Integrated Computer-Aided Engineering 24 (2017) 351–365
DOI 10.3233/ICA-170549
IOS Press

Noise tolerant probabilistic logic for statistical pattern recognition applications

V. Canals[a,b,*], C.F. Frasser[a], M.L. Alomar[a,b], A. Morro[a], A. Oliver[a], M. Roca[a], E. Isern[a], V. Martínez-Moll[b], E. Garcia-Moreno[a] and J.L. Rosselló[a]
[a]*Electronics Engineering Group, Department of Physics, University of Balearic Islands, Campus UIB, Majorca, Spain*
[b]*Energy Engineering Group, Department of Physics, University of Balearic Islands, Campus UIB, Majorca, Spain*

Abstract. The new generation of knowledge-based applications requires a large amount of computing power with minimal energy consumption. This has aroused the interest in the non-conventional computing methods capable to implement complex functions in a very simple way and which in turn are inherently noise tolerant, as is the case of probabilistic or stochastic computing architectures. This work analyzes the robustness against noise of the Extended Stochastic Logic (ESL) encoding, a recently proposed probabilistic computing methodology. Furthermore, the capabilities of the ESL encoding to implement complex computational functions in the field of statistical pattern recognition, as is the case of a Bayesian classifier, are presented. The ESL noise-tolerance is analyzed and tested in a FPGA by injecting a wide range of noise levels. The noise-tolerance results are compared with the archived by conventional circuits, with and without fault-tolerant capabilities. The ESL outperforms the conventional Triple Modular Redundancy (TMR) solutions as is show in the experimental results.

Keywords: Probabilistic computing, probabilistic logic, pattern recognition, field programmable gate arrays, robustness, error correction

1. Introduction

Nowadays, a new generation of knowledge-based applications [21] is taking importance in the technological world. Those applications use Computational Intelligence [53] methodologies to solve complex real-world problems in which traditional approaches are not feasible or ineffective; and generally its application falls in one of these three categories: pattern recognition [1,24], data mining [3,24,57] and digital synthesis [4,9]. All of these applications need to process large amounts of information, thus require of a large amount of computing power with a minimal consumption. All this has pushed the semiconductor industries to enhance chips performance [33] through the CMOS technology down-scaling and aggressive design practices (such as dynamic logic styles) those have resulted a current technology more noise-sensitive [19]. Unfortunately, the chip power consumption has also been continuously increasing [28]. A common technique for reducing the energy consumption is to decrease the power supply voltage, which implies a loss of noise robustness and therefore a circuit reliability decrease [44]. Therefore, decreasing the CMOS feature sizes causes the devices to be less reliable [32]. Other noise-related effects are the ground bounce node capacitance, node critical charge reduction, higher thermal noise, process variations, soft errors and noise margin diminution. Earlier, these effects have had little impact on the integrated circuits performance, but in the nanometric technologies era this relevance has increased significantly [23,37]. In particular, soft errors refer to non-permanent errors that can severely limit the reliability of CMOS circuits. They are produced by the charge injection due to a particle hit, from an alpha particle [31] or a neutron [56]. Traditionally, soft errors [27] were related with memories but with the technology downscaling has become as frequent in combi-

*Corresponding author: V. Canals, Electronics Engineering Group, Department of Physics, University of Balearic Islands, Cra. Valldemossa Km 7.5, Campus UIB, Mateu Orfila i Rotger Building, 07122, Palma of Majorca, Spain. Tel.: +34 971 172463; Fax: +34 971 173426; E-mail: v.canals@uib.es.

national circuits as in un-protected SRAM cells [52]. Numerous design methodologies have been developed to overcome the noise immunity loss as can be the SRAM [37] and combinational cells [59] hardening, the on-chip error checking and correction circuits [41], space redundancy and/or time redundancy. Space redundancy mainly includes Dual Modular Redundancy and Triple Modular Redundancy (TMR), this is the most common scheme to perform Single Event Upset (SEU) and electromagnetic waves hardening [18].

The current situation of an unreliable nanometric technology coupled with the raising of the knowledge-based applications demanding major computing capacity [7,39,42,49] have increased the interest in non-conventional computing methods capable to implement complex functions in a very simple way [34,45] and which in turn are inherently noise tolerant [10,15]. These methods are a complement to conventional parallel and distributed computing [2,5,6,8,25,26,48,54]. This is the case of probabilistic or Stochastic Computing architectures [11] which apply probabilistic laws to digital logic systems, thus performing pseudo-analog operations with stochastic pulse frames (stochastic signals) [14]. The main research lines on this unconventional technique are focused on some knowledge-based fields, such as: digital filters [50], image processing [40,58], Low Density Parity Check decoders used in wireless communications [20], neural networks [16,46,47], pattern recognition/classifiers [17, 29], data mining [35] and fault tolerant computing architectures [30,43,51].

This work aims to study theoretically and experimentally the noise tolerance of Extended Stochastic Logic (ESL) encoding, as well as analyzing its capabilities to implement complex computational functions in the field of statistical pattern recognition. The ESL encoding scheme and main blocks have been applied successfully by authors to implement highly reliable [15] probabilistic neural networks [16] using very few hardware resources.

ESL noise-tolerance capabilities are theoretical analyzed and discussed its robustness for different noise sources. The design of the main probabilistic rules is proposed and analyzed, such as addition, subtraction, multiplication, and conditional probability. To check the real statistical pattern recognition capabilities of the proposal, a Bayesian classifier circuit is presented and tested in a FPGA. To test the capacities of basic ESL blocks against noise (mainly electromagnetic noise), different rates of noise are injected (0 to 100%) to the FPGA inputs. In order to discuss the goodness of the obtained ESL results in front of the conventional systems, a conventional and a TMR 8 $\times$ 8-bit multiplier have been implement and tested under the same noise conditions. Finally, the ESL noise-tolerance dependency as function of the evaluation period (related with the computational capacity) is analyzed.

The rest of the paper is organized as follows: Section 2 briefly introduces ESL encoding and architecture, the ESL noise-tolerance is analyzed, the main statistical rules design is proposed, and finally the implementation of a Bayesian classifier is presented. Section 3 shows the obtained experimental noise-tolerance results by each basic ESL block and architecture, and compared with the conventional and TMR architectures. Finally, the conclusions are presented in Section 4.

2. Materials and methods

This section briefly presents ESL coding and probabilistic architecture. The ESL noise tolerance is mathematically analyzed. Then, the ESL blocks designs that implement the main statistical rules are presented. A Bayesian classifier is finally shown.

2.1. Extended stochastic logic

The ESL is a probabilistic computing encoding that represents the information by means of the ratio $x^* = p^*/q^*$ of the switching activity of two bipolar encoded stochastic signals [14], each one defined in the range $[-1, 1]$. The ratio x^* follows the Eq. (1), now being able to represent any real number $(-\infty, +\infty)$ related with the switching activities, p^* and q^*, of the P and Q (digital N-bit) values (as shown in Fig. 1a).

$$\begin{aligned}
&\begin{cases} p \in [0,+1] \\ q \in [0,+1] \end{cases} \rightarrow \begin{cases} p^* \in [-1,1] \\ q^* \in [-1,1] \end{cases} \\
&\Rightarrow \begin{cases} 0 \leqslant P \leqslant 2^N - 1 \\ 0 \leqslant Q \leqslant 2^N - 1 \end{cases} \\
&\begin{cases} p^* = 2 \cdot p - 1 \\ q^* = 2 \cdot q - 1 \end{cases} \Rightarrow x^* = \frac{p^*}{q^*} \rightarrow E\left(x^*\right) \\
&= E\left(\frac{p^*}{q^*}\right) = \frac{\left(2 \cdot P - 1 \cdot \left(2^N - 1\right)\right)/\left(2^N - 1\right)}{\left(2 \cdot Q - 1 \cdot \left(2^N - 1\right)\right)/\left(2^N - 1\right)} \\
&= \frac{2 \cdot P - \left(2^N - 1\right)}{2 \cdot Q - \left(2^N - 1\right)} = \frac{P^*}{Q^*}, \forall x^* \in (-\infty, +\infty)
\end{aligned} \tag{1}$$

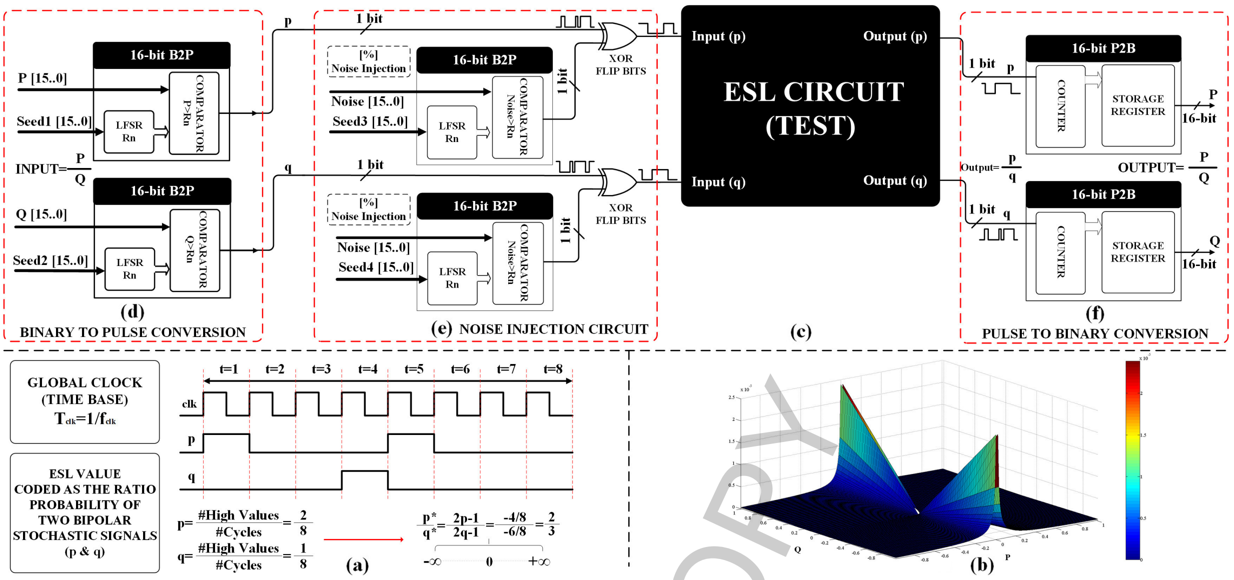

Fig. 1. (a) ESL encoding example. (b) ESL conversion error. (c) Probabilistic computing architecture. (d) Binary to pulse conversion. (e) Noise injection circuit. (f) Pulse to binary conversion.

On the bottom side of Fig. 1a we illustrate how to obtain an average value of the ratio $x^* = 2/3$ using a pair of pulsed signals with switching probabilities $p = 2/8$ and $q = 1/8$. On the other hand, the ESL coding conversion error is dependent on the P^* and Q^* encoded probability values according to:

$$X^* = \frac{P^*}{Q^*} \rightarrow Error(X^*) = \left|\frac{1}{Q^*}\right| \cdot Error(P^*) + \left|\frac{P^*}{{Q^*}^2}\right| \cdot Error(Q^*) \quad (2)$$

It should be noted that the use of small values of Q^* are necessary to represent large X^* values, which inevitably involves large error values independently of the conversion error associated with the p^* and q^* signals, as can be shown in Fig. 1b. This is one of the main drawbacks of the encoding. Therefore, P^* and Q^* must be set as small as possible in order to limit the magnitude of the error.

2.1.1. Probabilistic computing architecture

Probabilistic computing systems are composed at least by three basic stages, illustrated in Fig. 1c. The first stage converts digital values X^* to their respective pulsed signals x^*, to enable done any probabilistic computation. We call these blocks Binary-to-Pulse converter (B2P). With the ESL encoding as happens with traditional stochastic logic, values X^* must be coded as the quotient of P^* and Q^* binary numbers, and converted to stochastic signals (p^*, q^*) using a couple of B2P blocks [16].

The second stage implements the probabilistic computing circuit to carry out a certain task. The configuration and structure of this circuit is related to the codification being used (unipolar, bipolar or ESL) [14], which defines the representation range of the system.

Finally, the third stage is responsible to convert the pulsed signals resulting of the probabilistic computing systems into their equivalent binary values X^*. We named these blocks Pulse-to-Binary converter (P2B). To convert an ESL x^* probability again into a digital value, it is necessary to combine two P2B blocks [16] to obtain their equivalent binary values (P^*, Q^*). Storing both binary signals, the quotient X^* is obtained.

2.1.2. Noise-tolerance analysis

Conventional digital systems are extremely vulnerable to signal variations outside the specified voltage ranges for high and low logic values. In fact, a whole digital system can be halt as a result of noise. Noise can be induced by different ways, but electromagnetic waves are one of the most important noise sources, which may generate circuit malfunction. Generally, these have a greater impact and larger area of influence than soft errors induced by charge injection due to a particle hit, known as SEU. This is due to the fact that electromagnetic waves affect all the devices area while soft errors just affect at a single point. In addition, electromagnetic waves have much larger energy, and therefore they are more likely to cause multi-bit errors since a particle burst is generally unlikely to take place [36].

The ESL encoding not only makes feasible the implementation of probabilistic operations that exceed the representation range of the stochastic bipolar coding, but also expands the inherent noise-tolerance capacity of stochastic systems [11,43]. The mathematical development of the noise injection (with a noise rate r) in an ESL signal (coded as the ratio $x^* = p^*/q^*$) is presented in Eq. (3).

$$P \xrightarrow{\textit{Noise\_injection}}$$
$$\begin{cases} P' = P \cdot (1-r) + r \cdot (2^N - 1 - P) \\ = P - 2 \cdot P \cdot r + r \cdot (2^N - 1) \end{cases}$$
$$Q \xrightarrow{\textit{Noise\_injection}}$$
$$\begin{cases} Q' = Q \cdot (1-r) + r \cdot (2^N - 1 - Q) \\ = Q - 2 \cdot Q \cdot r + r \cdot (2^N - 1) \end{cases}$$
$$E'(x^*) = E'\left(\frac{p^*}{q^*}\right) = \frac{2 \cdot P' - (2^N - 1)}{2 \cdot Q' - (2^N - 1)}$$
$$= \frac{2 \cdot (P - 2 \cdot P \cdot r + r \cdot (2^N - 1)) - (2^N - 1)}{2 \cdot (Q - 2 \cdot Q \cdot r + r \cdot (2^N - 1)) - (2^N - 1)}$$
$$= \frac{2 \cdot P - (2^N - 1)}{2 \cdot Q - (2^N - 1)} \cdot \frac{1 - 2 \cdot r}{1 - 2 \cdot r}$$
$$= \frac{2 \cdot P - (2^N - 1)}{2 \cdot Q - (2^N - 1)} = E(x^*) \qquad (3)$$

When a noise rate r is injected into the input signals p and q, the associated binary values P and Q (average number of high values in the pulsed signal stream) are modified. But the encoded ratio obtained with the modified binary values P' and Q' remains unchanged since the factor $(1 - 2 \cdot r)$ appearing in both numerator and denominator can be cancelled. Nevertheless, this common factor approaches to zero for values of 100% of noise injection (related with a noise rate $r = 0.5$). A noise rate of $r = 0.5$ imply a 50% of probability of any bit being in a high or low value, which implies no information present in the digital signal. This corresponds to signal values of $p^* = q^* = 0$ and therefore an indeterminate value of the ESL quotient. In this case, the modified binary values P' and Q' become smaller, and the conversion error is increased due to the digital quantization of the information.

In addition, it can be assumed that an ESL signal is affected by a low intensity noise at a single point will continue to operate correctly, since a stochastic signal can be essentially considered as a signal generated by noise [43]. This is the case, of a SEU hit at sea level, which present a frequency of the order 10–12 upset/(bit · h) for memories [38]. For higher rates of noise (due to electromagnetic waves, SEUs at high altitudes or in space), we postulate that there are no reasons to believe that one of the two stochastic signals of an ESL magnitude will be exposed in average to different noise rates. However, we do not assume that both signals are affected by the noise in the same time instants. Then, Eq. (3) demonstrates that an ESL signal (coded as the ratio $x^* = p^*/q^*$) remains unchanged when the two switching signals are affected by the same noise level.

2.2. *ESL statistical rules blocks*

This sub-section presents the ESL design of the basic statistical rules and basic operations necessary to implement any general purpose statistical pattern recognition system and finally discusses the implementation of a Bayesian classifier.

2.2.1. *Rules of addition and subtraction*

The statistic *rule of addition* applies to the following situation. Assume the joint probability (X and Y) is given by:

$$P(X \cup Y) = P(X) + P(Y) - P(X \cap Y) \qquad (4)$$

Invoking the fact that $P(X \cap Y) = P(X) \cdot P(Y|X)$, the addition rule can also be expressed as:

$$P(X \cup Y) = P(X) + P(Y) - P(X) \cdot P(Y|X) \qquad (5)$$

The ESL addition/subtraction of probabilistic signals is performed using the design shown in Fig. 3a. The numerator of the sum of p^*/q^* and r^*/s^* (signal t^*) is assessed using two XNOR gates (cross multiplication between numerator and denominator of the two numbers) and a multiplexer (that provide the mean value of the two products).

Therefore, the result in the numerator is $t^* = (p^* - s^* + r^* - q^*)/2$. The denominator u^* is obtained using a three-input XNOR gate that multiplies the signals q^*, s^* and a signal with a switching activity $sig^* = 0.5$. Note that the difference between the addition using the traditional Stochastic Computing and the novel ESL methodology is that the former yields the mean value of two signals while the latter provides the real sum.

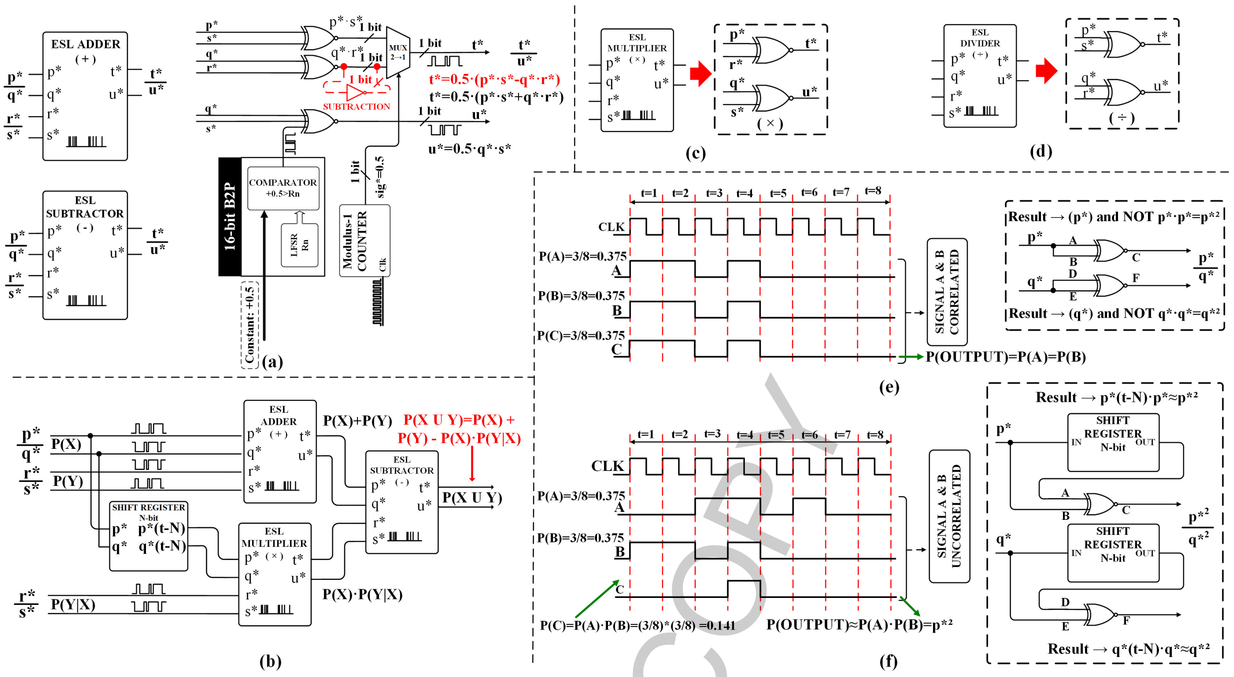

Fig. 2. (a) ESL adder/subtractor. (b) ESL statistical rule of addition. (c) ESL multiplier/rule of multiplication. (d) ESL divider. (e) ESL nonfunctional square circuit. (f) ESL functional square circuit.

$$out = \frac{t^*}{u^*} = x + y = \frac{\overline{(p \oplus s)} \cdot c + \overline{(r \oplus q)} \cdot \bar{c}}{\overline{(q \oplus s \oplus sig)}}$$

$$\Longrightarrow out^* = \frac{\frac{out\_t^*+1}{2}}{\frac{out\_u^*+1}{2}} = \ldots$$

$$\begin{cases} j^* = \overline{p \oplus s} = p^* \cdot s^* \\ k^* = \overline{r \oplus q} = r^* \cdot q^* \\ l^* = \overline{q \oplus s} = q^* \cdot s^* \qquad \ldots \\ c^* = 0 \\ sig^* = +0.5 \end{cases}$$

$$= \frac{\frac{j^*+1}{2} \cdot \frac{c^*+1}{2} + \frac{k^*+1}{2} - \frac{k^*+1}{2} \cdot \frac{c^*+1}{2}}{\left(1 - \frac{l^*+1}{2}\right) \cdot \frac{sig^*+1}{2} + \frac{l^*+1}{2} \cdot \left(1 - \frac{sig^*+1}{2}\right)}$$

$$= \ldots = \frac{\frac{1}{2} \cdot (c^* \cdot (j^* - k^*) + j^* + k^* + 2)}{\frac{1}{2} \cdot (1 - l^* \cdot sig^*)}$$

$$= \frac{0.5 \cdot j^* + 0.5 \cdot k^*}{0.5 \cdot l^*} =$$

$$= \frac{j^* + k^*}{l^*} = \frac{p^* \cdot s^* + r^* \cdot q^*}{q^* \cdot s^*} \tag{6}$$

The subtraction is performed in the same way as the addition but adding a NOT gate between the XNOR and the multiplexer (red dashed inverter in Fig. 2a), which changes the sign of one of the two values.

$$out = NOT\,(p) = \overline{p} = (1 - p) \rightarrow$$

$$\begin{cases} out^* = 2 \cdot out - 1 \\ p^* = 2 \cdot p - 1 \end{cases} \Rightarrow out = \frac{out^* + 1}{2}$$

$$= 1 - \frac{p^* + 1}{2} \Rightarrow out^* = 2 - 2 - p^* = -p \tag{7}$$

On the other hand, the ESL subtraction block is fundamental to implement the statistical *rule of subtraction,* which establishes that the probability of event X occurring is equal to one minus the probability of that event not occurring.

$$P(X) = 1 - P\,(X') \tag{8}$$

To perform the addition of two statistical rules Eq. (5), we need three ESL blocks: an adder, a subtractor and a multiplier, as shown in Fig. 2b.

Previously, this block has been shown to be essential to implement probabilistic neural networks [16], as it allows to perform the sum of the presynaptic contributions to evaluate the membrane potential u_i.

$$u_i = \sum_{j=0}^{n-1} w_{ij} \cdot x_j \leftarrow \begin{cases} w_{i0} = -b_i \\ x_0 = +1 \end{cases} \tag{9}$$

2.2.2. *Rule of multiplication and conditional probability*

The *Rule of multiplication* evaluates the probability of the intersection between two events; that is, the

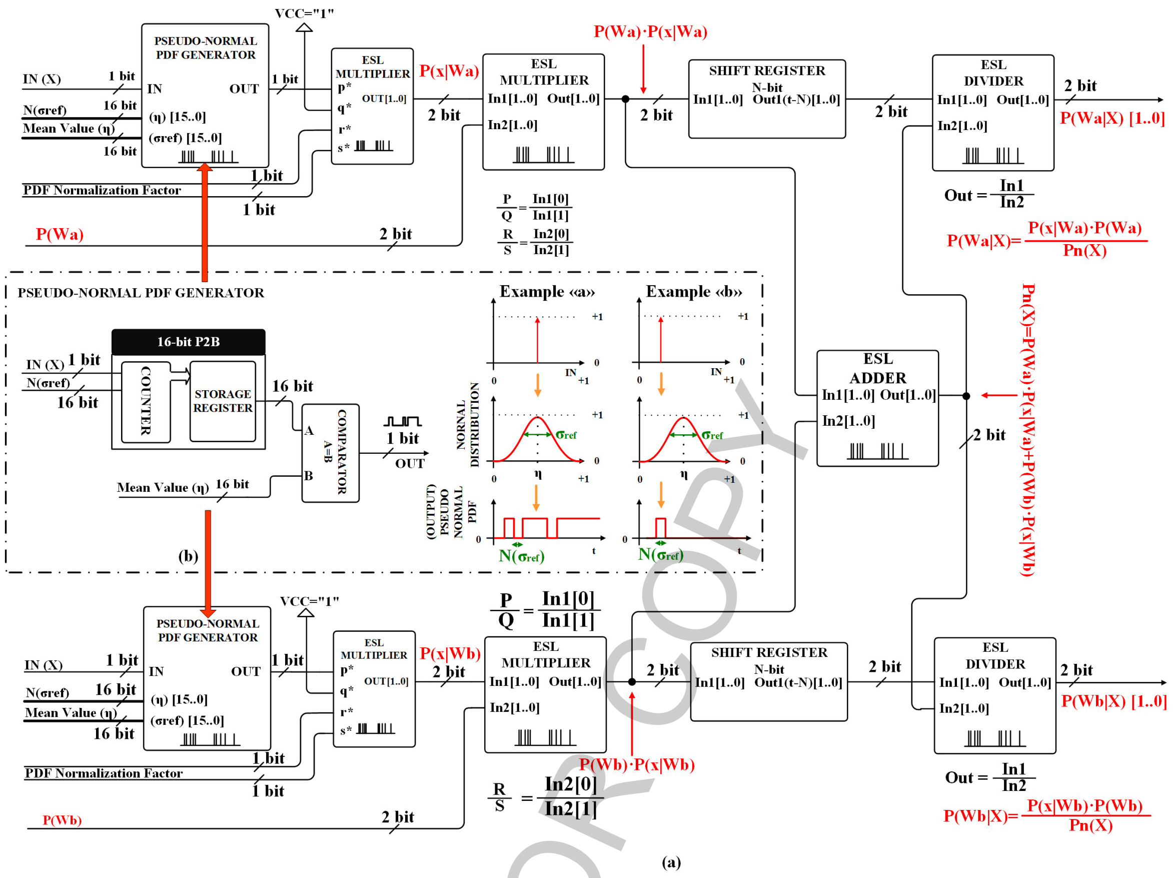

Fig. 3. (a) ESL Bayes rule design for two categories. (b) Stochastic pseudo-normal PDF generator.

probability of two events (event X and event Y) both occurring at the same time.

$$P\left(X\cap Y\right)=\begin{cases}P(X)\cdot P(Y), & \text{X \& Y independent}\\ P(X)\cdot P(Y|X)=P(Y)\cdot P(X|Y), & \text{other}\end{cases}\tag{10}$$

In addition, substituting the known values of $P(X)$ and $P(X\cap Y)$ into Eq. (10) is possible obtain the $P(Y|X)$, since the conditional probability can be expressed as.

$$P\left(Y|X\right)=\frac{P\left(X\cap Y\right)}{P(X)}\tag{11}$$

These operations can be implemented with the ESL encoding using a pair of XNOR gates to multiply or divide two probability values $X^*=p^*/q^*$ and $Y^*=r^*/s^*$ (see Fig. 2c for multiplication and Fig. 2d for division). The multiplication is performed by the direct product between both numerators and denominators as shown in Eq. (12). The same circuit can be used to perform the division doing the cross multiplication.

This basic function have been used to perform the product $w_{ij}\cdot x_j$ of a j-th input x_j by the i-th neuron connection weight w_{ij} in a probabilistic neuron [16].

$$\begin{aligned}out&=\frac{out\_u}{out\_v}=\overline{x\oplus y}=\frac{\overline{(p\oplus r)}}{\overline{(q\oplus s)}}=\frac{\overline{p}\cdot\overline{r}+p\cdot r}{\overline{q}\cdot\overline{s}+q\cdot s}\\&\xrightarrow[Var\_change]{}out^*=\frac{\frac{1+out\_u^*}{2}}{\frac{1+out\_v^*}{2}}=\\&\frac{1-\frac{p^*+1}{2}-\frac{r^*+1}{2}+2\cdot\frac{p^*+1}{2}\cdot\frac{r^*+1}{2}}{1-\frac{q^*+1}{2}-\frac{s^*+1}{2}+2\cdot\frac{q^*+1}{2}\cdot\frac{s^*+1}{2}}\\&=\frac{1+out\_u^*}{1+out\_v^*}=\frac{1+p^*\cdot r^*}{1+q^*\cdot s^*}\\&\rightarrow\frac{out\_u^*}{out\_v^*}=\frac{p^*\cdot r^*}{q^*\cdot s^*}=x^*\cdot y^*\end{aligned}\tag{12}$$

2.2.3. *Exponentiation*

The ESL exponentiation block is used to multiply a probability $P(X)$ by itself several times. To illustrate the exponentiation operation will be show the implementation of the square function $f(X^*) = (p^* \cdot p^*)/(q^* \cdot q^*)$. A crucial requirement of stochastic computing is that the pulsed signals to be operated must be un-correlated [17].

$$\begin{aligned} out &= \frac{out\_u}{out\_v} = \overline{x(t) \oplus x\,(t)} = \frac{\left(\overline{p(t) \oplus p\,(t)}\right)}{\left(\overline{q(t) \oplus q\,(t)}\right)} \\ &= \frac{\overline{p(t)} \cdot \overline{p(t)} + p(t) \cdot p(t)}{\overline{q(t)} \cdot \overline{q(t)} + q(t) \cdot q\,(t)} = \frac{\overline{p(t)} + p(t)}{\overline{q(t)} + q(t)} \\ &= \frac{1}{1} = 1 \end{aligned} \tag{13}$$

The importance of the temporal un-correlation [12] is shown in Fig. 2(e) with an example where a signal is operated with itself, using an XNOR gate. Since the signals at the inputs of the pair of XNOR gates ($p(t)$ or $q(t)$) are correlated (in this case they are indeed the same), the result is not the squared input but it is a constant value equal to '1' as shown in Eq. (13). A solution to eliminate the correlation between different signals is the use of an n-bit shift register to delay one of the signals with respect to the other as shown in Fig. 2f. This procedure makes possible to evaluate the product between the signals $p(t)$ or $q(t)$ and their corresponding time-delayed values $p(t-n)$ and $q(t-n)$, as shown below.

$$\begin{aligned} out &= \frac{out\_u}{out\_v} = \overline{x(t) \oplus x(t-n)} = \\ &\frac{\left(\overline{p(t) \oplus p(t-n)}\right)}{\left(\overline{q(t) \oplus q(t-n)}\right)} = \\ &= \frac{\overline{p(t)} \cdot \overline{p(t-n)} + p(t) \cdot p(t-n)}{\overline{q(t)} \cdot \overline{q(t-n)} + q(t) \cdot q(t-n)} \\ &\xrightarrow[\text{Var\_change}]{} out^* = \frac{p^* p^*}{q^* \cdot q^*} \\ &= x^* \cdot x^* = x^{*2} \end{aligned} \tag{14}$$

For the experimental implementation of the exponentiation block a 16-bit shift register has been used $n = 16$ as delay element.

2.2.4. Bayesian classifier implementation

In order to evaluate the ESL capabilities in the field of statistical pattern recognition a two-class Bayesian classifier is implemented. It's based on a likelihood function $P(x|w_i)$ for each class w_i with respect to an input/measurement x. Each class are obtained by supervised learning over a set of J patterns $\Omega = \{w_1, \ldots, w_J\}$, such that the properties of the set of classes are.

$$\begin{cases} w_i \cap w_j = \phi \\ \bigcup\limits_{i=1}^{J} w_i = \Omega \end{cases} \tag{15}$$

The solution that minimizes the decision error [55] about a given class w_i for a measurement x is estimated by the a posteriori probability function $P(w_i|x)$ using the Bayes formula.

$$P\,(w_i|x) = \frac{P\,(w_i) \cdot P\,(x|w_i)}{\sum\limits_{i=1}^{J} P\,(w_i) \cdot P\,(x|w_i)}, i = 1, \ldots, J \tag{16}$$

Showing how for a measurement x changes the decision based on the a priori probability of each class.

$$P(w_i) = \pi_j, i = 1, \ldots, J \tag{17}$$

The ESL encoding allows to overcome one of the major drawbacks found in the previous stochastic deployments [17] when evaluating the a posteriori probability of the different categories, related to the inability to normalize each class w_i likelihood function $P(x|w_i)$ to the unity.

$$\int_x P(x|w_i) = 1 \tag{18}$$

To evaluate the benefits of the ESL encoding in the statistical pattern recognition field, we implement the Bayes rule for two categories (A and B) as shown in Fig. 3a.

$$P(x|w_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu_i)^2}{2\sigma_i^2}}, x \in \Re$$

Where,

$$\begin{cases} \mu_i : \text{mean of the probability distribution} \\ \sigma_i^2 : \text{variance of the probability distribution} \end{cases} \tag{19}$$

The likelihood function of each class Eq. (19) is implemented by a pair of previously developed stochastic pseudo-normal Probability Density Function (PDF) generator blocks [43] (Fig. 3b), whose output is multiplied by an ESL normalization factor in order to set

Table 1
ESL Bayesian classifier hardware resources

Total logic elements (LE):	665/18,752 (3.5%)
Total combinational functions:	384/18,752 (2%)
Dedicated logic registers:	578/18,752 (3.1%)
Total memory bits:	0/239,616 (0%)
Embedded multiplier 9-bit element:	0/52 (0%)
FPGA main clock (MHz):	27
Number of clock cycles to obtain an output:	131,072
Bayes classifier evaluation period (ms):	4.855

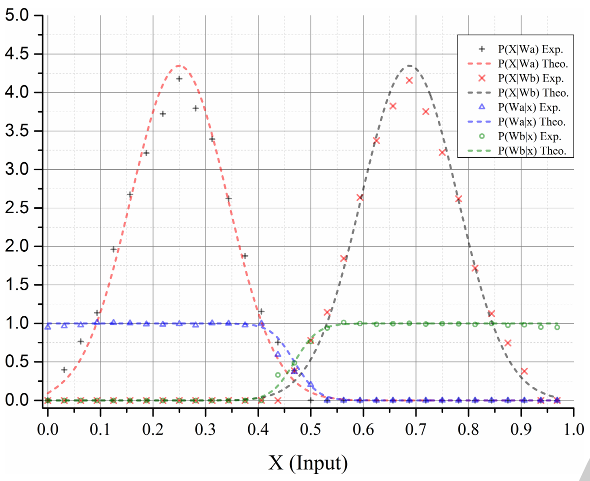

Fig. 4. ESL Bayesian classifier experimental results.

its area to one. Subsequently, we evaluate the product $P(x|w_i) \cdot P(w_i)$ for each category w_i through a pair of ESL multipliers. Then, the multipliers output is connected to the inputs of an ESL adder block to evaluate the normalization factor $P(x) = \Sigma P(x|w_i) \cdot P(w_i)$. Finally, before dividing each category contribution $P(x|w_i) \cdot P(w_i)$ by $P(x)$, we temporally uncorrelate both t signals with a pair of two-bit shift registers. Then, at the ESL dividers output the a posteriori probability of each category w_i is evaluated.

The two-category ESL Bayesian classifier circuit has been synthetized on an Altera FPGA with the following parameters: $\mu_A = 0.25, \mu_B = 0.688, \sigma_A = \sigma_B = 0.009$ and $P(w_A) = P(w_B) = 0.5$. The results are presented in Fig. 4, where we compare the theoretical expected behavior (dashed lines) and the circuit measurements (symbols). We find a misclassification between the experimental results and the theory of only 4.61% in the worst case $P(W_A|x)$. As can be appreciated, the circuit properly evaluates the a posteriori probability for both categories since the probabilities cross at the point where the conditional probabilities are equal. It is worth highlighting that the required number of logic gates for the implementation is very small, as shown in Table 1; which allows to replicate the circuit many times and to take advantage of hardware parallelism.

Although the results are good, one would expect them to be even better. This is due to the use of additional hardware associated with the probabilistic architecture (P2B and B2P blocks); that uses a considerable fraction of hardware resources especially when the stochastic circuit is so small.

On the other hand, the evaluation period for this circuit seems a priori very large according to the current computation time. However, probabilistic systems unlike conventional ones do not have a fixed period of computation; that depends on the conversion error that can be assumed for a given application. For example, in massive probabilistic pattern recognition applications usually uses only 4-bit (only 16 clock cycles) to evaluate the results [35].

Particularly, the proposed application intended to check the ESL capabilities to implement pattern recognition applications, and therefore the results should be compared with a computer floating-point one. This is the reason to use a slow architecture with a little conversion error (16-bit) a faster with a higher conversion error.

3. Experimental results and discussion

This section aims to study the experimental noise-tolerance archived by each basic ESL block. The results are compared with respect to a conventional TMR architecture.

In order to evaluate the noise-tolerance of the ESL architecture, different noise levels are injected. This procedure is repeated for the blocks configured with different evaluation periods (2^k clock cycles with $k =$ 12, 16, 20 and 24-bit) determining the noise-tolerance capabilities as a function of the architecture used.

The noise injection emulation will be done by introducing random changes to the inputs of the ESL blocks (Fig. 1d), i.e. generating random bit flips in the p and q signals who compose the ESL signal. For this purpose a pair of B2P blocks will be incorporated with 16-bit maximal-length LFSR as Random Number Generator [13], and different seeds. The setup ensures that there is no correlation between the two signals, and therefore emulates the fact that both signals are not affected by exactly the same bit changes.

Each B2P block combined with a XOR gate is in charge of flipping ($0 \rightarrow 1$ or $1 \rightarrow 0$) one of the stochastic signals composing the ESL magnitude with a given noise rate, as shown in Fig. 1e.

The dependency between computational performance (evaluation time) and noise-tolerance of the ESL computational blocks will be analyzed for different P2B blocks with 12, 16, 20 or 24-bit counters (Fig. 1f).

The experimental setup is composed by a low-range FPGA (Cyclone II, EP2C10F484C7N) assembled on the Altera Corp. Terasic DE1 educational board, a USB-Blaster to program the FPGA, a USB-to-RS232 converter to interconnect the PC with the development board, and a high end PC. The proposed ESL blocks are coded in VHDL and synthesized with the Altera Quartus II software suit (version 13.0SP1). In addition, an UART and a state machine on the FPGA to communicate the board with the PC has been incorporated, which allows the automated data acquisition with a MATLAB-based PC application.

3.1. *ESL blocks noise-tolerance*

Before presenting the noise-tolerance experimental results for the different ESL blocks, it is necessary to define the concept of noise rate and the mathematical function used to estimate the ESL blocks output error.

The maximum allowable noise ratio (100% of noise injection) of an ESL coded magnitude/probability is defined as the probability value of noise injection causing the complete loss of information transmitted by the pair of stochastic signals (p^* and q^*). This situation occurs when the number of ones in the stochastic signal sequences is equal to the number of zeros (probability of 50%), which corresponds to signal values of $p^* = q^* = 0$ and an indeterminate value of the ESL quotient. Therefore, the noise rate is given in the range from 0% to 100%, which corresponds to a swap of the probability r_{Noise} between [0, 0.5], as shown below.

$$Noise\_Rate[\%] = 200 \cdot r_{Noise}, \forall r_{Noise} \in [0, 0.5] \tag{20}$$

On the other hand, the ESL block output error for each injected noise rate will be calculated as the ratio between the mean absolute error and the average of the absolute value of the theoretical output (so that the negative values of the function are properly taken into account).

$$Error\,[\%] = \frac{\frac{1}{u}\sum_{i=1}^{u} \left|ESL_{out}\,(noise\%) - ESL_{out}\,(theo.)\right|}{\frac{1}{u}\sum_{i=1}^{u} \left|ESL_{out}\,(theo.)\right|} \cdot 100 \tag{21}$$

Table 2
ESL probabilistic blocks output errors for different noise rates (16-bit P2B)

	ESL adder		ESL multiplier		ESL square	
Noise injection [%]	Error [%]	σ	Error [%]	σ	Error [%]	σ
0	1.960	0.017	2.058	0.124	2.935	0.025
10	2.423	0.022	2.241	0.124	3.493	0.043
20	2.738	0.022	2.484	0.125	4.114	0.057
30	2.696	0.027	2.656	0.125	5.186	0.076
40	3.688	0.032	2.993	0.127	6.290	0.092
50	3.839	0.032	4.604	0.133	8.715	0.139
60	5.008	0.048	4.372	0.130	13.163	0.208
70	8.081	0.065	6.035	0.141	22.618	0.406
80	9.306	0.089	7.857	0.156	66.406	1.784
90	22.784	0.253	15.517	0.246	100	–
95	50.206	0.520	33.189	0.970	100	–
97.5	100	–	100	–	100	–
100	100	–	100	–	100	–

This ratio is finally multiplied by a factor of 100 to obtain the percentage of error as shown in Eq. (21), where u is the number of measurements considered for the error calculation.

3.1.1. *Adder & subtractor*

The experimental measurements of the ESL adder/subtractor block were performed by adding two ESL coded signals $z^* = y^* + x^*$. The first one $(y^* = r^*/s^*)$ was set to a constant value of 0.5 while the second one $(x^* = p^*/q^*)$ was varied in the $[-2, 2]$ range by steps of 0.015. These measurements were repeated for different noise rates with 10% increments in the range [0, 100%]. Some additional steps were made in the range of 90% to 100% to properly display the error in this interval. 16-bit P2B blocks have been used to convert the ESL pulsed output signals which corresponds to an evaluation period of 2^{16} FPGA global clock cycles @50 MHz.

The block operation results obtained for different noise ratio injection values (0%, 30%, 60% and 90%) are shown in Fig. 5a, where a good agreement can be observed between the block operation and theoretical one. All signals present nearly the same behavior and the error remains smaller than 5% until a noise injection rate of 60% is surpassed.

The complete set of error results of the ESL adder/subtractor obtained according to Eq. (21) for the different noise injection rates are presented in Table 2. The standard deviation has been calculated for the absolute error of the measurements (the deviation assessed using the error expression in Eq. (21) does not bring useful information due to the large values obtained for small errors).

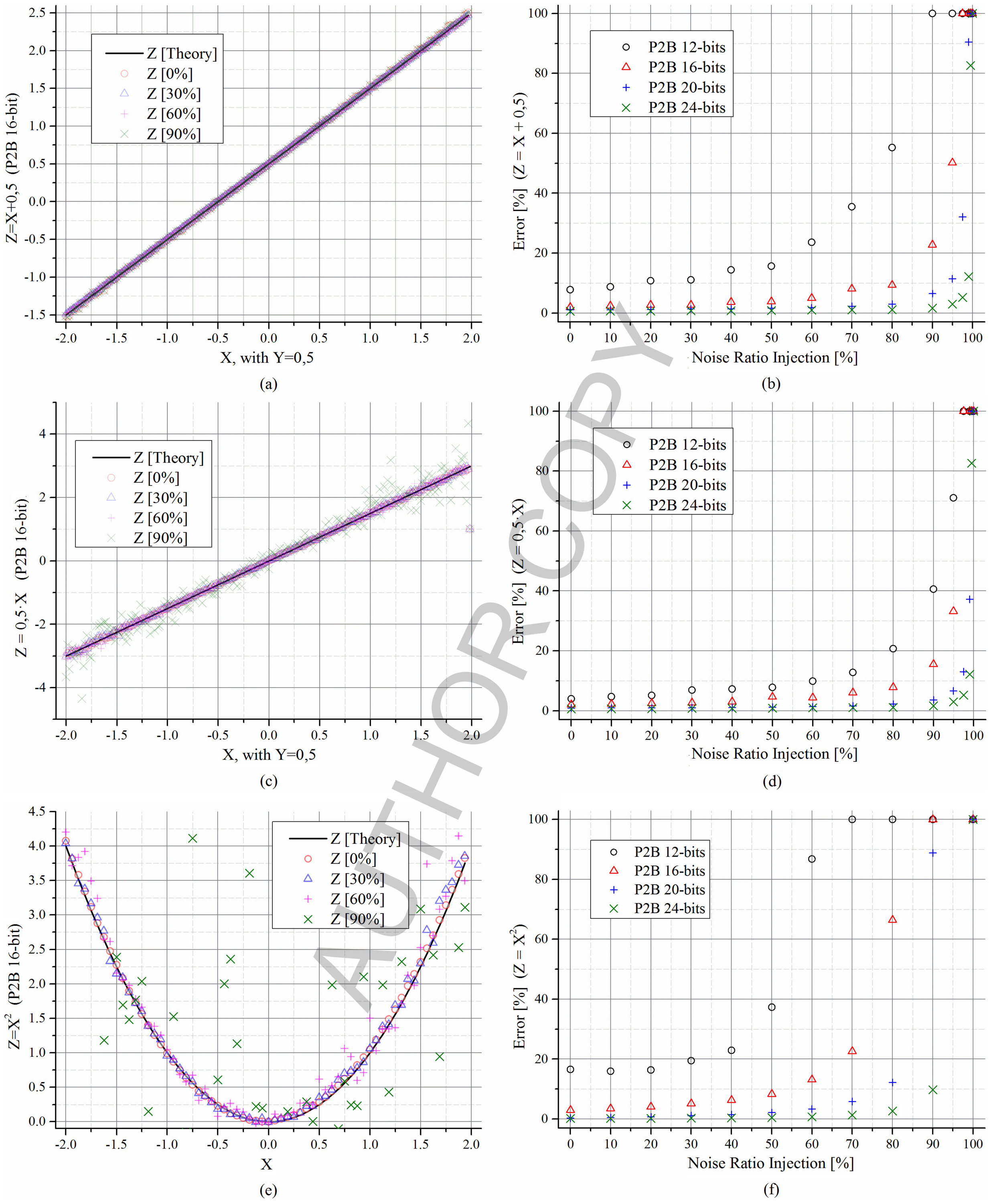

Fig. 5. (a) ESL adder output for different noise ratios. (b) ESL adder noise tolerance vs. P2B bit number (c) ESL multiplier output for different noise ratios. (d) ESL multiplier noise tolerance vs. P2B bit number. (e) ESL square output for different noise ratios. (f) ESL square noise tolerance vs. P2B bit number.

In addition, the noise-tolerance obtained by the ESL adder/subtractor block for different evaluation periods has been evaluated; configuring the P2B converters of the ESL architecture with different times: 2^{12}, 2^{16}, 2^{20} and 2^{24} clock cycles. The experimental results obtained for the different evaluation periods are shown in Fig. 5b, which illustrates how the inherent noise-tolerance of the ESL blocks can be adjusted with the architecture evaluation period. The worst result is obtained with the 12-bit counter and the best one with the 24-bit counter. The results using the 16, 20 and 24 bit counters are similar up to a 50% noise injection rate. It can be affirmed that noise-tolerance increases with the evaluation period.

3.1.2. *Multiplier & divider*

The measurements of the ESL multiplier/divider block were performed multiplying two ESL coded values $z^* = y^* \cdot x^*$. The first one $(y^* = r^*/s^*)$ was set to a constant of 1.5 while the second one $(x^* = p^*/q^*)$ was varied in the range [-2, 2] within 256 steps.

These measures were repeated for different noise injection rates with 10% increments in the range [0, 100%]. Some additional steps were made in the range from 90% to 100% to properly display the error in this interval. The results obtained with a 16-bit P2B block are shown in Fig. 5c. The experimental values are in accordance with the expected theoretical ones. All signals present nearly the same behavior and the error remains smaller than 5% until a noise injection rate of 60% is surpassed. The ESL multiplier set of error results is presented in Table 2.

The experimental results obtained for the different evaluation periods are shown in Fig. 5d. As expected, noise immunity increases with the evaluation period. It can be observed that the errors obtained with the ESL multiplier are significantly lower than those obtained with the ESL adder. This is mainly due to the fact that the ESL multiplier is made up of less logical elements than the ESL adder, since each logic gate has associated some uncertainty or error that adds to the block's output.

3.1.3. *Square*

The ESL square block experimental measurements were performed squaring an ESL value $z^* = (x^*)^2$, which input was varied in the range [-2, 2] within 64 steps. In this case, the 16-bit B2P used to generate the input signals was equipped with a 32-bit LFSR to avoid problems of temporal correlation between the signal and the delayed signal. The results using a 16-bit P2B block are depicted in Fig. 5e.

Table 3
ESL architecture bandwidth in function of P2B number of bit

	P2B number of bit			
	12-bit	16-bit	20-bit	24-bit
Period:	81.92 us	1.31 ms	20.97 ms	335.54 ms
Frequency:	12.21 kHz	762.93 Hz	47.68 Hz	2.98 Hz
# Averages:	0.063	1	16	256

Table 4
ESL noise-tolerance analysis as a function of P2B number of bits

	Adder [%]	Multiplier [%]	Square [%]
a^+	−0.92	−0.32	−0.80
b^+	39.40	14.82	36.81
c^+	−320.54	−70.75	−335.50
r^{2+}	0.98	0.99	0.99

$^+$Polynomial Fit Parameters: Noise tolerance $[\%] = a{\cdot}n^2{+}b{\cdot}n{+}c$, n: P2B number of bits.

As in previous cases, there is a good agreement between experimental values obtained and the theoretically expected. However, a 5% error is reached by this block with only a noise injection rate of the 30%. The ESL square complete set of error results are presented in Table 2. Additionally, the experimental results obtained for several architectures and evaluation periods are graphed in Fig. 5f. As expected, noise-tolerance increases with the evaluation period. The errors obtained for this block are significantly higher than those obtained for the previous blocks (at any noise rate and evaluation period).

3.2. *ESL noise-tolerance dependence with architecture evaluation period*

This subsection analyzes and discusses the relationship between the different ESL-block noise-tolerance reached and the evaluation period of the probabilistic computing architecture.

With this purpose, the maximum allowable noise rate for two maximum output error percentages set as 5% and 10% will be analyzed. The limiting noise is evaluated as a function of the number of bits of the P2B converter used for the stochastic to binary conversion (related to the evaluation period as shown in Table 3 using a system clock frequency of 50 MHz).

The obtained results are plotted in Fig. 6a (for a maximum error of 5%) and in Fig. 6b (for a maximum error of 10%). Notice how the noise immunity of the ESL blocks increases as a second order polynomial of number of bits of P2B converter used by stochastic architecture. As is shown in Fig. 6a and b the behavior of

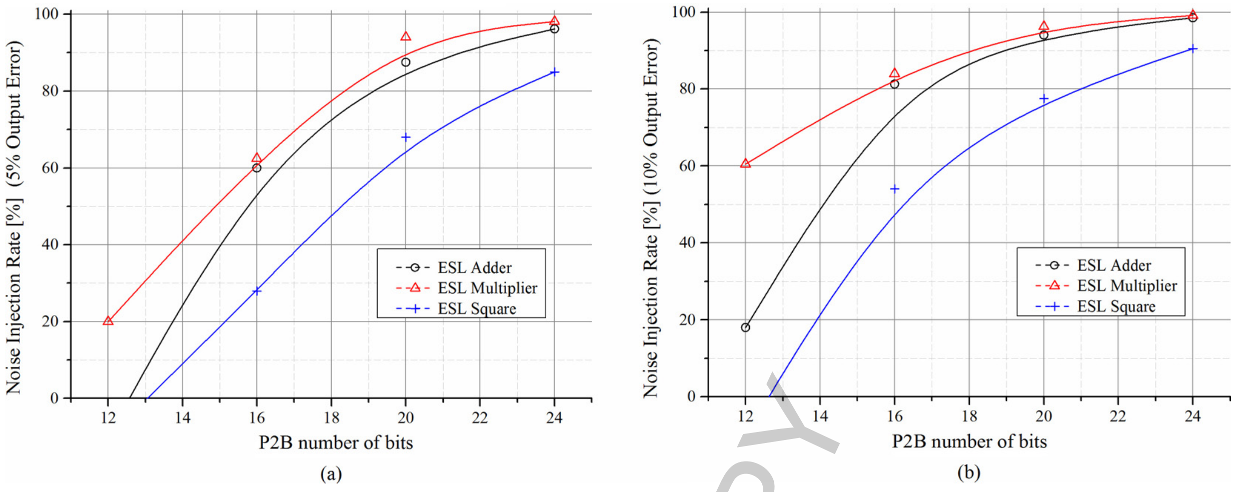

Fig. 6. (a) Noise tolerance archived vs. P2B bit number with 5% error. (b) Noise tolerance archived vs. P2B bit number with 10% error.

the noise immunity is very similar for both cases. The desired noise immunity levels are reached for all the analyzed ESL blocks with the architectures equipped with 16, 20 and 24-bit P2B modules. However, with the architectures equipped with a 12-bit P2B, the 5% maximum allowed error has been only reached by the multiplier, while for the 10% maximum error has been reached for all the blocks except the square function.

Meanwhile, Table 4 presents how the noise-tolerance reached by each ESL block follows a second order polynomial with the architecture P2B number of bits.

Finally, it is worth highlighting that for the 24-bit P2B architecture all the analyzed ESL blocks reach a noise robustness value higher than the 80% for a 5% of maximum output error, and noise robustness higher than the 90% for a 10% error.

3.3. *ESL noise-tolerance capabilities discussion*

This subsection aims to discuss the goodness of the ESL noise-tolerance results in front of the conventional circuits, with and without hardening. For this comparison, the ESL multiplier block has been chosen, combined with a 16-bit probabilistic architecture. It should be noted that the ESL multiplier is the simplest circuit analyzed. Therefore, as a conventional equivalent has been chosen an 8×8-bit combinational full-multiplier [22] that outputs a 16-bit value. This resolution is equivalent to the proposed 16-bit ESL architecture. To implement the conventional multiplier it has been interconnected with 64 Full Adder (FA) circuits.

Starting from the full-multiplier module has been developed a variant that incorporates space redundancy hardening. Specifically, the block includes 192 TMR FA and 64 voter circuits. The TMR is the most common scheme to perform SEU and electromagnetic waves hardening [22]. Both circuits have been synthesized on a Terasic DE1 educational board (Altera Corp.), equipped with an FPGA (Cyclone II, EP2C10F484C7N). The FPGA resources and circuit performance are presented in Table 5.

Finally, an additional circuit has been developed in order to randomly change bits from the input bytes to multiply. This is performed in a similar manner to the one used in ESL blocks testing. The noise-tolerance measurements have been performed in an automated way, taking 1000 measurements for each noise rate (256 k measurements), to subsequently evaluate the related error percentage for this noise rate. The error percentages obtained for the three modules versus the noise rates are presented in Fig. 7. These results show clearly how ESL probabilistic architecture is more noise-tolerant than the conventional solutions. The ESL encoding archives a multiplier output error of 10% for an 85% noise injection, while the conventional multiplier and the TMR reaches the same error for a noise injection of 0.4% and 2.7% respectively.

The hardware resources used are shown in Table 5 for the different implementations. The ESL consumes more resources than the conventional multiplier, although much less than the TMR.

For this estimation, the four B2P 16-bit modules and two P2B 16-bit modules to convert ESL signals has been considered. The ESL circuit presents the lower delay and therefore a greater operation frequency, at use less logic gates than the other two.

Table 5
Multipliers FPGA hardware resources and performance

	8 × 8 multiplier	8 × 8 TMR multiplier	ESL multiplier
Total logic elements (LE):	160/18,752 (0.853%)	444/18,752 (2.368%)	210/18,752 (1.120%)
Total combinational functions:	160/18,752 (0.853%)	444/18,752 (2.368%)	169/18,752 (0.901%)
Dedicated logic registers:	0/18,752 (0%)	0/18,752 (0%)	148/18,752 (0.789%)
Total memory bits:	0/239,616 (0%)	0/239,616 (0%)	0/239,616 (0%)
Embedded multiplier 9-bit element:	0/52 (0%)	0/52 (0%)	0/52 (0%)
Worst propagation delay[a] (ns):	23.451	36.095	9.756
F_{max} (MHz):	21.321	13.988	51.251
Number of clock cycles (@f_{max}) to obtain a correct circuit output:	1	1	65,536

[a]: Obtained with Altera TimeQuest Timing Analyzer. Slow-model worst-case timing paths, 1.2 V, 85°C.

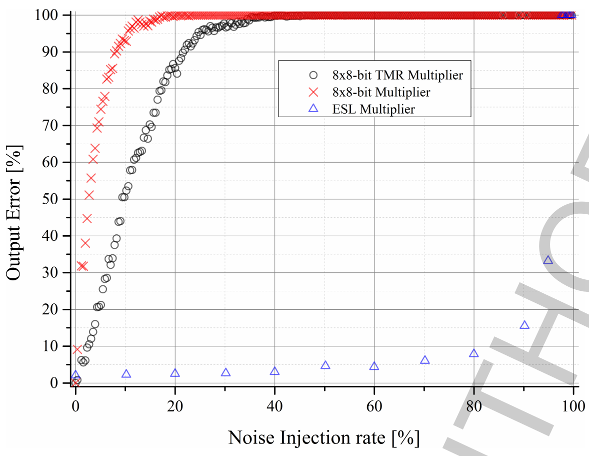

Fig. 7. Multipliers errors percentages vs. noise rate injection.

The evaluation period for ESL is very large related to the computation time. However, probabilistic systems unlike conventional ones do not have a fixed period of computation that depends on the resolution needed for a given application. In general, knowledge based applications implemented with probabilistic computing uses a low number of bits [35] for the conversion. This implies a minor noise-tolerance at the conversion, as discussed in the previous section.

4. Conclusions

This work analyzed the noise-tolerance and the statistical pattern recognition capabilities of the ESL. The ESL architecture demonstrated the ability to implement the main statistical rules and basic statistical classifiers, such as a two-class Bayesian classifier. The experimental results show that the proposed architecture can be useful to implement complex statistical pattern recognition applications.

The ESL has been shown to be an amazing noise-tolerant architecture due to its inherent redundancy. The use of two stochastic signals ratio to encode the information, one to express the numerator and the second for the denominators is the key to remove any presence of noise since it will be included in each signal in the same proportion. The analyzed ESL blocks exhibit an output error smaller to 10% for noise injections up to 90%, with 24-bit architectures. The architecture allows adjusting output noise-tolerance performance as a function of the size of the P2B converters (related to the evaluation period), as a second order polynomial, establishing a relationship between the computational performance and the noise-tolerance. Finally, a faithful comparison with conventional binary logic and TMR architectures is presented. The results show that, even that the total circuit area is not significantly increased when using the ESL architecture with respect the conventional binary logic (with a slight increase of a 30% in resources in comparison with a 176% of area increase for the TMR), its noise tolerance capabilities outperforms the TMR results. This is clearly shown in Section 3.2 where the output error obtained with the TMR solution crosses the 10% when the injected noise is higher than the 2.7% while the ESL architecture needs a noise level of the 85% to provide the same 10% of error. The ESL approach is a system-level noise-tolerant architecture and not a silicon level one, and therefore it can operate in any standard CMOS technologies or FPGAs. This approach can be complemented with other noise-hardening techniques, in order to protect the P2B converters registers and the auxiliary conventional digital circuits.

The main drawback of the proposed architecture is the low computational capacity when a high resolution or high noise-tolerance digital outputs a needed (2^{16} clock cycles for 16-bit P2B blocks needed to obtain a result), with respect to the conventional and TMR architectures.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (MINECO), the Regional European Development Funds (FEDER) under grant contract TEC2014-56244-R.

References

[1] Acharya UR, Bhat S, Adeli H, Adeli A. Computer-aided diagnosis of alcoholism-related EEG signals. Epilepsy Behav 2014 Dec; 41: 257-63.
[2] Adeli H, Park HS. Fully automated design of superhighrise building structure by a hybrid AI model on a massively parallel machine. AI Mag 1996; 17(3): 87-93.
[3] Adeli H, Samant A. An adaptive conjugate gradient neural network-wavelet model for traffic incident detection. Comput Civ Infrastruct Eng 2000 Jul; 15(4): 251-60.
[4] Adeli H. Parallel processing in computational mechanics. New York, NY, USA: Marcel Dekker, Inc. 1992.
[5] Adeli H. Supercomputing in engineering analysis. New York, NY, USA: Marcel Dekker, Inc. 1992.
[6] Adeli H, Kamal O. Parallel processing in structural engineering. London: Elsevier Applied Science, 1993.
[7] Adeli H, Hung S-L. Machine learning: Neural networks, genetic algorithms, and fuzzy systems. John Wiley and Sons, New York 1995. xii, 211.
[8] Adeli H, Kamal O. Concurrent optimization of large structures. I: Algorithms. J Aerosp Eng 1992 Jan; 5(1): 79-90.
[9] Adeli H, Kumar S. Distributed computer-aided engineering: For analysis, design, and visualization. 1st ed. Boca Raton, FL, USA: CRC Press, Inc. 1998.
[10] Alaghi A, Chan W-TJ, Hayes JP, Kahng AB, Li J. Trading accuracy for energy in stochastic circuit design. ACM J Emerg Technol Comput Syst 2017 Apr 20; 13(3): 1-30.
[11] Alaghi A, Hayes JP. Survey of stochastic computing. ACM Trans Embed Comput Syst 2013 May 1; 12(2s): 1-19.
[12] Alaghi A, Hayes JP. Exploiting correlation in stochastic circuit design. In: 2013 IEEE 31st International Conference on Computer Design (ICCD). IEEE 2013; 39-46.
[13] Alfke P. Efficient shift registers, LFSR counters, and long pseudo-random sequence generators. http://www.xilinx.com/bvdocs/appnotes/xapp052.pdf. 1998.
[14] Brown BD, Card HC. Stochastic neural computation. I. Computational elements. IEEE Trans Comput 2001; 50(9): 891-905.
[15] Canals V, Alomar ML, Morro A, Oliver A, Rossello JL. Noise-robust hardware implementation of neural networks. In: 2015 International Joint Conference on Neural Networks (IJCNN). IEEE 2015; 1-8.
[16] Canals V, Morro A, Oliver A, Alomar ML, Rossello JL. A new stochastic computing methodology for efficient neural network implementation. IEEE Trans Neural Networks Learn Syst 2016 Mar; 27(3): 551-64.
[17] Canals V, Morro A, Rosselló JL. Stochastic-based pattern-recognition analysis. Pattern Recognit Lett 2010 Nov 20; 31(15): 2353-6.
[18] D'Alessio M, Ottavi M, Lombardi F. Design of a nanometric CMOS memory cell for hardening to a single event with a multiple-node upset. IEEE Trans Device Mater Reliab 2014 Mar; 14(1): 127-32.
[19] Ding L, Mazumder P. On circuit techniques to improve noise immunity of CMOS dynamic logic. Very Large Scale Integr Syst IEEE Trans 2004 Sep; 12(9): 910-25.
[20] Dong QT, Arzel M, Jego C, Gross WJ. Stochastic decoding of turbo codes. IEEE Trans Signal Process 2010 Dec; 58(12): 6421-5.
[21] Engelbrecht A. Computational intelligence: An introduction, 2nd Edition. John Wiley & Sons Ltd. 2007; 628.
[22] Floyd T. Digital fundamentals. Pearson Education Limited 2009; 912.
[23] Hamilton TJ, Afshar S, van Schaik A, Tapson J. Stochastic electronics: A neuro-inspired design paradigm for integrated circuits. Proc IEEE 2014 May; 102(5): 843-59.
[24] Hirschauer TJ, Adeli H, Buford JA. Computer-aided diagnosis of parkinson's disease using enhanced probabilistic neural network. J Med Syst 2015 Nov 29; 39(11): 179.
[25] Hsu H-L, Adeli H. A microtasking algorithm for optimization of structures. Int J High Perform Comput Appl 1991 Jun 1; 5(2): 82-91.
[26] Hung SL, Adeli H. Parallel backpropagation learning algorithms on CRAY Y-MP8/864 supercomputer. Neurocomputing 1993 Nov; 5(6): 287-302.
[27] Keyes RW. Fundamental limits of silicon technology. Proc IEEE 2001 Mar; 89(3): 227-39.
[28] Kish LB. Moore's law and the energy requirement of computing versus performance. IEE Proc-Circuits, Devices Syst 2004; 151(2): 190.
[29] Li B, Najafi MH, Lilja DJ. Using stochastic computing to reduce the hardware requirements for a restricted boltzmann machine classifier. In: Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays – FPGA '16. New York, USA: ACM Press 2016; 36-41.
[30] Li X, Qian W, Riedel MD, Bazargan K, Lilja DJ. A reconfigurable stochastic architecture for highly reliable computing. In: Proceedings of the 19th ACM Great Lakes symposium on VLSI – GLSVLSI '09. New York, USA: ACM Press 2009; 315.
[31] May TC, Woods MH. Alpha-particle-induced soft errors in dynamic memories. Electron Devices, IEEE Trans 1979; 26(1): 2-9.
[32] Mezhiba AV, Friedman EG. Scaling trends of on-chip power distribution noise. Very Large Scale Integr Syst IEEE Trans 2004; 12(4): 386-94.
[33] Moore GE. Cramming more components onto integrated circuits, Reprinted from Electronics, volume 38, number 8, April 19, 1965, pp. 114 ff. IEEE Solid-State Circuits Society Newsletter 2006 Sep; 20(3): 33-35.
[34] Morro A, Canals V, Oliver A, Alomar ML, Galan-Prado F, Ballester PJ, et al. A stochastic spiking neural network for virtual screening. IEEE Trans Neural Networks Learn Syst 2017; 1-5.
[35] Morro A, Canals V, Oliver A, Alomar ML, Rossello JL. Ultra-fast data-mining hardware architecture based on stochastic computing. Lisacek F, editor. PLoS One 2015 May 8; 10(5): e0124176.
[36] Neuberger G, de Lima F, Carro L, Reis R. A multiple bit upset tolerant SRAM memory. ACM Trans Des Autom Electron Syst 2003 Oct 1; 8(4): 577-90.
[37] Nicolaidis M. Design for soft error mitigation. Device Mater Reliab IEEE Trans 2005 Sep; 5(3): 405-18.
[38] Normand E. Single event upset at ground level. Nucl Sci IEEE Trans 1996; 43(6): 2742-50.
[39] O'Reilly RC, Frank MJ, FéřŐster A, Alex Graves JS, Mayer H, Gomez F, Wierstra D, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. PLoS Comput Biol 2013; 7(1): 11.
[40] Onizawa N, Katagiri D, Matsumiya K, Gross WJ, Hanyu T. Gabor filter based on stochastic computation. IEEE Signal Process Lett 2015 Sep; 22(9): 1224-8.
[41] Osada K, Yamaguchi K, Saitoh Y, Kawahara T. SRAM immunity to cosmic-ray-induced multierrors based on analysis of an induced parasitic bipolar effect. Solid-State Circuits, IEEE J. 2004 May; 39(5): 827-33.
[42] Park HS, Adeli H. Data parallel neural dynamics model for

integrated design of large steel structures. Comput Civ Infrastruct Eng 1997 Sep; 12(5): 311-26.

[43] Qian W, Li X, Riedel MD, Bazargan K, Lilja DJ. An architecture for fault-tolerant computation with stochastic logic. IEEE Trans Comput 2011 Jan; 60(1): 93-105.

[44] Qin H, Cao Y, Markovic D, Vladimirescu A, Rabaey J. SRAM leakage suppression by minimizing standby supply voltage. In: Quality Electronic Design, 2004 Proceedings 5th International Symposium on 2004. 55-60.

[45] Rosselló JL, Alomar ML, Morro A, Oliver A, Canals V. High-density liquid-state machine circuitry for time-series forecasting. Int J Neural Syst 2016 Aug 14; 26(5): 1550036.

[46] Rossello JL, Canals V, Morro A. Hardware implementation of stochastic-based neural networks. In: The 2010 International Joint Conference on Neural Networks (IJCNN). IEEE 2010; 1-4.

[47] Rosselló JL, Canals V, Morro A, Oliver A. Hardware implementation of stochastic spiking neural networks. Int J Neural Syst 2012 Aug; 22(4): 1250014.

[48] Saleh A, Adeli H. Microtasking, and autotasking for structural optimization. J Aerosp Eng 1994 Apr; 7(2): 156-74.

[49] Saleh A, Adeli H. Robust parallel algorithms for solution of riccati equation. J Aerosp Eng 1997 Jul; 10(3): 126-33.

[50] Saraf N, Bazargan K, Lilja DJ, Riedel MD. IIR filters using stochastic arithmetic. In: Design, Automation & Test in Europe Conference & Exhibition (DATE 2014). New Jersey: IEEE Conference Publications 2014; 1-6.

[51] Sartori J, Sloan J, Kumar R. Stochastic computing: Embracing errors in architecture and design of processors and applications. In: Proceedings of the 14th International Conference on Compilers, Architectures and Synthesis for Embedded Systems – CASES '11. New York, USA: ACM Press 2011; 135.

[52] Shivakumar P, Kistler M, Keckler SW, Burger D, Alvisi L. Modeling the effect of technology trends on the soft error rate of combinational logic. In: Proceedings International Conference on Dependable Systems and Networks. IEEE Comput Soc 2002; 389-98.

[53] Siddique N, Adeli H. Computational intelligence. Oxford, UK: John Wiley & Sons Ltd 2013.

[54] Soegiarso R, Adeli H. Impact of vectorization on large-scale structural optimization. Struct Optim 1994 Feb; 7(1-2): 117-25.

[55] Theodoridis S, Koutroumbas K. Pattern recognition. Elsevier Inc. 2009.

[56] Tosaka Y, Satoh S, Suzuki K, Sugii T, Ehara H, Woffinden GA, et al. Impact of cosmic ray neutron induced soft errors on advanced submicron CMOS circuits. In: VLSI Technology, 1996 Digest of Technical Papers 1996 Symposium on 1996; 148-9.

[57] Wang N, Adeli H. Self-constructing wavelet neural network algorithm for nonlinear control of large structures. Eng Appl Artif Intell 2015 May; 41: 249-58.

[58] Yang Y-B, Li Y-N, Gao Y, Yin H, Tang Y. Structurally enhanced incremental neural learning for image classification with subgraph extraction. Int J Neural Syst 2014; 24(7): 1450024.

[59] Zhou Q, Mohanram K. Gate sizing to radiation harden combinational logic. Comput Des Integr Circuits Syst IEEE Trans 2006; 25(1): 155-66.