A Low Cost Fault Detection Technique for Full Adder Design

S.H.Mozafari[1], M. Fazeli[1], S. Hessabi, S. G. Miremadi[2]
Department of Computer Engineering, Sharif University of Technology, Azadi St., Tehran, Iran
[1]{sh.mozafari,m_fazeli}@ce.sharif.edu
[2] {hessabi,miremadi}@sharif.edu

Abstract

This paper proposes a low Cost Circuit Level fault detection technique called LCFD for an one-bit FA(Full Adder) as the basic element of adders circuits. To measure the fault detection coverage of the proposed technique, we have conducted an exhaustive circuit level fault injection experiments on all susceptible nodes of an FA. Experimental results show that the LCDF technique can detect about 83% of injected faults while having only 40% area and 25% power consumption overheads. In the LCDF technique, the fault detection latency does not affect the latency of the FA as the error detection circuit performs its job in parallel with the FA circuit.

1. Introduction

When a high energy neutron or an alpha particle strikes a sensitive region in a semiconductor device, a Single Event Upset (SEU) occurs that can alter the state of the system resulting in a soft error. Traditionally, SEUs were only an important issue in space applications[15], but Currently, smaller feature size, lower voltage levels and higher frequencies of deep sub-micron integrated circuits, make the circuits susceptible to the SEUs even at the ground level applications[1][2][3].

In addition, Packaging itself is the one of sources of alpha particles and as these particle can cause SEUs, shielding can't be an efficient method to protect against SEUs[12]. Since neutrons can easily penetrate through packages and cause SEUs, packaging couldn't be an effective way to prevent circuits against faults[13]. These facts augment the role of SEU tolerance techniques to increase reliability of digital circuits.

[12][13] categorized SEU-tolerant techniques into three levels. 1) *Device Level*: techniques which are applied in fabrication process to lessen particle strikes effects. 2) *Circuit Level:* these methods are based on robust circuit design techniques that are applied to circuits to decrease the probability of SEU occurrence[6][7][8][9][13][14][21][22]. 3) *System Level:* error detection and correction techniques[10],these techniques use time, hardware or information redundancy at system level in order to mitigate SEUs effects. One of the effective solution to tolerate faults is triplicating each latch i.e. TMR-latches[16][17][18][19][20]. This fault tolerant technique is an efficient way to reduce SEUs effects, but it suffers from at least 200% area and power overheads which are not appropriate for applications where cost and power consumption are important factors. However, reaching the reliability requirements of a system may have negative impact on other design objectives such as power consumption i.e. the more reliability is achieved, more power is consumed. Consequently, a fault tolerant technique is valuable if and only if it could provide an acceptable balance between design goals.

The importance of fault diagnosis in arithmetic components rises from this fact that these parts of systems are one of the busiest parts among other blocks. If any fault affects these parts it's more likely that the fault become an error and being distributed to other parts. As it mentioned above, arithmetic block is an active part of systems so not only the environmental disturbance could affect these blocks but also these blocks could affect each other by imposing cross talk noises to each other. So, arithmetic blocks placed in a noisy area which have high probability of affection by disturbance and need simple but powerful error detectors.

There are many approaches in order to design a fault-tolerant arithmetic circuit such as hardware, time or information redundancy. [23][24] have used a hardware redundancy method in order to mask the effect of SEUs. In theses work, the output of the arithmetic block is calculated in different ways and the result is voted in a special way that a single error can be masked. However, these methods impose high power and area overhead to the original arithmetic circuit. In [25][26][27], authors have reached to this conclusion that hardware redundancies techniques will be very costly and are not suitable for small arithmetic circuits. Consequently, they decided to exploit information redundancy methods, instead. The authors have suggested using residue codes, parity bits or Berger codes to detect errors. They have also used a time redundancy based approach to correct the errors. However, these techniques do not guarantee the detection of all errors.

Considering the above facts, we find out that although the previously proposed technique may detect errors, their high power and area overheads in basic arithmetic circuit such as one-bit FA, makes them inappropriate

technique in applications such as embedded applications where such overheads are important issues.
In this paper, a low power consuming and low area overhead technique for protecting FA, so called LCDF, is proposed. The LCDF technique can be used as a basic adder for complex arithmetic circuits. This design is based on this fact that some logical terms which exist inside a one-bit FA, would made constant logical relations with each other. Therefore, by checking these constant relations we can distinguish between corrected and faulty outputs. These constant relations between inner terms of a one-bit FA can be extracted by parity generators. Checking these constant logical terms can be accomplished concurrently with FA operation; this fault diagnose method have no timing overheads on the FA. However, using parity generator requires XOR gates that have much more delay and occupies much more area than other basic gates such as AND, OR, and NOR gates. To address this problem,
in this paper, we have used a simple but useful method to reduce XOR gate overheads. In this method, faulty results will be also corrected by a time redundancy approach.
The rest of the paper is organized as follows: In Section II, an overview of related work is presented. Our proposed fault tolerant FA (LCDF) is presented in Section III. The experimental results are discussed in Section IV. The benefits of our proposed fault tolerant techniques highlighted by experimental results presented in Section V. Section VI concludes the paper and presents some future work.

II. Related Work

In [25] after comparison of different fault tolerant techniques, such as time redundancy or information redundancy, the authors get to this conclusion that none of these method alone, could protect arithmetic circuits. In addition, use of hardware redundancy would be very costly. Therefore, the authors have proposed to use combination of these methods. He used residue codes as data redundancy method and corrected errors by time redundancy. In this paper it uses the residue code as follows, at first calculate the residue of inputs of a multi-bits FA to 3 (A%3, B%3) and then let the circuit to calculate the sum of 2 inputs, then calculate the output residue (Z%3) and expected that sum of residues be equal to output residue. $A\%3 + B\%3 = Z\%3$ (% means modulo). So in this method the designer cancalculate the sum and it's checking value simultaneously. If the residue check shows an error, it is concluded that at least one error has been occurred in sum or residue circuit. After finding the occurrence of an error, it's time to correct it. Writer suggested using time redundancy and recalculation of Adding operation. This method has only 45% overhead in 32 bits adders. Obviously this method has high hardware and power costs for small Adding operations and only can be used in big circuits. Though this method has high overheads in small adders but clearly shows that data redundancy methods are less costly than hardware redundancy. So, in our proposed design we didn't used information redundancy and we tried to solve the problem of data redundancy high costs by using low cost parity checkers.
The main idea for proposed method in [26] is based on this fact that Add operation is self dual function $\overline{f}(\overline{x}) = f(x)$. When an error is detected in a FA, we impose inverted inputs and get inverted outputs. So, if the origin error was due to hard (like SSA[1]) or soft errors, both of them could be corrected. In the proposed paper after comparing different methods for detecting errors in a one-bit FA the designer decides to use parity bits because of its low cost in comparison to other fault tolerant methods.

III. The Proposed FT Full Adder

One-bit FA is the base unit in arithmetic parts of every processing elements. Thus, if an error occurs in this part and is not detected, it can be propagated to other arithmetic elements resulting in program result failures.
A full adder (FA) has three inputs (a, b, Cin) and two outputs (sum, Cout). Eq.1 shows the relation of FA inputs and outputs.
sum = (a xor b) xor Cin **Eq.1**
Cout = (a and b) xor ((a xor b) xor Cin)

Figure 1 illustrates the gate level schematic of a full adder. As it can be seen from this figure, there are eight SEU susceptible nodes in the FA circuit. We call these nodes *Sensitive Points*.

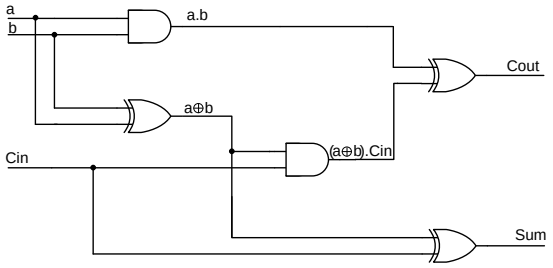

Figure 1. Schematic of proposed FA without error detection Circuit

These *Sensitive Points* maybe coincidently triggered by noises, if these *Sensitive Points* become faulty they would show same effects on outputs. This is very useful behavior of FA circuit for our proposed error detection technique.
In order to simulate the proposed FA in transistor level we need to know the structure of each gate. For proposed FA we need to know the structure of two gates: And gate and Xor gate. And is simulated in C-CMOS[2] logic style (6 transistors) and Xor gate has been simulated in Hybrid-CMOS logic style. The

[1] Single Stuck At
[2] Complementary CMOS

schematic of And gate and Xor gate are shown respectively in Figure 2 and Figure 3.

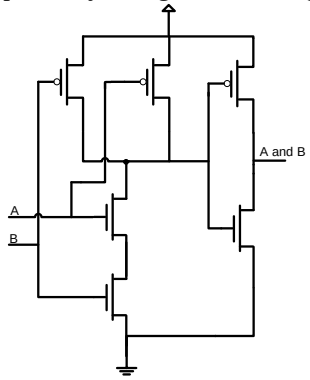

Figure 2. Complementary CMOS And gate

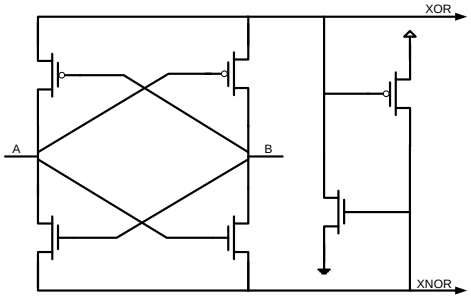

Figure 3. Hybrid Xor gate

This structure has been selected for Xor because most of Xors which would be used in implemented FAs use Hybrid-CMOS logic style [29]. Then these gate structures help us to get to real power estimations.

One of the popular methods in error detecting is *Data Redundancy*. In the proposed method for detecting errors we used a kind of *Data Redundancy* which is based on this fact that there are constant relations between *Sensitive Points*. So, once theses relations being denied the error detector alarms that an error occurred. if we want to have acceptable power and area overheads, these relations should be very simple and consume little power. In order to fulfill this goal we decided to use parity bits. In the Table1 you may see all *Sensitive Points* and their possible logical values in a one-bit FA.

Table1. the trust table of *Sensitive Points*.

a	b	Cin	a and b	a xor b	(a xor b) and Cin	Sum	Cout
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	1	0	1	0
0	1	1	0	1	1	0	1
1	0	0	0	1	0	1	0
1	0	1	0	1	1	0	1
1	1	0	1	0	0	0	1
1	1	1	1	0	0	1	1

Some groups of these *Sensitive Points* are listed below. As you can see through Table 1 even parity of these groups of signals have constant ‘0’ value. It means that always even number of these signals have ‘1’ value.

- {a, b, Cin, Sum}
- {a, b, (a xor b)}
- {(a and b), (a xor b) and Cin, Cout}
- {Cin, (a and b), (a xor b),(a xor b) and Cin, Sum, Cout}

Now as the relation between these *Sensitive Points* is cleared, it is time to implement the error detector circuit. After testing different groups of introduced signals for parity checking; we got to this conclusion that the best groups, for error detecting would be theses below two groups.

- {a, b, Cin, Sum}
- {a and b, (a xor b) and Cin, Cout}

The most challenging part of designing error detecting circuit would be its structure. Because as we know a single one-bit FA has only 5 elementary gates and if the error detector circuit be so big, it will become useless as it will consume more than acceptable power. So, it is important to design a simple parity checker with low power consumption. Inorder to get to this goal a parity checker with pass transistor logic style has been proposed. The schematic of this parity checker is shown in Figure 4.This circuit will Xor 4 signals: a, b, Cin, Sum.

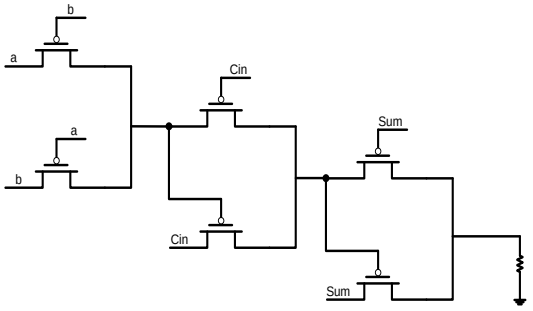

Figure 4. Parity checker circuit

As we used Pmos transistors as pass transistors the output couldn’t become full swing but with that transistor made resistance, the output became full swing. This resistance produces the zero logic whenever the output is high impedance. It seems that, this kind of parity checker design would have static power consumption, to prevent this waste we selected the groups of signals which have ‘0’ parity value. Therefore when the FA is working faultless, there will be no power consumption in parity checkers.

As it was mentioned before, in these two groups of selected signals 7 of 8 *Sensitive Points* have been participated in parity generators. This question may come to mind that maybe the omitted signal be faulty and not being discovered by these parity checkers. In reply we should say that not only the omitted *Sensitive Point* (a xor b) but also other point has overlap effects on each other. So, if an error occurs in one of them, this

error will distributed inside FA and will affect other points.
After diagnosing error occurrence in the FA, it's time to correct this error. For this process there are two ways. One is to use time redundancy and impose inputs once again. The other way is to impose inverted inputs and get the inverted outputs. As it is was mentioned the *sum* and *Cout* functions are self dual so inverting method would produce faultless outputs. The second way is better as it can also mask SSA faults which caused errors[26].

IV. Experimental Approach

A. Simulation System Setup

In order to be near to real working conditions it is needed to test our proposed FA in single platform this platform has been exhibit in Figure 5.

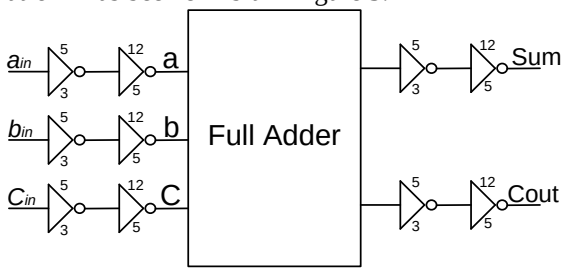

Figure 5. Simulation test bench

The proposed FA was designed by using the TSMC 0.13-µmCMOS technology, The threshold voltages of the PMOS and NMOS transistors are approximately 0.33 and 0.35V, respectively.this circuit was simulated using the BSIM3v3model with Level 49 technology file. Simulations were carried out using Star HSPICE. Environment temperature has been set to 27ºC. The introduced test bench used to simulate the FA circuits. This simulation environment has been commonly used to compare the circuit performances in [30][31][32].To simulate a real environment, input buffers for both inputs of the test circuit are used to generate a real waveform and output buffers for both outputs are used to generate output load. The buffers transistor sizes are specified in the figure.

In order to test all the possible transitions as inputs, an input test pattern with 56 transitions must be applied to a FA circuit. An input transition may or may not result in change at the output node. Even if there is no switching activity at the output node, some internal nodes maybe switching. Thus, for having an accurate result, all the possible input combinations are considered for all the test circuits [30]. Therefore, a group of input test patterns which offers all the 56 different transitions from one input combination to another are used as the input vectors for the FA circuit in 560ns duration. Each transition applied to circuit in 10ns.

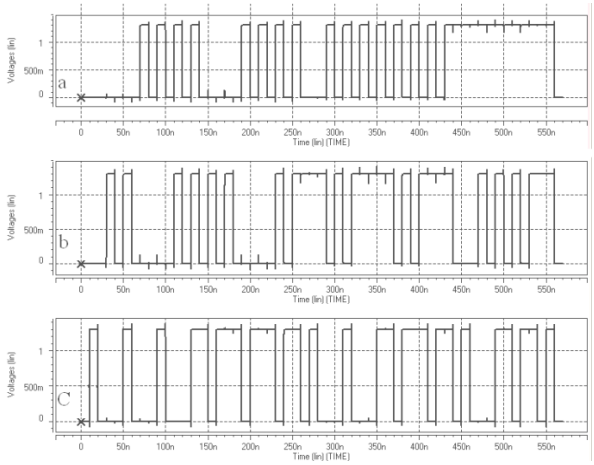

Figure 6. Buffered inputs and their transitions

B. Fault Injection Method

By imposing the described fault model to the circuit in different times, the outputs of FA is observed for error occurrence. If any error in sum or Cout signals occurs then we will check the error detector circuits. If they flip their logic, it shows that an error diagnosed and we will not use this output but if it doesn't detect any error, FT FA consider that no error occurred. You can see a 150 ns sample window for two *Sensitive Point* signals and parity generator in Figure 7. As it is exhibited, there are some faults on (a xor b) signal and these faults became errors and influenced the sum signal. While this happened to sum signal, the parity checker detected errors and flipped its value. In Figure 7, the Parity Checker signal has detected an error about 530ns which is not correct and it's a *False Alarm* from error detector circuit.

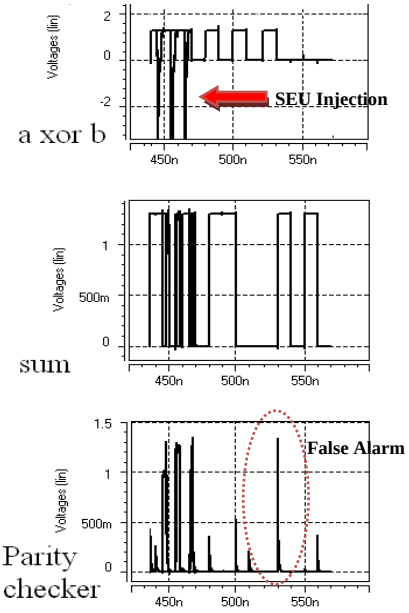

Figure 7. Signals' variation during fault injection

In General faults can be categorized into two classes: permanent faults and transient faults. One of the main

source for transient faults would be Particles strike, especially when circuit is designed in submicron technologies area.[36][37]. As technology sizes are shrinking the factor of SUTs effects is getting stronger than other causes of faults.

In order to model the particle strikes effects on digital circuits, an independent current source was used to represent the collected charges generated by a particle strike. Pervious works [36][38] have been used similar approach to model particle strikes effects.

Figure 8 shows the disturbance of current on a transistor's drain which is struck by a particle. The equation of current disturbance is presented in Eq. 2. This current equation depends on charge, a technology dependent parameter T, and time t [39][36]:

$$I_{Particle} = \frac{Q}{T\sqrt{\pi}}\sqrt{\frac{t}{T}}e^{\frac{-t}{T}} \qquad \textbf{Eq. 2}$$

In this equation T is technology dependent parameter and Q is the charge that we apply to electrical particle. This parameter is proportional to the deposit charge when a particle strikes at our circuit.

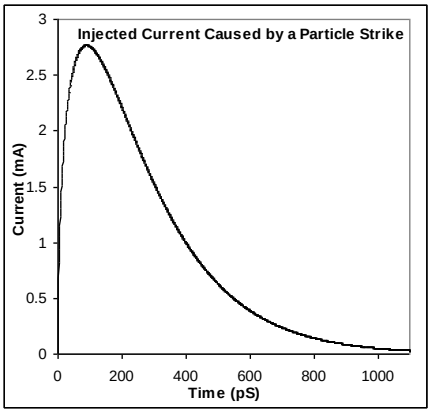

Figure 8. A particle strike current diagram

Parameter t will determine the time in which the particle has been affected the circuit.

In order to measure the SEU tolerance capability of different designs, the SEU injection experiments have been carried out for different values of Q ranging from -0.8pc to 0.8pc in different times, t, ranging from 0.15ns to 0.555ns with random T value which could varies from 0.4ns to 0.7ns. Since, a particle may has either a positive or negative charge, we consider both positive and negative values of Q in our SEU injection experiments, i.e. the value of Q varies between -0.8pc to 0.8pc by 0.2 step. Different simulated SUTs with different constant time and charge are applied to each node of FA. The total number of SEU injections for a specific design depends on the number of susceptible nodes in the design. Generally the total number of SEU injections can be measured by the Eq.3.

$$Totla\ Number\ of\ Injections = N_Q . N_T . N_{Node} \qquad \textbf{Eq.3}$$

Where N_Q is the number of values for Q, N_T is the number of values for T and N_{Node} is the number of susceptible nodes in the design. Considering the above equation 924 faults were injected to the circuit.

C. Transistor Sizing

In this section we suggest a way in order to determine each transistors size.

As there was no hint to determine size of the circuit's transistors, we decided to use transistor sizing method to fix the transistors sizes. As it is mentioned in [33], the transistor sizing for optimal performance, is technology dependent. For a certain technology, the channel lengths of all transistors are fixed at the minimal feature size. So, the only variable should be optimized, is the channel width of each transistor [34] Selecting the optimization factor is dependent on our needs. At present, we chose, Power-Delay Product (PDP) which is a metric for energy consumption of a circuit, and is vitally important. Thus the transistors have been sized to meet the minimum PDP. This factor will help us to have fair judgment on power consumption of error detector circuit, proportional to other parts of circuit. This method for transistor sizing will help us to lessen the consuming power of design and won't let maximum delay of circuit to be increased dramatically.

This transistor sizing algorithm is called SEA. In this algorithm we should grouped transistors and as its mentioned in [40] we grouped those transistors which have almost the same role in circuit. As we grouped these transistors according to their type and behavior we reached to 8 groups in our FA circuits. Each group should have its own width and all transistors in a group have the same size for their width but, group's width differ in each group of transistors.

In this algorithm for transistor sizing we begin with width of grouped transistors, as we call them W_i. We will put W_i of all groups to length of them which is set to the minimum size that is technology limit (0.13µ) and is shown with *Lmin*. In order to change each group of W_i we use Eq. 4.

$$Wi = ki * Wmin \qquad \textbf{Eq. 4}$$

In this equation the *Wmin* is a constant value which will be a coefficient for width of transistors. So the parameter that indicates the transistor width is k_i. This parameter will be determined during algorithm run.

At first SEA algorithm, will fix the *Wmin* parameter. It will do it by sweeping *Wmin*, from minimum length to multi *Lmin* when $k_{i=1}$. In this process the PDP of circuit will be calculated for each value of *Wmin* and minimum value will be selected in order to fix *Wmin* parameter. After this selection, it's time to determine k_i this will be done by mentioned algorithm in the body of SEA algorithm which is introduced in Figure 9.

1 *Group all the transistors in the circuit using Transistor Grouping Rules;*
2 *Initialize W_i , width of the transistors in group i , so that:*
3 $W_i = k_i \times W$;
4 $k_i = 1$; $i = 1,2,L,n$
5 $W = 0.13\ \mu m$;
6 *Initialize s , the step size and m, the number of step sizes;*
7 ***for***(t =1 to m) {
8 // *This loop sweeps W^* to obtain W_{opt}.*
9 $W^* = W^* + s$;
10 *Compute the target parameter, Θ and save it to an array; }*
11 W_{opt} *is the point where Θ has become minimum for that;*
12 $W = k \times W$
13 ***do {***
14 ***for***(i =1 to n) {
15 // *This loop sweeps k_i.*
16 ***for***(t =1 to m) {
17 $k_i = k_i + s$;
18 *Compute Θ and Save it to another array; } }*
19 k_i *is the point where Θ has become minimum for that;*
20 ***while****(all k_i's converge to an specific value);*
21 *Target parameter in this case is PDP.*

Figure 9 Simple Exact Algorithm (SEA) for optimizing the target parameter[40].

In order to run this algorithm we used 0.15 for ki steps (s= 0.15) and simulate circuit from k=1 to k=5. The simulation was running till PDP converge to a minimum value.

For calculating PDP parameter which is target parameter in this algorithm and is gotten from production of power and delay in a circuit, we used average consumption power which includes static power and switching power of circuit in all operating time(560 ns). In order to calculate delay of circuit we considered maximum delay which will produced in this way: that the calculated time is began from time of each transition in inputs and is end when the output of circuit is affected by that input.(those iterations which doesn't affect outputs will not be considered)After running SEA algorithm on our circuit. The transistor groups were sized and these results were gotten. PDP=1.409 fJ, Average Power=5.756 µW, Max Delay=0.244 fs.

V. Experimental Results

In simulations it was appeared that from 924 injected faults only 512 of them made error in outputs. Among these 512 effective faults, 428 were detected by Parity checkers and 37 correct outputs were wrongly detected as faulty outputs.

In Table 2 we categorized SEUs which has been implied to the fault tolerant FA circuit by their charges. These SEUs could have three different affect on FA outputs: First, they may not be detected. Second, they could be masked and Third, they could be detected wrongly.

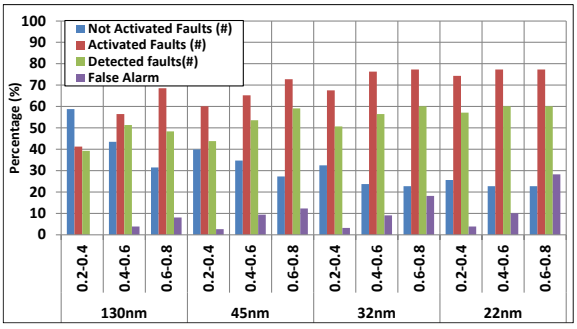

Figure 10 Fault injection results in the proposed circuit for total number of 308 fault injections.

Transistor sizing will enable us to have fair judgment on FA power consumption and Parity checker parts. Results indicate that we have about 25% power overhead for parity checkers while fault injection, parity checker circuits will consume 3% to 5% more power than fault free mode. If we consider the layout size of each Pmos transistor double of Nmos transistors, then the area overhead would be 40% for fault detection parts of this FA circuit style.

In order to have results of power consumption in newer technologies, we used low power PTM libraries[41], in 22, 32, 45 nanometer to simulate our proposed circuit in these technologies. In these simulations we used their nominal needed voltage, and transistor sized each circuit with new technology libraries by SEA algorithm. After transistor sizing we compared their power consumptions. As you can see in Table 3, average power consumption and maximum delay of FT circuit in each technology is reported.

Table 3 Power consumption and delay of proposed circuit in different Tech sizes

Tech Size[nm]	130	45	32	22
Average power[µW]	**5.756**	2.269	**1.672**	1.274
%power overhead	**24.5**	22.7	**19.9**	20.7
Max Delay[ns]	**0.244**	0.994	**1.684**	6.401
Power-Delay Product				

Reported results in Table 3 show that as technology size shrinks the power consumption of whole circuit would be lessen and the power portion of fault detector to rest of circuit will decrease.

In Figure 11 You can comparison the power consumption of proposed FA with fault diagnose parts.

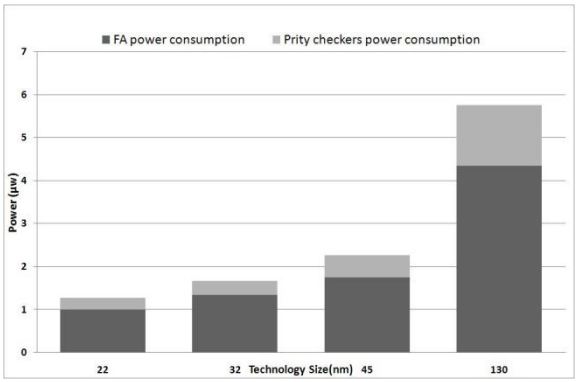

Figure 11. Power consumption of fault diagnose part & Non FT FA

As we used low power technology libraries delay of circuits will goes up. You can see chart of circuit's delay in Figure 12.

Figure 12. FA delays after transistor sizing

VI. Conclusion and Future Works

As it was mentioned, the proposed FT FA can detect SEUs and these errors can be corrected in the described method. In this paper we discussed about the strength of this FT FA for single error detection but this detection technique has the ability to detect multiple errors as well. This method of error detection will force a delay to get the correct result-time redundancy technic-, so it may be destructive for performance in some processing structures like single pipeline. In order to reduce time redundancy effect, it is recommended to use it in Superscalar processing units[35].

One of the defects which this method would have is that for every single one-bit FA we have 40% area overhead. This fact will increase number of transistor in big circuits like *Adder Based Multipliers* and *Compressors*. But this defect won't be critical as transistor's sizes are shirking. On the other hand these processing units have a small area portions respecting other parts of a processing element.

The proposed method for correcting errors is very simple and isn't robust against hard errors, especially whenever the fault happens in detection circuit. In order to solve this problem there are some suggestion which seems to be useful for future work. The main suggestion is a complete method for error detection and correction that is introduced here.

This method is base on this fact that faults are two kinds, transient and permanent. When a transient faults happens in a FA, by flipping inputs and outputs and recalculation it will be disappeared. But if the fault be a permanent fault, we confront with two cases.

1) Permanent fault, which we consider as SSA fault model, is happened in FA.
2) Permanent fault happened in error detection part.

In the first case the solution is like transient faults and by flipping inputs and outputs it will be solved (because of self-duality properties of FAs) but in the second case, error effect will not be disappeared by this method and this part should be changed by a correct one. So, we use a standby error detector. When an error occurs, first we use flipping method and go through below steps:

1) Recalculate the Adding operation with inverted inputs/outputs.

If an error detected once more, we think that two cases would happened, first we had multiple stuck at faults in FA, second we had faulty *error detection* part. So, in order to distinguish between these two parts we use these methods.

2) Recalculate the sum with standby error detection and no inverted inputs.

If these outputs were corrected we can see that error detection part was faulty. Otherwise we suppose that FA has permanent fault and should be replaced.

3) Recalculate the sum with standby FA and in this case we can conclude that our FA was faulty.

These methods can be used as correcting and reconfigure methods. These methods use time redundancy which will make bubble in processing sequence. Therefore these methods are suggested in multi-pipeline structures like Superscalar which *pipeline bubbling* do not have destructive affect on processing sequence.

7. References

[1] N. Cohen, T.S. Sriram, N. Leland, D. Moyer, S. Butler, R. Flatley, "Soft error considerations for deep-submicron CMOS circuit applications", International Electron Devices Meeting, Washington, DC, 1999, pp. 315–318.

[2] R.W. Keyes, "Fundamental limits of silicon technology", Proc. IEEE, vol. 89, no. 3, Mar. 2001, pp. 227–339.

[3] S. Mitra, T. Karnik, N. Seifert, M. Zhang, "Logic soft errors in sub-65nm technologies design and CAD challenges ", Design Automation Conference (DAC), Anaheim, CA, June 2005, pp. 2 – 4.

[4] S.W. Fu, A.M. Mohsen, T.C. May, "Alpha-particle-induced charge collection measurements and the effectiveness of a novel p-well protection barrier on VLSI memories," IEEE Trans. Electron Devices, vol. ED-32, no. 1, Jan. 1985. pp. 49–54.

[5] A. Mahmood, E.J. McCluskey, "Concurrent Error Detection Using Watchdog Processors – A Survey", IEEE Trans. on Computers, Feb. 1988, pp. 160 -174.

[6] A.J. Drake, A. KleinOsowski, A.K. Martin, "A Self- Correcting Soft Error Tolerant Flop-Flop", 12th NASA Symposium on VLSI Design, Coeur d'Alene, Idaho, USA, Oct. 4-5, 2005.

[7] R. Naseer, J. Draper, "The DF-DICE Storage Element for Immunity to Soft Errors", Proceedings of the 48th IEEE International Midwest Symposium on Circuits and Systems, August 2005.

[8] D.R. Blum, M.J. Myjak, J.G. Delgado-Frias, "Enhanced Fault-Tolerant Data Latches for Deep Submicron CMOS," The 2005 International Conference on Computer Design (ICCD), pp. 28-34, Las Vegas, June 2005.

[9] K.J. Hass, J.W. Gambles, B. Walker, M. Zampaglione, “Mitigating single event upsets from combinational logic,” 7th NASA Symposium on VLSI design, 1998.

[10] C.L. Chen, M.Y. Hsiao, “Error correcting codes for semiconductor memory applications: A state-of-the-art review,” IBM J. Res. Develop., vol. 28, no. 2, pp. 124–134, Mar. 1984.

[11] P. Hazucha, C. Svensson: "Impact of CMOS technology scaling on the atmospheric neutron soft error rate", IEEE Trans. On Nuclear Sc. Vol. 47, n. 6, Dec. 2000, pp. 2586-2594.

[12] Q. Zhou, K. Mohanram, "Gate sizing to radiation harden combinational logic", IEEE Transactions on Computer-aided Design of Integrated Circuits and Systems (TCAD), vol.25, Jan. 2006, pp. 155-166.

[13] T. Calin, N. Nicolaidis, R. Velazo, "Upset Hardened Memory Design for Submicron CMOS Technology", IEEE Trans. On Nuclear Sc. Vol. 43, n. 6, Dec. 1996, pp. 2874-2878.

[14] M.P. Baze, S.P. Buchner, D. McMorrow, "A Digital CMOS Design Technique for SEU Hardening", IEEE Trans. On Nuclear Sc. Vol. 47, n. 6, Dec. 2000, pp. 2603-2608.

[15] A. Ejlali, B.M. Al-Hashimi, M.T. Schmitz, P.Rosinger, S.G. Miremadi, "Combined time and information redundancy for SEU-tolerance in energy-efficient real-time systems", IEEE Trans. on Very Large Scale Integration Systems, Vol. 14, April 2006, Issue: 4, pp. 323- 335.

[16] F. Kastensmidt, L. Sterpone, M. SonzaReorda, L. Carro, "On the Optimal Design of Triple Modular Redundancy Logic for SRAM-Based FPGAs," DATE2005: IEEE Design, Automation and Test in Europe, 2005, pp. 1290-1295.

[17] M. Favalli, C. Metra, "TMR voting in the presence of crosstalk faults at the voter inputs", IEEE Transactions on Reliability, Sept. 2004, Volume: 53, Issue: 3, pages: 342 - 348, ISSN: 0018-9529.

[18] L. Sterpone, M. Violante, "Analysis of the robustness of the TMR architecture in SRAM-based FPGAs," IEEE Transactions on Nuclear Science, 2005, Vol. 52, No. 5, October 2005, pp. 1545 – 1549.

[19] V. Stachetti, J. Gaisler, G. Goller and C.L. Gargasson, “32-BIT Processing Unit For embedded Space Flight Applications”, IEEE Trans. on Nuclear Science, Vol. 43, No. 3, June 1996, p.p. 873-878.

[20] J. Gaisler, “A Portable and Fault-Tolerant Microprocessor Based on the SPARC V8 Architecture”, Proc. of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN’02), June 2002, p.p. 409 - 415.

[21] Y. Zhao, S. Dey, “Separate Dual-Transisto Registers –A Circuit Solution for On-Line Testing of Transient Error in UDSM-IC”, in Proc. of 9th IEEE Int. On-Line Testing Symp. (IOLTS’03), pp. 7-11, 2003.

[22] M. Omana, D. Rossi, C. Metra, “Novel Transient Fault Hardened Static Latch”, in Proc. of IEEE Int. Test Conference (ITC’03), pp. 886-892, 2003.

[23] Kai-Chiang Wu & Diana Marculescu. "Soft Error Rate Reduction Using Redundancy Addition and Removal.". *22nd IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems.*

[24] Alderighi, M.; D'Angelo, S.; Metra, C.; Sechi, G.R, “Novel fault-tolerant adder design for FPGA-based systems” *On-Line Testing Workshop, 2001. Proceedings. Seventh International"* Page(s): 54 – 58 2001.

[25] Veeravalli, V.S., “Fault tolerance for arithmetic and logic unit”,Southeastcon, 2009. SOUTHEASTCON '09. IEEE Page(s): 329 – 334 2009.

[26] Oikonomakos, P.; Fox, P., ”Error correction in arithmetic operations by I/O inversion”12th IEEE International On-Line Testing Symposium, 2006.

[27] E. J. Ossi, D. B. Limbrick, W. H. Robinson, B. L. Bhuva; “Soft-error Mitigation at the Architecture-Level Using Berger Codes and Instruction Repetition” 2009

[28] Forsati, R.; Faez, K.; Moradi, F.; Rahbar, A.,” A Fault Tolerant Method for Residue Arithmetic Circuits”Information Management and Engineering, ICIME '09, Page(s): 59 – 63 2009

[29] SumeerGoel, Ashok Kumar, Magdy A. Bayoumi, "Design of Robust, Energy-Efficient Full Adders for Deep-Submicrometer Design Using Hybrid-CMOS Logic Style" IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS,VOL. 14, NO. 12, pp. 1309-1321.

[30] S. Goel, A. Kumar, and M. A. Bayoumi, “Design of robust energy–efficient full adders for deep submicron design using Hybrid-CMOS logic style,” IEEE Trans. Very Large Scale Integr. (VLSI) Sys., vol. 14,no. 12, pp.1309-1321, Dec. 2006.

[31] A. M. Shams and M. A. Bayoumi, “A structured approach for designinglow power adders,” in Proc. 31st ASILOMAR Conf. Signals, Systems andComputers, Pacific Grove, 1998, pp. 757-761.

[32] M. Aguirre and M. Linares, “An alternative logic approach to implement high-speed low-power full adder cells,” in Proc. 18th Annual Symp.Integrated. Circuits and Syst. Design, Florianopolis, 2005, pp. 166-171.

[33] M. Sayed and W. Badawy, “Performance analysis of single-bit full adder cells using 0.18, 0.25 and 0.35µm CMOS technologies,” in Proc.35 IEEE Int. Symp. Circuits Syst., Scottsdale, 2002, pp. III-559-III-562.

[34] C.–H. Chang, J. Gu, and M. Zhang, “A review of 0.18-µm full adder performances for tree structured arithmetic circuits,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 13, no. 6, pp. 686-695, Jun. 2005.

[35] Ghosh, S.; Ndai, P.; Roy, K.; "Design, Automation and Test in Europe, 2008. DATE '08" Digital Object Identifier: 10.1109/DATE.2008.4484707 Publication Year: 2008 , Page(s): 366 – 371

[36] P. Hazucha, C. Svensson, “Impact of CMOS Technology Scaling on the Atmospheric Neutron Soft Error Rate,” IEEE Trans. on Nuclear Science. Vol. 47, No. 6, pp. 2586-2594, December 2000.

[37] K. Hass, J. Gambles, “Single Event Transients in Deep Submicron CMOS,” Proc. of Midwest Symposium on Circuits and Systems, 1999, pp. 122- 125.

[38] A. Ejlali, B.M. Al-Hashimi, M.T. Schmitz, P. Rosinger, S.G. Miremadi, “Combined Time and Information Redundancy for SEU-tolerance in Energy-Efficient Real-Time Systems,” IEEE Trans. on Very Large Scale Integration Systems, Vol. 14, No. 4, pp. 323-335, April 2006.

[39] L.B. Freeman, “Critical charge calculations for bipolar SRAM array,” J. Res. Develop., vol. 40, no. 1, pp. 119-129, January 1996.

[40] ToorajNikoubin, Poona Bahrebar, Sara Pouri, KeivanNavi, and VaezIravani, “Simple Exact Algorithm for Transistor Sizing of Low-Power High-Speed Arithmetic Circuits,” VLSI Design Volume 2010, Article ID 264390, 17 pages

[41] Predictive technology model : http://ptm.asu.edu./