http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 11 (10): 1442-1448, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.1442.1448

Fault Tolerance Structure of Radix 2 Signed Digital Adders

Fei Gu, Chunqing Ling, Jishun Kuang and Yingbo Zhou
College of Information Science and Engineering, Hunan University,
Changsha, Hunan, 410082, People's Republic of China

Abstract: In this study, structure of fault tolerance adder based on Radix 2 Signed Digital (SD) representation is proposed. The "carry-free" property of the SD adder that faults impact limited to a few digits can be used to fault detection which is based on parity checking assumed single fault set. Using an encoding scheme to get the parity value of digits involved in computing, this parity values can be exploited to check the circuit. An error information register is set to store the checking results and the bits of the register indicate the corresponding units faulty or not. According to the fault type, recomputation or reconfiguration is used to error correction. The hardware overhead appending Fault-Tolerant is about 120% and the maximum combinational path delay of the proposed adder is constant with the increase of operands.

Key words: SD adder, fault tolerance, self-checking, parity check

INTRODUCTION

With the development of semiconductors and technology, integrated circuits become more and more complex. In some special applications Integrated circuits have to face large amount of ionizing radiation. A single charged particle can knock thousands of electrons loose, resulting the digital circuits inaccurate or unintelligible (Yu *et al*., 2010). Due to the mentioned reasons above the self-checking or fault-tolerance of integrated circuit has become a hot item in the semiconductor industry. As an essential building block of arithmetic logic unit in data processing circuit, adder affects the reliability of the whole system strongly. How to design highly reliable adders with on-line error detection and correction capability has been an important research topic. In literatures, a number of self-checking adder implementations has been proposed. Self-checking is achieved by inclusion of redundancy within a circuit. The redundancy can be in the form of time redundancy, space redundancy, information redundancy or hybrid of these techniques. The space redundancy, also known as hardware redundancy, has the least impact on the speed of the circuit. However, its cost of circuit size and power are highly. The classic space redundancy techniques are DWC (Johnson *et al*., 2008) and TMR (D'Angelo *et al*., 1999). Information redundancy, such as parity codes (Hamidi *et al*., 2009; Ziabari *et al*., 2008), residue codes (Hosseinzadeh and Navi, 2007), M-out-of-N codes (Kharbash and Chaudhry, 2007) and Berger codes (Morozov *et al*., 2000) are widely used to implement self-checking logic. It increases the circuit size and power consumption moderate but it also increase the complexity of the circuit. Time redundancy (Townsend *et al*., 2003; Piuri and Swartzlander, 1999), require the minimum circuit area but it increases the circuit delay highly so that it is unsuitable for some real-time applications.

A system based on residue number is an unconventional system with parallel, carry free, high speed arithmetic capability (Barati *et al*., 2008). Signed digit number is a residue number representation. The carry propagation of signed digit number representation limits to one position during the addition operation (Avizienis, 1961). Currently most of the self-checking SD adders can just detect errors, little works proposing adders with both error detection and correction capabilities (Cardarilli *et al*., 2006). Cardarilli *et al*. (2006) presented a new procedure that locates the faulty digits by means of ad hoc algorithms which based on parity checker technique but the correction is partial. Alavi and Faez (2007) improved the correction capability to full correction by using a recomputation with triple shifted operands method. But it would waste too much time if a permanent fault attacks the circuit, because the correction procedure will be active every time when inputs change. Both of the algorithms are unsuitable for some real-time applications.

The purpose of this article was to present a new fault-tolerant radix 2 SD adder structure with hybrid redundancy. The error detection property in Alavi and Faez (2007) can be modified to locate error sites. Once catching any error, a two-stage diagnosis procedure is used to identify the fault. Recomputation and reconfiguration are decided by the diagnosis results used to identify the fault. Recomputation and reconfiguration are decided by the diagnosis results.

Corresponding Author: Chunqing Ling, College of Information Science and Engineering, Hunan University, Changsha, Hunan, 410082, People's Republic of China Tel: +86-15874093171

BACKGROUND

In this section, the basic theory of Binary Signed Digit Number (BSDN) is proposed. For a digit $a \in [-(2^n-1), 2^n-1]$, it can be described by the BSDN representation as:

$$a = \sum_{k=0}^{n-1} x_k 2^k \quad (1)$$

where, $x \in \{-1,0,1\}$, $(i = 0,..., n-1)$. The addition procedure of BSDN operands a_i and b_i can be partitioned to two operations:

$$w_i+2c_i = a_i+b_i \quad (2)$$

$$z_i = w_i+c_{i-1} \quad (3)$$

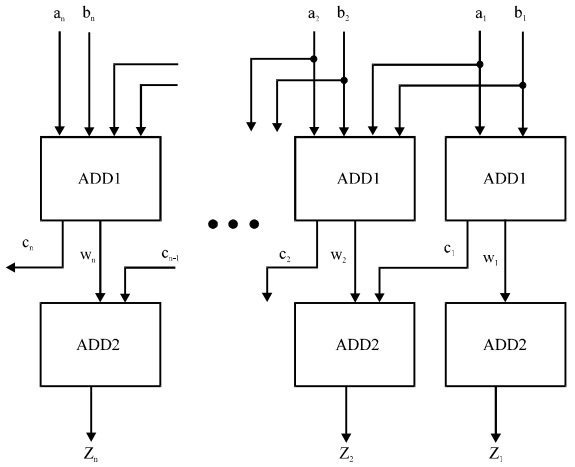

Fig. 1: Adder based on signed digit

Table 1: The function of ADD1

Addend, augend digits at position i	Sign info on digit at position i-1	Intermediate carry digit c_i	Intermediate sum digit w_i
-1,-1	Not used	-1	0
-1,0	$a_{i-1}+b_{i-1}<0$	-1	1
	Otherwise	0	-1
0,0	Not used	0	0
1,-1	Not used	0	0
1,0	$a_{i-1}+b_{i-1}>0$	1	-1
	Otherwise	0	1
1,1	Not used	1	0

Table 2: The function of ADD2

w_i	c_{i-1}	z_i
1	1	-
1	0	1
1	-1	0
0	1	1
0	0	0
0	-1	-1
-1	1	0
-1	0	-1
-1	-1	-

Where:

$$c_i = \begin{cases} 1 & \text{if } (a_i + b_i) \geq 1 \\ 0 & \text{if } (|a_i + b_i|) < 1 \\ -1 & \text{if } (a_i + b_i) \leq -1 \end{cases}$$

The addition can be implemented by the structure shown in Fig. 1 which contains two blocks, ADD1 and ADD2. The block ADD1 implements (2) and generates intermediate output c_i and w_i. The block ADD2 performs (3), providing the result z_i. Specific rules of arithmetic are shown in Table 1-2. This BSDN addition can be performed in parallel due to the carry propagation just limited to one digit.

PARITY CODING SCHEME

There are three different values for a binary signed digital number, thus need 2 bits to encode. The parity coding used in this paper is the same as Cardarilli *et al.* (2006). Digit '0' is represented by 00/11 and digits '1' and '-1' are signified by 01 and 10, respectively. This coding scheme has two advantages:

- Any Stuck-at fault will change the absolute value of the digit
- The outputs of ADD1 on position i relate with the sign of inputs on position i-1.With the chosen coding strategy only MSBs of inputs on position i-1 are needed and therefore, the ADD1 can be implemented with 6 input bits instead of eight

Using this coding strategy, the arithmetic results can be checked by the parity-check properties. Defined $P(a_i)$ as the XOR of the bits forming a_i, $P(b_i)$ as the XOR of the bits forming b_i, $P(c_i)$ and $P(z_i)$ as the parities of carry and partial sum and result, respectively, the following properties can be obtained:

Property 1:

$$P(z_i) = P(w_i) \oplus P(c_{i-1}) \quad (4)$$

The definition of ADD2 shows that w_i and c_{i-1} are not simultaneously equal to 1 or -1. Every possible values of z_i are shown in Table 3, from which the Property 1 can be extracted.

Table 3: Values of z_i in function of w_i and c_{i-1}

z_i	w_i	c_{i-1}	P (z_i)	P (w_i)	P (c_{i-1})
0	1	-1	0	1	1
1	1	0	1	1	0
-1	-1	0	1	1	0
0	0	0	0	0	0

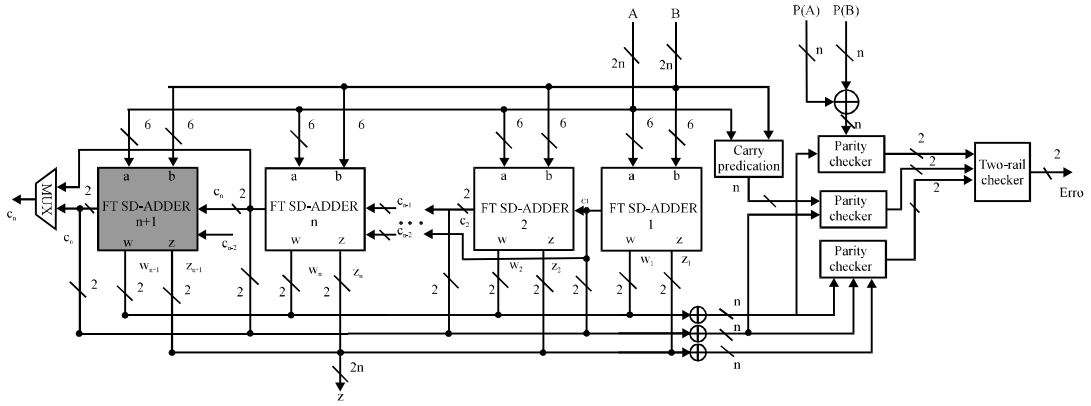

Fig. 2: Fault-tolerant SD adder structure

Table 4: Values of w_i in function of a_i and b_i

w_i	a_i	b_i	P (w_i)	P (a_i)	P (b_i)
±1*	0	1	1	0	1
±1*	0	-1	1	0	1
0	0	0	0	0	0
0	1	1	0	1	1
0	1	-1	0	1	1
0	-1	-1	0	1	1

*Sign depends on the values of a_{i-1} and b_{i-1}.

Property 2:

$$P(w_i) = P(a_i) \oplus P(b_i) \quad (5)$$

For every w_i, the relationship with input a_i and b_i is reported by Table 4 and property 2 can be demonstrated.

FAULT-TOLERANT SD ADDER STRUCTURE

Using (4) and (5) can detect all errors occurs at inputs or outputs or inside the circuit. Note that the wrong of c_i belongs to position i but it affects not the position i but position i+1 and using (4) to detect it may lead to wrong fault localization. In order to overcome this problem, it must be introduced another variable, Predication_P(c_i), to compare with the P(c_i) generating from addition operation:

$$\text{Predication\_P}(c_i) = P(c_i) \quad (6)$$

P(c_i) depends on six variables: f = (a_i[1], a_i[0], b_i[1], b_i[0], a_{i-1}[1], b_{i-1}[1]).

The architecture of our adder is shown in Fig. 2. Fault-tolerant SD-Adder contains ADD1 and ADD2 blocks which realizes two signed digit numbers adding together. This structure uses parity-check to detect errors and is composed of the following blocks:

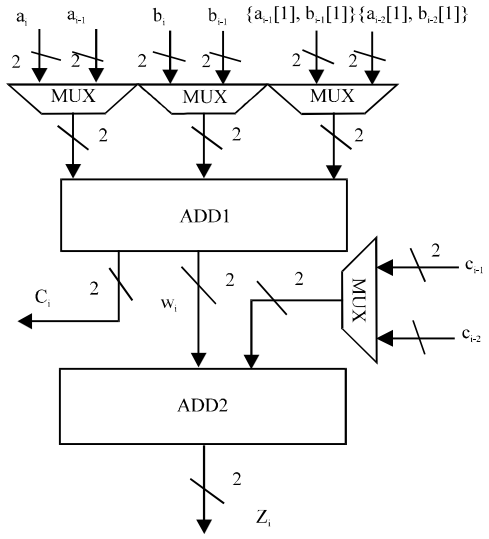

Fig. 3: FT SD-adder structure

- **XOR gates:** Generates the value of P(w_i), P(c_i) and P(z_i)
- **Carry parity prediction:** Generates the value of Prediction_P(c_i)
- **Checker:** Includes three parity checkers and a two-rail checker. Parity checker 1, 2 and 3 implements 5, 4 and 6, respectively and issues an error signal if any unequal occurs of the equation. The Two-rail checker protects the parity checking results
- **FT SD-adder:** Shown as Fig. 3 it contains standard SD adders and multiplexers. The Multiplexers are use to implement operation numbers left-shift when an SD-Adder has a permanent fault which will achieve the purpose of fault-tolerant. The gray block leftmost in Fig. 2 is a redundancy SD-adder which is disconnect when circuit is fault-free

Fault-tolerant generally includes three features: An error detection mechanism to catch the errors, a diagnosis

mechanism to locate the error sites, spare Redundancy configurable Units (RUs) and a reconfiguration mechanism for error correction (Yilmaz *et al*., 2006). The RU here is a 1-bit SD adder. This article provides protection at functional unit level. The control signals of multiplexers are generated by the error information register. When a fault is detected and diagnosed as permanent, the reconfiguration will be taken to eliminate the faulty unit.

The entire fault-tolerant process is shown as Fig. 4. Error detection is performed during the circuit normal operation. While an error is caught, the fault-tolerant flow distinguish transient or permanent fault by a two-stage diagnosis procedure. The detail steps are described in the following paragraphs.

Error detection: Using parity checking to detect a fault, it must avoid that the erroneous result have the same parity value as the correct one. With the chosen coding scheme, any stuck-at fault at the inputs or outputs of a block can only change the digit from 0 to ±1 or vice versa. Assumption that fault affection modifies at least one of the parity of w_i and c_i allows us to detect the fault. If a fault affects a logical cell which fan-out is greater than one, then due to the logic resource sharing in circuit design, it will change more than one bit of the same digit. This case that cannot be detected by parity checker should be avoided because P(-1) is equal to P(1). In order to overcome this problem, the two bits of the same digit should implement independent. In other words, both the block ADD1 and ADD2 should be split into two independently sub-blocks. One sub-block generates the MSB of digit and the other computes the LSB. Designing like this, for example, if a fault occurs inside block ADD1, it will change only one bit of w_i or of c_i or both of them and will certainly be detectable.

Table 5: Fault status register

0	0	… …	0	… …	0	0
n	n-1		i		2	1

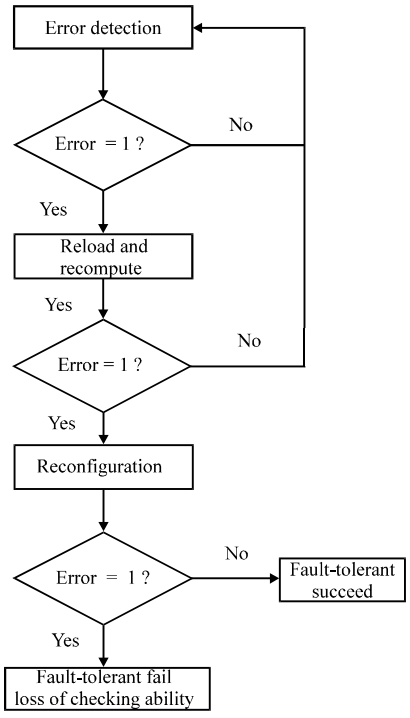

Fig. 4: Fault-tolerant flow chart

Fault diagnosis and localization: Before error recovery, it must diagnose the fault which means the fault site and the fault type should be determined. In order to localization faults, we introduce an extra n-bit register to store the checking results, where n is the length of operands. The bits in the register, labeled as fault bits, identify the current known fault status of each adder unit, 0 for fault-free and 1 for faulty. As Table 5 shows, all bits of the register are initialized to 0. When an error is caught by (4) or (5) or (6), the bit corresponding to it will be set to 1. Note that the localization is at functional level, unnecessary to know which node or wire is failure.

Measured of the duration, the fault is divided into transient fault and permanent fault. A transient fault that occurs on a cycle can result in the diagnosis incorrectly and make a fault-free RU identified as faulty. Although, the probability of this is small, it cannot be ignored. We can improve the diagnosis mechanism to 2-stage scheme. After the first diagnosis is completed, reloading the inputs and running a second diagnosis to compared with the former diagnosis result. If the results are different, it indicates that a transient error has invalidated the diagnosis process. Here, we consider the hold time of transient fault is shorter than one clock cycle. While this does not completely eliminate susceptibility to transient faults, we assume that the diagnosis consecutively affected by transient faults would hardly happen. If the two diagnosis results are equal, the permanent reconfiguration activates.

Error correction: The recovery is implemented though input shift controlled by multiplexers. The values of the multiplexer control signals are generated according to the fault status register and the value of the ith multiplexer control signal relates to the previous i-1 states. Defining C as the multiplexer control signals and b as the value of every bit in fault information register, then the relationship between C and b are as follow:

$$C_i = \sum_{k=1}^{i} b_k \qquad \text{where } 0 \le i \le n \qquad (7)$$

Assumption that a fault occurs at MSB of one input in position i, it will be cause the mismatching of output z_i and z_{i+1}, so the ith and (i+1)th bits are set to 1. According to (7), it can be calculated the former i-1 control values of 0 and the rest of 1. Previous i-1 SD adder units remain unchanged and from the beginning of ith unit each cell in turn backwards one unit of inputs. Note that the carry of (i-1)th unit is connecting to the (i+1)th unit. Through this shift in turn, the faulty unit which is the ith unit here, is discarded in order to achieve the purpose of eliminating failure. After recovery, the same operands are reloaded to the adder. If the error persists after recovery, it can infer that failure occurred in the carry predication or XOR gates or the checker. If this happens, the fault-tolerant structure forfeits checking ability because those blocks are unprotected.

EXPERIMENTAL EVALUATIONS AND DISCUSSION

Experimental results are presented in this section. All adders are coded in Verilog HDL, functionally simulated with MODELSIM SE and synthesized by Synopsys Design compiler. We are particularly interested in comparing among four different implementations:

- Non-redundancy structure. It is the original structure that performs signed digit number addition operation
- TMR (Triple Modular Redundancy) structure. TMR is the traditional redundancy structure in fault tolerance area which contains three identical parallel sources and a voter. The voter accepts the outputs from the sources and delivers the majority vote as its output
- Self-checking structure in the literature (Alavi and Faez, 2007). It uses parity properties for error detection. The error correction needs a specific controller which is not included in the structure
- The structure presented here

The details of the resource consumption and overhead are shown in Table 6. The digits in the table are number of LUT and the percentages are the hardware overhead rate on the basis of implementation 1. From Table 6 it can be known that the overhead of TMR is the largest, because it contains three function blocks and a voter. But from the perspective of testable, TMR can only eliminate errors, can't locate fault. Implementation 3, self-checking structure, has the least overhead. However, compared to TMR and fault tolerance it is short of error correction capability. Base on the Self-checking structure, the literature (Alavi and Faez, 2007) proposed an error correction algorithm. The error correction needs a specific controller which is not included in the circuit. Moreover, if a permanent fault attacks the circuit, the detection and correction procedure will be active every time when inputs change. It would waste too much time so that it is unsuitable for real-time applications. The structure proposed here can locate the fault at functional unit level and perform error correction, it is improved according to the structure of Alavi and Faez (2007) and an error correction mechanism is added. Hence, the resource overhead of this structure certainly exceeds implementation 3. Taking the 64 bit as an example, the overhead of proposed structure are decreased by 110% compared to implementation 2 and increased by 64% compared to implementation 3. The additional overhead ratio of the circuit is shown as Table 7. It can be divided into three parts: reconfiguration, error detection and diagnosis. Due to the adding of an parity checker (parity checker 3) and carry-predicate modules in proposed structure, the overhead of error detection (60%) is a bit larger than implementation 3 (56%). The others are due to the use of multiplexers and spare units in reconfiguration, register and the control logic in diagnosis.

Table 6: Hardware complexity and overhead comparisons

	Hardware overhead (%)			
Technique	8 bit	16 bit	32 bit	64 bit
Non-redundancy	80	160	320	640
TMR	258 (223)	514 (221)	1026 (321)	2050 (320)
Self-checking in Alavi and Faez (2007)	128 (60)	254 (59)	502 (157)	998 (156)
Proposed	180 (125)	356 (123)	707 (221)	1408 (220)

Values in brackets are percentage

Table 7: The extra overhead of 64 bit FT structure

Mechanism	Modules	Hardware overhead (%)
Reconfig.	Muxs and spare unit	55
Error Det.	Checkers and Parity gen. and carry prediction	60
Diagnosis	Reg and Control logic	5
Total	All extra logic	120

Table 8: Delay and overhead comparisons

	Delay overhead (%)			
Technique	8 bit	16 bit	32 bit	64 bit
Non-redundancy	5.05	5.05	5.05	5.05
TMR	5.65 (119)	5.65 (119)	5.65 (119)	5.65 (119)
Self-checking in Alavi and Faez (2007)	6.52 (129)	8.32 (165)	8.59 (170)	9.42 (184)
Proposed	6.44 (128)	7.37 (146)	7.88 (156)	7. 88 (156)

Values in brackets are percentage

Static timing analysis is used to analyze the maximum combinational path delay of different implementations. The detail data are presented in Table 8. The digits in the table are in *ns* and the percentages are the delay overhead rate compared to implementation 1. Because the SD adder is unrelated to the length of operands, the delay of implementation is a constant (5.05 nsec). As Table 8 shows, the TMR has the least time overhead (19%). Since, a voter is appended to original non-redundancy structures and it is just a two AND-or level logic structure, so the time consumption is always a constant. With the operands growing, the delay of Self-checking structure becomes worse. The parity values used for error detection are generated by XOR trees and the depth of XOR trees increases with the growing of operands therefore, the arithmetic speed will be more and more slowly. The delay of proposed structure is between the TMR and Self-checking structure. Although the parity properties are derived from the implementation 3, the parity values are generated by parallel XOR gates therefore, its time delay would not be as worse as implementation 3. Meanwhile, the FT adder units are identical and stable, so the time overhead, as Table 8 shows, tends to be constant (56%) with the operands increased.

CONCLUSION

This study proposes a new fault-tolerant structure of Radix 2 Signed Digit-Based adder. It can not only detect errors but also locate faults and correct errors. According to different types of the fault, error correction takes recomputation or shifting of operation mechanism. Compared to TMR, the hardware overhead of the structure decrease 100% with the capability of fault localization. Compared to self-checking structure Alavi and Faez (2007) although, the hardware overhead of the structure increase about 60%, the delay overhead is much more outstanding. Meanwhile, when the scale of circuit grows to some extent, the hardware and delay overhead percentages will be constant.

REFERENCES

Alavi, S.R. and K. Faez, 2007. Fault localization and full error correction in radix 2 signed digit-based adders. Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, August 22-24, 2007, Victoria, BC, Canada, pp: 214-218.

Avizienis, A., 1961. Signed-digit numbe representations for fast parallel arithmetic. IRE Trans. Electron. Comput., EC-10: 389-400.

Barati, A., M. Dehghan, A. Movaghar and H. Barati, 2008. Improving fault tolerance in ad-hoc networks by using residue number system. J. Applied Sci., 8: 3273-3278.

Cardarilli, G.C., M. Ottavi, S. Pontarelli, M. Re and A. Salsano, 2006. Fault localization, error correction and graceful degradation in radix 2 signed digit-based adders. IEEE Trans. Comput., 55: 534-540.

D'Angelo, S., C. Metra and G. Sechi, 1999. Transient and permanent fault diagnosis for FPGA-based TMR systems. Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, November 1-3, 1999, Albuquerque, NM, USA., pp: 330-338.

Hamidi, H., A. Vafaei and A.H. Monadjemi, 2009. Algorithm based fault tolerant and check pointing for high performance computing systems. J. Applied Sci., 9: 3947-3956.

Hosseinzadeh, M. and K. Navi, 2007. A new moduli set for residue number system in ternary valued logic. J. Applied Sci., 7: 3729-3735.

Johnson, J., W. Howes, M. Wirthlin, D.L. McMurtrey, M. Caffrey, P. Graham and K. Morgan, 2008. Using duplication with compare for on-line error detection in FPGA-based designs. Proceedings of the IEEE Aerospace Conference, March 1-8, 2008, Big Sky, MT, USA., pp: 1-11.

Kharbash, F. and G.M. Chaudhry, 2007. Binary signed digit adder design with error detection capability. Proceedings of the 9th International Symposium on Signal Processing and Its Applications (ISSPA), Feberuary 12-15, 2007, Sharjah, United Arab Emirates, pp: 1-4.

Morozov, A., V.V. Saposhnikov, V.V. Saposhnikov and M. Gossel, 2000. New self-checking circuits by use of Berger-codes. Proceedings of the 6th IEEE International On-Line Testing Workshop, July 3-5, 2000, Palma de Mallorca, Spain, pp: 141-146.

Piuri, V. and E.E. Swartzlander, 1999. Time-shared modular redundancy for fault-tolerant FFT processors. Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, November 1-3, 1999, Albuquerque, NM, USA., pp: 265-273.

Townsend, W.J., J.A. Abraham and E.E. Swartzlander Jr., 2003. Quadruple time redundancy adders. Proceedings of the 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, November 3-5, 2003, MA, USA., pp: 250-256.

Yilmaz, M., D.R. Hower and S. Ozev, 2006. Self-checking and self-diagnosing 32-bit microprocessor multiplier. Proceedings of the IEEE International Test Conference, October 22-27 2006, Santa Clara, CA USA., pp: 1-10.

Yu, F.X., J.R. Liu, Z.L. Huang, H. Luo and Z.M. Lu, 2010. Overview of radiation hardening techniques for IC design. Inform. Technol. J., 6: 1068-1080.

Ziabari, M., A.M. Kassai, A. Ziabari and S.E. Maklavani, 2008. Designing a hamming coder/decoder using QCAs. J. Applied Sci., 8: 2569-2576.