A Unifying Approach for Weighted and Diminished-1 Modulo $2^n + 1$ Addition

H. T. Vergos, *Member, IEEE*, and C. Efstathiou

***Abstract*—In this paper, it is shown that every architecture proposed for modulo $2^n + 1$ addition of operands that follow the diminished-1 representation can also be used in the design of modulo $2^n + 1$ adders for operands that follow the weighted representation. This is achieved by the addition of a constant-time operator composed of a simplified carry-save adder stage. The experimental results indicate that many architectures already proposed for the diminished-1 case, lead to very efficient adders for weighted operands, under this unifying approach.**

***Index Terms*—Diminished-1 representation, modulo $2^n + 1$ addition, residue arithmetic, residue number system.**

I. Introduction

RESIDUE arithmetic has been used in digital computing systems for many years. In particular, arithmetic modulo $2^n + 1$ appears to play an important role in a variety of applications. Modulo $2^n + 1$ arithmetic is most commonly met in the residue number system (RNS) [1], which is an arithmetic system well-suited to applications in which the operations are limited to addition, subtraction and multiplication; a common case for several digital signal processor (DSP) algorithms. The RNS has been used for the design of digital signal processors [1]–[4], finite-impulse response (FIR) filters [5], [6], and communication components [7], [8].

Three-moduli sets of the form $\{2^n - 1, 2^n, 2^n + 1\}$ have received significant attention as the RNS base, mainly because of the existence of efficient residue to binary converters [9]. Addition in such systems is performed using three channels, that, in fact, are a modulo $2^n - 1$ (equivalently one's complement), a modulo 2^n and a modulo $2^n + 1$ adder. From this, we conclude that the design of an efficient modulo $2^n + 1$ adder is a vital task in RNS-based applications that include a modulus of the $2^n + 1$ form. Unfortunately, in an RNS that uses a three moduli set $\{2^n - 1, 2^n, 2^n + 1\}$, the modulo $2^n + 1$ channel becomes the execution-rate bottleneck, since it has to deal with $(n + 1)$-bit operands, while the other two channels operate on n-bit ones.

The diminished-1 representation [10] 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. Let A and B denote two n-bit operands, such that $0 \leq A, B \leq 2^n$. In the diminished-1 representation, $A^\star$ and $B^\star$ are used to represent A and B, with $A^\star = |A - 1|_{2^n+1}$ and $B^\star = |B - 1|_{2^n+1}$. The diminished-1 sum $S^\star$ is then computed as $S^\star = |S - 1|_{2^n+1} = |A + B - 1|_{2^n+1} = |A^\star + B^\star + 1|_{2^n+1}$, by a diminished-1 adder, which is an adder that increments the integer sum of $A^\star$ and $B^\star$ whenever the carry flag of their respective integer addition is not set. A diminished-1 adder can be derived by connecting the inverted carry output of an integer adder back to its carry input. However, such solutions are inefficient due to the resulting oscillations [11]. Therefore, a number of efficient architectures that do not suffer from oscillations [12]–[14] have been proposed.

The need for handling zero operands and results separately, as well as the need for time and hardware consuming input (output) translators 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, including all applications apart from RNS implementations, modulo $2^n + 1$ adders with operands in weighted representation are more suitable. Efficient architectures for modulo $2^n + 1$ adders for operands in weighted representation have also been proposed [15], [16].

These two cases, namely modulo $2^n + 1$ adders that operate on operands in the diminished-1 representation (hereafter called *diminished-1 adders*) and those that operate on operands that follow a weighted representation (hereafter called *weighted adders*) have, so far, been considered distinct cases and efficient architectures for them have been studied independently. In this brief it is shown that these two alternatives can be unified. Given two $(n + 1)$-bit numbers A and B, the problem of computing two n-bit numbers Y and U, such that $Y + U + 1$ to be congruent to $A + B$ modulo $2^n + 1$, is attacked. It is shown that this problem has a constant time solution, enabling every architecture that has been or will be proposed for diminished-1 addition to also be used for addition of operands in the weighted representation. The required unifying arithmetic operator is just a simplified inverted end-around carry-save-adder (CSA) stage.

The rest of this paper is organized as follows. In Section II, an architecture for weighted adders that relies on the use of diminished-1 adders is formally derived. This architecture is then used in Section III along with already known architectures for diminished-1 adders, for exploring weighted adders with interesting area and delay tradeoffs. Our conclusions are drawn in Section IV.

Manuscript received September 20, 2007; revised February 7, 2008. Current version published October 15, 2008. This work was supported in part by the E.U. and by the Greek Government within the framework of the Educational and Initial Vocational Training program "Archimedes".

H. T. Vergos is with the Computer Engineering and Informatics Department, University of Patras, Patras 26500, Greece (e-mail: vergos@ceid.upatras.gr).

C. Efstathiou is with the Informatics Department, TEI of Athens, 12210 Egaleo, Athens, Greece (e-mail: cefsta@teiath.gr).

Digital Object Identifier 10.1109/TCSII.2008.2001964

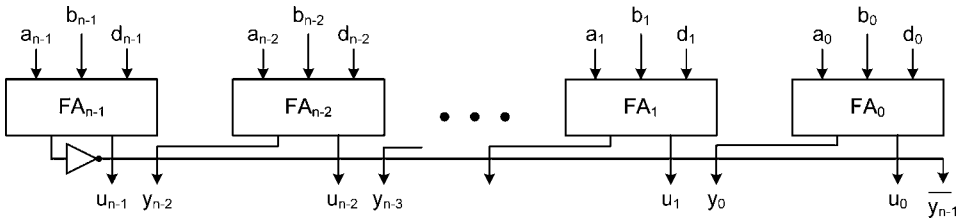

Fig. 1. CSA stage with inverted end-around carry.

II. Unified Design Derivation

Suppose that $A = a_n a_{n-1} a_{n-2}, \ldots, a_1 a_0$ and $B = b_n b_{n-1} b_{n-2}, \ldots, b_1 b_0$ denote two operands in the weighted representation, with $0 \leq A, B \leq 2^n$. Let A_n and B_n denote the n-bit vectors composed of the least significant bits of A and B, respectively. Equivalently, we have that $A = a_n \times 2^n + A_n$ and $B = b_n \times 2^n + B_n$. For the weighted modulo $2^n + 1$ addition of A with B it holds that

$$\begin{aligned} |A+B|_{2^n+1} &= |(2^n \times a_n + A_n) + (2^n \times b_n + B_n)|_{2^n+1} \\ &= |2^n \times (a_n + b_n) + A_n + B_n|_{2^n+1}. \end{aligned} \tag{1}$$

Let s_n and c_{n+1} denote the sum and carry bits of the $(a_n + b_n) \times 2^n$ addition contained in relation (1), respectively, and let $\overline{x}$ denote the complement of x. s_n and c_{n+1} are bits with weights 2^n and 2^{n+1}, respectively. Using them in (1) produces

$$\begin{aligned} |A+B|_{2^n+1} &= |2^{n+1} \times c_{n+1} + 2^n \times s_n + A_n + B_n|_{2^{n+1}} \\ &= |A_n + B_n - 2 \times c_{n+1} - s_n|_{2^n+1}. \end{aligned} \tag{2}$$

Using that for $x \in \{0, 1\}$ it holds that

$$\begin{aligned} |-x|_{2^n+1} &= |2^n + 1 - x|_{2^n+1} \\ &= |2^n + (1-x)|_{2^n+1} \\ &= |\overline{x} + 2^n|_{2^n+1} = |\overline{x} - 1|_{2^n+1} \end{aligned} \tag{3}$$

from (2), we can further derive that

$$\begin{aligned} |A+B|_{2^n+1} &= |A_n + B_n + 2 \times \overline{c_{n+1}} + \overline{s_n} - (2+1)|_{2^n+1} \\ &= |A_n + B_n + 2 \times \overline{c_{n+1}} + \overline{s_n} + 2^n - 2|_{2^n+1} \\ &= |A_n + B_n + (2^n - 4 + 2 \times \overline{c_{n+1}} + \overline{s_n}) \\ &\quad + 2|_{2^n+1} \\ &= ||A_n + B_n + D + 1|_{2^n+1} + 1|_{2^n+1} \end{aligned} \tag{4}$$

where $D = 2^n - 4 + 2 \times \overline{c_{n+1}} + \overline{s_n}$. D is represented by the n-bit vector $111 \ldots 1 \overline{c_{n+1}} \overline{s_n}$.

Suppose that $Y^\star = y_{n-1} y_{n-2} \ldots y_0$ and $U = u_{n-1} u_{n-2} \ldots u_0$ denote the carry and sum output vectors of the carry-save addition of A_n, B_n and D indicated in (4), respectively. It then holds that

$$\begin{aligned} |A+B|_{2^n+1} &= ||A_n + B_n + D + 1|_{2^n+1} + 1|_{2^n+1} \\ &= \left| \left| \sum_{i=0}^{n-1} (2^{i+1} \times y_i) + \sum_{i=0}^{n-1} (2^i \times u_i) \right|_{2^n+1} + 1 \right|_{2^n+1} \\ &= \left| \left| \sum_{i=1}^{n-1} \left(2^i \times (y_{i-1} + u_i)\right) + u_0 \right. \right. \\ &\quad \left. \left. + 2^n y_{n-1} + 1 \right|_{2^n+1} + 1 \right|_{2^n+1} \\ &= \left| \left| \sum_{i=1}^{n-1} \left(2^i \times (y_{i-1} + u_i)\right) + u_0 \right. \right. \\ &\quad \left. \left. - y_{n-1} + 1 \right|_{2^n+1} + 1 \right|_{2^n+1} \\ &= \left| \left| \sum_{i=1}^{n-1} \left(2^i \times (y_{i-1} + u_i)\right) \right. \right. \\ &\quad \left. \left. + u_0 + \overline{y}_{n-1} \right|_{2^n+1} + 1 \right|_{2^n+1} \\ &= ||Y + U|_{2^n+1} + 1|_{2^n+1} \\ &= |Y + U + 1|_{2^n+1} \end{aligned} \tag{5}$$

where $Y = y_{n-2} y_{n-3}, \ldots, y_0 \overline{y}_{n-1}$.

The previous analysis indicates that the vectors A_n, B_n, and D can be added by the inverted end-around carry save adder stage presented in Fig. 1. We can then drive its outputs Y and U to a diminished-1 adder, that will provide at its output $|Y + U + 1|_{2^n+1}$, that is, it will provide the n least significant bits of the weighted modulo addition of A and B.

The most significant bit of the weighted sum should be at 1, only when $|A + B|_{2^n+1} = 2^n$, or equivalently when $|Y + U + 1|_{2^n+1} = 2^n$, or equivalently (since $0 \leq Y, U \leq 2^n - 1$) when $Y + U = 2^n - 1$, that is, when Y and U are bit-wise complementary. This condition can be easily detected as the logical AND of the XOR of the bits of Y and U with the same weight. Since in every fast adder architecture there is a preprocessing stage that apart from the generate and propagate terms also computes the half-sum term, that is the XOR of the corresponding input operands bits, the extra hardware required for the most significant bit of the weighted addition is small; just ANDing the half-sum terms together. It should be noted that this operation will not add any delay on the critical path of the adder.

Fig. 2 presents the implementation that results from the previous analysis for a modulo $2^n + 1$ adder for operands in the weighted representation. It is composed of a diminished-1 adder and an inverted end-around-carry CSA stage. The full adders of the CSA stage perform the $|A_n + B_n + D|_{2^n+1}$ addition. The FAs at bit positions 2 up to $(n-1)$ are denoted as FA^+s since one of their operands coming from vector D is 1. Therefore,

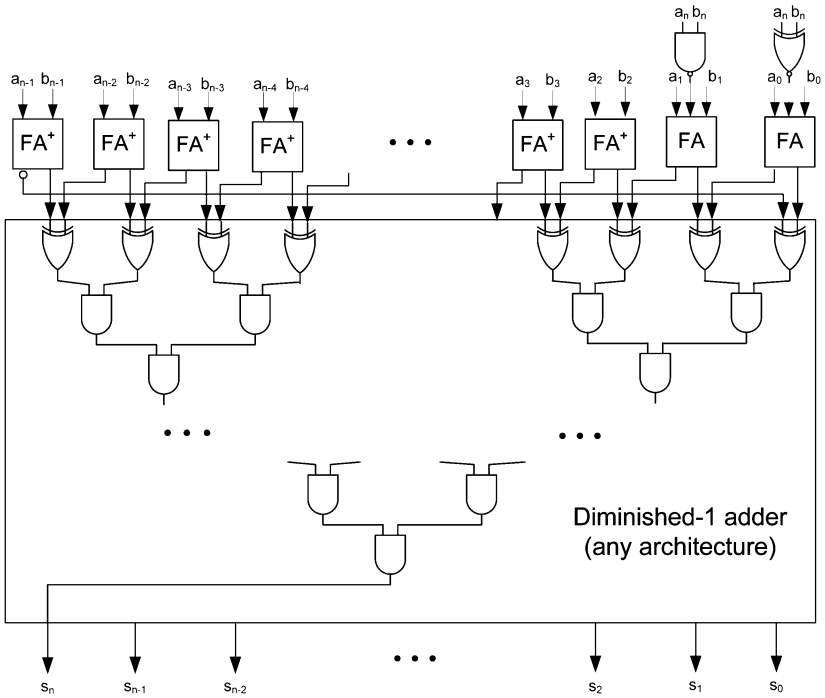

Fig. 2. Modulo $2^n + 1$ adder for weighted operands built using a diminished-1 adder.

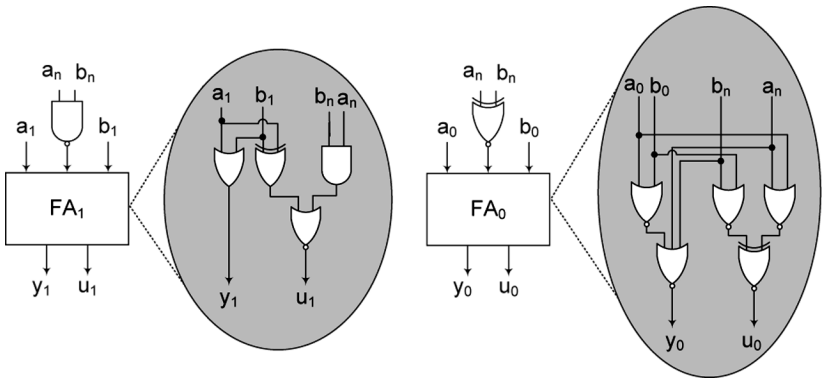

Fig. 3. Simplified circuitry for the least significant bit positions.

they have a hardware and delay complexity equal to that of a half adder.

Several simplifications can also be performed to the two rightmost FAs along with the accompanying NAND and XNOR gates by considering that $a_n(b_n)$ and a_1 or a_0 (and b_1 or b_0) can not be simultaneously at 1. Fig. 3 presents examples of simplified circuits at these least significant bit positions. However, even more aggressive simplification is possible within the preprocessing stage of the diminished-1 adder by considering that the propagate and generate signals at bit position 1 depend on signals y_0 and u_1 that are both dependent on a_n and b_n. The most significant carry bit produced by the CSA is inverted and then driven to the least significant position. The sum and carry bits of equal weight produced by the CSA stage are then driven to the diminished-1 adder. Obviously, any architecture proposed, can be used for the latter. The diminished-1 adder's result forms the n least significant bits of the weighted sum. The indication of complementary input vectors at the diminished-1 adder is the most significant bit of the weighted sum.

III. Emerging Architectures

A number of distinct architectures has already been proposed for diminished-1 modulo $2^n + 1$ addition. Using each of these architectures in the unified design of Fig. 2 provides a new architecture for weighted modulo 2^n+1 adder design with its own area and delay characteristics, which we explore in this section.

For our comparisons, we consider five different implementations for the diminished-1 adder of Fig. 2, namely as follows.

- The single and multi level carry-lookahead (CLA) architectures proposed in [13]. The resulting weighted-adders will be referred to as CLA weighted adders.
- The parallel-prefix architectures proposed in [12]. The resulting weighted adders will be referred to as Ladner–Fischer (LF) or Kogge–Stone (KS) weighted adders when a LF or a KS prefix algorithm is used in the carry computation unit, respectively.
- The parallel-prefix (PPD) architecture proposed in [13]. The resulting weighted-adders will be referred to as PPD weighted adders.

TABLE I
DELAY (NANOSECONDS) AND AREA (MICROMETERS SQUARED) RESULTS FOR WEIGHTED MODULO $2^n + 1$ ADDERS

	$n = 4$		$n = 8$		$n = 16$		$n = 32$	
Architecture	Delay	Area	Delay	Area	Delay	Area	Delay	Area
CLA	0.46	2,656.6	0.66	5,370.9	0.77	9,778.1	0.91	20,603.5
LF	0.55	2,415.1	0.67	4,065.9	0.77	8,147.9	0.89	16,987.7
KS	0.54	2,606.1	0.66	4,687.9	0.75	9,205.1	0.87	19,155.4
PPD	0.50	2,110.1	0.59	5,444.1	0.69	10,521.9	0.80	21,754.6
SP	Not applicable		0.62	4,632.4	0.73	9,033.8	0.84	19,008.6
PLNG	0.49	2,413.7	0.58	4,793.0	0.67	9,512.9	0.77	20,114.8
TPP	0.54	2,073.5	0.64	4,756.9	0.75	9,209.0	0.87	19,321.1
HIAS	0.63	1,719.8	0.81	4,561.6	0.90	9,546.2	0.99	19,796.6

- The select-prefix (SP) architecture proposed in [14]. We will refer to the resulting weighted adders using the SP weighted adders notation.
- The parallel-prefix architectures based on Ling carries explored in [17]. The resulting weighted-adders will be denoted as PLNG weighted adders.

We compare the area and delay characteristics of these emerging weighted adders against those of the totally parallel-prefix (TPP) adders, suggested in [16] (denoted as TPP adders), and the adders that rely on the multiplexing of the propagate and generate signals before used in a CLA carry computation unit, suggested in [15], which will be denoted as HIAS adders. The latter architectures are considered the current most efficient ones for weighted adder design.

For attaining our comparison results, we extended the generator of [18], for producing structural HDL descriptions for all adder architectures under comparison. In this way, HDL models for the examined adders with operand sizes of up to 32 bits were generated. After simulating every description produced, each design was subsequently mapped in a 0.18-μm technology library [19] assuming typical conditions (1.8 V, 25 °C). Each mapped design was iteratively optimized until no further delay savings were possible. The optimized designs then underwent successive area recovery steps. The same design constraints, such as maximum fan-out, output capacitance, and available input drive strength, were considered for all designs. The optimized netlists along with their constraints were then passed to a standard cell place and route tool. The floorplan initialization information was kept constant for each adder size. After a constraint driven place and route procedure the netlists were back annotated with the extracted information for gathering the reported results.

Table I presents the attained area and delay results for all examined architectures. The delay results are given in nanoseconds, while the area results in micrometers squared. The latter include both cell and interconnect area. The CLA and the SP lines of Table I represent the fastest designs achieved after examining a variety of possibilities. In the case of the CLA adders, it represents the best result out of single or double CLA levels. In the case of the SP adders it represents the fastest adder achieved after an exploration of the number of prefix blocks and the algorithm (LF or KS) used for them.

The results of Table I indicate that the resulting PLNG and PPD architectures lead to the fastest weighted adders in all examined cases, excluding the $n = 4$ case. The PLNG architecture outperforms the current fastest weighted adder architecture [16] by approximately 10.5% over the examined adder range, while the PPD one outperforms [16] by approximately 8%. This level of performance however, also requires increased implementation area, which is approximately increased by 6% and 10.8% by the PLNG and PPD architectures, respectively. The resulting CLA adders are the fastest proposed in the smallest examined case but inefficient in both area and delay terms in the wider operands cases. LF and KS adders are faster than the CLA ones and as fast as the fastest previously reported TPP adders for sufficiently wide operands. The SP adders are efficient in area and delay terms since in all examined cases offer smaller delay than the currently fastest known TPP architecture with smaller implementation area requirements. Considering the area $\times$ delay product as a metric, the SP adders offer more efficient designs than the TPP ones by approximately 5% in all the examined operands range. Finally, the HIAS architecture provides area efficient adders only at the narrower operands case.

IV. CONCLUSION

The problems of designing modulo 2^n+1 adders for operands in the normal weighted and in the diminished-1 representations have thus far been considered separate and studied independently. This paper has shown that every architecture that has been or will be proposed for diminished-1 adders can also be used in the design of weighted adders, by the addition of an inverted EAC CSA stage.

Our results indicate that using this unifying methodology and previously reported architectures for diminished-1 adders, more efficient architectures for weighted adders can be reached. The emerging PLNG and PPD architectures lead to the fastest reported weighted adders, while the CLA one to efficient adders for narrow operands. The emerging SP adders also offer the same or even higher operating speed than the fastest known proposal with lower implementation area complexity.

REFERENCES

[1] M. A. Soderstrand *et al., Residue Number System Arithmetic: Modern Applications in Digital Signal processing*. Piscataway, NJ: IEEE Press, 1986.
[2] F. Taylor, "A single modulus ALU complex for signal Processing," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 33, no. 5, pp. 1302–1315, Oct. 1985.
[3] E. D. Claudio, F. Piazza, and G. Orlandi, "Fast combinatorial RNS processors for DSP applications," *IEEE Trans. Comput.*, vol. 44, no. 5, pp. 624–633, May 1995.

[4] J. Ramirez *et al.*, "RNS-enabled digital signal processor design," *Electron. Lett.*, vol. 38, no. 6, pp. 266–268, 2002.
[5] J. Ramirez, "High performance, reduced complexity programmable RNS-FPL merged FIR filters," *Electron. Lett.*, vol. 38, no. 4, pp. 199–200, 2002.
[6] G. C. Cardarilli, A. Nannarelli, and M. Re, "Reducing power dissipation in FIR filters using the residue number system," in *Proc. IEEE 43rd IEEE Midw. Symp. Circuits Syst.*, 2000, pp. 320–323.
[7] J. Ramirez, A. Garcia, U. Meyer-Baese, and A. Lloris, "Fast RNS FPL-based communications receiver design and implementation," in *Proc. 12th Int. Conf. Field Program. Logic*, 2002, vol. 2438, pp. 472–481.
[8] T. Keller, T. H. Liew, and L. Hanzo, "Adaptive redundant residue number system coded multicarrier modulation," *IEEE J. Sel. Areas Commun.*, vol. C-18, no. 11, pp. 2292–2301, Nov. 2000.
[9] Y. Wang, "Residue to binary converters based on new chinese remainder theorems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 47, no. 3, pp. 197–205, Mar. 2000.
[10] L. M. Leibowitz, "A simplified binary arithmetic for the fermat number transform," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 24, no. 5, pp. 356–359, Oct. 1976.
[11] J. J. Shedletsky, "Comment on the sequential and indeterminate behavior of an end-around-carry adder," *IEEE Trans. Comput.*, vol. 26, no. 3, pp. 271–272, Mar. 1977.
[12] R. Zimmerman, "Efficient VLSI implementation of modulo $(2^n \pm 1)$ addition and multiplication," in *Proc. 14th IEEE Symp. Comput. Arithmetic*, Apr. 1999, pp. 158–167.
[13] H. T. Vergos, C. Efstathiou, and D. Nikolos, "Diminished-one modulo $2^n + 1$ adder design," *IEEE Trans. Comput.*, vol. 51, no. 12, pp. 1389–1399, Dec. 2002.
[14] C. Efstathiou, H. T. Vergos, and D. Nikolos, "Modulo $2^n \pm 1$ adder design using select prefix blocks," *IEEE Trans. Comput.*, vol. 52, no. 11, pp. 1399–1406, Nov. 2003.
[15] A. A. Hiasat, "High-speed and reduced area modular adder structures for RNS," *IEEE Trans. Comput.*, vol. 51, no. 1, pp. 84–89, Jan. 2002.
[16] C. Efstathiou, H. T. Vergos, and D. Nikolos, "Fast parallel-prefix modulo $2^n + 1$ adders," *IEEE Trans. Comput.*, vol. 53, no. 9, pp. 1211–1216, Sep. 2004.
[17] H. T. Vergos, "Fast modulo $2^n + 1$ adder architectures," in *Proc. 22nd Conf. Des. Circuits Integr. Syst.*, 2007, pp. 476–481.
[18] N. Kostaras and H. T. Vergos, "KoVer: A sophisticated residue arithmetic core generator," in *Proc. 16th IEEE Int. Workshop Rapid Syst. Prototyp.*, 2005, pp. 261–263.
[19] *UMC-18, eSi-route/11 0.18 Standard Cell Library*, VST, 2001.