On the Efficiency of Berger Codes Against Error Injection Attacks on Parallel Asynchronous Communication Channels

Ariel Burg and Osnat Keren
Faculty of Engineering, Bar-Ilan University, Ramat-Gan, Israel

ABSTRACT The paper deals with the efficiency of Berger codes under error injection attacks. Berger codes are unordered codes; that is, they are insensitive to propagation delays of individual bits in a codeword. Berger codes are commonly used (in concatenation with linear codes) to provide reliable communication over parallel asynchronous channels. The paper examines the robustness of Berger codes under error injection attacks on the communication channels. It is shown that if the number of information bits $k$ is not equal to $2^r - 1$, where $r$ is the number of redundancy bits, then the Berger code is robust; that is, any injected error can be detected.

KEYWORDS asynchronous communication, Berger code, fault analysis attacks, robust codes, security, undetected error probability

1. INTRODUCTION

Cryptographic devices have physically observable properties (e.g., EM radiation, temperature, power consumption). An adversary can use these properties to extract secret information from the hardware. Such acts are called noninvasive side channel attacks (Bar-El, Choukri, Naccache, Tunstall, & Whelan, 2006). A Differential Power Analysis (DPA) attack is a noninvasive attack in which the adversary extracts information by analyzing the power consumption of the device. One of the countermeasures against DPA attacks is use of multiple independent clock domains, which consume uncorrelated power. A Globally Asynchronous Locally Synchronous (GALS) system has multiple independent clock domains; it consists of locally synchronous islands that communicate using an asynchronous protocol (Wolinski & Belhadj, 1994).

The data transferred over asynchronous global communication parallel links without acknowledgment must be coded in a way that, despite possible propagation delays of individual bits, the receiver can deduce the correct code words sent by the sender in their original order. Consequently, the code must be insensitive to propagation delays of individual bits in a codeword. Unordered codes, such as the Berger code, are codes of this class (Berger, 1961;
Bose, 1984; Blaum & Bruck, 1992; Bose & Lin, 1985). They can indicate the receiver that a code word can be formed from a subset of the received signals.

Usually Berger codes are used in the context of reliable computing, either for detecting unidirectional errors caused by stuck-at faults in the monotone parts of a circuit or for detecting skews in asynchronous communication. In this paper, we study the codes in the context of secure communication and analyze their robustness against active side channel attacks and, in particular, against fault injection attacks which tamper the asynchronous transmission.

The paper is organized as follows: an introduction to parallel asynchronous communication is given in the next section. Necessary conditions for a code to provide reliable and secure communication are presented in section 3. Mathematical background on the Berger code and its properties is given in section 4. The robustness of the Berger code against attacks on the asynchronous transmission is analyzed in section 5. Section 6 concludes the paper.

2. TRANSMISSION OVER PARALLEL ASYNCHRONOUS CHANNELS

The communication between two components may be held over synchronous buses or over parallel asynchronous communication channels (links). The transmission over asynchronous communication channels is similar to rolling marbles that correspond to a logical 1 over parallel tracks. Although the marbles are sent simultaneously, they are received randomly and at different instants due to variations in their propagation time (Verhoeff, 1988). The delays in parallel asynchronous pipelined communication without acknowledgment cause skews. A skew happens when a certain number of marbles from the second message arrive before reception of the current message has been completed. Therefore, skews are considered as additive noncorrelated symmetric errors. Transitions from the second message that arrive before reception of the current message are considered as errors in the up direction, and the missing transitions are considered as errors in the down direction.

The difficulty of using parallel asynchronous communication without acknowledgement is demonstrated in Figure 1. Suppose the sender at the source sends the binary word \( x = (x_{15}, x_0) = (001010010111001) \) followed by the word \( y = (y_{15}, \ldots, y_0) = (000101000111000) \) over 16 parallel tracks. Assume that due to random delays, four marbles (logical 1's) from \( y \) (marbles 3, 5, 10, 12) have arrived to the receiver before the arrival of all the marbles from \( x \). Denote by \( z = (z_{15}, \ldots, z_1, z_0) \) a binary vector, where, \( z_i = 1 \) if a marble has arrived on track number \( i \), and it equals zero otherwise. For example, after the arrival of a second marble on track 5 (at time \( t_1 \)), \( z = (0000000100110001) \). Note that at this point, since the code is unordered, the receiver knows that a skew has happened. At time \( t_2 \), the vector \( z \) is \( (0011100100111001) \). Now, the receiver, who knows that a skew has happened, has to decide whether \( z \) covers a complete codeword, and if so, which. Namely, the receiver has to deduce the correct words that were sent in their original order from the order of arrival.

An additive symmetric error is defined by \( e = x \oplus z \), where \( \oplus \) stands for bit-by-bit XOR operation. The error reflects the propagation delays. Moreover, since \( z \) changes in time, so does the error. In our example,

\[
e(t_1) = (001010000100000),
\]

\[
e(t_2) = (000100000100100),
\]

\[
e(t_3) = (0001010000001000).
\]

3. NECESSARY CONDITIONS FOR RELIABLE AND SECURE PARALLEL ASYNCHRONOUS COMMUNICATION

A reliable and secure communication between components in GALS systems is one of the primary requirements in systems design. Reliability usually refers to immunity against random delays (small skews). In contrast, security refers to the immunity against malicious attacks on the hardware that may cause skews of arbitrary length.

In this paper we assume that the adversary is “weak”; that is, the induced error is uncorrelated with the correct word. This assumption is acceptable when the communication between two components is asynchronous, as it is difficult to synchronize the attack so to distort a specific word. Although the adversary is weak, we assume that a) he knows the set of (correct) words, and b) he can introduce delays on any number of tracks he chooses.
The set of correct (i.e., undistorted) words form a code. In what follows we review necessary conditions for codes that provide reliable and secure parallel asynchronous communication.

3.1. Necessary Conditions for Reliable Communication

The reliability of data transmitted over an asynchronous communication channel depends on the skew detecting and correcting capability of the code. A skew detecting code is defined as follows:

**Definition 1 (Skew detecting code; Blaum & Bruck, 1992):** Let \( t_1 \) and \( t_2 \) be two positive integers. We say that a code \( C \) is \((t_1, t_2)\)-skew-detecting if, for any pair of code words \( x, y \in C \) such that codeword \( x \) is transmitted followed by codeword \( y \), and the skew between \( x \) and \( y \) is limited by the following two conditions:

1. at most \( t_1 \) marbles may still be missing from \( x \) when a marble from \( y \) arrives; and
2. at most \( t_2 \) marbles from \( y \) may arrive before all the marbles from \( x \) have arrived;

then, \( C \) will correctly decode \( x \) when there is no skew between \( x \) and \( y \), and will detect at a certain point the presence of skew provided it does not exceed the \( t_1 \) and \( t_2 \) constraints.

A necessary condition for a code to detect a skew is that the code has to be unordered. Namely,

**Definition 2 (Unordered code):** Let \( u = (u_{n-1}, \ldots, u_0) \) and \( v = (v_{n-1}, \ldots, v_0) \) be two binary vectors of length \( n \). We say that \( u \) is covered by \( v \), or \( u \preceq v \), if \( u_i \leq v_i \) for all \( 0 \leq i \leq n-1 \). Two vectors \( u \) and \( v \) are unordered if \( u \not\preceq v \) and \( v \not\preceq u \), that is, the set of non-zero coordinates of one vector does not contain the set of nonzero coordinates of the other. A code \( C \) is said to be unordered if each pair of its code words is unordered.

If there is no skew, the delays are manifested as unidirectional errors. Therefore, for reliable asynchronous communication over skew-less tracks \((t_1 = t_2 = 0)\), one must use unordered codes — unordered codes can detect all unidirectional errors. However, this requirement is not sufficient in the presence of skews. Since the delays can be represented as symmetric errors, the codes must have sufficient error correcting capability. In Blaum and Bruck (1992), the authors presented the following necessary and sufficient conditions for a code to be a skew detecting code.

**Theorem 1.** (Blaum & Bruck, 1992). Let \( t_1 \) and \( t_2 \) be two positive integers. Let \( C \) be an unordered code with minimum distance \( d \geq t_1 + t_2 + 1 \). Then, \( C \) is \((t_1, t_2)\)-skew-detecting.

Unordered codes that detect (or correct) propagation delays and at the same time detect (or correct) a small number of errors caused by faults in the hardware were presented, for example, in Blaum and Bruck (1992, 2000), Bose (1984), Bose and Lin (1985), and Lala (2000).

In normal operation, that is, when the device is not under attack, the skew parameters \((t_1, t_2)\) are relatively small. However, when the device is under attack, the attacker can introduce any error he chooses, and hence, all the possible \((t_1, t_2)\) pairs have to be considered. In the next section we discuss necessary conditions for secure communication.

3.2. Necessary Conditions for Secure Communication

A skew can be represented as a symmetric error \( e \). If the skew is within the \((t_1, t_2)\) constraint, then the multiplicity (i.e. Hamming weight) of \( e \) is smaller or equal to \( t_1 + t_2 \). A \((t_1, t_2)\)-skew-detecting code will correctly decode the transmitted code word when there is no skew (i.e., when \( e = 0 \)), and will always detect at a certain point the presence of skew provided it does not exceed the \( t_1 \) and \( t_2 \) constraints. However, when a skew exceeds these constrains, in some cases the skew will be always detected, in other cases the skew will never be detected, or it will be undetected (masked) with a nonzero probability. Note that a skew represented by an error vector \( e \) is said to be masked by a code word \( c \) if the received word \( c \oplus e \) is a code word.

A code can be characterized by its error masking probability:

**Definition 3 (Error masking probability; Karpovsky & Taubin, 2004):** Let \( C \) be a binary code whose codewords are uniformly distributed. The error masking probability of the code, \( Q(e) \), is the probability that an error vector \( e \) maps a codeword to another codeword:

\[
Q(e) = \frac{|\{c \mid c, c \oplus e \in C\}|}{|C|}.
\]

The detection kernel of a code is denoted by \( K_d \). The kernel consists of all the error vectors that are never detected, that is,

\[
K_d = \{ e \mid Q(e) = 1 \}.
\]
Note that $|K_d| \geq 1$ for any given code $C$, since the all-zero error vector is always masked, that is, $Q(0) = 1$.

We distinguish between three types of codes: robust codes, partially robust codes, and codes whose detection kernel consists of all the code words. Codes of the third type are considered as the worst codes in terms of security. Robust and partially robust codes are defined as follows:

**Definition 4 (A $(t_1, t_2)$-skew-detecting robust code):** A $(t_1, t_2)$-skew-detecting code is called robust if any skew that does not exceed the $t_1$ and $t_2$ constraint is always detected, and any skew that exceeds this constraint is detected with nonzero probability; that is, for all $e$ there exists at least one codeword that can detect its presence ($|K_d| = 1$).

**Definition 5 (A $(t_1, t_2)$-skew-detecting partially robust code):** A partially robust code $a$ is a $(t_1, t_2)$-skew-detecting code whose detection kernel is of size $1 < K_d < |C|$. Clearly a robust code provides the maximal security level since it can detect any attack. The maximal error masking probability of a given robust code of length $n$ is lower bounded by (Karpovsky & Nagvajara, 1989):

$$\max_{e \neq 0} Q(e) \geq \max \left( \frac{2}{|C|}, \frac{|C|}{2^n} \right).$$

**Definition 6 (Optimum code):** An optimum code achieves Eq. (1) with equality.

Robust optimum codes and partially robust codes were presented in Engelberg and Keren (2011), Etzion and Vardy (1994), Karpovsky and Taubin (2004), Phelps (1983), and Wang, Karpovsky, and Kulikowski (2010). Unfortunately, all these codes are not unordered codes, and therefore, none of them can be used for reliable parallel asynchronous communication.

In this paper we examine the error masking probability of Berger codes and show that most of the codes are robust, however, they are not optimum.

### 4. THE BERGER CODE

Let $u$ be a $k$-bit binary vector. Denote by $V(u)$ the value of $u$, and by $Z_u$ the number of zeros in $u$. Namely,

$$V(u) = \sum_{i=0}^{k-1} u_i 2^i,$$

$$Z_u = k - wt_H(u),$$

where $wt_H(u)$ is the Hamming weight of $u$.

**Definition 7 (Berger code):** A Berger code $C(k, r)$ of dimension $k$ and $r = \lceil \log(k + 1) \rceil$ redundancy bits consists of the code words

$$C = \{ (x, w) | V(w) = Z_x \},$$

where $x$ denotes the $k$-bits information word and $w$ denotes the $r$-bits redundancy word.

**Example 2:** Consider a $C(8, 4)$ Berger code. The information word $x = 01001100$ has $Z_x = 5$, and therefore, $w = 0101$.

Berger codes are nonlinear unordered codes (i.e., any pair of code words is unordered). Therefore, they can detect all unidirectional errors (Lala, 2000). This property was used in Bose (1984), Blaum and Bruck (1992), and Bose and Lin (1985) to construct Berger-based skew detecting codes for asynchronous communication.

**Definition 8 (Minimum distance of a code; MacWilliams & Sloane, 1977):** The minimum distance of a code $C$ is the minimum Hamming distance between its code words, that is, the minimum number of places where any pair of codewords differ:

$$d = \min_{u, v \in C, u \neq v} \text{dist}(u, v) = \min_{u, v \in C, u \neq v} \text{wt}_H(u - v).$$

The minimum distance of Berger codes equals 2 since any two information words $x_i$ and $x_j$ that have the same number of zeros and differ in two positions, are encoded to code words $c_1$ and $c_2$, whereas, $\text{dist}(c_1, c_2) = \text{dist}(x_i, x_j) + \text{dist}(w_1, w_1) = 2$.

Since the Berger code has minimum distance of two, it is not strong enough to provide reliability (refer to Theorem 1). Therefore, it must be concatenated with a (linear) code that fulfills the minimum distance requirement. The linear part in a concatenated code provides reliability but it does not contribute to the security. Namely, it cannot improve (or even change) the immunity of the code against fault injection attacks. Therefore, the error masking probability of the concatenated code is solely determined by Berger code’s redundancy bits.

The following example demonstrates this statement.

**Example 3:** Let $C(k, r_d) = \{(x, u)\}$ be a Berger code, and let $C'(k, r_I) = \{xG\} = \{(x, xP)\}$ be a systematic binary linear code of dimension $k$, length $n = k + r_I$, and minimum distance $d$, where $G = [I, P]$ is its $k \times n$ generator matrix, and $I$ is the $k \times k$ identity matrix. Consider the concatenated code $\tilde{C}$,
of dimension $k$ and length $k + r_l + r_w$.

The minimum distance of $\tilde{C}$ is greater or equal to $\max(d, 2)$. Hence, the linear part provides extra reliability. However, in terms of security, the linear part does not affect the maximal error masking probability: An error of the form $e = (e_x, e_p, e_w)$ where $e_p \neq e_x P$ is always detected by checking the linear part, that is, $Q(e) = 0$. Consequently, the maximal error masking probability is determined by $Q(e)$ of errors of the form $e = (e_x, e_p = e_x P, e_w)$. Such errors are never detected by checking the linear part. Namely, $\max(Q(e))$ is solely determined by the error masking probability of the (nonlinear) Berger code.

5. ROBUSTNESS OF THE BERGER CODES

Let $C$ be a Berger code. Let $e = (x, w) \in C$ be a codeword and $e = (e_x, e_w)$ be an error vector. The error will be masked if

$$c \oplus e = (x \oplus e_x, w \oplus e_w) \in C.$$  

In other words, $e$ will be masked by $c$ if

$$Z_{x \oplus e_x} = V(w \oplus e_w). \tag{2}$$

Eq. (2) is called the error masking equation of the code. The number of solutions to this equation determines the error masking probability of the code. The number of solutions depends on the injected error:

Theorem 2. Let $C(k, r)$ be a Berger code. An error vector of the form $(e_x = 0, e_w \neq 0)$ is always detected. An error vector of the form $e = (e_x \neq 0, e_w)$ is masked with probability

$$Q(e) = \frac{\sum_{V(w)=0}^{\frac{a}{2}} \left( V(w)-b+a \right) \left( k-a \right) \left( V(w)+b-a \right)}{2^k} \tag{3}$$

where $a$ is the Hamming weight of $e_x$, and $b = V(w \oplus e_w)$.

Proof: There are three cases:

- If $e_x \neq 0$ and $e_w = 0$ then no error has occurred.
- If $e_x = 0$ and $e_w \neq 0$ then

$$Z_{x \oplus e_x} = Z_x = V(w) \neq V(w \oplus e_w),$$

and therefore the error will always be detected.

- If $e_x \neq 0$ and $e_w \neq 0$:

Denote by $a$ the Hamming weight of $e_x$, $1 < a \leq k$, and denote by $i_1, \ldots, i_a$ the positions of the ones in $e_x$. Define $x^a = (x_{i_1}, \ldots, x_{i_a})$. Using this notation we have

$$Z_{x^a \oplus e_x} = (a - Z_{x^a}) + (Z_x - Z_{x^a}) = a + V(w) - 2Z_{x^a}.$$  

Denote by $b$ the value of $w \oplus e_w$, that is, $b = V(w \oplus e_w)$.

The error vector is masked if

$$b = Z_{x^a \oplus e_x}.$$  

In other words, the error is masked by the code word $(x, w)$ if $Z_{x^a}$ is an integer that equals

$$Z_{x^a} = \frac{V(w) - b + a}{2}.$$  

Consequently, for any $w$, and an integer $Z_{x^a}$, there are

$$\left( a \right) \left( k - a \right) \left( V(w) - Z_{x^a} \right)$$

code words that mask this error vector. If $Z_{x^a}$ is not an integer then the error is detected by all the code words having this $w$.

By Theorem 2, the value of the error masking probability depends on $k$. The following theorem states that most of the Berger codes are robust.

Theorem 3. Let $C(k, r)$ be a Berger code. If $k = 2^r - 1$ then the code is partially robust. Otherwise, it is robust.

Proof: Let $C(k, r)$ be a Berger code. There are two cases:

- Case I: $k = 2^r - 1$,

A code is partially robust if $1 < |K_d| < |C|$. Namely, if there exists at least one error vector, say $e_1$, that is never detected and at least one error vector, say $e_2$, that is detected with probability. Let

$$e_1 = (e_{x_1}, e_{w_1}) = (11 \ldots 1).$$

This error is never detected since

$$a = wt_H(e_x) = k, b = 2^r - 1 - V(w) = k - V(w),$$

and thus,
On the other hand, let \( e_2 = (e_{x,2}, e_{w,2}) \) where
\[
e_{x,2} = (110 \ldots 0), \; e_{w,2} = (00 \ldots 0).
\]

This error vector is masked by all the \( x \)'s in which \( x_{k-1} + x_{k-2} = 1 \) and it is detected by all the \( x \)'s in which \( x_{k-1} + x_{k-2} = 0 \). In other words, \( e_2 \) is detected with probability, and thus the code is partially robust.

• Case II: \( 2^{r-1} \leq k < 2^r - 1 \).

In case where \( e_x = 0 \) and \( e_w \neq 0 \) every non-zero error vector \( e \) is detected with probability \( > 0 \). As Eqs. (3) and (4) imply, the error \( e = (e_x \neq 0, e_w) \) is always masked if
\[
\left( \frac{a}{V(w)} \right) \left( \frac{k-a}{V(w)+b-a} \right) = \left( \frac{k}{V(w)} \right)
\]
for all \( w \). This happens if \( a = k \) and \( V(w) = k - b \). As \( V(w) \) is nonnegative we require that for all \( w \)
\[
0 \leq b \leq k.
\]

However, for \( 0 \leq V(e_w) < 2^{r-1} \) and \( w = (0, \ldots, 0) \) we have
\[
b = V(w \oplus e_w) = V(e_w) < 2^{r-1} \leq k,
\]
which contradicts \( V(w) = k - b \).

For \( 2^{r-1} \leq V(e_w) \leq 2^r - 1 \), i.e., \( 0 \leq V(\bar{e}_w) < 2^{r-1} - 1 \), and \( w = \bar{e}_w \) we have
\[
b = V(w \oplus e_w) = V(\bar{e}_w \oplus e_w) = V(1, \ldots, 1)
\]
\[= 2^r - 1 > k,
\]
which again contradicts \( 0 \leq b \leq k \). Thereby, Eq. (4) is not satisfied by all the \( w \)'s; hence, any nonzero error vector is masked with probability \( < 1 \). Namely, the code is robust.

To fix ideas consider the following examples:

**Example 4 (Robust Berger code):** Let \( C(5,3) \) be a Berger code. The error masking probability of the code is
\[
Q(e) = \frac{\sum_{V(w) = 0}^k \left( \frac{k}{V(w)} \right)}{2^k} = \frac{2^k}{2^k} = 1.
\]

For instance, the error vector \( e_1 = (000000, 011) \) is always detected, whereas \( e_2 = (11010, 101) \) can be detected with probability 0.875 since the Hamming weight of \( e_x \) is \( a = 3 \), and we have the following six \( (V(w), b) \) pairs:
\[
w = (000) \Rightarrow V(w) = 0, b = V(w \oplus e_w) = 5
\]
\[
w = (001) \Rightarrow V(w) = 1, b = V(w \oplus e_w) = 4
\]
\[
w = (010) \Rightarrow V(w) = 2, b = 7
\]
\[
w = (011) \Rightarrow V(w) = 3, b = 6
\]
\[
w = (100) \Rightarrow V(w) = 4, b = 1
\]
\[
w = (101) \Rightarrow V(w) = 5, b = 0
\]
and
\[
Q(e_2) = \frac{1}{32}(0 + 2 + 0 + 0 + 2 + 0) = 0.125.
\]

For example, the codeword \( x_1 = 10000 \) 100 detects the presence of \( e_2 \), while the codeword \( x_2 = 1111001 \) masks it.

Indeed, for \( k = 5 \) and \( r = 3 \), \( Q(e) \) may take seven possible values. The values of \( Q(e) \) and the percentage of errors that are masked with this probability are shown in Table 1.

**Example 5 (Partially robust Berger code):** Let \( C(7,3) \) be a Berger code. The error masking probability of the code is
\[
Q(e) = \begin{cases} 1 & \text{if } e_x = 0, e_w = 0 \\ 0 & \text{if } e_x = 0, e_w \neq 0 \\ \frac{\sum_{V(w) = 0}^7 \left( \frac{a}{V(w)-b+a} \right) \left( \frac{5-a}{V(w)+b-a} \right)}{2^7} & \text{otherwise.} \end{cases}
\]

For instance, the error vector \( e_1 = (0000000, 111) \) cannot be detected since \( a = 7 \) and

<table>
<thead>
<tr>
<th>Percent errors</th>
<th>( Q(e) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>57.8</td>
<td>0.000</td>
</tr>
<tr>
<td>5.9</td>
<td>0.125</td>
</tr>
<tr>
<td>23.4</td>
<td>0.250</td>
</tr>
<tr>
<td>6.3</td>
<td>0.375</td>
</tr>
<tr>
<td>5.9</td>
<td>0.500</td>
</tr>
<tr>
<td>0.4 (e = 11111001)</td>
<td>0.625</td>
</tr>
<tr>
<td>0.4 (the all-zero error vector)</td>
<td>1.000</td>
</tr>
</tbody>
</table>

**Table 1** Error Masking Probabilities for \( k=5 \)
\[ w = (000) \Rightarrow V(w) = 0, b = V(w \oplus e_w) = 7 \]
\[ w = (001) \Rightarrow V(w) = 1, b = V(w \oplus e_w) = 6 \]
\[ w = (010) \Rightarrow V(w) = 2, b = 5 \]
\[ w = (011) \Rightarrow V(w) = 3, b = 4 \]
\[ w = (100) \Rightarrow V(w) = 4, b = 3 \]
\[ w = (101) \Rightarrow V(w) = 5, b = 2 \]
\[ w = (110) \Rightarrow V(w) = 6, b = 1 \]
\[ w = (111) \Rightarrow V(w) = 7, b = 0 \]

and

\[ Q(e_2) = \frac{1}{128} \left(1 + 7 + 21 + 35 + 35 + 21 + 7 + 1 \right) = 1 \]

Therefore, the detection kernel of the code consists of more than one error vector and thus the code \( C(7, 3) \) is partially robust, which complies with Theorem 3.

The maximal error masking probability \( Q(e) \), and the probability that an error vector is masked with this probability, is shown in Table 2 for several values of \( k \).

It is clear from Table 1 that the error masking probability is not uniform all over the errors, that is

\[
\max_{e \neq 0} Q(e) > \text{average}_{e \neq 0} Q(e).
\]

Therefore:

**Corollary 1.** A robust Berger code with \( k \neq 2^r - 1 \) is not an optimum code, that is, \( \max_{e \neq 0} Q(e) > 2^{-r} \).

Note that the known optimum codes are not unordered. Thus, they must be concatenated with unordered codes in order to enable a reliable asynchronous communication (even if the communication channel is skew-less). A Berger code is considered as a good candidate for concatenation since it is optimal in terms of code rate. However, since a Berger code is not an optimum code, any concatenation with any other code will result in a nonoptimum code. A concatenation can improve the error masking probability, but it cannot reduce it enough to meet the lower bound.

Although the \( C(k = 2^r - 1, r) \) Berger code is not robust, by concatenating it with a perfect nonlinear binary code, i.e., by adding a single redundancy bit, it may become an unordered robust code. The extended Berger code of length \( 2^r + r \) is defined as follows:

\[
\hat{C} = \{ \hat{c} = (c, f(c)) | c \in C(2^r - 1, r), f(c) \}
\]

\[
= \sum_{i=0}^{2^{r-1}-2} c_{2i} \cdot c_{2i+1} + c_{2r-2} \mod 2 \]

**Theorem 4.** The extended Berger code \( \hat{C} \) is a robust code with error masking probability \( \hat{Q}(\hat{e}) \leq \min(0.5, Q(e)) \) where \( Q(e) \) is the error masking probability of the \( C(k = 2^r - 1, r) \) Berger code.

**Proof:** The correctness of the theorem follows from the fact that an error is undetected only if it is undetected by both the Berger code and the additional redundancy check. Denote \( \hat{e} = (e_1, e_f) \) an error vector that distorts the code word \( \hat{c} = (c, f(c)) \). The function \( f \) is not a perfect nonlinear function since the error vector \( \hat{e}_1 = (e_1, 1) \) is never detected by \( f \) since \( f(e_1 + e_1) = f(e_1) + 1 \mod 2 \). However, this error vector is always detected by the Berger code since its left part \( e_1 \) flips a single bit in \( c \). Consequently, \( \hat{Q}(\hat{e}_1) = 0 \). All the other nonzero errors are detected by \( f \) with probability of 0.5. Thus we have, \( \hat{Q}(\hat{e}) \leq \min(0.5, Q(e)) \).

### 6. CONCLUSIONS

This paper deals with the robustness of Berger codes against error injection attacks. Berger codes can be used in Asynchronous Global Communication, since they are insensitive to propagation delays of individual bits. The paper examines the detection capability of Berger codes in cases where the errors are symmetric and presents conditions for the robustness of Berger codes against error injection attacks. It is shown that for \( k \neq 2^r - 1 \) the Berger codes are robust but not optimum, that is, any error can be detected, but the probability of detecting its presence is smaller than \( 1 - 2^{-r} \). For \( k = 2^r - 1 \) the codes are proved

<table>
<thead>
<tr>
<th>( P(\hat{Q} = \max Q) )</th>
<th>( \max_{e \neq 0} Q(e) )</th>
<th>( r )</th>
<th>( k )</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.91e – 03</td>
<td>0.625</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5.47e – 02</td>
<td>0.500</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>1.95e – 03</td>
<td>1.000</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>1.10e – 02</td>
<td>0.500</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>5.49e – 03</td>
<td>0.500</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>3.36e – 03</td>
<td>0.500</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>3.05e – 05</td>
<td>0.773</td>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>1.19e – 03</td>
<td>0.500</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>6.94e – 04</td>
<td>0.500</td>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>4.58e – 04</td>
<td>0.500</td>
<td>4</td>
<td>14</td>
</tr>
<tr>
<td>3.81e – 06</td>
<td>1.000</td>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>7.30e – 05</td>
<td>0.500</td>
<td>5</td>
<td>16</td>
</tr>
<tr>
<td>3.65e – 05</td>
<td>0.500</td>
<td>5</td>
<td>17</td>
</tr>
<tr>
<td>2.04e – 05</td>
<td>0.500</td>
<td>5</td>
<td>18</td>
</tr>
<tr>
<td>5.96e – 08</td>
<td>0.641</td>
<td>5</td>
<td>19</td>
</tr>
<tr>
<td>6.26e – 06</td>
<td>0.500</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

A. Burg and O. Keren
to be partially robust, that is, there exists an error vector that is never detected. However, the code can become robust by adding a single redundancy bit.

**FUNDING**

This research was supported by the Israel Science Foundation Grant No. 1200/12 and by the TRUDEVICE Cost Action IC1204.

**REFERENCES**


**BIOGRAPHIES**

**Osnat Keren** received a MSc Degree in Electrical Engineering from the Technion-Israeli Institute of Technology and a PhD Degree from the Tel-Aviv University, Israel, in 1988 and 1999, respectively. Between 1988 and 1994 she held a chip design and senior DSP engineer position at National Semiconductor, and between 1999 and 2003 she was the Senior Scientist of Millimetrix Broadband Networks. Since 2004, Dr. Keren has been with the Faculty of Engineering at Bar-Ilan University, Israel.

**Ariel Burg** received his Bachelor’s Degree in Computer Engineering from Bar-Ilan University, Israel, in 2011. Currently, he is studying toward a second degree in Electrical Engineering at Bar-Ilan University, Israel.