and Application," *Texts in Computer Science*, vol. 41, pp. 135-147, 2003.
- [9] S. Chai Beng, R. Gani, and V. Jeoti, "Wavelet-DCT based image coder for video coding applications," in *Intelligent and Advanced Systems, 2007. ICIAS 2007. International Conference on*, 2007, pp. 748-752.
- [10] M. Budagavi, A. Fuldseth, G. Bjontegaard, V. Sze, and M. Sadafale, "Core Transform Design in the High Efficiency Video Coding (HEVC) Standard," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 7, pp. 1029-1041, 2013.
- [11] A. Avizienis, "Signed-Digit Number Representations For Fast Parallel Arithmetic," *Electronic Computers and Transactions on*, vol. ec-10, pp. 389-400, 1961.
- [12] Hwang K, "Computer arithmetic-principles, architecture and design," New York: Wiley Press, 1979.
- [13] I. S. E. Chen and T. N. Rajashekhara, "A fast multiplier design using signed-digit numbers and 3-valued logic," in *Circuits and Systems, 1990., Proceedings of the 33rd Midwest Symposium on*, 1990, pp. 881-884 vol.2.
- [14] H. I. Jeon and M. A. G. Abushagur, "Digital optical modified signed-digit arithmetic based on symbolic substitution and its encoding scheme," in *System Theory, 1989. Proceedings., Twenty-First Southeastern Symposium on*, 1989, pp. 252-256.
- [15] M. Saremi and S. Timarchi, "Efficient 1-out-of-3 Binary Signed-Digit multiplier for the moduli set $\{2n-1, 2n, 2n+1\}$," in *Computer Architecture and Digital Systems (CADS), 2013 17th CSI International Symposium on*, 2013, pp. 123-124.
- [16] M. Joye and Y. Sung-Ming, "Optimal left-to-right binary signed-digit recoding," *Computers, IEEE Transactions on*, vol. 49, pp. 740-748, 2000.
- [17] H. Samuelli, "An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients," *Circuits and Systems, IEEE Transactions on*, vol. 36, pp. 1044-1047, 1989.
- [18] R. I. Hartley, "Optimization of canonic signed digit multipliers for filter design," in *Circuits and Systems, 1991., IEEE International Symposium on*, 1991, pp. 1992-1995.
- [19] R. I. Hartley, "Subexpression sharing in filters using canonic signed digit multipliers," *IEEE Trans.circuitsSyst.ii Analog Digit.signal Process*, vol. 43, pp. 677-688, 1996.