A Low-Cost Fault-Tolerant Technique for Carry Look-Ahead Adder

Alireza Namazi, Yasser Sedaghat, Seyed Ghassem Miremadi, Alireza Ejlali
Dependable Systems Laboratory
Department of Computer Engineering
Sharif University of Technology
Azadi St., Tehran, IRAN
E-mail: {namazi,y_sedaghat}@ce.sharif.edu, {miremadi,ejlali}@sharif.edu

***Abstract*— This paper proposes a low-cost fault-tolerant Carry Look-Ahead (CLA) adder which consumes much less power and area overheads in comparison with other fault-tolerant CLA adders. Analytical and experimental results show that this adder corrects all single-bit and multiple-bit transient faults. The Power-Delay Product (PDP) and area overheads of this technique are decreased at least 82% and 71%, respectively, as compared to adders which use traditional TMR, parity prediction, and duplication techniques.**

***Index Terms*—Carry Look-Ahead Adder, Fault Tolerance, Single -Event Transient**

I. Introduction

Embedded processors are widely used in various safety-critical systems [1], such as X-by-wire applications, in which failures could endanger human life or property [2]. These processors and VLSI circuits are very susceptible to transient faults [3] which are mostly caused by alpha particles and neutrons [4]. Particle strikes in a sensitive region of combinational circuits may lead to Single-Event Transient (SET) [5] or Multi-bit Event Transient (MET) [6, 9, 19] errors. Occurrence probabilities of these errors are very high in modern processors. On-line error detection techniques are widely used to detect transient faults in VLSI circuits [7].

Adders are the essential part of data processing systems in safety-critical applications [11, 12, 13, 15]. They are commonly found in the critical path of arithmetic and logical units (ALUs) and address generation units. Therefore, the design of adder structures with on-line error detection and correction capabilities is an important research topic [3, 9].

The Carry Look-Ahead (CLA) adder is one of the high speed adders among other adders like Carry Select Adder (CSA), carry skip adder and some implementations of the parallel-prefix adders [12, 14, 15, 16].

So far, few error detection techniques have been especially proposed for the CLA adder. Existing techniques usually use the parity prediction technique in combination with a two-rail code [17, 18]. In [8], the two-rail code is used for carries and the parity prediction is used for the outputs. Both the parity prediction and checkers of two-rail codes decrease the performance drastically, because both of them need XOR-tree structures. In [17], the parity code is used to detect input operands error and the two-rail code is used for the output error detection. In addition to these techniques, duplication with comparison [17] can also be used. This technique has about 100% area and power overheads. All of the mentioned techniques only detect errors and simply use re-execution, the most popular and simplest correction technique. The re-execution cannot correct permanent faults, and it increases the delay about two times. Furthermore, many single-bit errors cannot be detected in CLA adders, which are checked by arithmetic codes [16].

An alternative to re-execution [10] is TMR technique which is an almost traditional technique to cope with error occurrence in all circuits as well as the adders. This technique has high power consumption and area overheads; thus, a fault-tolerant adder is achieved at the cost of highly increasing the PDP and the area.

In this paper, a low-cost fault tolerance technique for CLA adders is proposed. The proposed technique detects or corrects both permanent and transient single-bit errors. The CLA adder protected by the proposed technique is compared with the conventional Triple Modular Redundancy (TMR) and time redundancy techniques as correction techniques and with the parity-prediction and duplication as detection techniques. The Power-Delay Product (PDP) and area overhead of the proposed technique are compared with efficient error detection or correction techniques. The proposed technique has 82% and 71% improvement in PDP and area overheads, respectively.

Organization of this paper is as follows. In section 2, the basic CLA adder is introduced. In section 3, the new fault-tolerant technique is proposed. Analysis and experimental results are presented in section 3 and 4, respectively. Section 5 concludes this paper.

II. The Basic CLA Adder

The main idea behind the CLA adder is an attempt to generate all incoming carries in parallel; therefore, it increases the performance. The CLA adders consist of three components: P/G generator, CLAU, and Sum generator (Fig. 1).

The First unit, P/G generator, generates P_i and G_i simultaneously using relations (1) and (2).

$$P_i = a_i \oplus b_i \,, 0 \le i \le n \quad (1)$$

$$G_i = a_i b_i \,, 0 \le i \le n \quad (2)$$

978-1-4244-4595-0/09/$25.00 ©

$$C_i = \begin{cases} C_{in} & ; i = 0 \\ G_{i-1} + \sum_{j=0}^{i-2}\left(\prod_{k=j+1}^{i-1} P_k\right)G_j + \left(\prod_{k=1}^{i-1} P_{k-1}\right)C_{in} & ; \begin{cases} i = 1 \\ : \\ i = n-1 \end{cases} \\ C_{out} & ; i = n \end{cases} \quad (3)$$

$$S_i = P_i \oplus C_i \quad ; 0 \le i \le n-1 \quad (4)$$

The Second unit, Carry Look-Ahead Unit (CLAU), generates carries in a parallel manner using Relation (3). The Third unit, Sum generator, calculates the sum products of the addition by exploiting Relation (4).

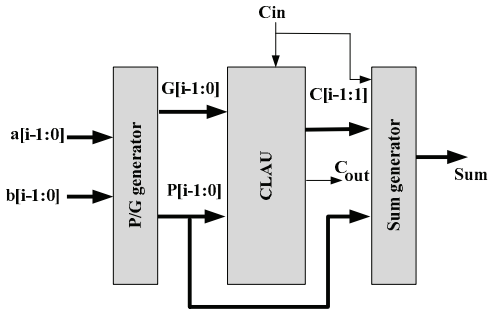

Fig. 1. General View Of the basic CLA adder

Considering Figure 1, delay estimation of the CLA adder is derived from (5). In (5), Δ_T denotes the total delay of the CLA adder, $\Delta_{P/G}$ denotes the delay of the P/G generator unit, Δ_{CLA} is the delay of CLAU and Δ_{sum} is the delay of sum generator unit.

$$\Delta_T = \Delta_{P/G} + \Delta_{CLA} + \Delta_{sum} \quad (5)$$

Values of $\Delta_{P/G} = \max\{\Delta_{xor}, \Delta_{and}\}$ and $\Delta_{sum} = \Delta_{xor}$ are identical in any n-bit CLA adder. Relation (3) shows that increasing bit-widths of the input operands does not affect the delay of the CLAU; because this delay completely depends on its implementation. Relation (3) shows that each carry can be implemented by two-level logics. This means the delay of each carry will be identical is equal to $\Delta_{CLA} = \Delta_{and} + \Delta_{or}$. High fan-in is the major drawback of this type of implementation. As a result, the designer uses hierarchical implementation for the CLAU to cope with high fan-in issue of the one-level logic implementation.

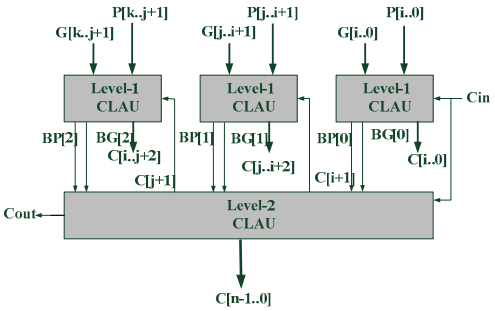

Fig. 2. Two-level hierarchical CLA schema

Hierarchical implementation results in increasing the delay and area of the CLA adder. The CLAU designed using the two-level hierarchical implementation is shown in Fig. 2. Both L1 and L2 CLAUs generate carries based on (3). Two extra signals which are called Block-Generate (BG) and Block-Propagate (BP) are generated in L1 CLAUs. The BP_i and BG_i signals can be derived from (6) and (7). In (6) and (7), Mn and Mx denote the LSB and MSB of the Ps and Gs entering the *i*th unit, respectively.

$$BP_i = \prod_{i=Mx}^{Mn} P_i \quad (6)$$

$$BG_i = G_{Mn} + \sum_{k=Mx}^{Mn-1} G_k \prod_{h=k+1}^{Mn} P_h \quad (7)$$

III. The Low-Cost Technique for Error Detection or Correction in the CLA

The major area of the basic CLA adder is occupied by the CLAU. Therefore, using fault-tolerant techniques without considering the importance of the CLAU, results in high overheads. This paper proposes a new fault-tolerant technique which concentrates on the CLAU (both one-level or hierarchical implementation of the CLAU).

The proposed method modifies Relation (3) to generate more than one adjacent carry signals simultaneously. For example, using Relation (8) results in one preceding carry signal to be generated in each carry generator logics. In (8), C_{i-1} is constructed based on (3).

$$C_i = G_{i-1} + P_{i-1}C_{i-1} = \quad (8)$$
$$\begin{cases} C_{in} & ; i = 0 \\ G_0 + P_0 C_{in} & ; i = 1 \\ G_{i-1} + P_{i-1}\left[\sum_{j=0}^{i-2}\left(\prod_{k=j+1}^{i-1} P_k\right)G_j + \left(\prod_{k=1}^{i-1} P_{k-1}\right)C_{in}\right] & ; \begin{cases} i = 2 \\ : \\ i = n-1 \end{cases} \\ C_{out} & ; i = n \end{cases}$$

Fig. 3 shows the structure of the modified carry generator logic, based on (8), which generates one preceding carry signal. This modification results in two gate-levels increase in the carry generator logic. This technique is called Low-Cost Error Detection (LCED).

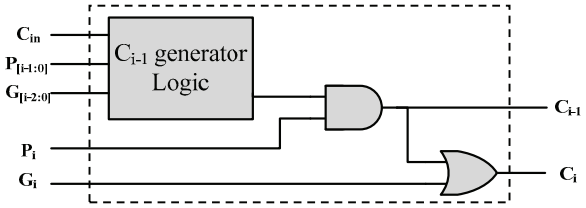

Fig. 3. LCED carry generator logic to obtain one extra carry simultaneously

One redundant carry signal only provides the error detection ability, but more than one redundant carries provide the ability of error correction, which is called Low-Cost Error Correction (LCEC). For example, in (10) two redundant carries are generated; the modified carry generator logic is shown in Fig. 4.

$$C_i = G_{i-1} + P_{i-1}G_{i-2} + P_{i-1}P_{i-2}C_{i-2} = \quad (9)$$

$$= G_{i-1} + P_{i-1}(G_{i-1} + P_{i-2}C_{i-2}) = \quad (10)$$

$$\begin{cases} C_{in} & ;i=0 \\ G_0 + p_0 C_{in} & ;i=1 \\ G_1 + G_0 P_1 + P_1 P_0 C_{in} & ;i=2 \\ G_{i-1} + P_{i-1}\left[G_{i-1} + P_{i-2}\left[\sum_{j=0}^{i-2}\left(\prod_{k=j+1}^{i-1} P_k\right)G_j + \left(\prod_{k=1}^{i-1} P_{k-1}\right)C_{in}\right]\right]; \begin{cases} i=3 \\ : \\ i=n-1 \end{cases} \\ C_{out} & ;i=n \end{cases}$$

Generation of two redundant carries results in single-bit error correction. If three or more redundant carries are extracted, multi-bit errors can be corrected.

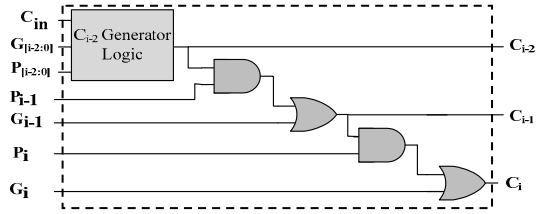

Fig. 4. LCEC carry generator logic to obtain two extra carries simultaneously

Relation (11) shows the gate-level increase in the CLAU; in this relation, D shows number of increased gate-levels, N_{ec} denotes number of extra carries which are extracted, and L is the hierarchical depth of the CLAU.

$$D = 2 * L * N_{ec} \quad (11)$$

$$\Delta_{HCLA} = (2L-1)(\Delta_{and} + \Delta_{or} + D) \quad (12)$$

Using (11), one redundant carry results in eight gate-levels increase in a two-level hierarchical implementation of the CLAU. Increasing gate-levels increases the delay of the CLA adder according to Relation (12). The Δ_{HCLA} denotes the delay of the hierarchical CLAU; L shows hierarchy levels of the CLAU, Δ_{or} and Δ_{and} are delays of OR-gate and AND-gate, respectively.

Till now main concepts of the LCED and LCEC techniques have been described. Although CLAU needs more attention than other units of a CLA adder and affects of other units in the reliability of the CLA adder cannot be ignored. Therefore, all units must be considered in our design. Fig. 5 shows the structure of the CLA adder with the Low-Cost Error Detection (CLA-LCED) technique. It uses the LCED technique in its CLAU and duplication in Sum and P/G generator units which is inevitable and is discussed later in Section IV.

In Fig. 5, $C_1[n]$ and $C_0[n]$ denote duplicates of C_{out} of CLA-LCED adder. Both hierarchical and non-hierarchical implementation of CLAU has effects on LCED technique. Fig. 6 shows both hierarchical and non-hierarchical implementations of CLAU with LCED technique. Fig. 6.a depicts a two-level hierarchical LCED-CLAU. Internal structure of level-1 LCED-CLAU is depicted in Fig. 6.a. Structure of level-2 LCED-CLAU is analogous to the structure shown in Fig. 6.b.

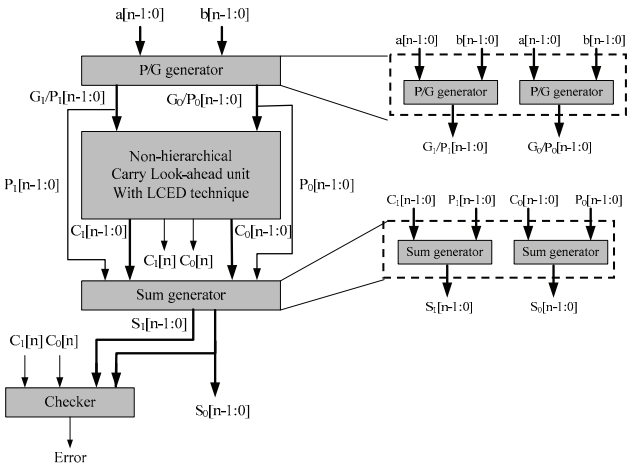

Fig. 5. Schematic view of the LCED-CLA adder

Fig. 6.b shows the non-hierarchical implementation of CLAU with LCED technique. In this figure $C_1[n]$ and $C_0[n]$ denote duplicates of C_{out}. As it is shown in Fig. 6.b, C[1] carry generator logic remains unchanged, and it just produces C[1]. Based on (9), all carry generator logics, except for the C[1] and $C_1[n]$, have increased two gate-levels. In Fig. 6, in C[j] carry generator logic parameter i= j mod 2. In the hierarchical implementation, there are two types of generator logics: 1) units with BP/BG signals and 2) units without BP/BG signals.

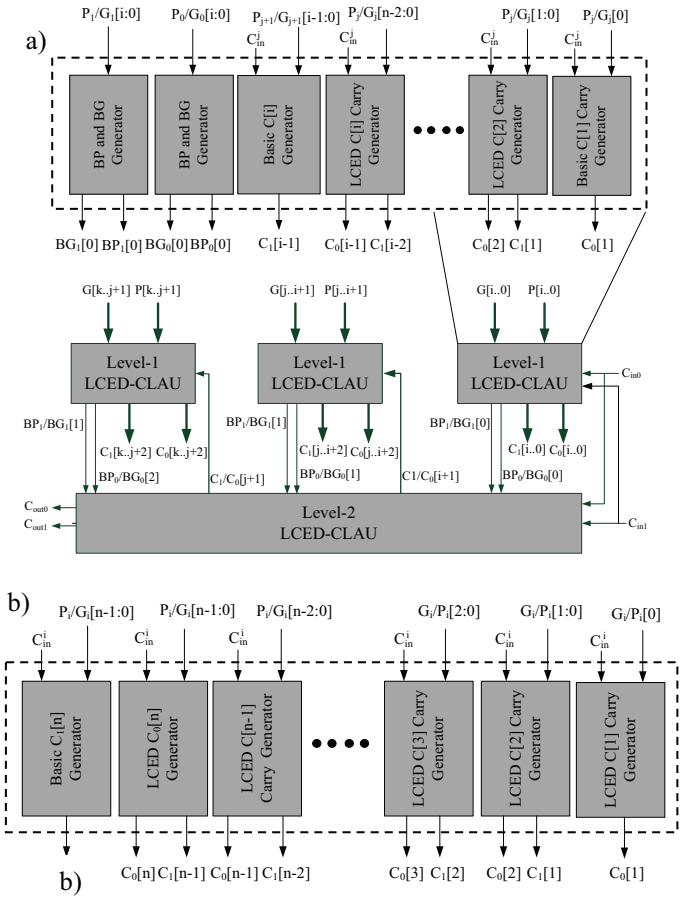

Fig. 6. Schematic view of the CLAU of the CLA-LCED with both a) hierarchical and b) non-hierarchical structure

As fault detection is not sufficient to have a reliable design, it should be followed by a correction mechanism. In general, re-execution is one of the best choices for coping with transient errors. Although re-execution may be useful for transient errors, it is not efficient in front of the permanent errors. Another design method for the fault-tolerant circuits is to apply an error correction technique to them in order to recover them from errors, i.e. TMR.

TMR is commonly used to mask errors which are occurred in the circuits. TMR can mask all single-bit transient and permanent faults. The LCEC technique which has been explained in the above is a low-cost technique which can be used in CLAU. Redundant Sum and P/G generator units have to be used in order to have a fault secure CLA adder with error correction characteristic, like CLA-LCED. The proposed structure is called Carry Look-Ahead adder with Low-cost Error Correction (CLA-LCEC).

Error correction ability in LCEC technique is achieved by setting N_{ec} greater or equal to two. Single-bit error correction ability is achieved by N_{ec}= 2. Considering N_{ec}= 2 results in four gate-levels increase in carry generator logics. Complete structure of the CLA-LCEC with N_{ec}=2 is depicted in Fig. 7.

As shown in Fig. 7 and discussed in Section IV, the P/G and Sum generator units are triplicate units. In addition, in Fig. 7, CLAU is modified by LCEC technique. The CLA-LCEC may seem to be similar with the CLA-LCED, but they differ slightly in their structures, i.e. the CLA-LCED has lower delay in comparison with the CLA-LCEC. Similar to the CLA-LCED, implementation of CLAU in the CLA-LCEC does not affect its reliability.

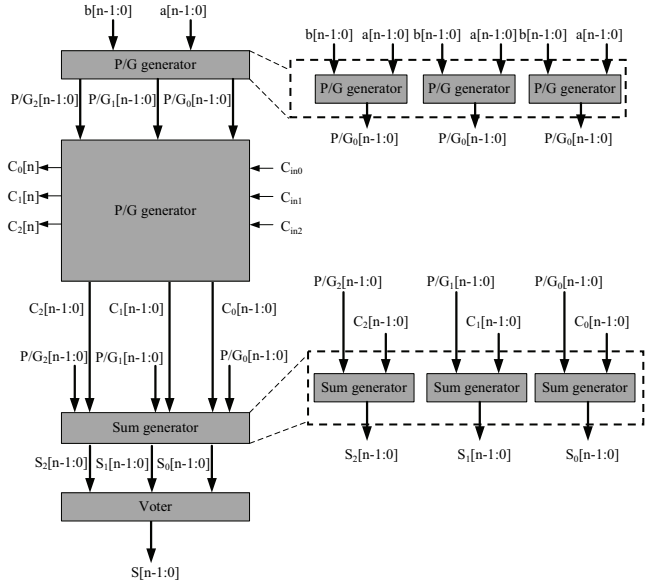

Fig. 7. The CLA-LCEC structure

Both hierarchical and non-hierarchical implementations of CLAU in the CLA_LCEC are depicted in Fig. 8. Fig. 8.a shows the hierarchical implementation of the LCEC-CLAU. It can be seen that the architecture which is proposed in Fig. 8.a is somehow the same as the architecture proposed in Fig. 6.a.

IV. Analysis and Results

In this section we are going to describe the effects of using LCED and LCED techniques on the CLA adder. The parameters which are going to be discussed are Fault tolerance, Performance, Power Consumption, and Area.

All circuits are simulated by Modelsim V6.0 and synthesized by Synopsis Design Compiler, Version X-2005.09-SP2 for Linux, and all power results are given by Synopsis Power Compiler, version X-2005.09-SP2 for Linux. 180nm technology is used for the synthesis.

Considering the architectures of the CLA-LCED, which is proposed and described in Section IV, the error detection coverage of the proposed technique is 100%. This can be proven by observing the behavior of the CLA-LCED adder in front of single-bit errors according to the following theorems:

Theorem 1, Occurrences of the single-bit errors in the P/G generator unit of the CLA- LCED adder can be detected.

Proof, Occurrence of single-bit errors in the P/G generator unit of the CLA-LCED adder results in one erroneous bit in P_0, G0, P1, or G_1signals. Considering Figures 10 and 11, it can be seen that none of these pairs are used continuously. This means that single-bit errors in P_0 or G_0 affect on LCED carry generator logics. Therefore, this error can only have an effect on C_1[i] or C_0[i], not both of them. This inequality results in unequal S_0[i] and S_1[i] or maybe C_0[n] and C_1[n]. Hence, single-bit errors occurring in the P/G generator unit of CLA-LCED can be detected.

Theorem 2, Occurrences of the single-bit errors in the CLAU of CLA-LCED adder can be detected.

Proof, Occurrence of single-bit error in this unit affects one of carry generator logics. Due to parallel behavior of carry generation in the CLA adder, in the worst case both of the produced carries of the LCED carry generator logic may be erroneous. Consider that C[i] generator logic produces erroneous results. In this situation C_0[i] and C_1[i-1] will be erroneous in the worst case. These types of error can be detected, because duplicated values of C_1[i] and C_0[i-1] are still correct. Therefore, all single-bit errors which are occurring in the CLAU can be detected.

Theorem 3, all single-bit errors in the sum generator unit of the CLA-LCED adder can be detected.

Proof, Occurrence of single-bit errors in the sum generator logic affects only one bit of result. Consider the situation that S_0[i] (S_1[i]) is affected. In this situation S_1[i] (S_0[i]) is still valid. Therefore, errors can be detected.

Theorems 1, 2 and 3 show that the CLA-LCED adder is completely fault secure and can detect any single-bit error which occurs in the circuit. These theorems include both permanent and transient errors. Behavior of this adder in the case of the multi-bit errors completely depends on where errors occur.

Using LCEC technique in the CLA adder results in 100% error correction, and it is described in Section V. Having high error detection or correction capability has indisputable effects on other parameters of the system.

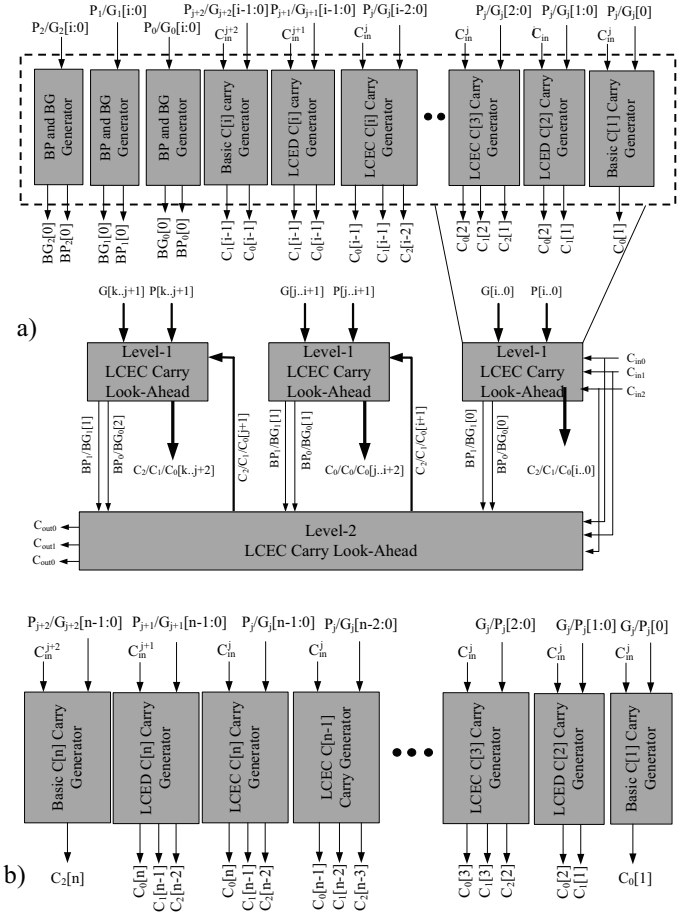

Fig. 8. Schematic view of the CLAU of the CLA-LCEC with both a) hierarchical and b) non-hierarchical structure

The effects of using LCED technique as the error detection technique in the CLA adder is compared with the parity prediction and duplication. These techniques are chosen because of two reasons; first, both of them have 100% error detection similar to the LCED technique. Second, they are more popular error detection techniques.

Fig. 9 shows the effects of applying error detection techniques on the CLA. As it can be seen, the power consumption and area overheads of the LCED technique are much less than the other error detection techniques. The results also show that average power consumption of the LCED technique is about 94.39% and 53.20% lower than the duplication and parity techniques, respectively. Notable decrease also can be seen in the area overheads of the LCED technique in comparison with duplication (91.10%) and parity (91.95%).

Fig. 9 also depicts the effects of the error detection techniques on the performance of the CLA adder. As it can be seen both duplication and LCED techniques have much less delay overheads than parity prediction technique. The duplication has lower delay overhead than LCED technique

Table 1 shows the delay overhead differences between LCED and duplication techniques. it shows that the maximum difference in delay overhead is 3.547% which belongs to the 4-bit CLA adder.

Table 1. Delay Overhead differences between LCED and duplication error detection techniques

	Bit Width				
	4	8	16	32	64
Delay Overhead Difference	3.54	2.22	0.99	0.20	0.15

Besides, increase in bit-width results in a noticeable decrease in delay overhead. Considering this, for 32-bit and 64-bit CLA adders which are mostly used in processors and DSPs, the delay overhead reduces up to 0.15% which is negligible.

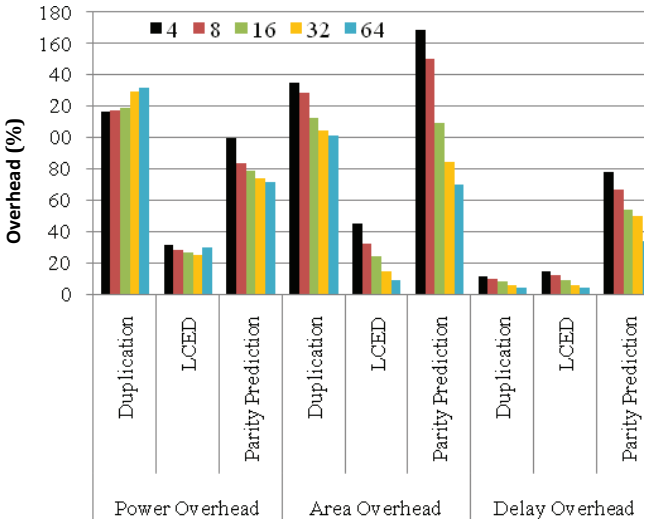

Fig. 9. Power consumption, area and delay overheads of three error detection techniques on the CLA adder

Although the delay overhead of LCED is higher than duplication (up to 3.54%), the Power-Delay Product (PDP) of the LCED is much better than the duplication and also parity prediction. Fig. 10 shows the PDP reduction of the LCED in comparison with parity prediction and duplication. This figure shows that the PDP reduction of LCED error detection technique is at least 43.48% while that of the parity prediction is 76.68%.

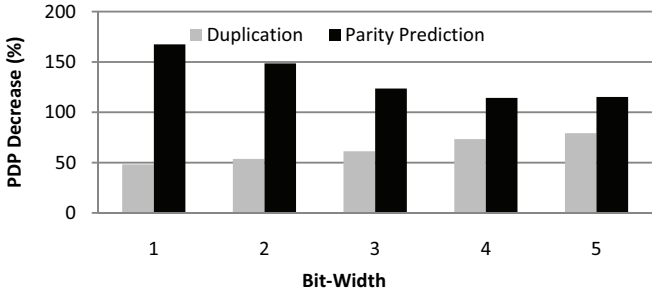

Fig. 10. PDP reduction of LCED in comparison with duplication and parity prediction

Considering derived results of above comparisons, it can be said that the LCED in comparison with the parity prediction and duplication is absolutely low-cost and beneficial. According to the derived results, LCED is absolutely low-cost

and beneficial in comparison with parity prediction and duplication.

In section III, it has been said that the proposed method can be configured into an error correction technique (LCEC). This error correction technique which consists of two extra redundant carries within the architecture and is discussed in the Section III has been implemented.

Table 2. Power consumption, Area, and performance decrease obtained by using LCEC in comparison with Conventional TMR

Bit-Width	Overhead Decrease (%)			
	Power	Area	Performance	PDP
4	159.6	198.4	5.1	168.3
8	167.7	192.1	4.7	146.2
16	173.4	187.5	4.0	104.9
32	186.3	183.1	3.7	63.9
64	189.9	179.3	2.5	46.7

The LCEC technique is compared with Triple Modular Redundancy (TMR). The TMR is almost the only error correction technique that can be applied to the CLA adder. Table 2 shows the effects of applying the LCEC and conventional TMR to the CLA adder.

As depicted in Table 2, it can be seen that LCEC acts exactly like the LCED technique. It is more beneficial in terms of power consumption, area and also PDP. Concerning the delay overheads, using LCEC increases the delay more than the conventional TMR, but the amount of this delay is ignorable because of the following reasons:

1- The maximum value of the delay overhead difference between conventional TMR and LCEC error correction techniques is 5.09%. The delay overhead decreases by increase in the bit width.
2- More than 175.32% decrease in PDP may cover the negligible decrease in performance.

Above results let us to this conclusion that using LCED and also LCEC techniques are more beneficial in order to create a fault-tolerant carry look-ahead adder with minimum cost regarding PDP and area overhead.

V. Conclusion

Conventional adders suffer from carry propagation problem which has a deep effect on performance and also fault tolerance. The CLA adder produces carries in parallel and is among high speed adders. In this paper, a new low-cost fault tolerant technique is proposed which can be applied either as an error correction (LCEC) or an error detection technique (LCED). LCED and LCEC both have focused on the CLAU to reduce the power consumption and area overheads. The LCED-CLA decreases the PDP up to 180% in comparison with parity prediction and duplication. The LCEC-CLA reduces the PDP up to 160% in comparison with the traditional TMR.

VI. References

[1] P. Marwedel, Embedded System Design, Springer, Netherlands, 2006.

[2] P.A. Hsiung, Y. Ru Chen, and Y.H. Lin, "Model Checking Safety-Critical Systems Using Safecharts", *IEEE Trans. on computer*, Vol. 56, No. 5, pp. 692-705, May 2007.

[3] A. Bakhoda, S. G. Miremadi, H. R. Zarandi, "Investigation of Transient Effects on FPGA-based Embedded Systems," *Proc. of 2nd International Conf. on Embedded Software and Systems (ICESS'05)*, pp. 231 - 236, Dec. 2005

[4] J. Benedetto, P. Eaton, K. Avery, D. Mavis, M. Gadlage, T. Turflinger, P. E. Dodd, and G. Vizkelethyd, "Heavy ion-induced digital single-event transients in deep submicron Processes," *IEEE Trans. on Nuclear Science* Vol. 51, no. 6, pp. 3480-3485, 2004.

[5] K. J. Hass and J. W. Ambles, "Single event transients in deep submicron CMOS," *Proc. of 42nd Midwest Sym. on circuits and Systems,* Vol. 1, 1999.

[6] R. Rajaraman, J. S. Kim, N. Vijaykrishnan, Y. Xie, and M. J. Irwin,"SEAT-LA: A soft error analysis tool for combinational logic*," Intl. Conf. on VLSI Design (VLSID), pp. 499-502, 2006.*

[7] P. Shivakumar, M. Kistler, S.W. Keckler, D. Burger, and L.Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," *Proc. of Intl. Conf. on Dependable Systems and Networks*, pp. 389-398, 2002.

[8] C. Zhao, X. Bai, and S. Dey, "Evaluating Transient Error Effects in Digital Nanometer Circuits," *IEEE Transactions on Reliability*, Vol. 56, no. 3, pp. 381-391, 2007.

[9] G.P. Saggese, N.J. Wang, Z.T. Kalbarczyk, S.J. Patel, R.K. Iyer, "An experimental study of soft errors in microprocessors," *IEEE Micro*, Vol. 25, Issue 6, pp. 30-39, 2005.

[10] P. Oikonomakos and P. Fox, "Error Correction in Arithmetic Operations by I/O Inversion," *Proc. of 12th IEEE International Symposium on On-Line Testing (IOLTS'06)*, pp. 287–292, 2006.

[11] V. Ocheretnij, D. Marienfeld, E. S. Sogomonyan, and M. Gössel, "Self-checking code-disjoint carry-select adder with low area overhead by use of add1-circuits," *Proc. of International On-Line Testing Symposium, 10th IEEE (IOLTS'04)*, pp. 31, 2004.

[12] Chen, C.-I. H. and Kumar, A., "Area-Time Optimal Digital BiCMOS Carry Look-Ahead Adder," *Proceedings of the IEEE Asia-Pacific Conference on Circuits and Systems (APCCAS'94*), pp. 115-120, 1994

[13] D. P. Vasudevan, P. K. Lala, and J. P. Parkerson, "Self-Checking Carry-Select Adder Design Based on Two-Rail Encoding," *IEEE Trans. on circuits and systems*, Vol. 54, No. 12, pp. 2696 – 2705, Dec. 2007

[14] Yu-Ting Pai, Yu-Kumg Chen, "The Fastest Carry Look-ahead Adder," *IEEE International Workshop on Electronic Design, Test and Applications,* pp. 434-436, Jan. 2004.

[15] K. Hwang, *Computer Arithmetic: Principles, Architecture, and Design*. New York: John Wiley and Sons, 1979.

[16] J. G. G. Langdon and C. K. Tang, "Concurrent error detection for group look-ahead binary adders," *IBM J. Res. Develop.*, pp. 563-573, Sept. 1970.

[17] V. Ocheretnij, E.S. Sogomonya, and M. Gossel, "A New Code-disjoint Sum-bit Duplicated Carry Look-ahead Adder for Parity Codes," *Proc. of the 10th Asian Test Symp.* (ATS'01), pp. 365, 2001

[18] M. Nicolaidis, "Carry Checking/Parity Prediction Adders and ALUs," *IEEE Trans. on Very Large Scale Integration (VLSI) Systems*, Vol. 11, Issue 1, pp. 121- 128, 2003.

[19] A. Maheshwari, W. Burleson, and R. Tessier, "Trading off transient fault tolerance and power consumption in deep submicron (DSM) VLSI circuits," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 12, no. 3, pp. 299-311, Mar. 2004.