Research Article

Error Immune Logic for Low-Power Probabilistic Computing

Bo Marr, Jason George, Brian Degnan, David V. Anderson, and Paul Hasler

School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

Correspondence should be addressed to Bo Marr, marr.bo@gmail.com

Received 27 May 2009; Accepted 19 November 2009

Academic Editor: Gregory D. Peterson

Copyright © 2010 Bo Marr et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Two novel theorems are developed which prove that certain logic functions are more robust to errors than others. These theorems are used to construct datapath circuits that give an increased immunity to error over other naive implementations. A link between probabilistic operation and ultra-low energy computing has been shown in prior work. These novel theorems and designs will be used to further improve probabilistic design of ultra-low power datapaths. This culminates in an asynchronous design for the maximum amount of energy savings per a given error rate. Spice simulation results using a commercially available and well-tested $0.25\ \mu\text{m}$ technology are given verifying the ultra-low power, probabilistic full-adder designs. Further, close to 6X energy savings is achieved for a probabilistic full-adder over the deterministic case.

1. Introduction

As digital technology marches on, ultra-low voltage operation, atomic device sizes, device mismatch, and thermal noise are becoming commonplace and so are the significant error rates that accompany them. These phenomena are causing ever increasing bit-error rates, and with billion-transistor digital chips being produced today, even a 1-in-100-million bit error rate becomes costly.

This paper will present a novel discovery of boolean logic that certain logic gates are more robust to error than others, and in fact it will be shown that some logic even *improves* the error rate just through natural computation. The paper will show how these principles translate into CMOS and other implementations, but these principles are *independent of technological implementation* since they are properties of boolean logic itself. Thus these design principles will stand the test of time.

The motivation behind studying computing architectures robust to error then becomes clear, especially as technological breakthroughs have recently shown that extreme power savings can be traded for a certain level of error—known as probabilistic computing [1]. Paying more attention to error rates is critical if scaling power consumption and devices is to continue [2].

Recently, ultra-low power computing has been achieved by lowering the supply voltage of digital circuits into near threshold or even the subthreshold region [1, 3]. Indeed a fundamental limit to voltage scaling technology has been proposed: the thermodynamic limit of these devices [4]. When the supply voltage becomes comparable to thermal noise levels in these types of ultra-low power designs, devices start to behave probabilistically giving an incorrect output with some nonzero probability [4, 5]. Kish predicts that the thermal noise phenomenon will result in the “death” of Moore’s law [6]. The International Technology Roadmap for Semiconductors predicts that thermal noise will cause devices to fail catastrophically during normal operation—without supply voltage scaling—in the next 5 years [6, 7].

A paradigm-shifting technology has been introduced in part by the authors called probabilistic CMOS or pcmos to combat the failure in voltage scaling due to the kT/q thermal noise limit. Experiments have already been completed that show that this thermal noise barrier can be overcome by computing with deeply scaled *probabilistic* CMOS devices. It was shown, in part by the authors, that probabilistic operation of devices allows for applications in arithmetic and digital signal processing that use a fraction of the power of their deterministic counterparts [1, 8]. This paper builds upon this work, and offers improved solutions for ultra-low power datapath units.

Logic gates are the fundamental building blocks of all digital technology, and it has been discovered that not all logic gates propagate error equally *regardless of technological implementation*. The main contributions of this paper are the following.

- (i) A novel property of boolean logic is presented showing that different types of boolean logic propagate errors much differently *regardless of implementation or technology generation*
- (ii) Several theorems will be given that offer a guideline for logic that most reduces error rates.
- (iii) A case study using full-adders is given using these principles to achieve reduced error rates strictly through intelligent implementation (e.g., without error correction logic).
- (iv) An analysis is given for asynchronous logic showing that a lower error rate is achieved when compared to its synchronous counterparts.
- (v) It is shown that reducing error rates can be translated into reduced power consumption via probabilistic computing.

This work is an expansion of the work proposed in [9]. The background to this research and related work is given in Section 2. Some definitions and assumptions are given in Section 3. A case study analyzing which implementation is optimal for noisy full-adders is given in Section 5. Theorems illustrating the reliability of logic circuits under a probabilistic fault model are presented in Section 4. Conclusions and future directions are discussed in Section 6.

2. Logic Tables and Device Physics

2.1. All Logic Is Not Created Equal. Boolean logic functions are most simply represented by a *Truth Table* mapping each input combination to an output. When a bit-error is present at the input, or one of the input bits is flipped in other words, if this new input combination is mapped to the same output, then no error results in calculating the given logic function. In this case the logic function did not *propagate* the error. If one assumes a small, given error rate per bit, then a single-bit error is more likely than two simultaneous bit errors which is more likely than three simultaneous errors, and so forth. A logic function that has the least input-output mappings where a single-bit error on an input causes a mapping to a different output will be the least likely to propagate a bit error. This phenomenon is shown in Figures 1(a) and 1(b) using an NAND function and an XOR function as an example.

Figure 1 illustrates the theme that not all logic gates propagate errors equally. If one calculates the average probability of error across all possible input combinations of NAND and XOR logic, an extremely interesting result emerges. Assume that a probability of error of $\epsilon = 0.2$ is present at the inputs of these gates. Further assume that full-adders are built such that a probability of error at the input $\epsilon = 0.2$ is also present and that one of the full-adders is built with an NAND-NAND implementation and

NAND truth table		NAND w/ input error			
		Bit flip	In AB	Out	Out error?
A	10	1	01	1	
None	00	1	01	1	
B	01	1	11	0	Error!
A	11	0	01	1	
None	01	0	11	0	Error!
B	00	0	11	0	Error!
A	00	1	10	1	
None	10	1	00	1	
B	11	0	00	1	
A	01	1	11	0	Error!
None	11	0	01	1	
B	10	1	01	1	
# output errors/possible					4/8

XOR truth table		XOR w/ input error			
		Bit flip	In AB	Out	Out error?
A	10	1	01	1	Error!
None	00	0	00	0	
B	01	1	11	0	Error!
A	11	0	01	1	Error!
None	01	0	11	0	Error!
B	00	0	11	0	Error!
A	00	0	10	1	Error!
None	10	1	00	1	Error!
B	11	0	00	1	Error!
A	01	1	11	0	Error!
None	11	0	01	1	Error!
B	10	1	01	1	Error!
# output errors/possible					8/8

FIGURE 1: The effect of a single-bit error or bit flip on the inputs of an NAND function and XOR function is shown. Note that if the new input combination resulting from a bit flip maps to the same output as the input combo with no bit flip, no error is seen at the output. A NAND function is much less likely to propagate an input bit flip to the output resulting in an error than an XOR function. (a) NAND function (b) XOR function.

the other is built with a standard XOR implementation. Figure 2 shows the drastically differing output probabilities δ that result. Note that in Figure 2, the gate itself has no probability of computing erroneously associated with it. This shows that different gates have differing abilities to mask errors regardless of input combination.

FIGURE 2: The probability of output error δ , of a NAND versus an XOR when a probability error of $\epsilon = 0.2$ is present at the input. Further, δ is compared for a full-adder implemented with NAND-NAND and with a standard XOR when $\epsilon = 0.2$ is present at the input. Note that not only is the probability of output error δ much lower for a NAND gate than an XOR gate, but that the error propagated from the NAND is *improved* over the input error. In a sense, a NAND “heals” the circuit. Note that the gate itself does not compute erroneously in this example.

As shown in Figure 2, the NAND gate has a much lower output probability of error, δ , than the XOR gate when under the same input error conditions. An interesting property also emerges that a NAND actually propagates an *improved* error probability due to its logic properties through its natural computation. Obviously, a $\epsilon = 0.2$ is orders of magnitude higher than the actual error rates seen in digital logic, but it will later be shown that the output error probability of logic gates monotonically decreases as a function of decreasing error rates and thus this concept holds true at more realistic bit-error rates. Subsequent sections will go into more depth explaining these properties.

2.2. MOSFET Implementation and Device Properties. There are two key issues that cause device failures in digital logic implemented with MOSFET transistors: device mismatch and thermal noise. This work will address the thermal noise problem; device mismatch will be addressed in future works. As for other types of noise, the switching speed of devices is faster than the frequencies typically seen for the $1/f$ type of noise (flicker noise), so there is no need to address that effect. A plot of thermal noise measured from a $0.35 \mu\text{m}$ chip is shown in Figure 3(a). The derivation of the probability of a digital “1” or “0” is seen in Figure 3(b) using a comparator.

Device mismatch continues to get worse with minimum size devices as process sizes scale down. Mismatch problems can be mitigated with larger device sizes, but this is not always reasonable, increases power dissipation, and might give loss of computing performance. Other techniques are possible for tuning digital gates and is the subject of much recent work [10, 11] that has shown promise for programmatically tuning devices with floating gates. With sufficient calibration using these tuning techniques, the device mismatch effect can be removed. This result is quite significant, as interest in subthreshold digital circuits continues to increase. Mismatch in threshold voltage impacts circuits by $e^{\kappa \Delta V_t / U_t}$, where $U_t = kT/q$ or 25.4 mV. Hence a mismatch of 20 mV, which happens for minimum devices in $0.35 \mu\text{m}$ CMOS process, could result in a change of current by

(a)

(b)

FIGURE 3: (a) Amplified thermal noise measured from a $0.35 \mu\text{m}$ chip. (b) Plot showing the derivation of probability of a digital “1” or “0” when thermal noise and a probability select signal are both put through a comparator. Data was collected to verify probability of error when CMOS logic is subject to noise.

a factor of 2 [12]. As we scale down, this level of V_t mismatch continues to increase. The failure in this case is that timing of the gates somehow does not occur when expected on a probabilistic basis.

Even a conservative estimate of the thermal noise problem is disconcerting. The thermal noise magnitude for a digital gate is approximated as kT/C noise. For a small 1 fF capacitor, the rms noise level is roughly 2 mV. However, because of a trend of larger interconnect capacitances, large digital buffers, and higher operating temperatures due to transistor density, thermal noise has been shown to increase to 200 mV in even nanoscale digital circuits [5]. In low-power digital logic, a supply voltage in the hundreds of mV is not unreasonable especially in subthreshold logic and hence the probability of an error becomes exceedingly likely in this case [3]. Of course, one would need to look at the probability over a large number of gates, say 1 billion gates, and never having an error all the sudden gets interesting.

An estimate given by Kish in [6] predicts untenable device failure due to thermal noise at the 22 nm transistor node assuming $V_t = 0.2$ V, creating an argument for a not so distant problem. So far, device mismatch issues are the larger effect. However, thermal noise is the eventual limit that will be reached in low-power circuit design, and thus merits the treatment given here.

2.3. Theoretical Background for Probabilistic Gates. Design with probabilistic gates is an old problem that has been given a new application via ultra-low power computing. In [13], von Neumann showed two important results. Reliable circuits can be built with noisy (probabilistic) gates with an arbitrarily high reliability, but this circuit will compute more slowly than the same circuit built with noiseless gates due to necessary error correction logic. von Neumann refers to triple modular redundancy (TMR) and others refer to N-modular redundancy (NMR) to achieve these results shown in [13]; however, these techniques often result in high circuit overhead of as much as 200% or more. Pippenger showed exactly how much error correction would be needed [14] to achieve arbitrary reliability.

Pippenger improved upon this result by showing that the fraction of layers of a computing element that must be devoted to error correction for reliable computation is $2\epsilon/\ln 3$, but he was unable to show how such a computing element could be built [14].

Others have addressed soft error rate (SER) reduction and error masking techniques through time redundancy which catches transient errors due to particle strikes and delay-based errors [15, 16]. This work differs from these others because here we consider bits that are truly probabilistic due to thermal noise properties and the error rate is independent of time. Therefore, the shadow latching techniques and timing redundancy techniques presented in the aforementioned work are not as effective. Not to mention that these techniques pose the same drawback as TMR for time-independent errors in that there is significant overhead circuit involved and often involve pipeline flushes if an error is detected. Alternatively, in this work we show that logic synthesis can be done such that the fabric of the logic itself reduces error propagation.

The work presented in [17] presents a probabilistic analysis framework similar to the one presented in this paper and addresses adders and logic gates also similarly to this paper. However, they neither come to the conclusion of the relative error propagation characteristics of different logic networks nor are they able to link probability to power the way this paper does. It also presents a framework for analysis and not necessarily a solution to probabilistic errors.

Krishnaswamy et al. in [18] address probabilistic gates by using logic transformations to reduce error rates by considering that observability do not care (ODC) sets. This work differs from the current paper in that probabilistic thermal noise faults are considered herein, not deterministic stuck-at-faults. This work is superior to previous methods in the metric of energy savings in that no error correction logic or overhead is used.

Synthesizing probabilistic logic into CMOS using Markov Random Fields was presented in [8]. However, the CMOS circuits presented are not standard and use far more transistors (20 transistors for an inverter) than the gates proposed here.

Finally, Chakrapani et al. in [19] have done work in synthesizing probabilistic logic for inherently probabilistic applications, but do not address probabilistic design for deterministic applications.

FIGURE 4: (a) NAND2 ITM (b) NAND2 PTM.

3. Defining Probabilistic Gates via Matrices

A tight analysis of the differing error rates present in boolean logic is given. To give a framework for the discussion two definitions are presented.

Definition 1. ϵ is the probability that a bit flip occurred on a circuit node.

Definition 2. δ is the probability that the output of a network of gates is incorrect as a function of ϵ .

The assumptions used in this work are similar to previous works on the subject [13, 14]: $0 < \delta < 0.5$, $0 < \epsilon < 0.5$, and each node is assumed to fail (flip) independently and with a uniform probability, ϵ . All input combinations are assumed to be equally likely. Stein in [5] and Cheemalavagu et al. in [4] show that thermal noise can be modeled as errors occurring with a precise probability defined by the noise to V_{dd} ratio and that this noise can be modeled at the input or output of each gate, thus ϵ is defined at the nodes of the circuit. The probability model used in this paper is the same as in [4], which has been verified by running an HSPICE circuit simulation and measurements from a $0.25\mu m$ chip. “Min”, “Maj”, and “AOI” will be used throughout this paper which stand for the Minority gate, Majority gate, and And-Or-Inv gates, respectively, which are shown in [20]. To review a minority gate simply outputs the value equal to the value present on the minority of inputs. Thus for a 3-input minority function with the input “001” or “000” the output would be “1”.

Transfer matrix methodology is used for calculating probabilities at the outputs of a complex network of gates. This methodology allows the probability to be calculated in a strict, mathematically defined way regardless of input [21]. The ideal transfer matrix (ITM) and PTM for a NAND2 gate can be seen in Figure 4 where each entry in the matrix is the probability of that input-output combination occurring. The top row of numbers in Figure 4(a): “00”, “01”, “10”, “11” are the possible input combinations and the columns on the left side are the possible output values. An ITM simply is a matrix with a “1” as an entry that corresponds to the correct output for a given input combination. In other words, $\epsilon = 0$ for an ITM.

The PTM methodology is used in the definition of δ , illustrated in Figure 5. To calculate δ , all inputs are considered to be equally likely. Once the final probability transfer

FIGURE 5: This circuit illustrates the definitions for ϵ and δ . Symbol ϵ is the error probability for a node and δ is the probability that a network of gates is erroneous. For a single gate, δ is the error probability of that gate inclusive of the fact that the inputs may be erroneous. Each gate is represented by a matrix and the matrices from each level of the circuit are multiplied to get the final PTM.

matrix is calculated for the circuit, giving a probability of each output occurring for each input combination, the probability of error is then calculated by summing the probability of error for each input combination and then dividing by the number of input combinations. In Figure 5, there are 2 inputs to the circuit yielding 4 possible input combinations, thus the final sum is divided by 4 to get the average probability of error.

The algorithm described in [22] shows how to calculate an overall circuit PTM from individual gate PTMs. Briefly, one calculates the final PTM from the output. Each level of the circuit is represented by a matrix. Each level is then matrix multiplied with the level before it. If there are more than one gate at a given level, the PTM is calculated using the kronecker product of all gates.

4. Theorems for Noisy Gates

Two theorems proving properties of noisy gates are outlined here, which can be used as a guideline for designing circuits with minimal probability of error. The first theorem proves which gate types have the lowest probability of propagating an error, and the second one proves that the probability of an error at the output of a circuit increases as the depth of that circuit increases.

Theorem 1. *A noisy gate of 2 or 3 inputs will have a minimal probability of error, δ , when the cardinality of its ON-Set or OFF-Set is 1.*

Note that the cardinality of the ON-Set of an AND gate is 1 and the cardinality of the OFF-Set of a NAND gate is 1. In other words, there is 1 input-output mapping that gives a value of ON (1) and OFF (0), respectively.

Proof. The probability of an *correct* value at the output of the gate, $1 - \delta_{\text{gate}}$, is calculated in (1).

$$1 - \delta_{\text{GI}} = \frac{1}{C} \sum_{r=1}^R \sum_{c=1}^C \text{ITM}_{\text{GI}}(r, c) * \text{PTM}_{\text{GI}}(r, c), \quad (1)$$

where GI = Gates + Inputs, C = number of columns in transfer matrix, R = number of rows in transfer matrix.

The PTM for a gate including the inputs is the dot product of the PTM of the gate itself with the kronecker product of the PTM for inputs.

$$\text{PTM}_{\text{GI}} = \text{PTM}_{\text{G}} \cdot (A_1 \otimes A_2 \otimes \dots \otimes A_n), \quad (2)$$

where $A_1 \dots A_n$ are PTMs for inputs A_1 through A_n , G = Gates.

The δ error function is a monotonic function with respect to ϵ and thus so is $1 - \delta$.

The equation for the minimal δ as calculated from (1) is shown for each possible value of $y = \min(|F^{\text{ON}}|, |F^{\text{OFF}}|)$ in (3). For a 3-input function, $y \in \{1, 2, 3, 4\}$.

$$\begin{aligned} 1 - \delta_{y=1} &= \frac{1}{8} [8(1 - \epsilon)^4 + 18\epsilon(1 - \epsilon)^3 \\ &\quad \dots + 24\epsilon^2(1 - \epsilon)^2 + 12\epsilon^3(1 - \epsilon) + 2\epsilon^4], \\ 1 - \delta_{y=2} &= \frac{1}{8} [8(1 - \epsilon)^4 + 16\epsilon(1 - \epsilon)^3 \\ &\quad \dots + 20\epsilon^2(1 - \epsilon)^2 + 16\epsilon^3(1 - \epsilon) + 4\epsilon^4], \\ 1 - \delta_{y=3} &= \frac{1}{8} [8(1 - \epsilon)^4 + 14\epsilon(1 - \epsilon)^3 \\ &\quad \dots + 20\epsilon^2(1 - \epsilon)^2 + 16\epsilon^3(1 - \epsilon) + 6\epsilon^4], \\ 1 - \delta_{y=4} &= \frac{1}{8} [8(1 - \epsilon)^4 + 16\epsilon(1 - \epsilon)^3 \\ &\quad \dots + 16\epsilon^2(1 - \epsilon)^2 + 16\epsilon^3(1 - \epsilon) + 8\epsilon^4], \end{aligned} \quad (3)$$

where

$\delta_{y=k}$ δ for 3-input function

$$y = \min(|F^{\text{ON}}|, |F^{\text{OFF}}|), \quad (4)$$

ϵ probability of error at gate node.

The equation for δ for a each value of y for 2 input functions is given in (5).

$$\begin{aligned} 1 - (\delta_{y=1}) &= \frac{1}{4} [4(1 - \epsilon)^3 + 4\epsilon(1 - \epsilon)^2 + 6\epsilon^2(1 - \epsilon) + 2\epsilon^3], \\ 1 - (\delta_{y=2}) &= \frac{1}{4} [4(1 - \epsilon)^3 + 4\epsilon(1 - \epsilon)^2 + 4\epsilon^2(1 - \epsilon) + 4\epsilon^3]. \end{aligned} \quad (5)$$

FIGURE 6: δ versus ϵ for a NAND2 gate and an XOR2 gate.

For $0 < \epsilon < 0.5$, the probability of error, δ , for 2 and 3 input gates is minimal for $y = 1$. $\square$

Corollary 1. *NAND and NOR have the minimal probability of error, δ , of any gate type subjected to noise for 2 and 3 inputs. Among the set of 2 and 3 input gates subjected to noise for the set: {NAND, NOR, XOR, AOI, OAI, Maj, Min}, NAND and NOR have a lower probability of error than any other gate in this set.*

Proof. NAND and NOR are gates where $y = \min(|F^{ON}|, |F^{OFF}|) = 1$, and the other gates in the set are such that $y > 1$. Therefore, from Theorem 1, NAND and NOR have the minimal probability of error, δ , which is less than δ of the XOR, AOI, OAI, Maj, and Min gates. $\square$

It is easy to conclude from Figure 6 that the XOR2 gate has a higher error, δ , than the NAND2 gate for all values $0 < \epsilon < 0.5$. Thus it could be said that XOR2 gates propagate errors in a circuit with a higher probability than a NAND2 gate. As per Theorem 1, NAND2 will in fact have the lowest error rate of any gate for any $0 < \epsilon < 0.5$.

Another implication of Theorem 1 and its corollary is that the results can be extended to circuits of an arbitrary size. That is to say that any logic function will have the least probability of error when implemented with gates where the cardinality of those gates' ON-Set or OFF-Set is 1. Further, the smaller the minimum cardinality of either the ON-Set or OFF-Set of these gates, the smaller the error probability of the network of gates.

Secondly, the error rate of a NAND gate decreases as a function of number of inputs, for example going from a 2-input to a higher input version, whereas the opposite is true for an XOR gate.

Theorem 2. *The probability of error at an output of a noisy circuit δ increases as the logic depth of that circuit increases.*

FIGURE 7: δ versus Logic Depth for an inverter chain for $\epsilon = \{0.01, 0.05, 0.1\}$.

TABLE 1: δ for 2 and 3-input Gates for $\epsilon = .005$.

Gate	δ	Gate	δ
NAND2	.0099	XOR2	.0149
NAND3	.0087	XOR3	.0197
AOI3	.0112	MIN3	.0124
NOR2	.0099	NOR3	.0087

Proof. Let A be the probability transfer matrix (PTM) for a given noisy circuit of depth d . Depth is defined as the longest path in number of gates from primary input to primary output. To increase the length of this circuit to $d + 1$, additional noisy logic is added to compute the inputs of A . Let the kronecker product of the PTMs of the additional input logic to A be B . The resulting PTM of the new circuit of length $d + 1$ is $A \cdot B$ according to [21].

Assume $\delta_{A \cdot B} \leq \delta_A$. Then according to (1), the noisy circuit B would have to have $\epsilon \leq 0$, a contradiction. Therefore, $\delta_{A \cdot B} > \delta_A$. $\square$

The results for increasing logic depth of an inverter chain with different values of ϵ can be seen in Figure 7.

From Figure 7, one can see that no matter the value of ϵ , the error of the circuit, δ , increases as the logic depth increases. Another result that is available from the experiment is that the rate at which δ increases with an increase in logic depth is proportional to ϵ . This implies that the importance of logic depth becomes increasingly important with the level of noise present in the circuit. This further implies that as deep submicron transistor generations are explored, logic cones and pipeline stages must become increasingly smaller to sustain the same level of reliability.

Table 1 shows the different probabilities of error δ of gates given $\epsilon = 0.005$.

Several observations of regarding logic synthesis under a probabilistic fault model were observed.

TABLE 2: Single-Bit Flips (SBF) at the input that could cause a flip in the output of the carryout calculation of a dual-rail asynchronous adder.

A	B	C	A_t	A_f	B_t	B_f	C_t	$\overline{C_{out}_t}$	SBF Error
$\emptyset$	$\emptyset$	$\emptyset$	0	0	0	0	0	1	$\emptyset$
0	0	0	0	1	0	1	0	1	$\emptyset$
0	0	1	0	1	0	1	1	1	A_t, B_t
0	1	0	0	1	1	0	0	1	C_t
0	1	1	0	1	1	0	1	0	C_t
1	0	0	1	0	0	1	0	1	C_t, B_t
1	0	1	1	0	0	1	1	0	C_t, A_t
1	1	0	1	0	1	0	0	0	B_t, A_t
1	1	1	1	0	1	0	1	0	$\emptyset$

TABLE 3: Full-Adder Implementations used in experiment.

FA	C_{out}	Sum	No. Transis.
async	—	—	42
f28	Min3-Inv	Min3-Min4-Inv	28
fmaj	Inv-Min3	XOR3	32
fbase	XOR2-And2-Or2	XOR2-XOR2	42

TABLE 4: simulation results for 4 different full-adder implementations in TSMC 0.25 μm at $\epsilon = .005$ ($V_{dd} = 1.2 V$, $U_n = 0.3 V$).

FA Type	Energy (pJ)	C_{out} delay (ns)	C_{out} δ	Energy savings versus deterministic
async	0.691	1.07	.0159	5.91X
f28	1.409	1.96	.0173	2.90X
fmaj	1.549	2.22	.0196	2.64X
fbase	4.087	1.19	—	—

- (i) $\delta_{\text{Minority-Inv}} = \delta_{\text{Majority}}$: a minority gate followed by an inverter will have the same δ value as a majority gate.
- (ii) For $x_1 \oplus x_2 \oplus x_3$: $\delta_{\text{XOR2-XOR2}} > \delta_{\text{NAND2-NAND2}} > \delta_{\text{XOR3}}$: XOR3 has the minimal error rate, δ .
- (iii) $\delta_{\text{NAND2-NAND2}} > \delta_{\text{NAND4}}$.
- (iv) For $x_1 \cdot x_2 + x_3 \cdot x_4$: δ_{Majority} is minimal.

Table 1 confirms the results predicted by the theory given earlier in this section. Namely, NAND and NOR type gates have the lowest probability of error, δ , of any gate. Further, for a given number of inputs, as $y = \min(|F^{ON}|, |F^{OFF}|)$ increases, so does the probability of error δ . So for example, a gate that has $y = 1$ such as a NAND3 gate has a minimal error. But for $y = 3$ such as an AOI3 gate δ is increased, and δ further increases for $y = 4$ as in the XOR3 gate.

5. Case Study: The Full-Adder

The full-adder is a primary building block in digital arithmetic computation, present in nearly all digital adders and multipliers, and is one of the most well-studied circuits with a large variety of implementations. Its diverse array of circuit

FIGURE 8: Circuit diagram of “f28” or “mirror” full-adder. Minority gate circuit used for the carry-out function of the 40-transistor adder as well is shown.

implementations and relative importance in creating low-power datapath units make it a prime candidate for study.

5.1. Full-Adder Microarchitecture. In [20, 23], a CMOS full-adder is presented that is optimized for transistor count, which is known as the 28-transistor implementation or “mirror adder”; it will be referred to as “f28”. In [23], the authors claim the f28 is not only faster but consumes less power than the low-power carry-pass logic (CPL) implementation. The transistor level implementation of the “f28” full-adder is shown in Figure 8

It should also be noted that a majority gate is nothing more than a minority gate with an inverted output. A majority gate can also be achieved by inverting the inputs to the minority gate, which is useful should the inverted

FIGURE 9: A minority gate with an inverted output is logically equivalent to a majority gate, which is also logically equivalent to a minority gate with inverted input. However, the probability of a bit error at the output is not equivalent based on the properties presented here.

FIGURE 10: The transistor level schematic of the carryout bit calculation using dual-rail asynchronous logic. Each bit is represented by a true and false line, A_t and A_f , for example, where $A = 1 \rightarrow A_t = 1, A_f = 0$. If $A = 0 \rightarrow A_t = 0, A_f = 1$. The carryout bits are left as inverted because in dual-rail asynchronous adders, every other adder can calculate using inverted carry-in bits.

inputs be needed for some other part of the circuit. This phenomenon is illustrated in Figure 9.

Finally a dual-rail asynchronous adder is also presented for comparison. Asynchronous logic can be visited more in depth in [24]. The circuit for carry-out computation is shown in Figure 10.

In terms of the theorems proven earlier, dual-rail asynchronous logic is quite attractive because of a unique property. This type of logic switches between a *valid* state and an *invalid* state, and when the logic is in an invalid state, between one and many bit-flips on the input can be sustained before an output error occurs. These states are defined such that if bit A is *invalid* then $A_t = 0, A_f = 0$. Otherwise, if $A_t = 1$ or $A_f = 1$, then A is in *valid* state. Table 2 summarizes

the result of single-bit flips (SBF) on the inputs of dual-rail asynchronous logic for each relevant input/output mapping.

Table 2 shows the single-bit flips at the input that would cause the output to erroneously flip for the carryout calculation of the dual-rail asynchronous adder. For example, no single-bit flip would cause an output error when the async-adder is in *invalid* state because at the very least the output will float. The async-adder is designed to leak back to the previous input in the case of a floating output. As another example if the async-adder is in state $ABC = 001$, a bit flip on A_t or B_t would cause $\overline{C_{out}_t}$ to flip. This is because the output is supposed to be pulled up to 1 but with the input $C_t = 1$ a bit flip on either of A_t or B_t will create a DC path to ground, pulling the output down to 0. As it turns out, the async-adder has the least single-bit flip errors on the input that can cause an output bit flip of any adder.

Several other adders were built including a “fmaj” full-adder built with an XOR3 gate for the sum bit and an Maj3 for the carryout. A baseline adder, measured at full voltage so that no probabilistic errors would occur called “fbase”, was built for comparison. This adder, “fbase” was sized and built with several stages according to logical effort to maximize speed.

Simulations were run using the PTM algorithm in [22] to assess which full-adder types were most robust to error. These 4 full-adders were built in Cadence with TSMC 0.25 μm technology, and voltage scaled (except for the deterministic “fbase” adder) such that probabilistic errors occurred at each node in the circuit due to thermal noise, and then measured with the HSPICE circuit simulator. The results can be seen in Table 4. This particular V_{dd} was chosen because it was shown in [1] that adders and multipliers can be built to compute successfully with probabilities of error up to $\delta = .03$ and still successfully be used for image processing.

By scaling down supply voltages, the circuits will of course run more slowly; however, the energy-performance product metric showed a gain of up to 2.5X in these experiments which is not shown in Table 4 for simplification.

6. Conclusion

It was shown that probabilistic design principles can be used for ultra-low power computing. Probability and energy become closely linked in ultra-low power regimes because supply voltage is lowered so much so that it becomes comparable to the thermal noise level rendering the circuit probabilistic.

As shown in Table 4, error-aware logic design can produce a great reduction in the error rate and thus energy consumption over the baseline case which is optimized for speed. It was shown that in fact an asynchronous adder had the best performance producing both the lowest probability of error δ for a given ϵ and least energy consumption as well.

The principles behind the efficiency improvement were proven to be two important theorems developed under the probabilistic transfer matrix model. All gates do not propagate errors equally and that the depth of a digital circuit has a direct effect on the efficiency in terms of

energy consumed for a given error rate. Future work includes extending these theorems to a general logic synthesis algorithm, and continued work on tuning threshold voltages through floating gate technology to mitigate other device effects such as device mismatch.

Acknowledgments

The authors would like to thank the National Science Foundation for supporting this research and would also like to thank the reviewers for their insightful comments.

References

- [1] J. George, B. Marr, B. E. S. Akgul, and K. V. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in *Proceedings of the International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES '06)*, pp. 158–168, Seoul, Korea, 2006.
- [2] T. Sakurai, "Perspectives on power-aware electronics," in *Proceedings of the IEEE International Solid-State Circuits Conference (ISSCC '03)*, pp. 19–29, San Francisco, Calif, USA, February 2003.
- [3] R. Hegde and N. R. Shanbhag, "Energy-efficient signal processing via algorithmic noise-tolerance," in *Proceedings of the International Symposium on Low Power Electronics and Design*, pp. 30–35, San Diego, Calif, USA, August 1999.
- [4] S. Cheemalavagu, P. Korkmaz, K. V. Palem, B. E. S. Akgul, and L. N. Chakrapani, "A probabilistic cmos switch and its realization by exploiting noise," in *Proceedings of the IFIP International Conference on Very Large Scale Integration of System-On-Chip (VLSI-SoC '05)*, Perth, Australia, October 2005.
- [5] K.-U. Stein, "Noise-induced error rate as a limiting factor for energy per operation in digital ICs," *IEEE Journal of Solid-State Circuits*, vol. 12, no. 5, pp. 527–530, 1977.
- [6] L. B. Kish, "End of Moore's law: thermal (noise) death of integration in micro and nano electronics," *Physics Letters A*, vol. 305, no. 3-4, pp. 144–149, 2002.
- [7] ITRS 2005 edition, <http://www.itrs.net/Links/2009ITRS/Home2009.htm>.
- [8] K. Nepal, R. I. Bahar, J. Mundy, W. R. Patterson, and A. Zaslavsky, "Designing logic circuits for probabilistic computation in the presence of noise," in *Proceedings of the Design Automation Conference (DAC '05)*, pp. 485–490, Anaheim, Calif, USA, June 2005.
- [9] B. Marr, J. George, P. Hasler, and D. V. Anderson, "Increased energy efficiency and reliability of ultra-low power arithmetic," in *Proceedings of the Midwest Symposium on Circuits and Systems (MWSCAS '08)*, pp. 366–369, Knoxville, Tenn, USA, August 2008.
- [10] A. Basu, C. M. Twigg, S. Brink, et al., "RASP 2.8: a new generation of floating-gate based field programmable analog array," in *Proceedings of the Custom Integrated Circuits Conference (CICC '08)*, pp. 213–216, San Jose, Calif, USA, September 2008.
- [11] B. Marr, A. Basu, S. Brink, and P. Hasler, "A learning digital computer," in *Proceedings of the Design Automation Conference (DAC '09)*, pp. 617–618, San Francisco, Calif, USA, July 2009.
- [12] C. Mead, *Analog VLSI and Neural Systems*, Addison-Wesley, Reading, Mass, USA, 1989.
- [13] J. von Neumann, *Automata Studies*, Princeton University Press, Princeton, NJ, USA, 1956.
- [14] N. Pippenger, "Reliable computation by formulas in the presence of noise," *IEEE Transactions on Information Theory*, vol. 34, no. 2, pp. 194–197, 1988.
- [15] S. Krishnamohan and N. R. Mahapatra, "A highly-efficient technique for reducing soft errors in static CMOS circuits," in *Proceedings of the IEEE International Conference on Computer Design: VLSI in Computers and Processors (ICCD '04)*, pp. 126–131, San Jose, Calif, USA, October 2004.
- [16] S. Das, S. Pant, D. Roberts, et al., "A self-tuning DVS processor using delay-error detection and correction," in *Proceedings of IEEE Symposium on VLSI Circuits*, pp. 258–261, Kyoto, Japan, June 2005.
- [17] J. P. Hayes, I. Polian, and B. Becker, "An analysis framework for transient-error tolerance," in *Proceedings of the IEEE VLSI Test Symposium (VTS '07)*, pp. 249–255, Berkeley, Calif, USA, May 2007.
- [18] S. Krishnaswamy, S. M. Plaza, I. L. Markov, and J. P. Hayes, "Enhancing design robustness with reliability-aware resynthesis and logic simulation," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '07)*, pp. 149–154, San Jose, Calif, USA, November 2007.
- [19] L. N. Chakrapani, B. E. S. Akgul, S. Cheemalavagu, P. Korkmaz, K. V. Palem, and B. Seshasayee, "Ultra-efficient (embedded) SOC architectures based on probabilistic CMOS (PCMOs) technology," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '06)*, vol. 1, Munich, Germany, March 2006.
- [20] D. Harris and N. H. Weste, *CMOS VLSI Design: A Circuits and Systems Perspective*, Pearson Education, Upper Saddle River, NJ, USA, 2005.
- [21] K. N. Patel, J. P. Hayes, and I. L. Markov, "Evaluating circuit reliability under probabilistic gate-level fault models," in *Proceedings of the 12th International Workshop on Logic and Synthesis (IWLS '03)*, pp. 59–64, Laguna Beach, Calif, USA, May 2003.
- [22] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Accurate reliability evaluation and enhancement via probabilistic transfer matrices," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE '05)*, vol. 1, pp. 282–287, Munich, Germany, 2005.
- [23] R. Zimmermann and W. Fichtner, "Low-power logic styles: CMOS versus pass-transistor logic," *IEEE Journal of Solid-State Circuits*, vol. 32, no. 7, pp. 1079–1090, 1997.
- [24] A. J. Martin, "Asynchronous datapaths and the design of an asynchronous adder," *Formal Methods in System Design*, vol. 1, no. 1, pp. 117–137, 1992.