Efficient Implementations of Radix-4 Parallel-Prefix Trees

Stefania Perri, Pasquale Corsonello
Department of Electronics, Computer Science and Systems
DEIS-UNICAL
perri@deis.unical.it; p.corsonello@unical.it

Abstract—**This paper presents novel dynamic circuits purpose-designed to realize parallel-prefix adder trees with computational delay and power consumption lower than the conventional domino logic implementations. The proposed circuits increase speed by reducing the complexity of the pull-down networks of each dynamic gate; and save power by reducing the number of dynamic stages within the overall structure of the generic parallel-prefix tree. When the ST 45nm 1V CMOS technology is used, 32-bit radix-4 Brent-Kung, Han-Carlson and Ladner-Fischer trees designed as proposed here achieve, respectively, a computational delay of 148ps, 129.6ps and 117.2ps; dissipate 194fJ, 240fJ and 209fJ; and shows a silicon area requirement of 160um^2, 190um^2 and 170um^2.**

Keywords- Adders; VLSI circuits; parallel-prefix trees.

I. Introduction

Addition is the basic operation of any digital system. Therefore, the design of high-speed, low-power and area efficient binary adders always receives a great deal of attention. Among the hundreds adder architectures known in the literature, when high performances are mandatory, parallel-prefix trees are generally preferable [1-12].

Optimizing a parallel-prefix tree architecture and its transistor-level implementation for a specific design is not trivial since the designer has to choose: i) the radix-of the carry tree (i.e. the number of carries grouped in each step of the computation); ii) the tree architecture; iii) the logic style. As is well known, all these choices are crucial for both speed and power. In fact, higher radices determine a lower number of stages needed in the tree to compute the output carry signals, but they require more complex gates. Furthermore, at a given radix r, dense architectures, such as the Kogge-Stone tree [13], reach the minimum logic depth, but they require a large number of gates and consume a large amount of power. On the contrary, sparse trees, like the Brent-Kung [14], the Han-Carlson [15] and the Ladner-Fischer [16], do not assure obtaining the minimum logic depth, but they save hardware resources and power. Last but not least, logic style significantly affects delay and energy. As shown in [5-7, 12], parallel-prefix trees realized using dynamic domino logic achieve higher speed performances at the expense of consumed energy; whereas, using static logics lowers power consumption, but sacrifices computational speed.

This paper proposes a novel approach to optimize the implementation of the basic logic modules, namely the preprocessing stage and the associative dot operator, typically used within parallel-prefix adders. The basic idea exploited in the proposed designs consists of: 1) increasing speed by reducing the complexity of the pull-down networks (PDNs) of each dynamic gate; and 2) saving power by reducing the number of dynamic stages within the overall structure of the generic parallel-prefix tree.

Further advantages are taken by using the compound domino logic (CDL) [17]. The latter was used as an efficient alternative to the purely dynamic and static logic design styles also in [1], [5], [6], and [12]. The CDL replaces the inverter stages used in common domino circuits to invert the precharged nodes with more complex inverting static CMOS gates.

Purpose-designed CDL gates are proposed to realize efficient radix-4 parallel-prefix adder trees. Gates designed with the approach proposed here are implemented using the ST 45nm 1V CMOS technology.

The novel circuits were used to implement 32-bit parallel-prefix trees based on the Brent-Kung, Han-Carlson and Ladner-Fischer architectures. Comparison with conventional domino counterparts demonstrate that, due to the innovations introduced, up to ~40% lower computational delay is achieved with up to ~44.7% lower energy consumption and up to ~44.8% lower silicon area requirement.

The paper is organized as follows: in Section 2, a brief background on the parallel-prefix adder trees is provided and conventional domino gates implementations are also shown; the novel circuits are then described in Section 3 where comparison results are also presented and discussed; finally, conclusions are drawn.

II. Background

Let us consider two n-bit addends $A = a_{n-1} \dots a_0$ and $B = b_{n-1} \dots b_0$. A parallel-prefix adder computes the sum $S = A + B = s_{n-1} \dots s_0$ through the following three steps: i) the preprocessing stage computes the auxiliary signals propagate and generate; ii) the carry propagation stage groups the propagate and generate signals r by r, with r being the radix of the adder; iii) the produced carries are then used by the final stage to calculate the sum bit s_i, with $i=n-1,\dots,0$.

In Fig.1, examples of 32-bit radix-4 parallel-prefix trees are depicted. By observing these examples and others numerous trees known in the literature, it can be easily seen that the basic modules needed to design a parallel-prefix tree are those indicated as Group1, Group2, Group3 and Group4.

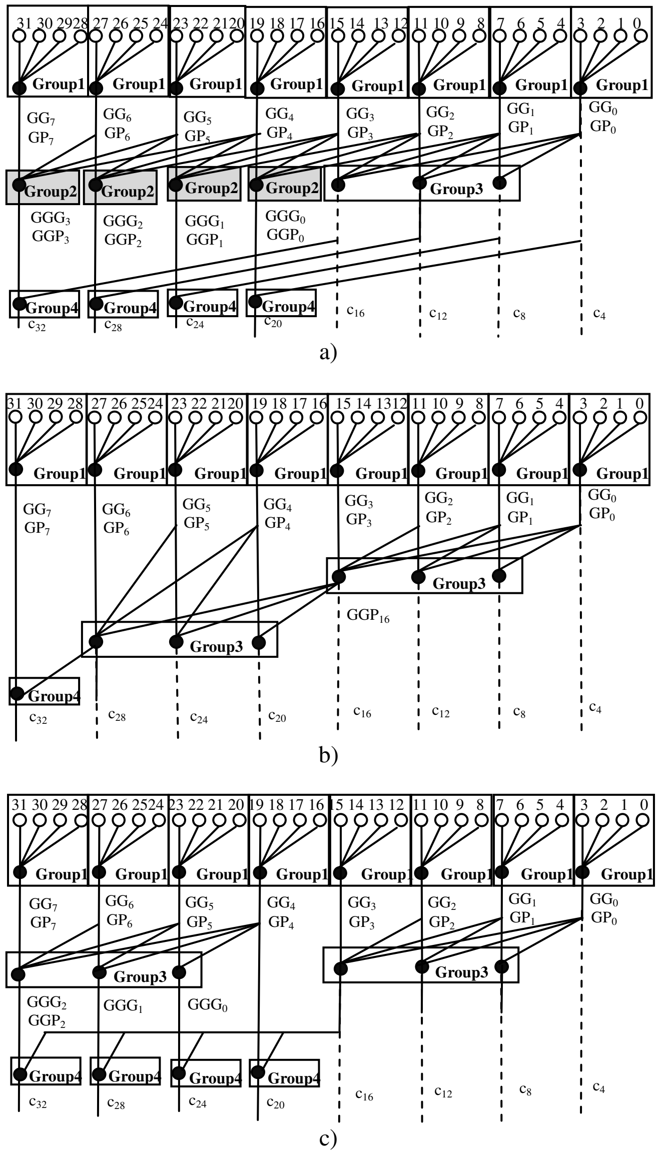

Figure 1. Examples of parallel prefix adders: (a) Han-Carlson; (b) Brent-Kung; (c) Ladner-Fischer.

Their conventional domino logic implementations are shown in Fig.2 that also reports transistor widths referred to the ST 45nm 1V CMOS technology. They were obtained considering that 0.12um is the minimum transistor width allowed by the technology used. The minimum size criterion was applied to the elementary 2-input OR and AND gates that are not shown in the Figure, whereas the progressive transistor sizing with a 1.5 tapering factor was exploited within the Manchester carry-chains and gates with higher fan-in. All the inverters on the dynamic nodes are minimum sized with an aspect ratio of 4/3.

The module Group1 of Fig.2a preliminary computes propagate and generate signals at the *i*-th bit position as shown in (1), where *i=0,..., n-1*.

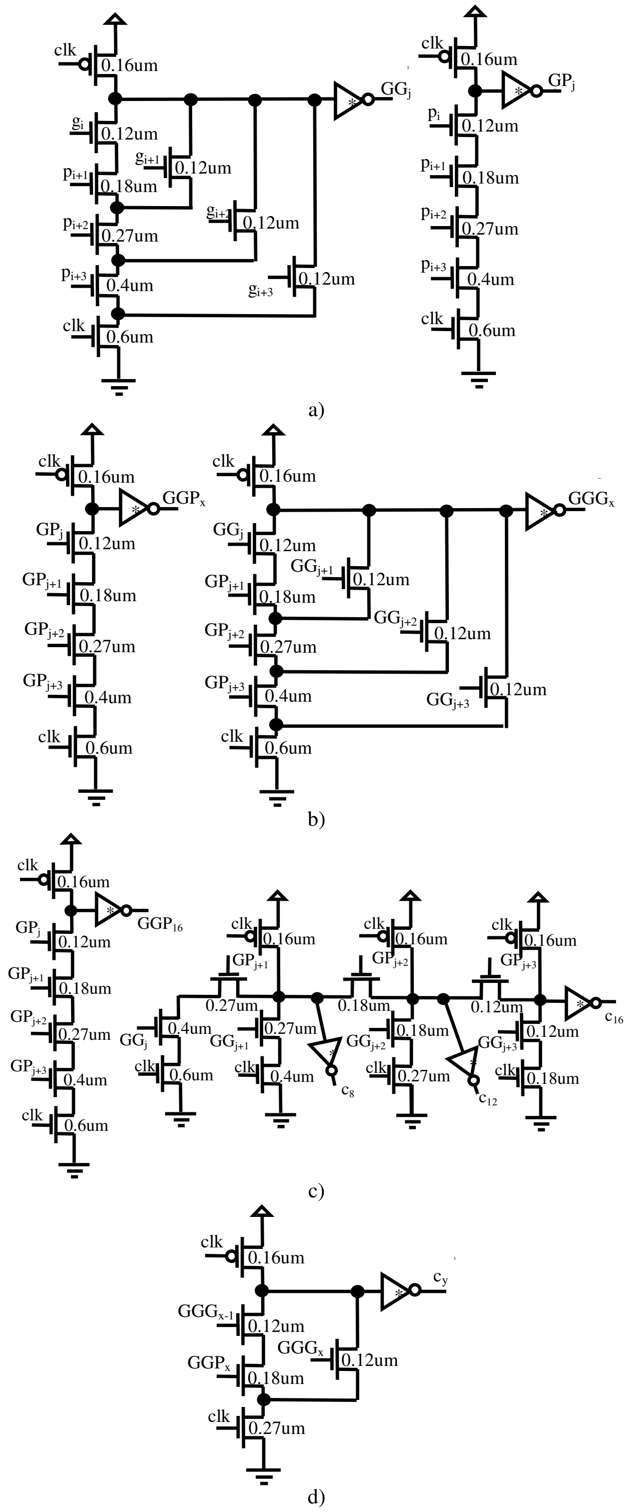

Figure 2. Conventional implementations of the modules: (a) Group1; (b) Group2; (c) Group3; (d) Group4.

$$p_i = a_i + b_i \qquad (1)$$
$$g_i = a_i \cdot b_i$$

Propagate and generate signals are then grouped four by four implementing the classical carry-look-ahead equations reported in (2), where $j = \frac{i}{4}$, and $j=0,...,7$.

$$GP_j = p_{i+3} \cdot p_{i+2} \cdot p_{i+1} \cdot p_i$$
$$GG_j = g_{i+3} + p_{i+3} \cdot g_{i+2} + p_{i+3} \cdot p_{i+2} \cdot g_{i+1} + \qquad (2)$$
$$+ p_{i+3} \cdot p_{i+2} \cdot p_{i+1} \cdot g_i$$

The module Group2 is used to implement equations (3), where $x=j-1$, and $x=0,...,3$ for the tree in Fig.1a.

$$GGP_x = GP_{j+3} \cdot GP_{j+2} \cdot GP_{j+1} \cdot GP_j$$
$$GGG_x = GG_{j+3} + GP_{j+3} \cdot GG_{j+2} + \qquad (3)$$
$$GP_{j+3} \cdot GP_{j+2} \cdot GG_{j+1} + GP_{j+3} \cdot GP_{j+2} \cdot GP_{j+1} \cdot GG_j$$

For all the referred tree architectures, the carry signals c_8, c_{12} and c_{16} are computed by the module Group3 that, as illustrated in Fig.2c, also provides the grouped propagate signal GGP_{16}. The same module is used in the Ladner-Fischer sparse tree of Fig.1c also to compute the grouped generate signals GGG_2, GGG_1, GGG_0 and the grouped propagate signal GGP_2, and in the Brent-Kung architecture of Fig.1b to compute the carry signals c_{20}, c_{24} and c_{28}.

Some of the final carries of the referred 32-bit trees are computed further grouping the signals GGG_x and GGP_x two by two following the classical carry-look-ahead logic shown in (4) and implemented by the module Group4 depicted in Fig.2d.

$$c_y = GGG_x + GGP_x \cdot GGG_{x-1} \qquad (4)$$

Equations (1)-(3) are specialized for 32-bit radix-4 trees. However, they can be easily extended to different wordlengths and radices.

III. THE NOVEL CIRCUITS

This Section proposes novel transistor-level implementations of the basic modules used within parallel-prefix trees. The novel circuits are purpose-designed to increase speed performance and to reduce energy consumption with respect to their conventional domino logic counterparts. The main innovations here introduced consist in: i) reducing the number of dynamic nodes within each module with the objective of reducing the power consumption and ii) simplifying the pull-down networks (PDNs) of each dynamic stage to reduce the computational delay.

The number of dynamic nodes within each module is reduced mainly by avoiding the computation of useless intermediate signals. To better explain how this is possible let us examine the module Group1 implemented with the conventional domino logic as illustrated above in Fig.2a. It is easy to verify that the computation of the generic GG_j and GP_j signals involves ten dynamic nodes: four belong to the 2-input OR gates computing the propagate signals p_i, p_{i+1}, p_{i+2} and p_{i+3}; further four dynamic nodes belong to the 2-input AND gates producing the generate signals g_i, g_{i+1}, g_{i+2} and g_{i+3}; finally, two dynamic nodes are used within the gates required to group 4 by 4 the propagate and generate signals.

The novel module Group1_new was designed to produce only one propagate and one generate signal by each two bit positions. As visible from the transistor-level implementation illustrated in Fig.3, in this way only two propagate intermediate signals, $\overline{P_{32\_j}}$ and $\overline{P_{10\_j}}$, and two generate intermediate signals, $\overline{G_{32\_j}}$ and $\overline{G_{10\_j}}$, are required, thus reducing the overall number of dynamic nodes involved in the computation of the signals GG_j and GP_j to four. It can be observed that, in order to simplify the PDNs of dynamic stages, the CDL style is exploited and static CMOS inverters used in conventional dynamic circuits to achieve the domino behavior are replaced with more complex static CMOS gates.

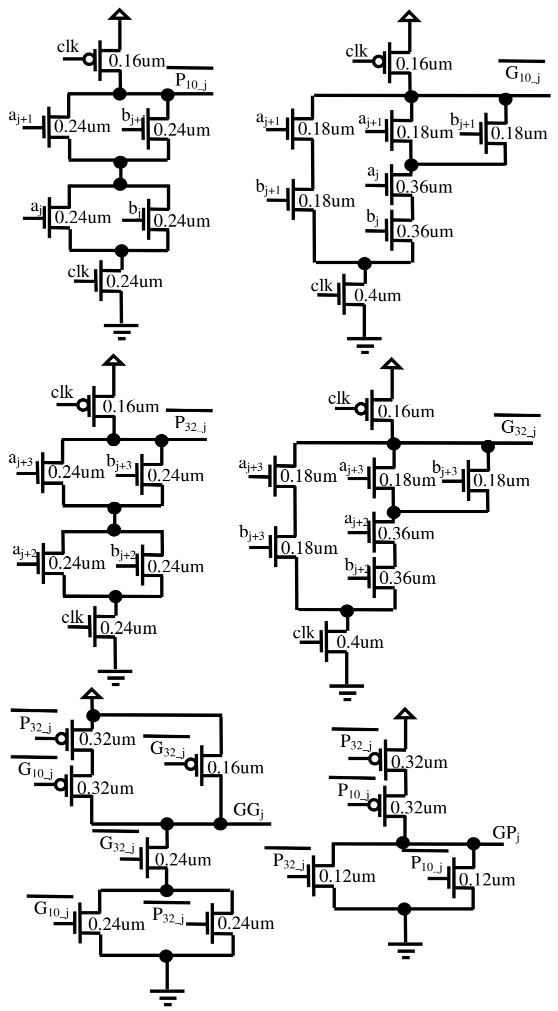

Figure 3. The Group1_new module.

The basic gates used in the novel modules Group2_new and Group3_new are illustrated in Fig.4. There, GGG_x and GGP_x correspond to c_{16} and GGP_{16} in Group3_new. The latter also uses the gates enclosed in the dashed box to compute the carry signals c_8 and c_{12}. The approach used to design these gates does not reduce the number of dynamic nodes with respect to the conventional implementations, but it allows the PDNs of dynamic stages to be simplified. In fact, from Fig.4 it can be seen that the PDNs of dynamic stages inside the modules Group2_new and Group3_new contain no more than three series transistors, whereas the conventional modules Group2 and Group3 of Figs.2b and 2c use PDNs with up to five series transistors.

Due to its simplicity, for the module Group4 depicted in Fig.2d a CDL implementation does not make sense.

Finally, it is worth emphasizing that the transistors of the novel circuits are sized ensuring that the input capacitances of a parallel-prefix tree designed as proposed here are mainly unchanged with respect to the conventional implementation. In this way, front-end modules providing the operands *A* and *B*, are not influenced by the adopted innovations.

A. Parallel-prefix trees implementation and comparison results

The novel modules described above were exploited to realize the 32-bit radix-4 architectures depicted in Fig.1. In the following, the Han-Carlson, Brent-Kung and Ladner-Fischer trees realized using the novel modules are named HC_new, BK_new and LF_new, respectively.

For purposes of comparison, conventional domino logic implementations of the referenced trees (in the following named HC_conv, BK_conv and LF_conv) were also carried out and they were compared to the novel implementations in terms of worst-case delay T_W and energy consumption. Pre-layout Corner Analysis was performed loading each carry output signal with a 1fF capacitance. The latter was chosen referring to the input capacitance of a positive edge-triggered D flip-flop with 9x drive strength available within the standard cells library of the used 45nm technology.

TABLE I. PRE-LAYOUT SIMULATION RESULTS

Tree	T_W [ps]			Energy [pJ]		
	TT	FF	SS	TT	FF	SS
HC_conv	126.4	100.3	158.5	0.296	0.332	0.282
HC_new	115	89.7	145.7	0.214	0.255	0.192
BK_conv	179	142.1	225	0.262	0.295	0.249
BK_new	134.6	104.6	170.8	0.185	0.223	0.165
LF_conv	129	101.8	161.4	0.27	0.303	0.257
LF_new	103.3	80.2	130.9	0.193	0.231	0.173

TABLE II. POST-LAYOUT SIMULATION RESULTS

Tree	T_W [ps]			Energy [pJ]			Area
	TT	FF	SS	TT	FF	SS	[um^2]
HC_conv	175.7	138.3	223.8	0.392	0.416	0.381	320
HC_new	129.6	101	164	0.24	0.29	0.231	190
BK_conv	248	196.6	313.5	0.351	0.373	0.341	290
BK_new	148	115	188.5	0.194	0.221	0.186	160
LF_conv	175.4	139.2	222.3	0.36	0.383	0.351	306
LF_new	117.2	91.5	148.5	0.209	0.248	0.2	170

Simulation results reported in Table I demonstrate that, for all the examined trees, the novel circuits lead to consistent speed improvement and energy reduction.

All the above compared parallel-prefix trees were laid out using the full-custom layout approach. Results obtained through the post-layout Corner Analysis are reported in Table II, which demonstrates how the advantages introduced by the novel circuits are maintained also in the laid out trees. As an example, the BK_new tree exhibits computational delay, energy consumption and silicon area occupancy ~40%, ~44.7% and ~44.8% lower than the conventional domino implementation BK_conv. Similar improvements are achieved also for the Han-Carlson and Ladner-Fischer parallel-prefix trees, thus demonstrating that the proposed circuits are advantageous in several parallel-prefix architectures.

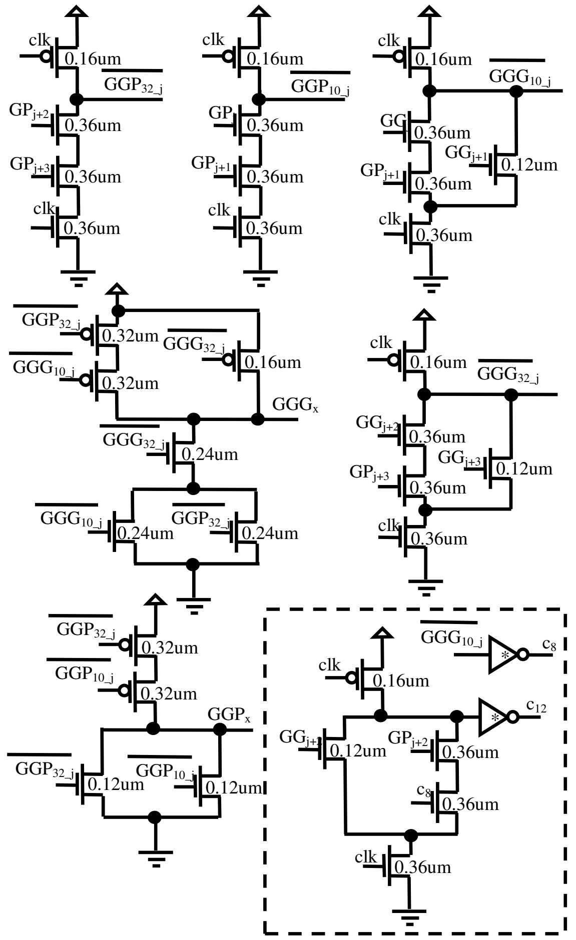

Figure 4. The novel modules Group2_new and Group3_new.

IV. CONCLUSIONS

A novel design approach was presented to implement efficient sparse parallel-prefix adder trees using nanometer technologies. The basic idea exploited in the proposed designs consists of: 1) reducing the complexity of the pull-down networks (PDNs) of each dynamic gate; and 2) minimizing the number of dynamic nodes within the overall structure of the generic parallel-prefix tree. The innovations here introduced allow reducing both the computational time and the average power consumption with respect to conventional domino logic implementations.

As an example, a 32-bit radix-4 Brent-Kung tree designed as proposed here achieves a computational delay of only 148ps, dissipates just 194fJ and occupies a 160um^2 silicon area, that are ~40%, ~44.7% and ~44.8% lower than the conventional domino implementation.

REFERENCES

[1] J.Park, H.C.Ngo, J.A.Silberman, S.H.Dhong, “470ps 64bit Parallel Binary Adder”, *Proc. IEEE Symposium on VLSI Circuits*, June 2000, Honolulu, Hawaii, pp.192-193.

[2] P.Corsonello, S.Perri, M.Margala, “Efficient Addition Circuits for Modular Design of Processors-in-Memory”, *IEEE Transactions on Circuits and Systems I: Regular Papers*, Vol.52, n°8, pp.1557-1567, 2005.

[3] F.Frustaci, M.Lanuzza, P.Zicari, S.Perri, P.Corsonello, “Designing High-Speed Adders in Power-Constrained Environments”, *IEEE Transactions on Circuits and Systems II: Express Briefs*, Vol.56, n°2, pp.172-176, 2009.

[4] S.Das, S.P.Khatri, “A Novel Hybrid Parallel-Prefix Adder Architecture With Efficient Timing-Area Characteristic”, *IEEE Transactions on VLSI Systems*, Vol.16, n°3, pp.326-331, 2008.

[5] V.G.Oklobdzija, B.R.Zeydel, H.Q.Dao, S.Mathew, R.Krishnamurthy, “Comparison of High-Performance VLSI Adders in the Energy-Delay Space”, *IEEE Transactions on VLSI Systems*, Vol.13, n°6, pp.754-758, 2005.

[6] R.Zlatanovici, S.Kao, B.Nikolic, “Energy-Delay Optimization of 64-bit Carry-Lookahead Adders with a 240ps 90nm CMOS Design Example”, *IEEE Journal of Solid-State Circuits*, Vol.44, n°2, pp.569-583, 2009.

[7] J.Grad, J.E.Stine, “Low Power Binary Addition Using Carry Increment Adders”, *Proc. of the IEEE International Symposium on Circuits and Systems*, May 2006, Island of Kos, Greece, pp.17-20.

[8] G.Dimitrakopoulos, D.Nikolos, “High-Speed Parallel-Prefix VLSI Ling Adders”, *IEEE Transactions on Computers*, vol.54, n°2, pp.225-231,2005.

[9] F.Liu, F.F.Forouzandeh, O.A.Mohamed, G.Chen, X.Song, Q.Tan, “A Comparative Study of Parallel Prefix Adders in FPGA Implementation of EAC”, *Proc. of the EUROMICRO Conference on Digital System Design, Architetcures, Methods and Tools*, August 2009, Patras, Greece, pp.281-286.

[10] S.Das, S.P.Khatri, “A Novel Hybrid Paralle-Prefix Adder Architecture With Efficient Timing-Area Characteristic”, *IEEE Transactions on VLSI Systems*, vol.16, n°3, pp.326-331, 2008.

[11] S.Ghosh, K.Roy, “Novel Low Overhead Post-Silicon Self-Correction Technique for Parallel Prefix Adders Using Selective Redundancy and Adaptive Clocking”, *IEEE Transactions on VLSI Systems,* Forthcoming.

[12] B.R.Zeydel, D.Baran, V.G.Oklobdzija, “Energy-Efficient Design Methodologies: High-Performance VLSI Adders”, *IEEE Journal of Solid-State Circuits*, vol.45, n°6, pp.1220-1233, 2010.

[13] P.M.Kogge, H.S.Stone, “A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations”, *IEEE Transactions on Computers*, Vol.C-22, n°8, pp.786-793, 1973.

[14] R.P.Brent, H.T.Kung, “A regular layout for parallel adders”, *IEEE Transactions on Computers*, vol.C-31, n°3, pp.260-264, 1982.

[15] T.Han, D.A.Carlson, “Fast area-efficient VLSI adders”, *Proc. IEEE Symposium on Computer Arithmetic*, May 1987, Como, Italy, pp.49-56.

[16] R.E.Ladner, M.J.Fischer, “Parallel Prefix Computation”, *Journal of the ACM*, Vol.27, n°4, pp.831-838, 1980.

[17] K.Bernstein, K.M.Carrig, C.M.Durham, P.R.Hansen, D. Hogenmiller, E.J.Nowak, N.R.Rohrer, *High-Speed CMOS Design Styles*, Kluwer Academic, 1998.