An Optimized Modified Booth Recoder for Efficient Design of the Add-Multiply Operator

Kostas Tsoumanis, *Student Member, IEEE*, Sotiris Xydis, Constantinos Efstathiou, Nikos Moschopoulos, and Kiamal Pekmestzi

***Abstract*—Complex arithmetic operations are widely used in Digital Signal Processing (DSP) applications. In this work, we focus on optimizing the design of the fused Add-Multiply (FAM) operator for increasing performance. We investigate techniques to implement the direct recoding of the sum of two numbers in its Modified Booth (MB) form. We introduce a structured and efficient recoding technique and explore three different schemes by incorporating them in FAM designs. Comparing them with the FAM designs which use existing recoding schemes, the proposed technique yields considerable reductions in terms of critical delay, hardware complexity and power consumption of the FAM unit.**

***Index Terms*—Add-Multiply operation, arithmetic circuits, Modified Booth recoding, VLSI design.**

I. Introduction

MODERN consumer electronics make extensive use of Digital Signal Processing (DSP) providing custom accelerators for the domains of multimedia, communications etc. Typical DSP applications carry out a large number of arithmetic operations as their implementation is based on computationally intensive kernels, such as Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT), Finite Impulse Response (FIR) filters and signals' convolution. As expected, the performance[1] of DSP systems is inherently affected by decisions on their design regarding the allocation and the architecture of arithmetic units.

Recent research activities in the field of arithmetic optimization [1], [2] have shown that the design of arithmetic components combining operations which share data, can lead to significant performance improvements. Based on the observation that an addition can often be subsequent to a multiplication (e.g., in symmetric FIR filters), the Multiply-Accumulator (MAC) and Multiply-Add (MAD) units were introduced [3] leading to more efficient implementations of DSP algorithms compared to the conventional ones, which use only primitive resources [4]. Several architectures have been proposed to optimize the performance of the MAC operation in terms of area occupation, critical path delay or power consumption [5]–[7]. As noted in [8], MAC components increase the flexibility of DSP datapath synthesis as a large set of arithmetic operations can be efficiently mapped onto them. Except the MAC/MAD operations, many DSP applications are based on Add-Multiply (AM) operations (e.g., FFT algorithm [9]). The straightforward design of the AM unit, by first allocating an adder and then driving its output to the input of a multiplier, increases significantly both area and critical path delay of the circuit. Targeting an optimized design of AM operators, fusion techniques [10]–[13], [23] are employed based on the direct recoding of the sum of two numbers (equivalently a number in carry-save representation [14]) in its Modified Booth (MB) form [15]. Thus, the carry-propagate (or carry-look-ahead) adder [16] of the conventional AM design is eliminated resulting in considerable gains of performance. Lyu and Matula [10] presented a signed-bit MB recoder which transforms redundant binary inputs to their MB recoding form. A special expansion of the preprocessing step of the recoder is needed in order to handle operands in carry-save representation. In [12], the author proposes a two-stage recoder which converts a number in carry-save form to its MB representation. The first stage transforms the carry-save form of the input number into signed-digit form which is then recoded in the second stage so that it matches the form that the MB digits request. Recently, the technique of [12] has been used for the design of high performance flexible coprocessor architectures targeting the computationally intensive DSP applications [17]. Zimmermann and Tran [13] present an optimized design of [10] which results in improvements in both area and critical path. In [23], the authors propose the recoding of a redundant input from its carry-save form to the corresponding borrow-save form keeping the critical path of the multiplication operation fixed.

Although the direct recoding of the sum of two numbers in its MB form leads to a more efficient implementation of the fused Add-Multiply (FAM) unit compared to the conventional one, existing recoding schemes are based on complex manipulations in bit-level, which are implemented by dedicated circuits in gate-level. This work focuses on the efficient design of FAM operators, targeting the optimization of the recoding scheme for direct shaping of the MB form of the sum of two numbers (Sum to MB – *S-MB*). More specifically, we propose a new recoding

Manuscript received January 22, 2013; revised April 29, 2013; accepted July 09, 2013. Date of publication January 13, 2014; date of current version March 25, 2014. This research was co-funded by the European Union (European Social Fund) and Greek national resources under the framework of the "Archimedes III: Funding of Research Groups in TEI of Athens" project of the "Education & Lifelong Learning" Operational Programme. This paper was recommended by Associate Editor S.-Y. Chien.

K. Tsoumanis, N. Moschopoulos, and K. Pekmestzi are with the School of Electrical and Computer Engineering, National Technical University of Athens, Greece (e-mail: kostastsoumanis@microlab.ntua.gr; Nikos@microlab.ntua.gr; pekmes@microlab.ntua.gr).

S. Xydis is with the Dipartimento di Elettronica e Informazione, Politechnico Di Milano, Italy (e-mail: xydis@elet.polimi.it).

C. Efstathiou is with the Technological Institute of Athens, Greece (e-mail: cefsta@teiath.gr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSI.2013.2283695

[1]Within this paper performance refers to the area, delay and power metrics.

technique which decreases the critical path delay and reduces area and power consumption. The proposed *S-MB* algorithm is structured, simple and can be easily modified in order to be applied either in signed (in 2's complement representation) or unsigned numbers, which comprise of odd or even number of bits. We explore three alternative schemes of the proposed *S-MB* approach using conventional and signed-bit Full Adders (FAs) and Half Adders (HAs) as building blocks.

We evaluated the performance of the proposed *S-MB* technique by comparing its three different schemes with the state-of-the-art recoding techniques [12], [13], [23]. Industrial tools for RTL synthesis [18] and power estimation [19] have been used to provide accurate measurements of area utilization, critical path delay and power dissipation regarding various bit-widths of the input numbers. We show that the adoption of the proposed recoding technique delivers optimized solutions for the FAM design enabling the targeted operator to be timing functional (no timing violations) for a larger range of frequencies. Also, under the same timing constraints, the proposed designs deliver improvements in both area occupation and power consumption, thus outperforming the existing *S-MB* recoding solutions.

The rest of the paper is organized as follows: In Section II, we discuss the motivation and present a technical background for the implementation of FAM units. In Section III, the proposed *S-MB* recoding scheme is presented. In Section IV, both theoretical analysis and experimental evaluation are given, clearly identifying the advantages of the proposed schemes with respect to area complexity, critical delay and power dissipation. Section V concludes our work.

II. Motivation and Fused AM Implementation

A. Motivation

In this paper, we focus on AM units which implement the operation $Z = X \cdot (A+B)$. The conventional design of the AM operator (Fig. 1(a)) requires that its inputs A and B are first driven to an adder and then the input X and the sum $Y = A + B$ are driven to a multiplier in order to get Z. The drawback of using an adder is that it inserts a significant delay in the critical path of the AM. As there are carry signals to be propagated inside the adder, the critical path depends on the bit-width of the inputs. In order to decrease this delay, a Carry-Look-Ahead (CLA) adder can be used which, however, increases the area occupation and power dissipation. An optimized design of the AM operator is based on the fusion of the adder and the MB encoding unit into a single datapath block (Fig. 1(b)) by direct recoding of the sum $Y = A + B$ to its MB representation [10]–[13], [23]. The fused Add-Multiply (FAM) component contains only one adder at the end (final adder of the parallel multiplier). As a result, significant area savings are observed and the critical path delay of the recoding process is reduced and decoupled from the bit-width of its inputs. In this work, we present a new technique for direct recoding of two numbers in the MB representation of their sum.

B. Review of the Modified Booth Form

Modified Booth (MB) is a prevalent form used in multiplication [15], [20], [24]. It is a redundant signed-digit radix-4 encoding technique. Its main advantage is that it reduces by half the number of partial products in multiplication comparing to any other radix-2 representation.

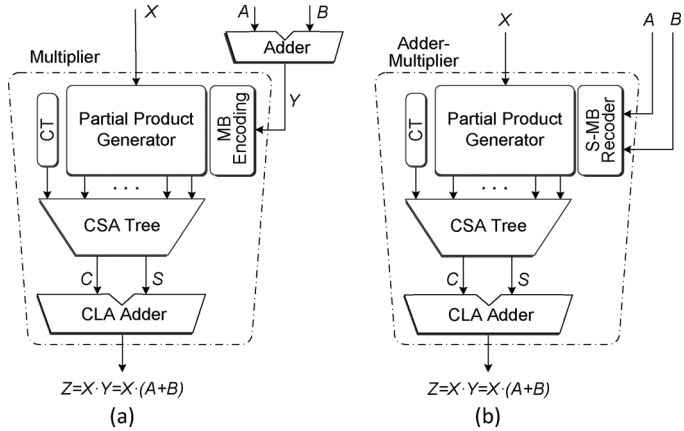


Fig. 1. AM operator based on the (a) conventional design and (b) fused design with direct recoding of the sum of A and B in its MB representation. The multiplier is a basic parallel multiplier based on the MB algorithm. The terms CT, CSA Tree and CLA Adder are referred to the Correction Term, the Carry-Save Adder Tree and the final Carry-Look-Ahead Adder of the multiplier.

Let us consider the multiplication of 2's complement numbers X and Y with each number consisting of $n = 2k$ bits. The multiplicand Y can be represented in MB form as:

$$Y = \langle y_{n-1}y_{n-2}\cdots y_1y_0\rangle_{2's} = -y_{2k-1}\cdot 2^{2k-1} + \sum_{i=0}^{2k-2} y_i \cdot 2^i$$

$$= \left\langle \mathbf{y}_{k-1}^{MB}\mathbf{y}_{k-2}^{MB}\cdots\mathbf{y}_1^{MB}\mathbf{y}_0^{MB}\right\rangle_{MB} = \sum_{j=0}^{k-1}\mathbf{y}_j^{MB}\cdot 2^{2j} \quad (1)$$

$$\mathbf{y}_j^{MB} = -2y_{2j+1} + y_{2j} + y_{2j-1}. \quad (2)$$

Digits $\mathbf{y}_j^{MB} \in \{-2, -1, 0, +1, +2\}$, $0 \le j \le k-1$, correspond to the three consecutive bits y_{2j+1}, y_{2j} and y_{2j-1} with one bit overlapped and considering that $y_{-1} = 0$. Table I shows how they are formed by summarizing the MB encoding technique. Each digit is represented by three bits named s, *one* and *two*. The sign bit s shows if the digit is negative ($s = 1$) or positive ($s = 0$). Signal *one* shows if the absolute value of a digit is equal to 1 ($one = 1$) or not ($one = 0$). Signal *two* shows if the absolute value of a digit is equal to 2 ($two = 1$) or not ($two = 0$). Using these three bits we calculate the MB digits $\mathbf{y}_j^{MB}$ by the following relation:

$$\mathbf{y}_j^{MB} = (-1)^{s_j}\cdot[one_j + 2\cdot two_j]. \quad (3)$$

Fig. 2(a) shows the Boolean equations on which the implementation of the MB encoding signals is based (Fig. 2(b)) [20], [24].

C. FAM Implementation

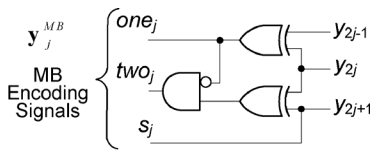
In the FAM design presented in Fig. 1(b), the multiplier is a parallel one based on the MB algorithm. Let us consider the product $X \cdot Y$. The term $Y = \langle y_{n-1}y_{n-2}\ldots y_1y_0\rangle_{2's}$ is encoded based on the MB algorithm (Section II.B) and multiplied with $X = \langle x_{n-1}x_{n-2}\ldots x_1x_0\rangle_{2's}$. Both X and Y consist of

TABLE I
MODIFIED BOOTH ENCODING TABLE.

Binary			$\mathbf{y}_j^{MB}$	*MB Encoding*			*Input Carry*
y_{2j+1}	y_{2j}	y_{2j-1}		sign=s_j	×1=one$_j$	×2=two$_j$	$c_{in,j}$
0	0	0	**0**	0	0	0	0
0	0	1	**+1**	0	1	0	0
0	1	0	**+1**	0	1	0	0
0	1	1	**+2**	0	0	1	0
1	0	0	**-2**	1	0	1	1
1	0	1	**-1**	1	1	0	1
1	1	0	**-1**	1	1	0	1
1	1	1	**0**	1	0	0	0

$one_j = y_{2j-1} \oplus y_{2j}$
$two_j = (y_{2j+1} \oplus y_{2j}) \cdot \overline{one_j}$
$s_j = y_{2j+1}$

(a)



(b)

Fig. 2. (a) Boolean equations and (b) gate-level schematic for the implementation of the MB encoding signals.

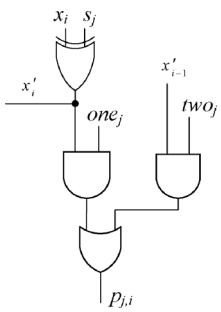


Fig. 3. Generation of the i-th bit $p_{j,i}$ of the partial product PP_j for the conventional MB multiplier.

$n = 2k$ bits and are in 2's complement form. Equation (4) describes the generation of the k partial products:

$$PP_j = X \cdot \mathbf{y}_j^{MB} = \overline{p}_{j,n} 2^n + \sum_{i=0}^{n-1} p_{j,i} \cdot 2^i. \quad (4)$$

The generation of the i-th bit $p_{j,i}$ of the partial product PP_j is based on the next logical expression while Fig. 3 illustrates its implementation at gate level [20], [24]:

$$p_{j,i} = ((x_i \oplus s_j) \wedge one_j) \vee ((x_{i-1} \oplus s_j) \wedge two_j). \quad (5)$$

For the computation of the least and the most significant bits of the partial product we consider $x_{-1} = 0$ and $x_n = x_{n-1}$ respectively. Note that in case that $n = 2k + 1$, the number of the resulting partial products is $\lfloor n/2 \rfloor + 1 = k + 1$ and the most significant MB digit is formed based on sign extension of the initial 2's complement number.

After the partial products are generated, they are added, properly weighted, through a Wallace Carry-Save Adder (CSA) tree [21] along with the Correction Term (CT) which is given by the following equations:

$$Z = X \cdot Y = CT + \sum_{j=0}^{k-1} PP_j \cdot 2^{2j} \quad (6)$$

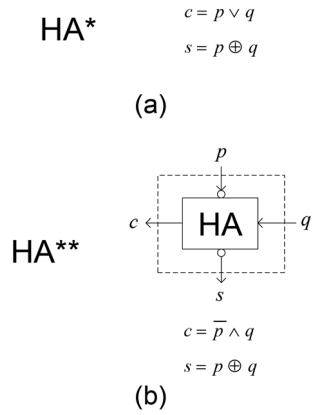


Fig. 4. Boolean equations and schematics for signed (a) HA* and (b) HA**.

$$CT = CT(low) + CT(high) = \\ = \sum_{j=0}^{k-1} c_{\text{in},j} \cdot 2^{2j} + 2^n \left(1 + \sum_{j=0}^{k-1} 2^{2j+1}\right) \quad (7)$$

where $c_{\text{in},j} = (one_j \vee two_j) \wedge s_j$ (see Table I).

Finally, the carry-save output of the Wallace CSA tree is leaded to a fast Carry Look Ahead (CLA) adder to form the final result $Z = X \cdot Y$ as shown in Fig. 1(b).

III. SUM TO MODIFIED BOOTH RECODING TECHNIQUE (*S-MB*)

A. Defining Signed-Bit Full Adders and Half Adders for Structured Signed Arithmetic

In *S-MB* recoding technique, we recode the sum of two consecutive bits of the input A (a_{2j}, a_{2j+1}) with two consecutive bits of the input B (b_{2j}, b_{2j+1}) into one MB digit $\mathbf{y}_j^{MB}$. As we observe from (2), three bits are included in forming a MB digit. The most significant of them is negatively weighted while the two least significant of them have positive weight. Consequently, in order to transform the two aforementioned pairs of bits in MB form we need to use signed-bit arithmetic. For this purpose, we develop a set of bit-level signed Half Adders (HA) and Full Adders (FA) considering their inputs and outputs to be signed.

More specifically, in this work, we use two types of signed HAs which are referred as HA* and HA**. Tables II - IV are their truth tables and in Fig. 4 we present their corresponding Boolean equations. Considering that p, q are the binary inputs and c, s are the outputs (carry and sum respectively) of a HA* which implements the relation $2 \cdot c - s = p + q$ where the sum s is considered negatively signed (Table II, Fig. 4(a)), the output takes one of the values $\{0, +1, +2\}$. In Table III, we also describe the dual implementation of HA* where we inversed the signs of all inputs and outputs and, consequently, changed the output values to $\{-2, -1, 0\}$. Table IV and Fig. 4(b) show the operation and schematic of HA** which implements the relation $2 \cdot c - s = -p + q$ and manipulates a negative (p) and a positive (q) input resulting in the output values $\{-1, 0, +1\}$.

Also, we design two types of signed FAs which are presented in Table V and VI and Fig. 5. The schematics drawn in Fig. 5(a) and (b) show the relation of FA* and FA** with

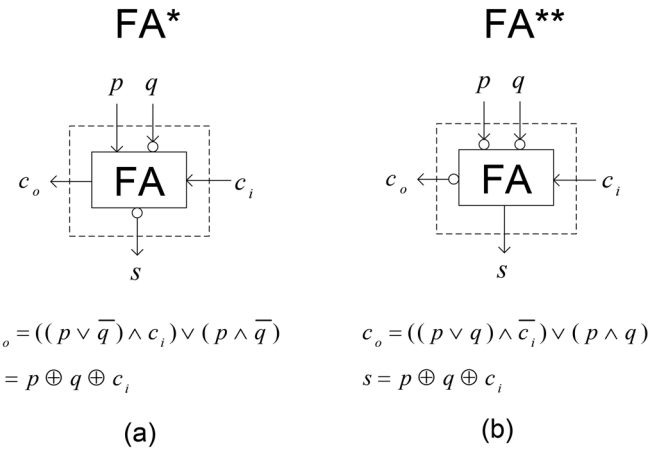


Fig. 5. Boolean equations and schematics for signed (a) FA* and (b) FA**.

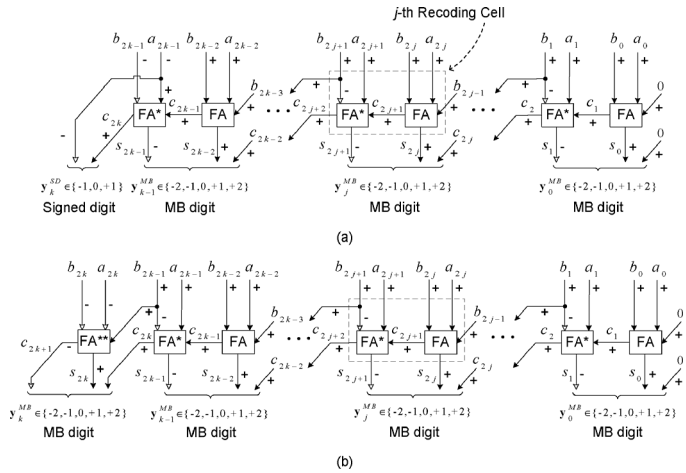


Fig. 6. *S-MB*1 recoding scheme for (a) even and (b) odd number of bits.

the conventional FA. Assuming that p, q and c_i are the binary inputs and c_o, s are the output carry and sum respectively, FA* implements the relation $2 \cdot c_o - s = p - q + c_i$ where the bits s and q are considered negatively signed (Table V, Fig. 5(a)). Table V (truth table of FA*) shows that the output values of FA* are $\{-1, 0, +1, +2\}$. In the case of FA**, the two inputs p, q are negatively signed and FA** implements the relation $-2 \cdot c_o + s = -p - q + c_i$ (Table VI, Fig. 5(b)). The output values become $\{-2, -1, 0, +1\}$. As shown in Fig. 5, the signed FAs are implemented using the conventional FA with the negative inputs and outputs inverted.

TABLE II
HA* BASIC OPERATION.

Inputs		*Output*	*Outputs*	
p (+)	q (+)	*Value*[1]	c (+)	s (-)
0	0	0	0	0
0	1	+1	1	1
1	0	+1	1	1
1	1	+2	1	0

[1] $OutputValue = 2 \cdot c - s = p + q$

B. Proposed S-MB Recoding Techniques

We use both conventional and signed HAs and FAs of Section III.A in order to design and explore three new alternative schemes of the *S-MB* recoding technique. Each of the three schemes can be easily applied in either signed (2's complement representation) or unsigned numbers which consist of odd or even number of bits.

In all schemes we consider that both inputs A and B are in 2's complement form and consist of $2k$ bits in case of even or $2k+1$ bits in case of odd bit-width. Targeting to transform the sum of A and B $(Y = A + B)$ in its MB representation, we consider the bits a_{2j}, a_{2j+1}, and b_{2j}, b_{2j+1} as the inputs of the j recoding cell in order to get at its output the three bits that we need to form the MB digit $\mathbf{y}_j^{MB}$ according to (2).

1) S-MB1 Recoding Scheme: The first scheme of the proposed recoding technique is referred as *S-MB*1 and is illustrated in detail in Fig. 6 for both even (Fig. 6(a)) and odd (Fig. 6(b)) bit-width of input numbers. As can be seen in Fig. 6, the sum of A and B is given by the next relation:

$$Y = A + B = \mathbf{y}_k \cdot 2^{2k} + \sum_{j=0}^{k-1} \mathbf{y}_j^{MB} \cdot 2^{2j}$$
$$\text{where} \quad \mathbf{y}_j^{MB} = -2s_{2j+1} + s_{2j} + c_{2j}. \tag{8}$$

The encoding of the MB digits $\mathbf{y}_j^{MB}$, $0 \le j \le k-1$, of (8) is based on the analysis of Section II.B (see (2)).

Both bits s_{2j+1} and s_{2j} are extracted from the j recoding cell of Fig. 6. A conventional FA with inputs a_{2j}, b_{2j} and b_{2j-1} produces the carry $c_{2j+1} = (a_{2j} \wedge b_{2j}) \vee (b_{2j-1} \wedge (a_{2j} \vee b_{2j}))$ and the sum $s_{2j} = a_{2j} \oplus b_{2j} \oplus b_{2j-1}$. As the bit s_{2j+1} needs to be negatively signed, we use a FA* (Table V, Fig. 5(a)) with inputs a_{2j+1}, b_{2j+1} $(-)$ and c_{2j+1}, which produces the carry c_{2j+2} and the sum s_{2j+1} $(-)$:

$$c_{2j+2} = (a_{2j+1} \wedge \overline{b}_{2j+1}) \vee (c_{2j+1} \wedge (a_{2j+1} \vee \overline{b}_{2j+1}))$$
$$s_{2j+1} = a_{2j+1} \oplus b_{2j+1} \oplus c_{2j+1}. \tag{9}$$

Note that, based on equation $b_{2j+1} = 2 \cdot b_{2j+1} - b_{2j+1}$, b_{2j+1} is driven to the FA* as negatively signed while it is also used with positive sign as an input carry of the subsequent recoding cell. We consider the initial values $b_{-1} = 0$ and $c_0 = 0$.

When we form the most significant digit (MSD) of the *S-MB*1 recoding scheme, we distinguish two cases: In the first case, the bit-width of A and B is even (Fig. 6(a)), while in the second case, both A and B comprise of odd number of bits (Fig. 6(b)). In the first case, the MSD $\mathbf{y}_{k,even}^{SD}$ is a signed digit and is given by the next algebraic equation:

$$\mathbf{y}_{k,even}^{SD} = -a_{2k-1} + c_{2k}. \tag{10}$$

In the second case, the MSD $\mathbf{y}_{k,odd}^{MB}$ is a MB digit that is formed based on c_{2k+1}, s_{2k} and c_{2k}. The carry c_{2k+1} $(-)$ and the sum s_{2k} are produced by a FA** with inputs a_{2k} $(-)$, b_{2k} $(-)$ and b_{2k-1} (Table VI, Fig. 5(b)).

The critical path delay of *S-MB*1 recoding scheme (Fig. 6) is constant in respect to the input bit-width and is given by the equation:

$$T_{S-MB1} = T_{FA,carry} + T_{FA^*,sum} \tag{11}$$

TABLE III
HA* DUAL OPERATION.

Inputs		*Output*	*Outputs*	
p (-)	q (-)	*Value*[2]	c (-)	s (+)
0	0	0	0	0
0	1	-1	1	1
1	0	-1	1	1
1	1	-2	1	0

[2] $OutputValue = -2 \cdot c + s = -p - q$

TABLE IV
HA** OPERATION.

Inputs		*Output*	*Outputs*	
p (-)	q (+)	*Value*[3]	c (+)	s (-)
0	0	0	0	0
0	1	+1	1	1
1	0	-1	0	1
1	1	0	0	0

[3] $OutputValue = 2 \cdot c - s = -p + q$

TABLE V
FA* OPERATION.

Inputs			*Output*	*Outputs*	
p (+)	q (-)	c_i (+)	*Value*[1]	c_o (+)	s (-)
0	0	0	0	0	0
0	0	1	+1	1	1
0	1	0	-1	0	1
0	1	1	0	0	0
1	0	0	+1	1	1
1	0	1	+2	1	0
1	1	0	0	0	0
1	1	1	+1	1	1

[1] $OutputValue = 2 \cdot c_o - s = p - q + c_i$

TABLE VI
FA** OPERATION.

Inputs			*Output*	*Outputs*	
p (-)	q (-)	c_i (+)	*Value*[2]	c_o (-)	s (+)
0	0	0	0	0	0
0	0	1	+1	0	1
0	1	0	-1	1	1
0	1	1	0	0	0
1	0	0	-1	1	1
1	0	1	0	0	0
1	1	0	-2	1	0
1	1	1	-1	1	1

[2] $OutputValue = -2 \cdot c_o + s = -p - q + c_i$

where $T_{FA,carry}$ is the delay of shaping the output carry of a conventional FA and $T_{FA^*,sum}$ is the delay of forming the sum of a signed FA*.

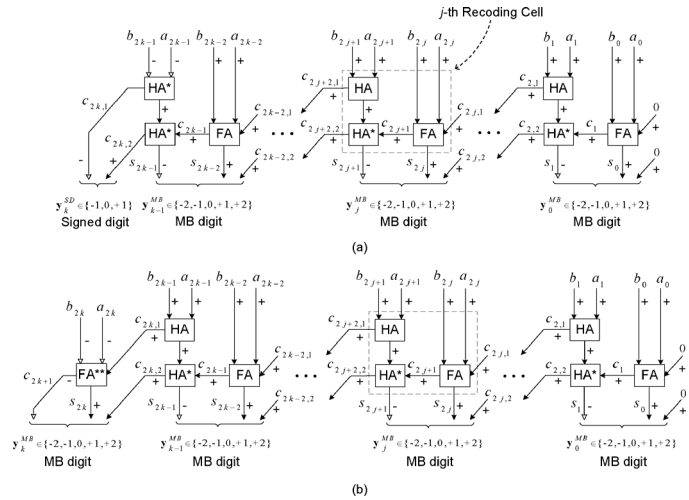


Fig. 7. *S-MB*2 recoding scheme for (a) even and (b) odd number of bits.

2) S-MB2 Recoding Scheme: The second approach of the proposed recoding technique, *S-MB*2, is described in Fig. 7 for even (Fig. 7(a)) and odd (Fig. 7(b)) bit-width of input numbers. We consider the initial values $c_{0,1} = 0$ and $c_{0,2} = 0$. The digits $\mathbf{y}_j^{MB}$, $0 \leq j \leq k-1$, are formed based on s_{2j+1}, s_{2j} and $c_{2j,2}$ according to (8). As in the *S-MB*1 recoding scheme, we use a conventional FA to produce the carry c_{2j+1} and the sum s_{2j}. The inputs of the FA are a_{2j}, b_{2j} and $c_{2j,1}$. The bit $c_{2j,1}$ is the output carry of a conventional HA which is part of the $(j-1)$ recoding cell and has the bits a_{2j-1}, b_{2j-1} as inputs. The bit s_{2j+1} is the output sum of a HA* (basic operation – Table II, Fig. 4(a)) in which we drive c_{2j+1} and the sum produced by a conventional HA with the bits a_{2j+1}, b_{2j+1} as inputs. The HA* is used in order to produce the negatively signed sum s_{2j+1} and its outputs are given by the following Boolean equations:

$$\begin{aligned} c_{2j+2,2} &= c_{2j+1} \vee (a_{2j+1} \oplus b_{2j+1}) \\ s_{2j+1} &= a_{2j+1} \oplus b_{2j+1} \oplus c_{2j+1}. \end{aligned} \tag{12}$$

In case that A and B comprise of even number of bits (Fig. 7(a)), a_{2n-1} and b_{2n-1} are negatively weighted and the conventional HA of the $(n-1)$ recoding cell is replaced by the dual HA* analyzed in Table III. The MSD $\mathbf{y}_{k,even}^{SD}$ is a signed digit and is given by the relation:

$$\mathbf{y}_{k,even}^{SD} = -c_{2k,1} + c_{2k,2}. \tag{13}$$

In case that the number of bits of the inputs A and B is odd (Fig. 7(b)), the MSD $\mathbf{y}_{k,odd}^{MB}$ is a MB digit that is formed based on c_{2k+1}, s_{2k} and $c_{2k,2}$. The carry c_{2k+1} $(-)$ and the sum s_{2k} are produced by a FA** with inputs a_{2k} $(-)$, b_{2k} $(-)$ and $c_{2k,1}$ (Table VI, Fig. 5(b)).

The critical path delay of *S-MB*2 recoding scheme is calculated as follows:

$$T_{S-MB2} = T_{HA,carry} + T_{FA,carry} + T_{HA^*,sum} \tag{14}$$

where $T_{HA,carry}$ and $T_{FA,carry}$ are the delays of shaping the output carry of a conventional HA and FA respectively and $T_{HA^*,sum}$ is the delay of forming the sum of a signed HA*.

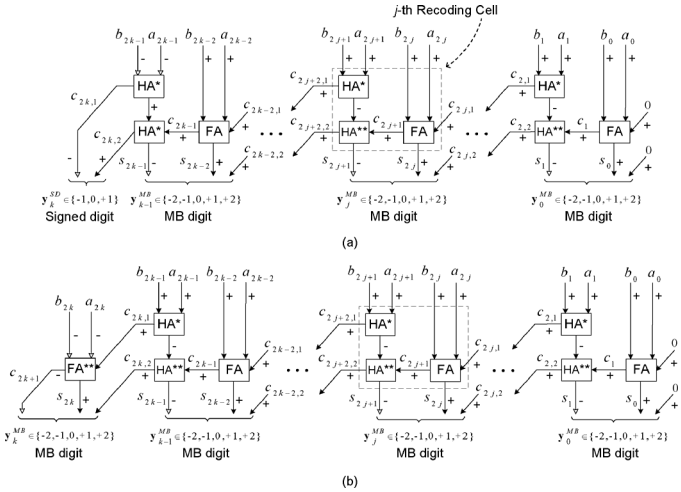


Fig. 8. *S-MB*3 recoding scheme for (a) even and (b) odd number of bits.

3) S-MB3 Recoding Scheme: The third scheme implementing the proposed recoding technique is *S-MB*3. It is illustrated in detail in Fig. 8 for even (Fig. 8(a)) and odd (Fig. 8(b)) bit-width of input numbers. We consider that $c_{0,1} = 0$ and $c_{0,2} = 0$. We build the digits $\mathbf{y}_j^{MB}, 0 \leq j \leq k-1$, based on s_{2j+1}, s_{2j} and $c_{2j,2}$ according to (8). Once more, we use a conventional FA to produce the carry c_{2j+1} and the sum s_{2j}. The bit $c_{2j,1}$ is now the output carry of a HA* (basic operation – Table II, Fig. 4(a)), which belongs to the $(j-1)$ recoding cell and has the bits a_{2j-1}, b_{2j-1} as inputs. The negatively signed bit s_{2j+1} is produced by a HA** (Table IV, Fig. 4(b)) in which we drive c_{2j+1} and the output sum (negatively signed) of the HA* of the j recoding cell with the bits a_{2j+1}, b_{2j+1} as inputs. The carry and sum outputs of the HA** are given by the following Boolean equations:

$$\begin{aligned} c_{2j+2,2} &= c_{2j+1} \wedge (\overline{a_{2j+1} \oplus b_{2j+1}}) \\ s_{2j+1} &= a_{2j+1} \oplus b_{2j+1} \oplus c_{2j+1}. \end{aligned} \tag{15}$$

In case that both A and B comprise of even number of bits (Fig. 8(a)), a_{2k-1} and b_{2k-1} are negatively weighted and we use the dual implementation of the HA* (Table III, Fig. 4(a)) in the $(k-1)$ recoding cell. Consequently, the output sum of the HA* becomes positively weighted and the HA** that follows has to be replaced with a HA*. The most significant digits for both cases of even and odd bit-width of A and B, are formed as in *S-MB*2 recoding scheme.

The critical path delay of *S-MB*3 recoding scheme is calculated as follows:

$$T_{S-MB3} = T_{HA^*,carry} + T_{FA,carry} + T_{HA^{**},sum} \tag{16}$$

where $T_{HA^*,carry}$ and $T_{FA,carry}$ are the delays of shaping the output carry of a signed HA* and FA respectively and $T_{HA^{**},sum}$ is the delay of forming the sum of a signed HA**.

4) Unsigned Input Numbers: In case that the input numbers A and B are unsigned, their most significant bits are positively signed. Figs. 9–11 present the modifications that we have to make in all *S-MB* schemes for both cases of even (the two most significant digits change) and odd (only the most significant digit change) bit-width of A and B, regarding the signs of the most significant bits of A and B. The basic recoding block in all schemes remains unchanged.

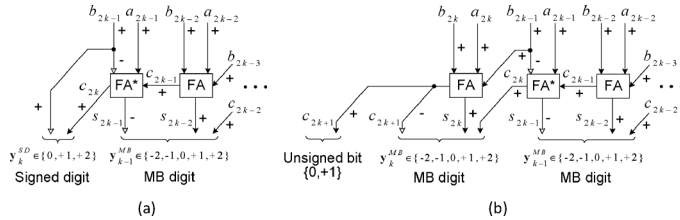


Fig. 9. Implementation of the MSD of the *S-MB*1 recoding scheme in case of unsigned input numbers for (a) even and (b) odd bit-width.

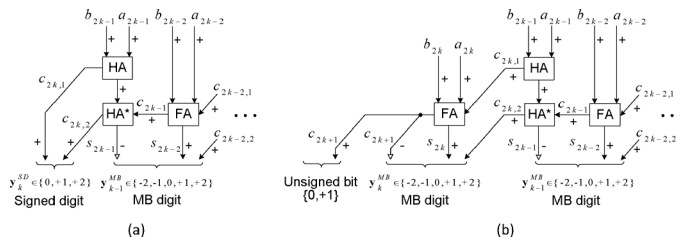


Fig. 10. Implementation of the MSD of the *S-MB*2 recoding scheme in case of unsigned input numbers for (a) even and (b) odd bit-width.

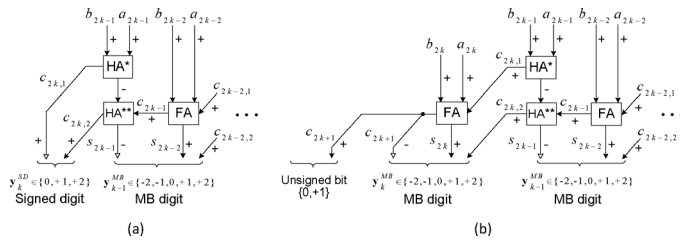


Fig. 11. Implementation of the MSD of the *S-MB*3 recoding scheme in case of unsigned input numbers for (a) even and (b) odd bit-width.

IV. Performance Evaluation

A. Theoretical Analysis

In this section, we present a theoretical analysis and comparative study in terms of area complexity and critical delay among the three recoding schemes that we described in Section III and the three existing recoding techniques [12], [13] and [23]. Our analysis is based on the unit gate model [16]. More specifically, for our quantitative comparisons the 2-input primitive gates (NAND, AND, NOR, OR) count as one gate equivalent for both area and delay, whereas the 2-input XOR, XNOR gates count as two gate equivalents [16]. The area of a FA and a HA is 7 and 3 gate equivalents respectively. The delays of the sum and carry outputs of a FA are 4 and 3 gate equivalents respectively, while those of a HA are 2 and 1. All the aforementioned information is summarized in Table VII.

In Figs. 12–14 we illustrate the recoding schemes of [12], [13] and [23] respectively at the level of the basic recoding cell including the MB encoder. In each of the Figs. 12–14 we draw a bold line showing the critical path of the corresponding recoding scheme which is selected to be the one that contains the highest number of XOR gates. A second bold line is drawn in Figs. 12 and 13 showing an alternative critical path which consists of the

TABLE VII
AREA AND DELAY OF VARIOUS COMPONENTS IN THE UNIT GATE MODEL.

Components	*Area (Gate equivalents)*	*Delay (Gate equivalents)*
NAND-2, NOR-2	A_g	T_g
NAND-3, NOR-3	$2A_g$	$2T_g$
XOR, XNOR	$2A_g$	$2T_g$
HA	$3A_g$	$T_{HA,carry}=T_g$, $T_{HA,sum}=2T_g$
FA	$7A_g$	$T_{FA,carry}=3T_g$, $T_{FA,sum}=4T_g$

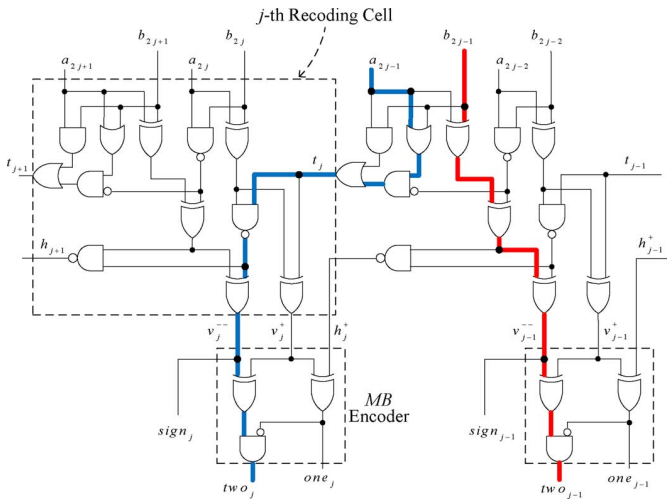


Fig. 12. Recoding scheme of [12] at the level of basic recoding cell and its alternative critical paths.

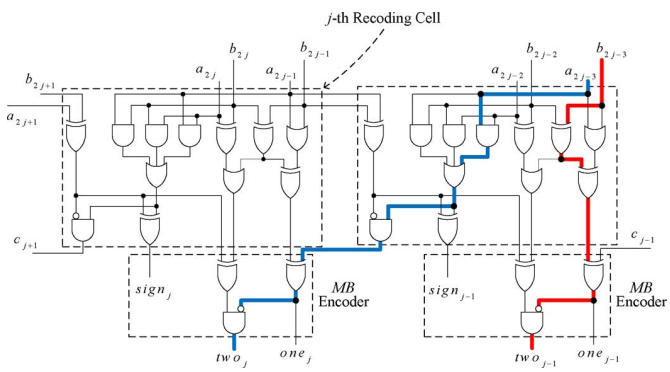


Fig. 13. Recoding scheme of [13] at the level of basic recoding cell and its alternative critical paths.

highest number of primitive gates. In both figures, the two alternative critical paths have the same delay which is equal to $9T_g$ in Fig. 12 and $7T_g$ in Fig. 13. In Fig. 15(a) and (b) we also provide the basic cells of the proposed recoders *S-MB*1 and *S-MB*2 respectively and draw their critical paths (bold lines). The proposed *S-MB*3 recoding scheme is equivalent to the *S-MB*2 one, in the unit gate model, in terms of area and delay. Table VIII summarizes the area complexity and the critical delay of the proposed recoding schemes (based on (11), (14) and (16)) and the recoders presented in [12], [13] and [23].

Based on Table VIII, we observe that all the proposed recoding schemes (*S-MB*1, *S-MB*2 and *S-MB*3) achieve area reduction of up to $5A_g$ compared to the existing recoders described in [12], [13] and [23]. In terms of critical delay, the proposed schemes appear to be faster by a T_g than [12] and [23] and

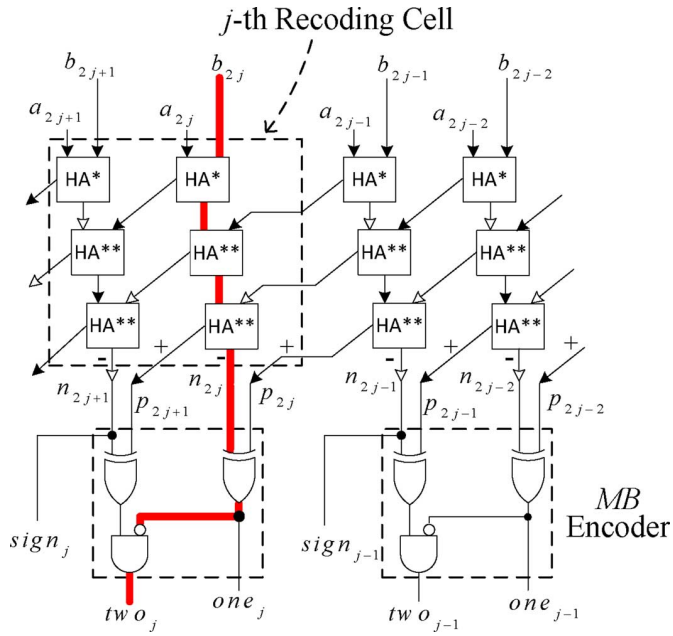


Fig. 14. Recoding scheme of [23] at the level of basic recoding cell and its critical path.

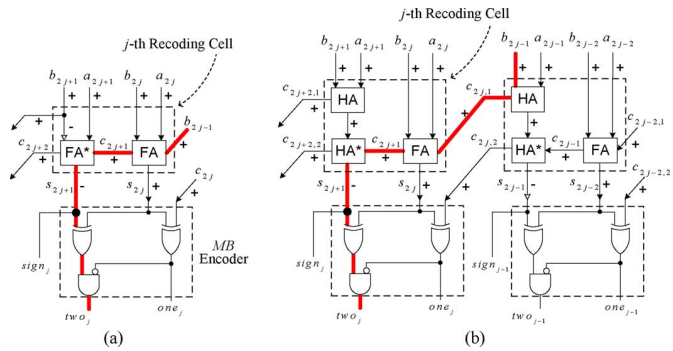


Fig. 15. Critical path of the proposed (a) *S-MB*1 and (b) *S-MB*2 recoding scheme.

TABLE VIII
AREA AND DELAY COMPARISON OF THE PROPOSED RECODING SCHEMES WITH EXISTING RECODERS.

Design		*Area Complexity*	*Critical Delay*
[12]		$7A_{xor}+8A_g=22A_g$	$3T_{xor}+3T_g=9T_g$ or $2T_{xor}+5T_g=9T_g$
[13]		$7A_{xor}+9A_g=23A_g$	$T_{xor}+5T_g=7T_g$ or $3T_{xor}+3T_g=7T_g$
[23]		$6A_{HA}+2A_{xor}+A_g=23A_g$	$T_{HA,sum}+T_{HA,sum}+T_{HA,sum}+T_{xor}+T_g=9T_g$
Proposed	*S-MB*1	$2A_{FA}+2A_{xor}+A_g=19A_g$	$T_{FA,carry}+2T_g+T_{xor}+T_g=8T_g$ †
Proposed	*S-MB*2	$A_{FA}+2A_{HA}+2A_{xor}+A_g=18A_g$	$T_{HA,carry}+2T_g+T_{HA,sum}+T_{xor}+T_g=8T_g$ ‡
Proposed	*S-MB*3	$A_{FA}+2A_{HA}+2A_{xor}+A_g=18A_g$	$T_{HA,carry}+2T_g+T_{HA,sum}+T_{xor}+T_g=8T_g$

†In Fig. 15a, the $T_{FA,carry}$ and the $T_{FA*,sum}$ of (11) are overlapped. So, the $T_{FA*,sum}$ is reduced from $4T_g$ to $2T_g$. The last two terms T_{xor} and T_g correspond to the delay of the MB encoder.
‡In the same way, in Fig. 15b, the $T_{HA,carry}$ and the $T_{FA,carry}$ of (14) are overlapped. So, the $T_{FA,carry}$ is reduced from $3T_g$ to $2T_g$.

very close to the delay of [13]. Thus, we resort to the synthesis of all recoding schemes using industrial tools in order to clarify the differences among them in terms of critical delay and verify

TABLE IX
AREA, DELAY AND POWER MEASUREMENTS OF FAM DESIGNS (EVEN BIT-WIDTH).

Design	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*	$A^\dagger$ *(um²)*	$P^\ddagger$ *(mW)*
8 bits																
	T=0.91ns		T=0.92ns		T=0.94ns		T=0.95ns		T=0.98ns		T=1.10ns		T=1.20ns		T=1.40ns	
*S-MB*1	-	-	-	-	-	-	4418	5.58	**4140**	4.75	3284	**3.15**	**2898**	**2.67**	2644	**2.27**
*S-MB*2	-	-	-	-	**4693**	**5.84**	4669	5.58	4281	4.92	**3278**	3.21	2944	2.82	**2631**	2.31
*S-MB*3	**5214**	**6.88**	**4887**	**6.29**	4765	6.17	**4407**	**5.42**	4161	**4.71**	3343	3.16	2943	2.73	2651	2.30
[12]	-	-	-	-	-	-	-	-	4659	5.83	3410	3.36	3078	2.97	2697	2.41
[13]	-	-	5452	6.86	5051	6.19	4934	5.97	4381	4.84	3610	3.35	3066	2.88	2699	2.44
[23]	-	-	-	-	-	-	-	-	4686	5.81	3435	3.40	3273	3.05	2803	2.49
12 bits																
	T=1.05ns		T=1.06ns		T=1.07ns		T=1.09ns		T=1.12ns		T=1.20ns		T=1.30ns		T=1.50ns	
*S-MB*1	**10191**	**12.30**	**9641**	**10.70**	**9392**	**11.00**	9006	10.20	8253	8.71	6747	5.80	6017	4.88	5070	**3.59**
*S-MB*2	-	-	-	-	9646	11.40	**8765**	**10.40**	**7797**	**8.24**	**6521**	**5.62**	**5847**	**4.59**	**5058**	3.69
*S-MB*3	-	-	-	-	-	-	-	-	8623	9.47	6792	5.79	5915	4.55	5102	3.68
[12]	-	-	-	-	-	-	-	-	9653	11.40	7392	7.01	6472	5.27	5274	3.97
[13]	-	-	-	-	-	-	9117	10.90	8476	9.37	6838	6.16	5938	4.70	5153	3.67
[23]	-	-	10425	12.80	9769	11.80	9392	11.30	8329	9.26	6998	5.93	6172	5.11	5192	3.90
16 bits																
	T=1.16ns		T=1.19ns		T=1.20ns		T=1.22ns		T=1.25ns		T=1.30ns		T=1.40ns		T=1.60ns	
*S-MB*1	**15573**	**18.80**	14441	**15.30**	**14012**	**14.10**	**12642**	**12.20**	**12047**	**10.40**	**10227**	**8.08**	**9246**	**6.48**	8205	5.54
*S-MB*2	-	-	**14303**	15.70	14064	15.60	13421	14.00	12258	11.40	10416	8.16	9286	6.96	**8192**	5.50
*S-MB*3	-	-	-	-	-	-	14366	16.00	13244	13.40	10639	9.21	9353	7.03	8207	5.44
[12]	-	-	-	-	-	-	15171	17.30	14080	15.00	12346	11.50	9567	7.42	8340	5.83
[13]	-	-	-	-	14831	16.30	13481	14.40	12877	13.10	10865	9.47	9369	6.98	8234	**5.29**
[23]	-	-	-	-	14796	16.20	13672	14.30	12719	12.10	10573	8.96	9681	7.21	8375	5.93
24 bits																
	T=1.36ns		T=1.39ns		T=1.41ns		T=1.43ns		T=1.45ns		T=1.50ns		T=1.60ns		T=1.80ns	
*S-MB*1	-	-	26269	26.10	23656	21.00	**21318**	18.20	21437	17.60	20273	15.30	18444	13.20	16563	10.70
*S-MB*2	-	-	24756	22.20	22470	**19.90**	21366	**17.30**	**20971**	**16.80**	**19857**	**14.40**	18389	13.30	16601	10.60
*S-MB*3	**28465**	**29.60**	26385	25.00	24569	22.40	24770	21.50	21104	17.40	20200	15.10	18451	13.00	**16408**	10.80
[12]	-	-	-	-	-	-	26030	25.40	26244	25.50	21101	17.30	18951	13.40	16662	11.20
[13]	-	-	**22613**	**20.40**	**22405**	20.20	21853	18.50	21368	16.90	19943	14.40	**17965**	**12.40**	16492	**10.40**
[23]	-	-	-	-	27538	27.20	24804	23.50	21484	17.80	20034	15.20	18252	12.90	16644	10.80
32 bits																
	T=1.50ns		T=1.51ns		T=1.53ns		T=1.55ns		T=1.60ns		T=1.70ns		T=1.90ns		T=2.00ns	
*S-MB*1	**40919**	**37.20**	**39284**	34.00	38617	33.30	35940	29.70	33475	24.10	31075	20.20	28253	17.30	27137	16.70
*S-MB*2	-	-	40690	35.10	38206	**31.20**	**35591**	**28.50**	33914	24.80	30739	20.40	28886	17.00	27312	**15.90**
*S-MB*3	-	-	-	-	**37841**	31.40	35855	29.10	**33254**	**23.90**	**30480**	20.60	28439	17.60	27557	17.00
[12]	-	-	-	-	-	-	39406	34.40	34648	27.80	31965	22.50	28723	17.70	27402	17.00
[13]	-	-	40444	36.20	39393	34.40	38285	30.50	33855	24.40	30894	**20.10**	28668	17.90	27593	16.50
[23]	-	-	39422	**33.40**	38983	32.20	36540	30.10	33619	24.50	30918	20.70	**27945**	**17.00**	**27136**	16.30

$A^\dagger$=area occupation, $P^\ddagger$=power consumption.

the area advantages that the proposed schemes deliver over the existing ones.

B. Experimental Evaluation

In this section, we compare the performance of the three proposed recoding schemes explored in Section III with the three schemes described in [12], [13], [23]. We included each of the recoding schemes in a fused Add-Multiply (FAM) operator (Fig. 1(b)) and implemented them using structural Verilog HDL for both cases of even and odd bit-width of the recoder's input numbers. The Wallace CSA tree and the fast CLA adder have been imported from the Synopsys DesignWare Library. We used Synopsys Design Compiler [18] and the Faraday 90 nm standard cell library [22] to synthesize the evaluated designs. In order to have a fair comparison among all FAM designs, they are implemented identically except for the recoding module. The synthesis was conducted considering the highest degree of optimization in Synopsys Design Compiler and ungrouping the hierarchy of the designs. Since the purpose of this work is to compare the performance of the aforementioned designs, we synthesized each FAM design at the highest achievable frequency. We also synthesized all designs at lower frequencies in order to explore how they behave considering different timing constraints in terms of area, timing and power consumption. For each timing constraint, we simulated the designs using Modelsim for the same set of 2^{16} pairs of input numbers in 2's complement representation which were generated randomly with equal probability of a bit to be 0 or 1. Finally, we used Synopsys PrimeTime-PX [19] to calculate power consumption.

The performance of the FAM designs that include the proposed recoding schemes is considered with respect to the bit-width of the input numbers. Tables IX and X show the area and power measurements of all FAM units for different clock periods for even (i.e., 8, 12, 16, 24 and 32 bits) and odd (i.e., 7, 11, 15, 23 and 31 bits) bit-width of the recoder's inputs respectively. The lowest area and power values of each clock period are notified in bold. Note that the measurements presented in both Tables IX and X should be evaluated considering that the *S-MB* recoder is only a part of the overall circuit of a FAM design.

As shown in Table VIII, there are timing differences among the three existing recoders ([12], [13] and [23]). The recoder of [13] is by $2T_g$ faster than the recoders of [12] and [23] which is verified by the experimental results where the [13] based FAM design achieves lower critical delays compared to the ones that incorporate the recoders of [12] and [23]. The recoding scheme of [23] includes subcomponents while the recoders of [12] and [13] are designed directly in gate-level. The ungrouping of the hierarchy of the recoder of [23] leads to larger timing optimizations than the ones of [12] and [13]. Thus, in both Tables IX and X we observe that the [23] based FAM design can achieve lower critical delays than the FAM design which uses the recoder of [12].

TABLE X
AREA, DELAY AND POWER MEASUREMENTS OF FAM DESIGNS (ODD BIT-WIDTH).

Design	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*	$A^{\dagger}$ *(um²)*	$P^{\ddagger}$ *(mW)*
7 bits																
	T=0.87ns		T=0.88ns		T=0.91ns		T=0.93ns		T=0.95ns		T=1.10ns		T=1.20ns		T=1.40ns	
*S-MB*1	**4156**	**5.66**	**3934**	**5.01**	3584	**4.48**	**3494**	**4.11**	**3163**	**3.47**	**2400**	**2.34**	**2202**	**2.10**	**2078**	**1.79**
*S-MB*2	-	-	-	-	3844	4.83	3657	4.12	3424	3.85	2513	2.51	2224	2.18	2088	1.91
*S-MB*3	-	-	3995	5.38	**3566**	4.51	3494	4.54	3227	4.12	2483	2.51	2237	2.25	2096	1.90
[12]	-	-	-	-	-	-	-	-	3674	4.80	2680	2.72	2403	2.42	2118	1.91
[13]	-	-	-	-	4037	5.38	4025	4.78	3600	4.25	2638	2.62	2310	2.33	2093	1.94
[23]	-	-	-	-	-	-	3811	4.86	3621	4.76	2711	2.83	2463	2.35	2160	2.00
11 bits																
	T=1.03ns		T=1.04ns		T=1.05ns		T=1.06ns		T=1.08ns		T=1.20ns		T=1.30ns		T=1.50ns	
*S-MB*1	**7426**	**8.29**	**7489**	**8.74**	**7248**	**8.32**	**7168**	**8.07**	6764	**7.17**	**5216**	**4.26**	4752	**3.95**	4327	**3.20**
*S-MB*2	8426	10.10	8415	10.00	7551	8.40	7625	8.91	6943	7.61	5349	4.79	**4744**	4.01	**4312**	3.29
*S-MB*3	-	-	-	-	-	-	7584	8.88	6944	8.20	5364	4.53	4841	4.04	4321	3.27
[12]	-	-	-	-	-	-	-	-	7898	9.07	5697	5.09	4990	4.20	4423	3.40
[13]	-	-	7940	9.64	8176	10.10	7555	8.60	**6762**	7.60	5376	4.45	4811	3.76	4391	3.36
[23]	-	-	-	-	7915	9.41	7581	8.83	7066	7.86	5318	4.43	4887	4.10	4474	3.48
15 bits																
	T=1.14ns		T=1.16ns		T=1.17ns		T=1.18ns		T=1.20ns		T=1.30ns		T=1.40ns		T=1.60ns	
*S-MB*1	-	-	**12691**	**14.30**	12973	14.20	12061	12.30	**10573**	**9.58**	8844	7.13	7983	5.94	**7112**	**4.77**
*S-MB*2	**13773**	**17.00**	13278	15.60	**12665**	**13.70**	12127	13.30	10926	11.60	8823	7.24	**7908**	**5.82**	7144	4.88
*S-MB*3	-	-	-	-	12767	14.50	**11540**	**12.30**	11159	11.20	**8731**	7.01	8237	6.42	7188	5.09
[12]	-	-	-	-	-	-	-	-	13889	16.70	9421	8.32	8252	6.31	7356	5.15
[13]	-	-	-	-	-	-	12392	14.00	10796	10.90	8825	**6.78**	8090	6.04	7236	4.91
[23]	-	-	13180	15.70	12595	14.80	11798	13.10	10770	10.80	9093	7.40	8247	6.36	7294	5.35
23 bits																
	T=1.32ns		T=1.34ns		T=1.35ns		T=1.36ns		T=1.38ns		T=1.50ns		T=1.60ns		T=1.80ns	
*S-MB*1	-	-	-	-	-	-	22610	22.30	21358	21.80	18179	13.90	16513	11.80	**14948**	9.85
*S-MB*2	25237	25.90	**22402**	21.70	**21293**	20.20	**21582**	20.10	20317	**17.80**	**17566**	**13.00**	16524	11.70	14952	9.88
*S-MB*3	-	-	22711	22.20	22708	22.70	22483	21.30	**20316**	18.90	17818	13.10	**16310**	11.60	15059	10.30
[12]	-	-	-	-	-	-	-	-	22257	22.80	18756	14.80	17251	12.40	15230	10.30
[13]	**24107**	**24.90**	23005	**21.60**	22372	**20.10**	21754	**20.00**	21142	18.70	18132	13.90	16525	11.60	15005	**9.62**
[23]	-	-	-	-	22745	21.70	22920	21.10	20517	18.90	18437	13.90	16409	**11.40**	15004	9.83
31 bits																
	T=1.48ns		T=1.49ns		T=1.51ns		T=1.52ns		T=1.60ns		T=1.70ns		T=1.90ns		T=2.00ns	
*S-MB*1	-	-	39244	35.10	36420	31.10	36630	30.70	32704	25.40	28429	19.20	26473	16.40	25518	15.50
*S-MB*2	**37920**	**32.80**	**36806**	**30.70**	37786	30.40	36026	28.10	32227	23.50	28344	**18.80**	**26132**	**15.30**	**25314**	**15.00**
*S-MB*3	-	-	39061	33.10	35884	29.50	36414	30.00	32364	23.80	28696	19.20	26263	16.00	25941	15.60
[12]	-	-	-	-	-	-	39114	36.00	32542	26.40	29950	21.10	27609	16.70	25848	16.10
[13]	38635	33.50	37866	31.10	**35757**	**28.80**	**34906**	**27.80**	32341	**22.60**	28671	18.90	26982	17.10	26034	16.30
[23]	-	-	-	-	36980	30.60	36694	31.10	**31854**	25.30	**27899**	19.10	26750	16.10	26003	15.60

$A^{\dagger}$=area occupation, $P^{\ddagger}$=power consumption.

In order to confirm the consistency between the theoretical analysis and the experimental evaluation, we should always consider the results of Tables VIII–X for the area of the recoding schemes in relation with the corresponding delay values. In Table VIII, the existing recoders have similar area complexity. However, in Tables IX and X we observe that the [13] based FAM design is more area efficient than those based on [12] and [23]. This was expected considering it can be synthesized in lower clock periods, which in turn leads to area reduction while the clock period is being relaxed. Also, in the highest clock periods, we observe that the area differences among the FAM designs which incorporate existing recoders are small verifying the theoretical analysis (Table VIII).

As in the case of the recoder of [23], the proposed recoding schemes (*S-MB*1, *S-MB*2 and *S-MB*3) contain a number of sub-components. Thus, although in Table VIII the recoder of [13] is by a T_g faster than the proposed recoding schemes, the experimental results of Tables IX and X show that the FAM designs which incorporate the proposed recoders (mainly the *S-MB*1 and *S-MB*2 FAM designs) are amenable to timing optimizations and deliver improvements in the critical delay compared to the FAM designs based on the existing recoders.

In Figs. 16 and 17 we present a comparison among all FAM designs, in terms of area and power consumption respectively, for even bit-width. For each case that we explored, we focus on the lowest clock period where all FAM designs are synthesized.

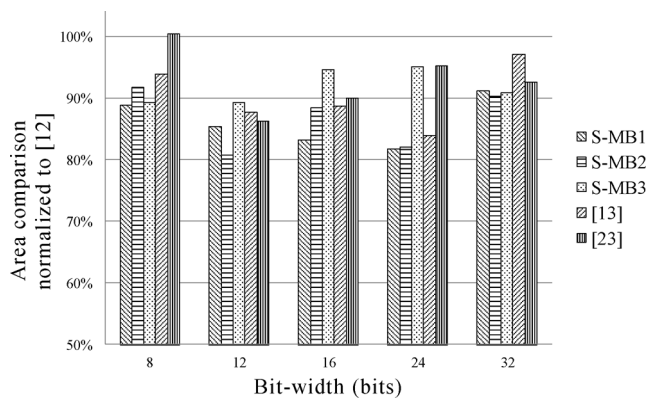


Fig. 16. Area comparison for even bit-width with all values normalized to the corresponding ones of [12].

All values are normalized to the corresponding ones of [12]. As observed, the proposed recoding technique outperforms [12]. In addition, in all cases there is at least one FAM design that incorporates a proposed recoding scheme and outperforms both [13] and [23]. The corresponding figures for odd bit-width are omitted as the results of comparison for both area and power consumption metrics are similar to the ones for even bit-width.

Based on Tables IX and X and considering the measurements for all bit-widths and clock periods, we summarize in Table XI the average area and power gains that the *S-MB*1, *S-MB*2 and

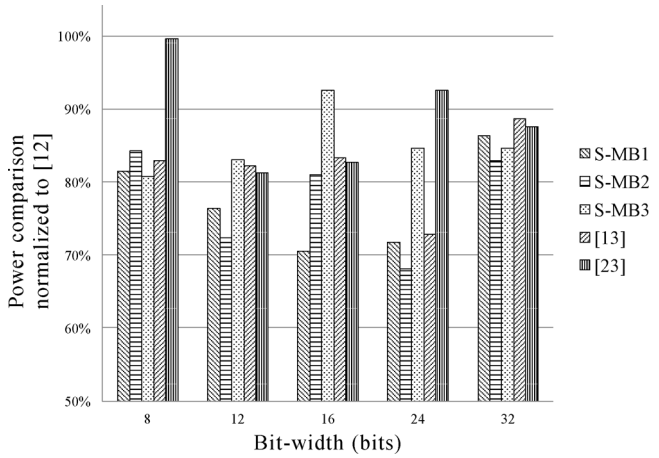


Fig. 17. Power comparison for even bit-width with all values normalized to the corresponding ones of [12].

TABLE XI
AVERAGE GAINS OF AREA COMPLEXITY AND POWER CONSUMPTION.

Even Bit-width						
Design	[12]		[13]		[23]	
	$A^{\dagger}$ (um^2)	$P^{\ddagger}$ (*mW*)	$A^{\dagger}$ (um^2)	$P^{\ddagger}$ (*mW*)	$A^{\dagger}$ (um^2)	$P^{\ddagger}$ (*mW*)
*S-MB*1	7.41%	13.95%	2.10%	3.04%	4.24%	7.50%
*S-MB*2	7.47%	13.68%	2.39%	3.26%	4.13%	6.54%
*S-MB*3	5.66%	10.85%	0.30%	-0.63%	2.10%	3.42%
Odd Bit-width						
Design	[12]		[13]		[23]	
	$A^{\dagger}$ (um^2)	$P^{\ddagger}$ (*mW*)	$A^{\dagger}$ (um^2)	$P^{\ddagger}$ (*mW*)	$A^{\dagger}$ (um^2)	$P^{\ddagger}$ (*mW*)
*S-MB*1	6.36%	11.28%	2.41%	2.40%	2.93%	5.10%
*S-MB*2	6.04%	10.70%	1.50%	1.62%	2.33%	4.85%
*S-MB*3	5.67%	8.72%	1.48%	-0.83%	2.16%	2.30%

$A^{\dagger}$=area occupation, $P^{\ddagger}$=power consumption.

*S-MB*3 based FAM designs present over the ones which include the recoding schemes of [12], [13] and [23]. Table XI shows that the proposed recoding schemes deliver considerable improvements in both area and power dissipation and verifies the theoretically expected (Table VIII) area reductions. Also, as we observe in Table XI, the *S-MB*1 and *S-MB*2 based FAM designs are on average more efficient than the ones based on the *S-MB*3 and the existing schemes. So, we propose the *S-MB*1 and *S-MB*2 recoding schemes as the most efficient ones (regarding their overall performance which includes critical delay, area complexity and power dissipation) since Table XI shows that the *S-MB*1 and *S-MB*2 based FAM designs achieve the highest gains for even and odd bit-width respectively.

V. CONCLUSION

This paper focuses on optimizing the design of the Fused-Add Multiply (FAM) operator. We propose a structured technique for the direct recoding of the sum of two numbers to its MB form. We explore three alternative designs of the proposed *S-MB* recoder and compare them to the existing ones [12], [13] and [23]. The proposed recoding schemes, when they are incorporated in FAM designs, yield considerable performance improvements in comparison with the most efficient recoding schemes found in literature.

ACKNOWLEDGMENT

The authors would like to thank the reviewers for their constructive comments.

REFERENCES

[1] A. Amaricai, M. Vladutiu, and O. Boncalo, "Design issues and implementations for floating-point divide-add fused," *IEEE Trans. Circuits Syst. II–Exp. Briefs*, vol. 57, no. 4, pp. 295–299, Apr. 2010.
[2] E. E. Swartzlander and H. H. M. Saleh, "FFT implementation with fused floating-point operations," *IEEE Trans. Comput.*, vol. 61, no. 2, pp. 284–288, Feb. 2012.
[3] J. J. F. Cavanagh, *Digital Computer Arithmetic*. New York: McGraw-Hill, 1984.
[4] S. Nikolaidis, E. Karaolis, and E. D. Kyriakis-Bitzaros, "Estimation of signal transition activity in FIR filters implemented by a MAC architecture," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 19, no. 1, pp. 164–169, Jan. 2000.
[5] O. Kwon, K. Nowka, and E. E. Swartzlander, "A 16-bit by 16-bit MAC design using fast 5: 3 compressor cells," *J. VLSI Signal Process. Syst.*, vol. 31, no. 2, pp. 77–89, Jun. 2002.
[6] L.-H. Chen, O. T.-C. Chen, T.-Y. Wang, and Y.-C. Ma, "A multiplication-accumulation computation unit with optimized compressors and minimized switching activities," in *Proc. IEEE Int, Symp. Circuits and Syst.*, Kobe, Japan, 2005, vol. 6, pp. 6118–6121.
[7] Y.-H. Seo and D.-W. Kim, "A new VLSI architecture of parallel multiplier–accumulator based on Radix-2 modified Booth algorithm," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 2, pp. 201–208, Feb. 2010.
[8] A. Peymandoust and G. de Micheli, "Using symbolic algebra in algorithmic level DSP synthesis," in *Proc. Design Automation Conf.*, Las Vegas, NV, 2001, pp. 277–282.
[9] W.-C. Yeh and C.-W. Jen, "High-speed and low-power split-radix FFT," *IEEE Trans. Signal Process.*, vol. 51, no. 3, pp. 864–874, Mar. 2003.
[10] C. N. Lyu and D. W. Matula, "Redundant binary Booth recoding," in *Proc. 12th Symp. Comput. Arithmetic*, 1995, pp. 50–57.
[11] J. D. Bruguera and T. Lang, "Implementation of the FFT butterfly with redundant arithmetic," *IEEE Trans. Circuits Syst. Il, Analog Digit. Signal Process.*, vol. 43, no. 10, pp. 717–723, Oct. 1996.
[12] W.-C. Yeh, "Arithmetic Module Design and its Application to FFT," Ph.D. dissertation, Dept. Electron. Eng., National Chiao-Tung University, , Chiao-Tung, 2001.
[13] R. Zimmermann and D. Q. Tran, "Optimized synthesis of sum-of-products," in *Proc. Asilomar Conf. Signals, Syst. Comput.*, Pacific Grove, Washington, DC, 2003, pp. 867–872.
[14] B. Parhami, *Computer Arithmetic: Algorithms and Hardware Designs*. Oxford: Oxford Univ. Press, 2000.
[15] O. L. Macsorley, "High-speed arithmetic in binary computers," *Proc. IRE*, vol. 49, no. 1, pp. 67–91, Jan. 1961.
[16] N. H. E. Weste and D. M. Harris, "Datapath subsystems," in *CMOS VLSI Design: A Circuits and Systems Perspective*, 4th ed. Readington: Addison-Wesley, 2010, ch. 11.
[17] S. Xydis, I. Triantafyllou, G. Economakos, and K. Pekmestzi, "Flexible datapath synthesis through arithmetically optimized operation chaining," in *Proc. NASA/ESA Conf. Adaptive Hardware Syst.*, 2009, pp. 407–414.
[18] [Online]. Available: http://www.synopsys.com/Tools/Implementation/RTLSynthesis/DCUltra/Pages/default.aspx
[19] [Online]. Available: http://www.synopsys.com/Tools/Implementation/SignOff/PrimeTime/Pages/default.aspx
[20] Z. Huang, "High-Level Optimization Techniques for Low-Power Multiplier Design," Ph.D., University of California, Department of Computer Science, Los Angeles, CA, 2003.
[21] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron. Comput.*, vol. EC-13, no. 1, pp. 14–17, 1964.
[22] [Online]. Available: http://www.faraday-tech.com/main/IPonline/category.do?method=showProcessIPList&process=90&categoryID=3089&categoryName=Standard%20Cell
[23] M. Daumas and D. W. Matula, "A Booth multiplier accepting both a redundant or a non redundant input with no additional delay," in *Proc. IEEE Int. Conf. on Application-Specific Syst., Architectures, and Processors*, 2000, pp. 205–214.
[24] Z. Huang and M. D. Ercegovac, "High-performance low-power left-to-right array multiplier design," *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 272–283, Mar. 2005.

Kostas Tsoumanis (S'12) received the Diploma at the Department of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece, in 2010, where he is currently pursuing the Ph.D. degree.

His research interests include hardware-efficient implementation of arithmetic operations and low-power design of Digital Signal Processing algorithms. He is a co-author of a number of research papers published in international conferences.

Sotiris Xydis received the Diploma and the Ph.D. degree in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2005 and 2011, respectively.

Currently, he holds a Post-Doctoral Research Fellow position with Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy. His research interests include design space exploration for system level and datapath synthesis, multi-/many-core anf reconfigurable architectures, design and optimization of arithmetic VLSI circuits and power/thermal aware design. He has published over 40 technical and research papers in scientific books, international journals and conferences.

Dr. Xydis is the recipient of the two best paper awards from the NASA/ESA/IEEE International Conference on Adaptive Hardware and Systems 2007 (AHS 2007) and from the 4th Workshop on Parallel Programming and Run-Time Management Techniques for Many-core Architectures (PARMA 2013).

Constantinos Efstathiou received the B.S. degree in physics in 1979 and the M.Sc. degree in electronics in 1982, both from the University of Athens, Athens, Greece. In 1985, he received the Ph.D. degree in computer science from the University of Thessaloniki, Thessaloniki, Greece.

He has worked as Digital System Engineer in the Research Center of the Hellenic Air Force and as a Computer System Engineer in the Department of Informatics Development of the Greek Government. He also worked as a Digital System Engineer in the Hellenic Telecommunication Organization (OTE). Since 1994, he has been a Professor in the Department of Informatics of the Technological and Educational Institute (TEI) of Athens, where he is responsible for the "Digital Design" and "Computer Systems Organization" courses of the undergraduate program. He is the owner of an International Patent and the author of several scientific articles in International Journals and Proceedings of International Conferences. He also serves as an invited reviewer in several well-known international conferences and journals. His published work is referred to in international edition books, journal articles, articles published in conference proceedings, as well as cited in many M.Sc. and Ph.D. theses.

Nikos Moschopoulos received the Diploma from the Department of Computer Engineering and Informatics, the University of Patras, Patras, Greece, in 1996 and the Ph.D. degree from the Department of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece, in 2001.

In 2001, he joined Atmel Corp as an IC Engineer and an ASIC Project Leader, where he taped out more than 7 complex asics, in leading technologies. In 2007, he co-founded Sitel Semiconductor Hellas, a subsidiary of Sitel Semi B.V., a semiconductor leader in cordless chips, and initiated the Voice over IP (VoIP) asics product line. Since April 2011, he is a Lecturer with the Department of Electrical & Computer Engineering, National Technical University of Athens. His research interests include SoC architecture, low power design and cryptography in hardware.

Kiamal Pekmestzi received the Diploma from the National Technical University of Athens, Athens, Greece, in 1975, and the Ph.D. degree from the University of Patras, Patras, Greece, in 1981, both in electrical engineering.

From 1975 to 1981, he was a Research Fellow with the Electronics Department, Nuclear Research Center "Demokritos". From 1983 to 1985, he was a Professor with the Higher School of Electronics, Athens, Greece. Since 1985, he has been with the National Technical University of Athens, where he is currently a Professor with the Department of Electrical and Computer Engineering. His research interests include efficient implementation of arithmetic operations, design of embedded and microprocessor-based systems, architectures for reconfigurable computing, VLSI implementation of cryptography, and digital signal processing algorithms.