INTEGRATION, the VLSI journal 43 (2010) 42–48

Contents lists available at ScienceDirect

INTEGRATION, the VLSI journal

journal homepage: www.elsevier.com/locate/vlsi

Fast modulo 2^n+1 multi-operand adders and residue generators

H.T. Vergos [a,*], D. Bakalis [b], C. Efstathiou [c]

[a] Computer Engineering and Informatics Department, University of Patras, 26500 Patras, Greece
[b] Department of Physics, University of Patras, 26500 Patras, Greece
[c] Informatics Department, ATEI of Athens, 12210 Athens, Greece

ARTICLE INFO

Article history:
Received 14 May 2008
Received in revised form
29 April 2009
Accepted 29 April 2009

Keywords:
Residue arithmetic
Modulo 2^n+1 addition
Residue number system
Diminished-1 adder
Carry-save addition
Multi-operand addition
Residue generation

ABSTRACT

In this manuscript novel architectures for modulo 2^n+1 multi-operand addition and residue generation are introduced. The proposed arithmetic components consist of a translation stage, an inverted end-around-carry carry-save-adder tree and an enhanced diminished-1 modulo 2^n+1 adder. Qualitative and quantitative results indicate that the proposed architectures result in significantly faster and in several cases smaller circuits than the previously proposed.

1. Introduction

Residue arithmetic has been used in digital computing systems for many years. In particular, modulo 2^n+1 arithmetic appears to play an important role in a variety of applications since it is used in pseudorandom number generation, cryptography [1–3] and convolution computations without round-off errors [4]. Moreover, it is most commonly met as a part of a residue number system (RNS) [5,6] which is an arithmetic system well suited to applications in which the operations are limited to addition, subtraction and multiplication. The RNS has been adopted in the design of digital signal processors [6–9], FIR filters [10,11] and communication components [12,13], offering enhanced operation speed and increased low-power characteristics.

In an RNS application which uses as its base the most frequently used three moduli set $\{2^n, 2^n-1, 2^n+1\}$ the execution delay is dictated by the modulo 2^n+1 channel because this has to handle $(n+1)$-bit wide operands. The diminished-1 representation [14] was introduced to alleviate this problem by having each operand represented decreased by one compared to its weighted representation and by deriving the results in an alternative manner when one or both operands or the results are zero. Hence the operands that are used in the computation units are n-bit wide. The need for handling zero operands and results separately, as well as, the need for converters from/to the weighted to/from the diminished-1 representation, make the use of the diminished-1 representation efficient only when a large number of calculations take place before a new conversion is required. In all other cases, modulo 2^n+1 components for operands in weighted representation are more suitable.

Several architectures have been presented for modulo 2^n+1 arithmetic components assuming the weighted operands representation. Among them, there are architectures for designing multi-operand modulo adders (MOMAs) [15–19] and residue generators (RGs) [18,19]. In this paper we present novel architectures for designing these arithmetic components that result in faster, and in several cases smaller, circuits than the previously proposed.

The rest of the paper is organized as follows. The next section presents an overview of multi-operand modulo adders and residue generators. The proposed architecture for MOMAs is analytically derived in Section 3. Its use in the design of efficient RGs is investigated in Section 4. Qualitative and quantitative comparison results against the architectures proposed in [18], that are currently considered the most efficient ones, are given in Section 5. Finally, the last section concludes the paper.

2. Overview of MOMAs and RGs

Hardware support for multi-operand modulo addition is highly appreciated in several multiply-and-add intensive computations performed over an RNS base such as digital filtering, convolution estimation and FFT transforms. Let $|X|_M$ denote the residue of X taken modulo M. A MOMA is a circuit that accepts k operands,

* Corresponding author.
E-mail address: vergos@ceid.upatras.gr (H.T. Vergos).

0167-9260/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.vlsi.2009.04.002

suppose $X_1, X_2, \ldots, X_k$, with $0 \le X_1, X_2, \ldots, X_k < M$ and computes the residue of their sum taken modulo M, that is, it computes $S = |X_1 + X_2 + \cdots + X_k|_M$, with $0 \le S < M$. In the following such a multi-operand modulo adder will be denoted as MOMA(k, M). Hwang [20, p. 99] suggested for the first time in the open literature a MOMA for $M = 2^n - 1$ implemented by a carry-save adder (CSA) tree with end-around carry (EAC). This MOMA has the same delay complexity as an integer multi-operand adder. The first effort for a modulo $2^n + 1$ MOMA for inputs that follow the weighted representation (hereafter denoted as *weighted MOMA*) appeared in [15] but the proposed architecture required several parallel-adders connected in series. The problem of designing MOMAs for generalized moduli was considered in [16–19]. The architecture proposed for weighted MOMAs in [18] has been shown to be more efficient than those of [16,17]. However, it still requires two parallel adders connected in series (the second adder actually performs a constant subtraction) to provide an unbiased result and the CSA tree needs to be carefully designed for every distinct number of operands because of its irregularity. In [19] the need for two parallel adders connected in series was canceled at the cost of doubling each CSA stage of the tree and requiring four carry lookahead (CLA) units at the final stage that operate in parallel. Unfortunately, [19] does not effectively exploit the properties of arithmetic modulo $2^n + 1$. It therefore results in weighted MOMAs clearly inferior than those of [18].

Applications that rely on modulo arithmetic need a circuit that accepts as input a k-bit binary operand X and produces at the output the residue of this operand taken modulo M, that is, it computes $|X|_M$. This arithmetic component is called a residue generator and will be denoted as RG(k, M) in the following. Efficient RGs for the generic modulo case as well as for specific moduli cases, such as $2^n + 1$, have been proposed in [18]. The RGs of [18], which are considered the most efficient ones for the modulo $2^n + 1$ case, follow the same architecture as the MOMAs proposed in [18]. That is, they are composed of a CSA tree and a parallel adder that since it may output the desired residue in biased format, it needs to be followed by a constant subtractor and a multiplexer to select between the result of the adder or the subtractor. Hence, two parallel adders connected in series have to be used in those circuits too.

In this manuscript we introduce a new architecture for designing weighted MOMAs. The proposed architecture relies on n-bit vectors computations. To this end, we introduce a translator circuit that enables to express the modulo $2^n + 1$ sum of two $(n+1)$-bit operands as a congruent modulo $2^n + 1$ sum of n-bit operands. Each translator circuit accepts two $(n+1)$-bit weighted operands, A_i and B_i, with $0 \le A_i, B_i \le 2^n$ and computes two n-bit vectors U_i and Y_i, such that $|A + B|_{2^n+1} = |Y + U + 1|_{2^n+1}$. The formal derivation of the translator circuit is given in Section 3.1. It is shown that the translator circuit is a simplified CSA stage; therefore, it has small area requirements and a constant execution delay. For a k operand MOMA $\lceil k/2 \rceil$ translator circuits are used in parallel. An inverted EAC CSA adder tree is then used for reducing the outputs of these translators along with a vector that represents a total correction term in two final addends. The total correction factor accounts for both the $+1$ term introduced by each translator and the correction due to the adder tree itself. The formal introduction of the inverted EAC CSA adder tree along with the analytical derivation of the total correction term are given in Section 3.2. The final module used in the proposed architecture is an enhanced diminished-1 modulo $2^n + 1$ parallel adder. This module accepts the outputs of the adder tree and computes the $(n+1)$-bit result of the multi-operand addition. The modifications required over a normal diminished-1 adder are presented in Section 3.3 and are shown to have very small implementation area while they do not contribute to the critical path of the diminished-1 adder.

Table 1 summarizes the modules used in the proposed MOMA architecture. For each one it presents its inputs and outputs and briefly describes its functionality. The proposed architecture requires just one parallel adder and simple CSA stages; therefore, it outperforms all previous proposals on designing weighted MOMAs in terms of delay. In several cases it also provides more compact multi-operand adders.

The proposed MOMA architecture can also be used in the design of a modulo $2^n + 1$ RG circuit. The resulting residue generators are also faster than those of [18]. Finally, in both the MOMA and RG cases, the proposed circuits are built around a completely regular inverted EAC CSA tree for every modulus value and number of operands or number of bits, respectively, resulting in more efficient CMOS VLSI realizations.

Table 1
Inputs, outputs and functionality of the modules used in the proposed MOMA architecture.

Translator module	
Inputs	Two $(n+1)$-bit vectors A_i and B_i, with $0 \le A_i, B_i \le 2^n$
Outputs	Two n-bit vectors Y_i and U_i
Functionality	$\lvert A_i + B_i \rvert_{2^n+1} = \lvert Y_i + U_i + 1 \rvert_{2^n+1}$ Enables to express the modulo $2^n + 1$ sum of two $(n+1)-$bit operands as a congruent modulo $2^n + 1$ sum of n-bit operands
Notes	1. A MOMA$(k, 2^n + 1)$ requires $\lceil k/2 \rceil$ translators that are used in parallel 2. The outputs of the translators are driven to an inverted EAC CSA tree
Inverted EAC CSA tree	
Inputs	The n-bit vectors $Y_1, Y_2, \ldots, Y_{\lceil k/2 \rceil}$, $U_1, U_2, \ldots, U_{\lceil k/2 \rceil}$ produced by the translators and the n-bit vector $COR = \lvert - \lceil k/2 \rceil \rvert_{2^n+1}$
Outputs	Two n-bit vectors F and J
Functionality	$\lvert \sum_{i=1}^{\lceil k/2 \rceil} Y_i + \sum_{i=1}^{\lceil k/2 \rceil} U_i + \lceil k/2 \rceil \rvert_{2^n+1} = \lvert (F + J + 1) - \lceil k/2 \rceil - COR \rvert_{2^n+1}$ $= \lvert F + J + 1 \rvert_{2^n+1}$ Reduces the $(2 \times \lceil k/2 \rceil + 1)$ n-bit input addends in two final addends
Notes	1. *COR* accounts for all required correction terms 2. The outputs F and J of the tree are driven to the enhanced diminished-1 modulo $2^n + 1$ adder
Enhanced diminished-1 modulo $2^n + 1$ *adder*	
Inputs	The n-bit vectors F and J
Outputs	The $(n+1)$-bit vector S
Functionality	$S = \lvert F + J + 1 \rvert_{2^n+1}$
Notes	1. A diminished-1 adder is used for the n least significant bits of S 2. The most significant bit of S is set when F and J are bit-wise complementary.