A Family of Area-Time Efficient Modulo $2^n + 1$ Adders

H. T. Vergos
Computer Engineering & Informatics Dept.,
University of Patras, 26500, Rio, Greece.

***Abstract*—A family of diminished-1 modulo $2^n + 1$ adders is proposed in this manuscript. All members of the family use a sparse carry computation unit for deriving only some of the carries in $\log_2 n$ prefix levels, while all the rest carries are computed in an extra one. The proposed adders offer significant area and power savings compared to earlier proposals, while maintaining a high operation speed.**

I. Introduction

A modulo 2^n+1 adder is an important arithmetic component for a variety of applications ranging from random number generation and cryptography up to convolution / correlation computation without rounding and truncation error. It is also a vital part of almost every residue number system (RNS). Most commonly, the diminished-1 representation is used for the adder inputs and output. A diminished-1 adder is equivalent to an inverted end-around carry (EAC) adder [1], [2]. In [1] diminished adders have been proposed that make use of a parallel-prefix carry computation unit that can either follow the Ladner-Fischer (LF) or the Kogge-Stone (KS) algorithm along with an extra prefix level for handling the inverted EAC. Figure 1 presents the proposal of [1] by means of a prefix graph. The operation of the different nodes used in the graph is also given. The well-known carry generate, g_i, carry propagate, p_i and half-sum h_i bits are used in these nodes. In [2] it has been shown that the recirculation of the inverted EAC can be performed within the existing prefix levels, that is, in parallel with the carries' computation. Therefore, the need of an extra prefix level is canceled. Figure 2 presents the proposal of [2]. It uses a parallel-prefix computation unit that for several carries needs a double computation tree. One tree is used to associate generate and propagate signals in their normal form, whereas the second for associating the complemented form of them. By comparing Figures 1 and 2 it becomes obvious that the increased speed of [2] comes at the penalty of heavily increased cell and interconnection area. The same is also true for the full parallel prefix (FPP) and reduced area parallel prefix (RAPP) architectures proposed in [3], that follow a similar to [2] prefix algorithm, but for Ling carries.

II. Proposed Diminished Adders

The proposed adders stem from considering the proposals of [1] and [2], as the two end cases of the number of diminished addition carries that are computed within the first $\log_2 n$ prefix levels. In the first case, only one diminished-1 carry is computed within $\log_2 n$ prefix levels, while in the second case every carry is computed. The proposed adders are derived considering the alternative of computing only some

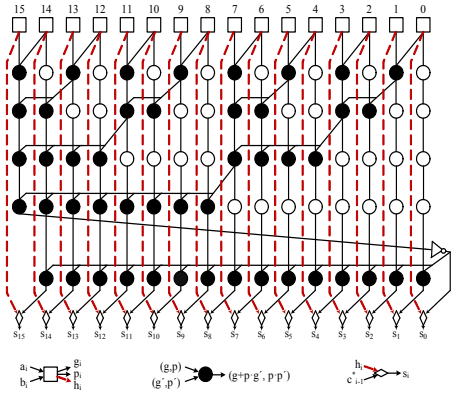

Fig. 1. Diminished-1 modulo $2^{16} + 1$ adder of [1].

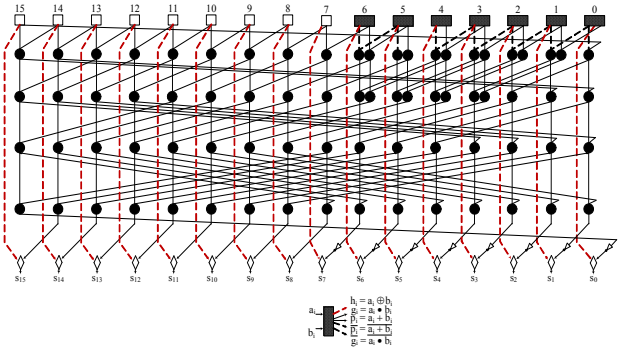

Fig. 2. Diminished-1 modulo $2^{16} + 1$ adder of [2].

of the carries within the first $\log_2 n$ prefix levels. They are based on showing that the modulo $2^n + 1$ carry at bit position $i + 1$, $c^\star_{i+1}$, can be computed based on $c^\star_i$ and available generate and propagate terms. More specifically it can be shown that (the proof is omitted due to space limitations) $c^\star_{i+1} = g_{i+1} + p_{i+1} \cdot c^\star_i$. This reveals that the next carry of the diminished addition can be computed straightforwardly, by associating in a prefix operator the (g_{i+1}, p_{i+1}) pair of generate and propagate terms and the carry of the previous position. Since $c^\star_{i+2}$ can be similarly computed using $c^\star_{i+1}$, which as before can be computed based on $c^\star_i$, it becomes obvious that we can compute every carry $c^\star_{i+k}$ of the diminished addition associating $c^\star_i$ and the $(g_{k:i+1}, p_{k:i+1})$ pair of group generate and propagate terms in a prefix operator. As a result, we can compute in $\log_2 n$ prefix levels any number of the diminished addition carries and then use as many of them as we wish to compute the rest in a further prefix level. In this way a whole family of diminished adders is derived. While all adders of the family have a carry computation unit

978-0-7695-4076-4/10 $26.00 © 2010 IEEE
DOI 10.1109/ISVLSI.2010.35

TABLE I
EXPERIMENTAL RESULTS FOR DIMINISHED-1 ADDERS

	[1] LF				[1] KS				[2]				[4]			
n	Delay	Area	Power	$A \times T^2$	Delay	Area	Power	$A \times T^2$	Delay	Area	Power	$A \times T^2$	Delay	Area	Power	$A \times T^2$
4	299	768.37	0.245	1.29	298	761.55	0.226	1.27	260	790.66	0.240	1.00	N/A			
8	371	1671.57	0.574	1.35	369	1978.23	0.682	1.58	324	2023.46	0.665	1.25	350	1513.00	0.456	1.09
16	431	3805.50	1.198	1.54	445	4656.65	1.562	2.01	388	5168.50	1.807	1.70	411	3746.26	1.141	1.38
32	509	8343.34	2.594	1.49	544	11368.73	3.962	2.33	460	13152.81	4.870	1.92	492	7767.00	2.503	1.30
	[3] FPP				[3] RAPP				Prop-$\frac{n}{2}$				Prop-$\frac{n}{4}$			
n	Delay	Area	Power	$A \times T^2$	Delay	Area	Power	$A \times T^2$	Delay	Area	Power	$A \times T^2$	Delay	Area	Power	$A \times T^2$
4	N/A				N/A				286	650.96	0.182	1.00	293	677.77	0.198	1.09
8	302	1952.76	0.629	1.05	335	1748.09	0.514	1.15	342	1499.52	0.468	1.03	346	1421.76	0.429	1.00
16	370	5118.35	1.711	1.53	413	4650.92	1.558	1.73	403	3289.95	1.057	1.17	397	2906.73	0.860	1.00
32	442	14986.88	5.354	2.02	481	12054.81	4.322	1.93	471	7799.13	2.594	1.20	463	6750.10	2.080	1.00

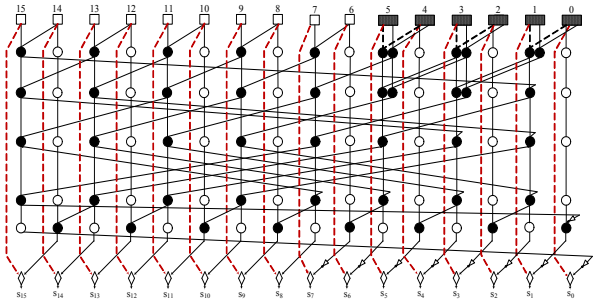

Fig. 3. Prop-$\frac{n}{2}$ diminished-1 modulo $2^{16}+1$ adder.

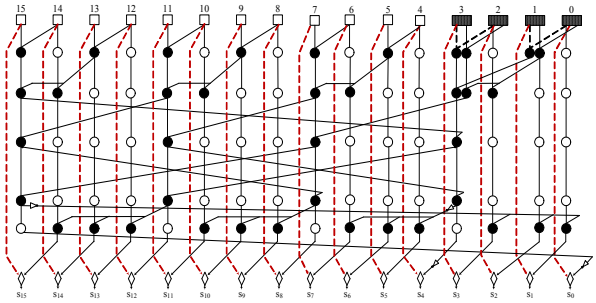

Fig. 4. Prop-$\frac{n}{4}$ diminished-1 modulo $2^{16}+1$ adder.

composed of $\log_2 n+1$ prefix levels, each has its own fan-out requirements and cell and interconnection area. The notation Prop-k is hereafter used, to denote the proposed diminished-1 modulo 2^n+1 adders in which k out of the total n carries of the diminished addition are computed in the first $\log_2 n$ prefix levels. Under this definition, the adders proposed in [1] are the Prop-1, while the adders of [2] are the Prop-n members of the family. Figures 3 and 4 present the proposed Prop-$\frac{n}{2}$ and Prop-$\frac{n}{4}$ diminished adders. In the Prop-$\frac{n}{2}$ adder case only the odd numbered carries are computed in $\log_2 n$ prefix levels, while the even numbered ones in the last prefix level. This adder offers a fan-out equal to 2 and has a similar structure to the area-time efficient adders derived by the Han-Carlson algorithm for integer addition. The Prop-$\frac{n}{4}$ adder on the other hand has a fan-out equal to 4 but requires significantly less prefix operators along with their interconnections than the Prop-$\frac{n}{2}$ adder. It is noted that the elimination of several prefix operators also removes their associated interconnections. As a result the wiring complexity is also significantly reduced.

III. COMPARISONS

The proposed diminished adders were quantitatively compared against the proposals of [1]–[4] for $n = 4, 8, 16$ or 32 by implementations in a $90nm$ technology. Table I lists the attained results. Delay results are given in ps, area results in μm^2, and average power results in mW. The proposed family of adders outperforms the earlier proposals of [1] and [4] in delay, area and average power consumption terms. They also outperform the adders designed according to the RAPP architecture of [3] in all terms in the two widest examined cases. On the other hand, they can not reach the speed of the adders proposed in [2] and the ultimate speed of the FPP adders [3]. However, in both these proposals, this level of speed performance is achieved at a very high area and average power consumption price. More specifically, the totally parallel-prefix adders of [2] require from 21% up to 95% more implementation area and consume from 32% up to 134% more power than the proposed adders, while the FPP adders of [3] require from 37% up to 122% more implementation area and consume from 47% up to 157% more power. As a result, the proposed adders are the most efficient of all examined architectures when the area $\times$ delay^2 ($A \times T^2$) is considered (normalized values are listed in each case of Table I with respect to the best offered). Under this metric, the proposed adders are also more efficient than the LF and the KS adders of [1] by 29% up to 54% and by 27% up to 132%, respectively. They also outperform the adders of [2] by up to 92%, the adders of [4] by up to 32% and the FPP and RAPP proposals of [3] by up to 102% and 93%, respectively.

REFERENCES

[1] R. Zimmerman, "Efficient VLSI Implementation of Modulo $(2^n \pm 1)$ Addition and Multiplication," in *Proc. of the* 14^{th} *IEEE Symposium on Computer Arithmetic*, April 1999, pp. 158–167.

[2] 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, December 2002.

[3] H. T. Vergos and C. Efstathiou, "Efficient Modulo 2^n+1 Adder Architectures," *Integration, the VLSI Journal*, vol. 42, no. 2, pp. 149–157, February 2009.

[4] 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, November 2003.