Modular TSC Checkers for Bose-Lin and Bose Codes

X. Kavousianos & D. Nikolos
Dept. of Comp. Eng. & Inf.University of Patras &
Computer Technology Institute, Patras, Greece

Abstract

It is well known that the most common errors in VLSI circuits are unidirectional in nature. Many applications need protection against up to t unidirectional errors, while some other against burst unidirectional errors. Bose-Lin codes are systematic t-unidirectional error detecting codes while Bose codes are burst unidirectional error detecting codes. In this paper we propose a modular method for designing double output checkers for Bose-Lin and Bose codes. The proposed checkers are Totally Self Checking (TSC) with respect to a realistic fault model including stuck-at, transistor stuck-open, transistor stuck-on, resistive bridging faults and breaks. The method is applicable to every code information length and the checkers are very compact and fast.

Introduction

The most common errors in VLSI circuits are unidirectional in nature [1, 2]. The optimal systematic code that can detect all unidirectional errors is the Berger code [3]. However, many applications need a code with detection capability of up to t unidirectional errors. Some of the known systematic t-Unidirectional Error Detecting (t-UED) codes have been presented in [4-6]. In certain applications the unidirectional errors tend to occur in a burst, i.e., a cluster of adjacent bits up to a certain length is affected. Burst Unidirectional Error Detecting (BURD) codes have been proposed by Bose [7] and Blaum [8]. The Blaum code for a specific number r, r≥4, of check bits detects burst unidirectional errors with longer length than the codes given by Bose [7]. However encoding and decoding in the Blaum codes is significantly more complicated than in Bose codes. The suitability of a code for use in a computer system, apart from its ability to cope with errors, heavily depends on the existence of a simple and fast encoder and decoder .

Self Checking Circuits (SCC) provide concurrent error detection and thus can detect transient, intermittent as well as permanent faults. Since transient faults have become increasingly dominant in VLSI circuits [9], providing protection against them has become very important. The reliability of a SCC depends on the ability of its checker to behave correctly despite the possible occurrence of internal faults and this is achieved when the checker satisfies either the Totally Self Checking (TSC) or the Strongly Code Disjoint (SCD) [10] property. In this paper we take into account the TSC property. A checker is TSC if it is self-testing, fault-secure and code disjoint [11].

TSC checkers for Bose-Lin t-UED codes [5] and Bose BUED codes [7] under the single stuck-at fault model were proposed in [12-14]. The single stuck-at fault model is inadequate for the CMOS technology [15]. CMOS is the current dominant technology for manufacturing VLSI circuits, thus new TSC checker designs are required that will take into account a more realistic fault model, including apart from stuck-at, transistor stuck-open, transistor stuck-on, resistive bridging faults and breaks. Such TSC checkers have been recently presented in [16]. However, they are limited to applications with short information length due to their sensitivity in statistical variations of the manufacturing process parameters.

In this paper a modular method for designing TSC checkers for t-UED Bose-Lin codes and BUED Bose codes is proposed. The checkers designed according to this method are TSC with respect to stuck-at, transistor stuck-on, transistor stuck-open, resistive bridging faults and breaks and they are more efficient, with respect to area and speed, than the corresponding already known TSC checkers [5, 7, 12-14]. The checkers presented in [16] require a little less area for their implementation and they are slightly faster than the checkers of this paper but they can be designed only for codes with very small information length. On the contrary the proposed checkers can be designed for every information length.

Throughout this paper the following notations are used :

- $I_1, I_2, \ldots, I_k$ ($C_0, C_1, \ldots, C_{r-1}$) are information (check) bits.
- $W^0(X)$ and $W^1(X)$ denote the number of zeroes and ones respectively of the vector X.
- V_{OHMIN} (V_{OLMAX}) is the minimum HIGH (maximum LOW) voltage at the output of a circuit.
- V_{tn} (V_{tp}) is the threshold voltage of nmos (pmos) transistor
- KP_n (KP_p) is the Spice parameter for $\mu_n \cdot C_{ox}$ ($\mu_p \cdot C_{ox}$).
- W_{qi}/L_{qi} (W_{pi}/L_{pi}) is the ratio of nmos (pmos) transistor i.
- $[x]$ denotes the integer part of x.
- $|A|$ denotes the cardinality of set A.
- $\lceil x \rceil$ denotes the smallest integer greater than or equal to x

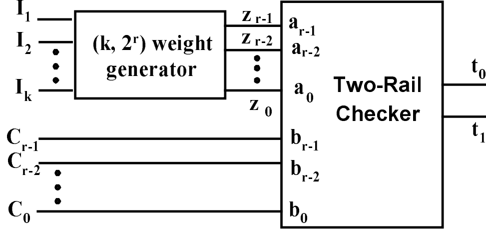

Figure 1. Bose BUED Code checker

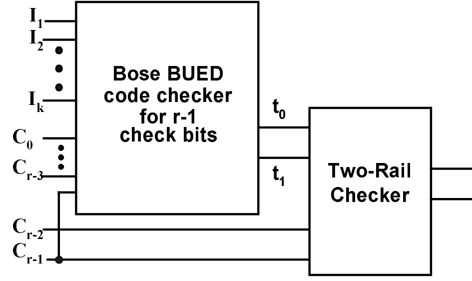

Figure 2. Bose-Lin code checker for r≥4.

II. Preliminaries

In this section we will briefly describe the methods given in [7] and [5] for designing BUED and t-UED codes respectively.

Bose BUED Codes. This code can detect a burst unidirectional error in up to 2^{r-1} bits. The check symbol $CS=(C_0, C_1, ..., C_{r-1})$ is obtained as $CS = W^0(I_1, I_2, \ldots, I_k) \bmod 2^r$ and the bits of the code word are arranged as follows: $I_1 I_2 \ldots I_{k-2^{r-1}} C_{r-1} I_{k-2^{r-1}+1} \ldots I_{k-1} I_k C_{r-2} C_{r-3} \ldots C_0$.

Bose-Lin t-UED Codes. Bose and Lin gave optimal t-UED codes with t=2, 3 and 6 using 2, 3 and 4 bits respectively. The check symbol CS for these codes is derived as follows:

2-UED code with r=2: $CS = W^0(I_1, I_2, \ldots I_k) \bmod 4$,

3-UED code with r=3: $CS = W^0(I_1, I_2, \ldots I_k) \bmod 8$,

6-UED code with r=4: $CS = (W^0(I_1, I_2, \ldots I_k) \bmod 8) + 4$ or equivalently $CS=C_3C_2C_1C_0$ where $C_3=D_2$, $C_2=\overline{D_2}$, $C_1=D_1$, $C_0=D_0$ and $D_2D_1D_0$ is the binary representation of $W^0(I_1, I_2, \ldots I_k) \bmod 8$.

For r ≥ 5, Bose-Lin gave two methods of deriving the t-UED codes [5]. For r≥6 the codes designed by the second method detect more unidirectional errors. However, the encoder and decoder of the codes designed by the first method is simpler and faster than the codes designed by the second method. According to the first method the check symbol is given as $CS = (W^0(I_1, I_2, \ldots I_k) \bmod 2^{r-1}) + 2^{r-2}$ or equivalently $CS= C_{r-1}C_{r-2}C_{r-3}\ldots C_0$ where $C_{r-1}=D_{r-2}$, $C_{r-2}=\overline{D_{r-2}}$, $C_{r-3}=D_{r-3}$, …, $C_0 = D_0$ and $D_{r-2}D_{r-3}\ldots D_0$ is the binary representation of $W^0(I_1, I_2, \ldots I_k) \bmod 2^{r-1}$. This code can detect up to $2^{r-2}+r-2$ unidirectional errors.

III. Design Method

The checker for the Bose BUED code is shown in figure 1. The $(k, 2^r)$-weight-generator receives k inputs I_1, I_2, $\ldots I_k$ and produces the 1's complement of the binary representation of $W^0(I_1, I_2, \ldots, I_k) \bmod 2^r$, that is,

$$Z_{r-1}2^{r-1}+\ldots+Z_0 2^0 = 2^r-1-W^0(I_1, I_2, \ldots, I_k) \bmod 2^r \qquad (1)$$

When the checker receives a Bose BUED code word then $C_{r-1}2^{r-1}+\ldots+C_0 2^0 = W^0(I_1, I_2, \ldots, I_k) \bmod 2^r$ and taking into account the above relation we get $C_{r-1}2^{r-1}+\ldots+C_0 2^0+ Z_{r-1}2^{r-1}+\ldots+Z_0 2^0 = 2^r-1$ that is, $C_{r-1}C_{r-2}\ldots C_0$ and $Z_{r-1}Z_{r-2}\ldots Z_0$ are bitwise complementary, hence the output of the two rail checker will be two-rail encoded.

When the received word is not a Bose BUED code word then $C_{r-1}2^{r-1}+\ldots+C_0 2^0 \neq W^0(I_1,\ldots, I_k) \bmod 2^r$ and from relation 1 we conclude that the two-rail checker receives at least one non two-rail encoded input and hence gives a non two-rail encoded output.

It is evident that the checker of figure 1 is also a checker for the Bose-Lin codes with r=2 and r=3 check bits. The checker for Bose-Lin codes with r ≥ 4 is given in figure 2. For the reasons presented in the introduction for r≥5 we give checkers for the codes designed by the first method. The Two-Rail checkers of figures 1, 2 are designed as proposed in [18] in order to be TSC under a realistic fault model.

In the sequel we will give a method for designing $(k, 2^r)$-weight-generators. Consider a module, hereby denoted (n, 2)-weight-generator, that receives a set of n inputs I_1, I_2,…, I_n and gives one output $Y_0= W^0(I_1,\ldots, I_n) \bmod 2$, as well as a module, denoted (n, 4)-weight-generator, that receives a set of n inputs I_1, I_2,…, I_n and gives two sets of outputs Y_0Y_1 and $D_0D_1...D_{s-1}$, where

$$2\,\overline{Y_1} + \overline{Y_0} = W^0(I_1,\ldots, I_n) \bmod 4, \text{ and}$$

$$W^0(D_0,\ldots, D_{s-1}) = [W^0(I_1,\ldots, I_n)/4].$$

We postpone the design of the (n, 4) and (n, 2)-weight-generators until subsection C. However, we have to note here that due to manufacturability problems, that will be explained in subsection D, the value of n must be less than or equal to 8. In the next subsection we will give a method for designing (k, 4) and (k, 2) weight generators with k>8 using respectively (n, 4) and (n, 2)-weight-generators with n≤8, while in subsection B we will give a method for designing $(k, 2^r)$-weight-generators using (k, 4) and (k, 2)-weight-generators with k>8 and r>2.

A. Design of (k, 4) and (k, 2)-weight-generators using (n, 4) and (n, 2)-weight-generators, n≤k.

Consider that we have k inputs X_1, X_2,…, X_k. We split the set of inputs $\{X_1, X_2,\ldots, X_k\}$ into $c_1=\lceil k/n \rceil$ subsets A , A^1_2, …, $A^1_{c_1}$ such that each subset A^1_i has less than or equal to n elements, or in other words $|A^1_i| \leq n$. It is well

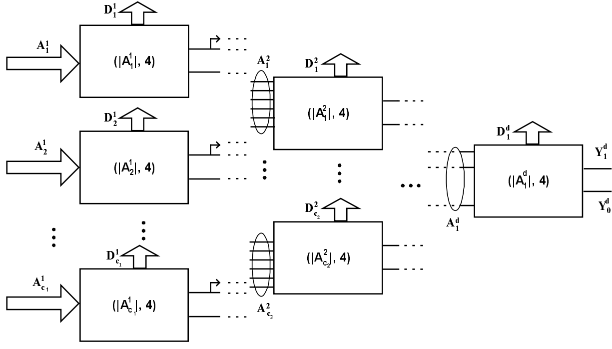
Figure 3. (k, 4)-weight-generator module

known that $W^0(X_1, \ldots, X_k) \bmod 4 = (W^0(A_1^1) \bmod 4 + W^0(A_2^1) \bmod 4 + \ldots + W^0(A_{c_1}^1) \bmod 4) \bmod 4$ (2)

Therefore each term $W^0(A_i^1) \bmod 4$ is calculated by one $M_i^1(|A_i^1|,4)$-weight-generator with outputs $D^{M_i^1}=\{D_0^{M_i^1}, D_1^{M_i^1}, \ldots\}$ and $Y^{M_i^1}=\{Y_1^{M_i^1}, Y_0^{M_i^1}\}$. Then $W^0(A_i^1) \bmod 4 = 2\overline{Y_1^{M_i^1}} + \overline{Y_0^{M_i^1}} = W^0(Y_1^{M_i^1}, Y_1^{M_i^1}, Y_0^{M_i^1})$ and from relation 2 we get $W^0(X_1,\ldots,X_k) \bmod 4 = (W^0(Y_1^{M_1^1}, Y_1^{M_1^1}, Y_0^{M_1^1}) + \ldots + W^0(Y_1^{M_{c_1}^1}, Y_1^{M_{c_1}^1}, Y_0^{M_{c_1}^1})) \bmod 4 =$
$W^0(Y_1^{M_1^1}, Y_1^{M_1^1}, Y_0^{M_1^1}, \ldots, Y_1^{M_{c_1}^1}, Y_1^{M_{c_1}^1}, Y_0^{M_{c_1}^1}) \bmod 4$.

In the same way we split the set $\{Y_1^{M_1^1}, Y_1^{M_1^1}, Y_0^{M_1^1}, \ldots, Y_1^{M_{c_1}^1}, Y_1^{M_{c_1}^1}, Y_0^{M_{c_1}^1}\}$, into $c_2=\lceil 3 \cdot c_1/n \rceil$ subsets A_1^2, A_2^2, …, $A_{c_2}^2$ and we drive the lines of each A_i^2 subset to the inputs of one M_i^2 ($|A_i^2|$, 4)-weight-generator. The above procedure is repeated until we get a set $A^d=\{Y_1^{M_1^{d-1}}, Y_1^{M_1^{d-1}}, Y_0^{M_1^{d-1}}, \ldots\}$ with $|A^d| \le n$. Then $W^0(X_1, \ldots, X_k) \bmod 4 = W^0(A^d) \bmod 4$. Therefore using the $M_1^d(|A_1^d|,4)$-weight-generator, with inputs the lines belonging to the set A_1^d, we get the 1s complement of the binary representation of $W^0(X_1, \ldots, X_k) \bmod 4$. Figure 3 presents the design of the (k, 4)-weight-generator. The (k, 2)-weight-generator can be designed in the same way using (n, 2)-weight-generators.

In the following we use the notation:

- D^t: the union of the $D^{M_i^t}$ sets of all modules M_i^t of level t, that is $D^t=\bigcup_i D^{M_i^t}$, $i \in [1, c_t]$
- D: the union of the D^t sets of all levels 1…t, or equivalently $D=\bigcup_t D^t$, $t \in [1, d]$.

We have proved that the zeroes weight of the values of the outputs belonging to the set D of the (k, 4)-weight-generator, is equal to $[W^0(X_1, \ldots, X_k)/4]$.

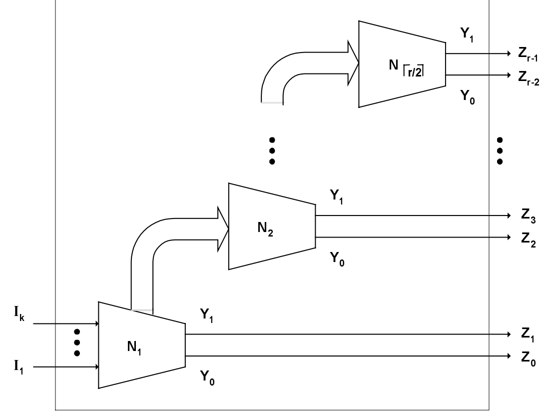

Figure 4. (k, 2^r)-weight-generator, r≥2

B. Design of a (k, 2^r)-weight-generator, r>2, using (k, 4) and (k, 2)-weight-generators.

We can easily see that $W^0(I_1, \ldots, I_k) \bmod 2^r =$
$((4 \cdot [W^0(I_1,\ldots, I_k)/4]) \bmod 2^r + (W^0(I_1,\ldots, I_k) \bmod 4)) \bmod 2^r$
Then taking into account that
$(4x) \bmod 2^a = 4(x \bmod 2^{a-2})$ with x, a∈{1, 2, ...,∞}, a>2 (3)
we get $W^0(I_1,\ldots,I_k) \bmod 2^r =$
$4([W^0(I_1,\ldots,I_k)/4] \bmod 2^{r-2}) + W^0(I_1,\ldots, I_k) \bmod 4$ (4)
The term $W^0(I_1, \ldots, I_k) \bmod 4$ can be calculated by the N_1 (k, 4)-weight-generator, with inputs $I_1, \ldots, I_k$ and outputs $D^{N_1}=\{D_0^{N_1}, D_1^{N_1}, \ldots\}$ and $Y^{N_1}=\{Y_1^{N_1}, Y_0^{N_1}\}$. Then
$2\overline{Y_1^{N_1}} + \overline{Y_0^{N_1}} = W^0(I_1,\ldots,I_k) \bmod 4$ and
$W^0(D^{N_1}) = [W^0(I_1,\ldots, I_k)/4]$ and from relation (4) we get
$W^0(I_1,\ldots,I_k) \bmod 2^r = 4(W^0(D^{N_1}) \bmod 2^{r-2}) + 2\overline{Y_1^{N_1}} + \overline{Y_0^{N_1}}$ (5)
In the same way we get $W^0(D^{N_1}) \bmod 2^{r-2} =$
$((4 \cdot [W^0(D^{N_1})/4]) \bmod 2^{r-2} + (W^0(D^{N_1}) \bmod 4)) \bmod 2^{r-2}$
and from relation 3 we get $W^0(D^{N_1}) \bmod 2^{r-2} =$
$4([W^0(D^{N_1})/4] \bmod 2^{r-4}) + W^0(D^{N_1}) \bmod 4$ (6)
The term $W^0(D^{N_1}) \bmod 4$ can be calculated by the N_2 ($|D^{N_1}|$, 4)-weight-generator, with inputs D^{N_1}, the output set of the N_1 circuit, and outputs $D^{N_2}=\{D_0^{N_2}, D_1^{N_2}, \ldots\}$ and $Y^{N_2}=\{Y_1^{N_2}, Y_0^{N_2}\}$. Then $2\overline{Y_1^{N_2}} + \overline{Y_0^{N_2}} = W^0(D^N) \bmod 4$ and $W^0(D^{N_2}) = [W^0(D^N)/4]$ and taking into account relation 6, relation 5 becomes $W^0(I_1, \ldots, I_k) \bmod 2^r =$
$16(W^0(D^{N_2}) \bmod 2^{r-4}) + 8\overline{Y_1^{N_2}} + 4\overline{Y_0^{N_2}} + 2\overline{Y_1^{N_1}} + \overline{Y_0^{N_1}}$ (7)
Lets assume that r is an even number, then we apply the above procedure r/2 times and at step i we append the N_i circuit which is a ($|D^{N_{i-1}}|$, 4)-weight-generator, with inputs $D^{N_{i-1}}$, and outputs $D^{N_i}=\{D_0^{N_i}, D_1^{N_i}, \ldots\}$ and $Y^{N_i}=\{Y_1^{N_i}, Y_0^{N_i}\}$ where $D^{N_0}=\{I_1, \ldots, I_k\}$. After the ith step relation (7) becomes,

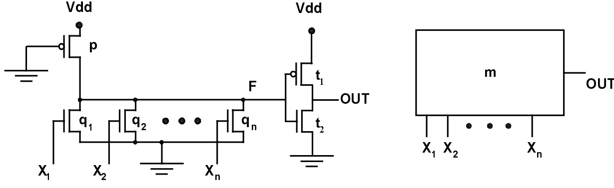

Figure 5. m-ones threshold circuit

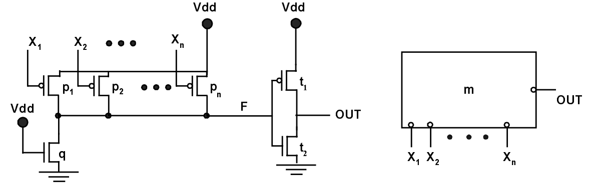

Figure 6. m-zeroes threshold circuit

$W^0(I_1, \ldots, I_k) \bmod 2^r = 4^i(W^0(D^{N_i}) \bmod 2^{r-2i}) + 2^{2(i-1)+1}\overline{Y_1^{N_i}} + 2^{2(i-1)}\overline{Y_0^{N_i}} + \ldots + 8\overline{Y_1^{N_2}} + 4\overline{Y_0^{N_2}} + 2\overline{Y_1^{N_1}} + \overline{Y_0^{N_1}}$ where $i<r/2$.

After the r/2-1 step, that is i=r/2-1 the above relation becomes $W^0(I_1, \ldots, I_k) \bmod 2^r = 4^{r/2-1}(W^0(D^{N_{r/2-1}}) \bmod 4) + 2^{r-3}\overline{Y_1^{N_{r/2-1}}} + 2^{r-4}\overline{Y_0^{N_{r/2-1}}} + \ldots + 4\overline{Y_0^{N_2}} + 2\overline{Y_1^{N_1}} + \overline{Y_0^{N_1}}$

and using at the r/2 step the $N_{r/2}$-weight-generator, we get $W^0(D^{N_{r/2-1}}) \bmod 4 = 2\overline{Y_1^{N_{r/2}}} + \overline{Y_0^{N_{r/2}}}$, so the above relation becomes $W^0(I_1, \ldots, I_k) \bmod 2^r = 2^{r-1}\overline{Y_1^{N_{r/2}}} + 2^{r-2}\overline{Y_0^{N_{r/2}}} + 2^{r-3}\overline{Y_1^{N_{r/2-}}} + 2^{r-4}\overline{Y_0^{N_{r/2-}}} + \ldots + 4\overline{Y_0^{N_2}} + 2\overline{Y_1^{N}} + \overline{Y_0^{N}}$.

It is obvious that the output vector $Y_1^{N_{r/2}}, Y_0^{N_{r/2}}, Y_1^{N_{r/2-1}}, Y_0^{N_{r/2-1}}, \ldots, Y_1^{N_1}, Y_0^{N_1}$ is the 1s complement of the binary representation of $W^0(I_1, \ldots, I_k) \bmod 2^r$.

When r is an odd number, we repeat the procedure $\lceil r/2 \rceil$ times and at the last repetition we use the $N_{\lceil r/2 \rceil}$ weight generator which is a $(\left| D^{N_{\lceil r/2 \rceil}} \right|, 2)$-weight-generator.

Figure 4 presents a $(k, 2^r)$-weight-generator for the case that r is an even number. In the case that r is odd module $N_{\lceil r/2 \rceil}$ is a (k, 2)-weight-generator with only one output.

C. (n, 4) and (n, 2)-weight-generators, with n≤8.

Definition 1. A circuit with n inputs, $X_1, X_2, \ldots, X_n$ and one output, OUT, is called m-ones threshold circuit, if it operates as follows: when $W^1(X_1, X_2, \ldots, X_n) \geq m$ then OUT is High else OUT is Low.

In [17] we have proved that the circuit of figure 5 is an m-ones threshold circuit if the following relations are satisfied: $W_{q_1}/L_{q_1} = W_{q_2}/L_{q_2} = \ldots = W_{q_n}/L_{q_n}$ and

$$(m-1)Q_1 \cdot W_{q_1}/L_{q_1} \leq W_p/L_p \leq mQ_2 \cdot W_{q_1}/L_{q_1} \quad (8)$$

where $Q_1 = KP_n/KP_p(2(V_{dd}-V_{tn})V_{OHMIN} - V_{OHMIN}^2)/(V_{dd}+V_{tp})^2$

$Q_2 = KP_n/KP_p \cdot (2(V_{dd}-V_{tn})V_{OLMAX} - V_{OLMAX}^2)/(V_{dd}+V_{tp})^2$.

The ones-weight, $TW^1(C)$, of an m-ones threshold circuit is by definition equal to m.

Definition 2. A circuit with n inputs, $X_1, X_2, \ldots, X_n$ and one output, OUT, is called m-zeroes threshold circuit, if it operates as follows: when $W^0(X_1, X_2, \ldots, X_n) \geq m$ then OUT is Low else OUT is High.

Following the method given in [17] we can easily see that the circuit of figure 6 is an m-zeroes threshold circuit if the following relations are satisfied:

$W_{p_1}/L_{p_1} = W_{p_2}/L_{p_2} = \cdots = W_{p_n}/L_{p_n}$ and

$$(m-1) \cdot 1/Q_2 \cdot W_{p_1}/L_{p_1} \leq W_q/L_q \leq m \cdot 1/Q_1 \cdot W_{p_1}/L_{p_1} \quad (9)$$

The zeroes-weight, $TW^0(C)$, of an m-zeroes threshold circuit is by definition equal to m.

Definition 3. A circuit C with n+z inputs $X_1, X_2, \ldots, X_n$ and $Y_1, Y_2, \ldots, Y_z$ and one output OUT, is called an $(V_1, V_2, \ldots, V_z)$ aggregate-ones threshold circuit, with $V_1, V_2, \ldots, V_z \in N^*$ if for each vector $Y=(Y_1\ Y_2 \ldots Y_z)$ the circuit operates as follows: OUT is High when $W^1(X_1,X_2,\ldots, X_n) \geq \overline{Y_1}V_1 + \overline{Y_2}V_2 + \cdots + \overline{Y_z}V_z$ else OUT is Low.

The sum $\overline{Y_1}V_1 + \overline{Y_2}V_2 + \cdots + \overline{Y_z}V_z$ is called the aggregate-ones-weight, $AW_C^1(V, Y)$, of the circuit for the vector Y. It can be easily proved that the circuit C of Figure 7 is an $(V_1, V_2, \ldots, V_z)$ aggregate-ones threshold circuit, if the following relations are satisfied:

$W_{q_1}/L_{q_1} = W_{q_2}/L_{q_2} = \cdots = W_{q_n}/L_{q_n}$ and for i=1, …, z

$$(V_i - 1/z) \cdot W_{q_1}/L_{q_1} \cdot Q_1 \leq W_{p_i}/L_{p_i} \leq V_i \cdot W_{q_1}/L_{q_1} \cdot Q_2 \quad (10)$$

Figures 5, 6 and 7 show also the symbols for an m-ones threshold circuit, an m-zeroes threshold circuit, and a $(V_1, V_2, \ldots, V_z)$ aggregate-ones threshold circuit respectively, that will be used throughout this paper.

The (n, 4)-weight-generator is shown in figure 8, where $m_1=[n/4]$, $m_2=[n/2]-[n/4]$, $m_3=[n/2]$, $m_4=n-[n/2]$. From definitions 1, 2 it is very easy to see that $T_i=0$ when $W^0(I_1, \ldots, I_n) \geq i$, with $i \in [1, n]$. From definition 3 we have that $Y_1=0$ if and only if

$\overline{Gnd} \cdot 1 + \overline{T_4} \cdot 1 + \overline{T_8} \cdot 1 + \cdots + \overline{T_{4m_1}} \cdot 1 \leq W^1(\overline{T_2}, \overline{T_6}, \cdots, \overline{T_{4m_2-2}})$

or equivalently $(\overline{T_2} - \overline{T_4}) + (\overline{T_6} - \overline{T_8}) + \ldots + (\overline{T_{4i-2}} - \overline{T_{4i}}) + \ldots > 0$.

In Tables 1 and 2 for any value of $W^0(I_1,\ldots, I_n)$ we give the values of T_i (as they are derived from definitions 1 and 2). From Table 1 we get $Y_1=0$ if and only if

$$W^0(I_1,\ldots,I_n) \in \{2, 3\} \cup \{6, 7\} \cup \ldots \cup \{4i-2, 4i-1\} \cup \ldots \quad (11)$$

In the same way, $Y_0=0$ if and only if

$(\overline{T_1} - \overline{T_2}) + (\overline{T_3} - \overline{T_4}) + \ldots + (\overline{T_{2i-1}} - \overline{T_{2i}}) + \ldots > 0$.

Now from Table 2 we get $Y_0=0$ if and only if

$$W^0(I_1, \ldots, I_n) \in \{1, 3, 5, 7, \ldots, 2i-1, \ldots\} \quad (12)$$

Table 3 summarises relations 11, 12. We get $2\overline{Y_1} + \overline{Y_0} = W^0(I_1,\ldots,I_n) \bmod 4$ hence (Y_1, Y_0) is the 1's complement of the binary representation of $W^0(I_1,\ldots,I_n) \bmod 4$.

Theorem 2. The zeroes weight of the set $\{T_4, T_8, T_{12},$

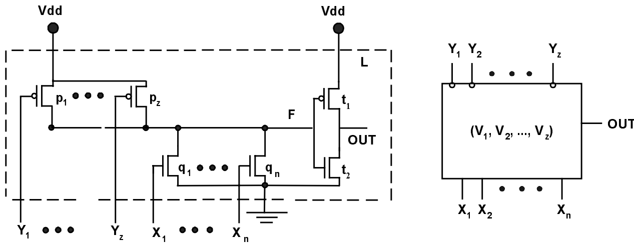

Figure 7. $(V_1, V_2, \ldots, V_z)$ aggreg.-ones threshold circuit

Table 1.

$W^0(I_1, \ldots, I_n)$	T_{4i-2}	T_{4i}	$(\overline{T_{4i-2}} - \overline{T_{4i}})$
[0, 4·i-2)	1	1	0
[4·i-2, 4·i)	0	1	1
[4·i, n]	0	0	0

Table 2.

$W^0(I_1, \ldots, I_n)$	T_{2i-1}	T_{2i}	$(\overline{T_{2i-1}} - \overline{T_{2i}})$
[0, 2·i-1)	1	1	0
2·i-1	0	1	1
[2·i, n]	0	0	0

Table 3.

$W^0(I_1 \ldots I_n)$	Y_1	Y_0
0, 4, 8, ...	1	1
1, 5, 9, ...	1	0
2, 6 ,10, ...	0	1
3, 7, 11, ...	0	0

$\ldots, T_{4m_1}\}$ of the (n, 4)-weight-generator of Figure 8 is equal to $[W^0(I_1, \ldots, I_n)/4]$.

*P***roof.** When $W^0(I_1, \ldots, I_n) \geq 4\cdot i$ then $T_{4i} = 0$ else $T_{4i} = 1$, thus we have $T_{4i} = 0$ and $T_{4i+1} = 1$ if and only if $4\cdot i \leq W^0(I_1,\ldots,I_n) < 4\cdot(i+1)$ or equivalently $[W^0(I_1,\ldots,I_n)/4]=i$. We also have that $T_{4j} = 0$ for all $j\in[1, i]$ and $T_{4j} = 1$ for $j>i$, hence $W^0\{T_4, T_8, T_{12}, \ldots, T_{4m_1}\}=i$ therefore,

$W^0\{T_4, T_8, T_{12}, \ldots, T_{4m_1}\} = [W^0(I_1, \ldots, I_n)/4]$ ■

Based on theorem 2, we see that the set $D=\{D_0D_1 \ldots D_{s-1}\}$ of outputs of the (n, 4)-weight-generator is the set $\{T_4, T_8, \ldots, T_{4m_1}\}$. The (n, 2)-weight-generator is derived from the (n, 4)-weight-generator if we remove module A.

D. On the manufacturability of the (k,2^r)-weight-generator

In subsections A and B we have seen that the $(k, 2^r)$-weight-generator can be designed using as building blocks the (n, 4) and (n, 2)-weight-generators with $n\leq 8$. Therefore the manufacturability of a $(k, 2^r)$-weight-generator depends on the manufacturability of the (n, 2) and (n, 4)-weight-generator. If these circuits can be manufactured, then the $(k, 2^r)$-weight-generator can be manufactured for every value of k and r. The (n, 2) and (n, 4)-weight-generators are ratioed circuits. A problem of a ratioed circuit is that its correct operation depends on the conductance values of nmos and pmos transistors as well as other circuit parameter's values. It is well known that fluctuations in integrated circuit manufacturing processes cause deviations on the actual values of the parameters from their nominal values. Designing the (n, 2) and (n, 4)-weight-generators, we choose the values of W_p, L_p, W_q, L_q, W_{p_i} and L_{p_i} so that the values of W_p/L_p, W_q/L_q and W_{p_i}/L_{p_i} to be in the middle of the ranges given by relations (8), (9) and (10) respectively. Then due to statistical variations of the device characteristics the range can be shortened or shifted to the left or to the right but the value of the ratio will remain within the range, therefore the manufactured IC will operate correctly. As the value of n becomes greater, the ranges defined by relations (8), (9) and (10) become shorter and the circuit is more sensitive

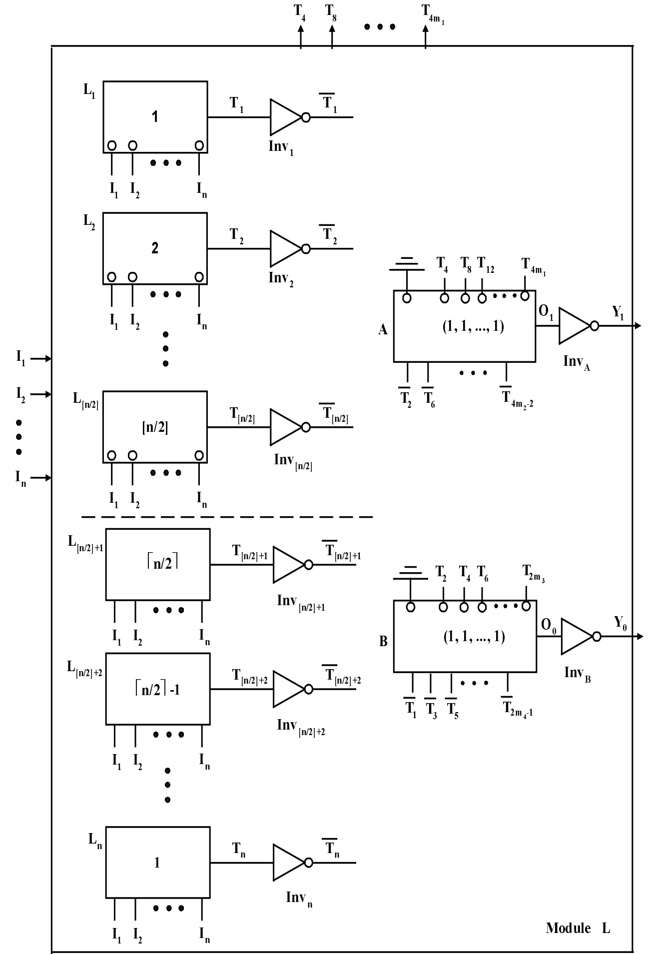

Figure 8. (n, 4)-weight-generator module

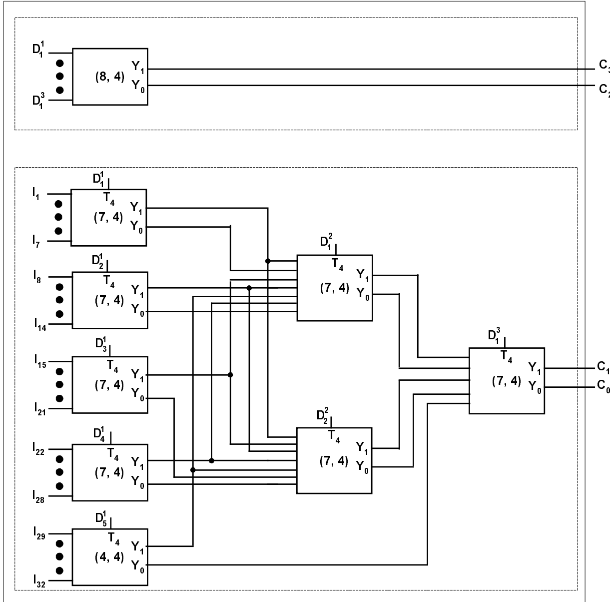

Figure 9. (32, 16)-weight-generator.

to the statistical variations of the device parameters. This is the reason that we have kept $n \leq 8$. The (n, 2) and (n, 4)-weight-generators of figure 8 consist of threshold circuits with weights less than or equal to $[n/2]=\lceil n/2 \rceil=4$.

We have run Monte Carlo simulations for the (n, 2) and (n, 4)-weight-generator with $n \leq 8$ as well as for $(k, 2^r)$-weight-generators with $k \leq 64$ and $r \leq 3$ for circuit parameter deviations up to 10% and verified the correct operation of the circuits. Apart from the above we have verified that for all cases the noise margins are above 0.7 volts.

E. General design guidelines.

In figure 3 practically there are many ways (without taking into account the ordering) to split the f inputs of level i of figure 3, into groups A_1^i, A_2^i, … of inputs with $|A_j^i| \leq n$ for j=1, 2, ... For testability issues the outputs Y_1 and Y_1 (we count it twice) of the ($|A_j^i|$, 4)-weight-generator must be assigned to different sets $A_{j_1}^{i+1}$, $A_{j_2}^{i+1}$, with $j_1 \neq j_2$. This is not possible at the last level of each tree where we have a single set. For that reason we modify slightly the (n, 4)-weight-generator of the last level as it is shown in Appendix.

For testability issues in figure 4 all N_j, with $j < \lceil r/2 \rceil$, must be designed using (n, 4)-weight-generators with $n \leq 7$ while the last $N_{\lceil r/2 \rceil}$ module can be designed using (n, 2) or (n, 4)-weight-generators with $n \leq 8$.

With respect to the speed of a (k, 4)-weight-generator we have to note that the (a, 4)-weight-generator is faster than the (b, 4)-weight-generator if a<b, while among two (k, 4)-weight-generators, the one with the smaller number of levels is faster. In Figure 9 we present the (32, 16) weight generator.

IV. Testability Analysis

All the stuck-at, transistor stuck-on and transistor stuck-open faults of the (n, 4) weight generator are detectable except of the following:

1. q or p or t_1 transistor of module L_i, $i \in [1, n]$ stuck-on.
2. pmos transistor of Inv_i or Inv_A or Inv_B stuck-on.
3. p_1 or t_1 transistor of modules A or B stuck-on.

After the occurrence of any one of the undetected faults, the checker remains code disjoint. Furthermore if they are followed by one of the other considered faults, the resulting fault is detectable. All the inverters are designed with n-dominate logic.

We have to note here that the testability analysis of the modified (n, 4)-weight-generator (Appendix) is the same with the testability analysis of the normal (n, 4)-weight-generator except the undetected stuck open fault on transistor $p_{i,j}$ of module L_1 and stuck at 1 fault on input $I_{i,j}$ of module L_1. This modified module is used only at the last level of each tree.

The self-checking capability with respect to resistive bridging faults (RBFs) between two transistor terminals or between two inputs has been evaluated with extensive circuit-level simulations. All RBFs with connecting resistance $R \in [0,R_{max}]$ are detected, where R_{max} depends on the sizing of the transistors and the information length. The proposed checkers are also Self Testing for all break faults on device terminals.

All the (n, 2) and (n, 4)-weight-generator modules of a $(k, 2^r)$-weight-generator, receive all possible binary combinations at their inputs, therefore receive their test set. Therefore, any detectable fault of any one of the (n, 2) and (n, 4) modules is detected. This was also verified with the use of a specific simulator developed in our lab.

Since $n \leq 8$, we have $[W^0(I_1, \ldots, I_8)/4] = W^0(D_0, D_1)$. According to the design of the $(k, 2^r)$-weight-generator with r>2, the ith tree receives as inputs the D output set of the $(i\text{-}1)^{th}$ tree. The problem of this approach is that the pair (D_0, D_1) does not receive the value (1, 0) so there are some undetected faults in the ith tree. For that reason we limit the value of n to be less than or equal to 7 for all trees, except the last one. In that way each (n, 4)-weight-generator produces only the $\{D_0\}$ output which receives all possible values (0 and 1). We have verified with extensive simulations that if we apply the above rule, each tree N_i (i>1) (figure 4) receives its test set and it is completely tested, therefore the $(k, 2^r)$-weight-generator is completely tested for all values of k and r.

Table 4. Improvement of proposed over Piestrak[14].

(k, r)	Delay	Area
(8, 4)	25,8%	62%
(16, 4)	11,8%	67,8%
(32, 8)	14%	66%

V. Comparisons and Conclusions.

In this paper we presented a novel method for designing modular TSC checkers for BUED Bose and t-UED Bose-Lin codes. The proposed checkers are TSC under a realistic fault model including stuck-at, transistor stuck-on, transistor stuck-open, resistive bridging faults and breaks and they are applicable for all possible values of k and r. The corresponding already known from the open literature checkers [12-14] have been proved to be TSC under the stuck-at fault model. Among them, the checkers given in [14] are the most efficient with respect to the area required for their implementation and the delay. TSC checkers under a realistic fault model were recently proposed in [16]. They are the most efficient with respect to area and speed among all the already known checkers, but their applicability is limited to short information lengths due to their sensitivity to statistical variations of the circuit parameters during the manufacturing process. To this end we compare our checkers to the checkers given in [14]. We have implemented some of the proposed TSC checkers as well as the corresponding checkers given in [14] with λ=1μm technology. The comparison results are given in Table 4. We have to note that the area has been estimated as the sum of WxL of the transistors, that is, the routing has not been taken into account. We can easily see that the routing in the proposed design is less than the routing required for the implementation of the checkers given in [14]. From Table 4 we can easily see that the proposed checkers are significantly more efficient, with respect to the area required for their implementation and the speed than the checkers given in [14].

Appendix.

The (n, 4)-weight-generator module of figure 8 is designed to receive n independent inputs. When some of the inputs are not independent, for example I_i, I_j are driven by the same line, lets say $I_{i,j}$, then some testability problems may arise. For that reason we modify the L_1, L_2, …L_n modules in the following way: instead of having two transistors q_i, q_j (figure 5) and p_i, p_j (figure 6) driven by the input $I_{i,j}$, we use one $q_{i,j}$ and one $p_{i,j}$ respectively, with

$$W_{q_{i,j}} / L_{q_{i,j}} = W_{q_i} / L_{q_i} + W_{q_j} / L_{q_j} \text{ and}$$

$$W_{p_{i,j}} / L_{p_{i,j}} = W_{p_i} / L_{p_i} + W_{p_j} / L_{p_j}$$

driven by the input $I_{i,j}$. In that way we modify all the modules L_1, L_2, …L_n and the functionality of the circuit remains the same.

References.

[1] D. K. Pradhan and J. I. Stiffler, "Error correcting codes and self-checking circuits in fault tolerant computers", Computer, pp.27-37, Mar. 1980.

[2] E. Fujiwara and D. K. Pradhan, "Error-Control Coding in Computers", Computer , vol. 23, pp. 63-72, July 1990.

[3] J. M. Berger, "A note on error detection codes for asymmetric binary channels", Inform. Contr., vol. 4, pp 68-73 , Mar. 1961.

[4] H. Dong , " Modified Berger codes for detection of unidirectional errors", IEEE Trans. Comput., vol. C-33, pp. 572-575, June 1984.

[5] B. Bose and D. J. Lin, "Systematic unidirectional error-detecting codes," *IEEE Trans. Comput*., vol. C-34, pp. 1026-1032, Nov. 1985.

[6] N. K. Jha and M. B. Vora, " A systematic code for detecting t-unidirectional errors," in Proc. Int. Symp. Fault-Tolerant Comput., Pittsburgh, PA, pp. 96-101, June 1987.

[7] B. Bose, "Burst unidirectional error detecting codes," *IEEE Trans. Comput*., vol. C-35, pp. 350-353, Apr. 1986.

[8] M. Blaum, "On systematic burst unidirectional error detecting codes", IEEE Trans. Comput., vol. 37, pp. 453-457, Apr. 1988.

[9] Y. Savaria, N. C. Rumin, J. F. Hayes, and V. K. Agarwal, " Soft-error filtering: A solution to the reliability problem of future VLSI digital circuits," Proc. IEEE, vol. 74, no. 5, pp. 669-683, May 1986.

[10] M. Nicolaidis and B. Courtois, "Strongly Code Disjoint Checkers", IEEE Trans. Comp., Vol. 37, pp 751-756, June 1988.

[11] T.R.N. Rao, E. Fujiwara, "Error-Control coding for computer systems", Prentice-Hall, Englew. Cliffs, NJ, 1989.

[12] N. K. Jha, " Totally self-checking checker designs for Bose-Lin, Bose and Blaum codes", IEEE Trans Comp.-Aided Des., vol. CAD-10, pp. 136-143, Feb. 1991.

[13] S. J. Piestrak, "Efficient encoding/decoding circuitry for systematic unidirectional error-detecting codes," in Proc. 5th Int. Conf. Fault-Tolerant Computing Systems, FRG, Sept. 25-27, 1991, Springer-Verlag, Berlin, pp. 181-193.

[14] S. J. Piestrak, "Design of encoders and self-testing checkers for some systematic unidirectional error detecting codes", Proc. of 1997 IEEE Int. Symposium on Defect and Fault Tolerance in VLSI Systems, October 20-22, 1997 Paris, France, pp. 119-127.

[15] J. Shen,W. Maly and F. Ferguson, "Inductive Fault Analysis of MOS Intergrated Circuits", IEEE Design & Test of Computers, pp.26-33, December 1985.

[16] X. Kavousianos & D. Nikolos, "Novel TSC Checkers for Bose-Lin and Bose Codes" 3rd IEEE Int. On-Line Testing Workshop, July 6-8, 1998, Capri, Italy, pp. 172-176.

[17] X. Kavousianos, D. Nikolos "Self-Exercising, Self-Testing k-order Comparators" 15th IEEE VLSI Test Symposium, pp. 216-221, April 27-May 1, 1997 Monterey California.

[18] J. C. Lo, "A Novel Area-Time Efficient Static CMOS TSC Comparator", IEEE Journal of Solid-State Circuits, Vol. 28, No. 2, pp. 165-168, Feb. 1993.