Low-power Properties of the Logarithmic Number System

V. Paliouras and T. Stouraitis
Electrical and Computer Engineering Department
University of Patras, Greece

Abstract

In this paper, the potential of reducing power dissipation in a digital system using the Logarithmic Number System (LNS) is investigated. To provide a quantitative measure of power savings, the equivalence of an LNS to a linear fixed-point system is initially explored. The bit assertion activity of an LNS encoded signal is studied for both uniform and correlated Gaussian inputs. It is shown that LNS reduces the average bit assertion probability by more than 50%, in certain cases, over an equivalent linear representation. Finally, the impact of LNS on the hardware architecture and, by means of that, to power dissipation, is discussed. It is found that the average number of logic transitions is reduced by several times, for certain arithmetic operations and word lengths, thus compensating the power-dissipation overhead due to the unavoidable linear-to-logarithmic and logarithmic-to-linear conversion.

1 Introduction

Power dissipation minimization is sought at all levels of design abstraction, ranging from system-level software-hardware partitioning down to technology-related issues. A wide variety of design techniques have been proposed [1], aiming to reduce the clock frequency, the total switching capacitance, the supply voltage, and the average activity in a clock period. The successful selection of the number system and the proper design of arithmetic circuits has been proposed as a power dissipation minimization technique [2]. For example, Ramprasad *et al.* report that the sign-magnitude representation demonstrates minimal activity, among the various number representations (unsigned, two's complement, one's complement) [3].

A variety of LNS processors have been reported [4, 5], demonstrating the feasibility of LNS-based systems. The main drawback in LNS processing is the complexity of logarithmic addition and, in particular, of logarithmic subtraction. The hardware implementation of such operations relies on memory look-up tables and several techniques that reduce their size have been proposed [5]. Furthermore, different approaches to mitigating the logarithmic subtraction complexity problem have been suggested, such as the redundant LNS by Arnold *et al.* [6] and the co-transformations by Arnold *et al.* [7].

Most of the previous works on LNS, generally, do not consider the impact of logarithmic arithmetic on power dissipation. However, Sullivan [8] reports that an LNS-based DSP can dissipate 29% less power than a linear DSP of comparable dynamic range and SNR. The application of LNS processing to a hearing-aid instrument is described by Morley *et al.* [9]. More recently, Sacha and Irwin studied the impact of LNS on QRDRLS adaptive filtering in terms of switched capacitance and numerical accuracy [10].

In this paper, it is shown that the adoption of the LNS [11] can lead to substantial power dissipation savings, since it both reduces the average bit activity and it simplifies certain arithmetic operations, directly cutting the corresponding power dissipation. An introduced probabilistic analysis shows that reduced bit activity should be expected in the most significant part of an LNS word, a theoretical result found to be in agreement with experimental results. Moreover, this paper shows that complicated LNS operations require less power than complicated fixed-point operations, i.e., LNS addition/subtraction is less power demanding than fixed-point multiplication. As a result, combined operations such as multiplication-addition, are shown to be efficiently implemented in LNS. A preliminary version of some of the results presented in this paper are included in [12].

2 Linear/LNS Equivalence

The LNS representation maps a real number, X, to a triplet, as

$$X \xrightarrow{\text{LNS}} (z, s, x = \log_b |X|)\,, \tag{1}$$

where b is the base of the logarithm, z is the zero flag, and s is the sign of X. A zero flag is required, because $\log_b |X|$ is not finite at $X = 0$. Similarly, since the logarithm of a negative number is not a real number, the sign information of X is separately stored in the sign bit s. Logarithm $x = \log_b |X|$ is encoded as a binary number (two's complement or sign-magnitude), expressed as $x = I.F$, where I is the integral and F is the fractional part.

LNS has been considered as an extreme case of floating-point arithmetic [13]. However, floating-point arithmetic is not a common choice for power-constraint systems; instead fixed-point arithmetic is preferred due to its simplicity. Hence, in this paper that focuses on power dissipation aspects, LNS is compared to an n-bit linear fixed-point representation and it is shown to provide substantial improvement in terms of power dissipation. There are two main issues in a finite word length number system, namely the *range* of the representable numbers and the *precision* of the representation [13].

Let k and l be integers which denote the word length of the integral and fractional part I and F, respectively. Let (k, l, b)-LNS denote an LNS of integral and fractional word lengths k and l, respectively, and of logarithm base b. The problem of equivalence between a (k, l, b)-LNS and an n-bit linear fixed-point system is formulated here as the computation of k and l

0-7695-1150-3/01 $10.00 © 2001 IEEE

so that the two representations satisfy a suitably defined criterion, for a particular base b.

Several quantities have been used for investigating the equivalence between representations, including the relative step size, the Average Relative Representational Error (ARRE), and the Signal-to-Noise ratio. Koren [13] suggests the use of the relative step size, ϵ_{rss}, to compare LNS precision with floating-point precision. Let ϵ_{rss} be defined as

$$\epsilon_{\text{rss}} = \frac{x_{i+1} - x_i}{x_i}, \tag{2}$$

where $\{x_i\}$ is the sequence of representable numbers. Assuming an integer linear system with a word length n, it holds that $x_{i+1} = x_i + 1$; hence (2) gives

$$\epsilon_{\text{rss,fxp}} = \frac{1}{x_i}, \tag{3}$$

where fxp denotes a fixed-point system. The relative step size for an (k, l, b)-LNS is

$$\epsilon_{\text{rss,LNS}} = \frac{b^{x_{i+1}} - b^{x_i}}{b^{x_i}} = \frac{b^{x_i + 2^{-l}} - b^{x_i}}{b^{x_i}} = b^{2^{-l}} - 1. \tag{4}$$

It can be noted that, while $\epsilon_{\text{rss,fxp}}$ depends on the corresponding representable value x_i, $\epsilon_{\text{rss,LNS}}$ does not. In order to overcome the particular difference and be able to compare the precision of the two representations, the notion of average relative step size is used and the following two restrictions are posed: the two representations should a) cover equivalent data ranges and b) should exhibit equal average representational error. The average relative step size, ϵ_{ave}, is defined as

$$\epsilon_{\text{ave}} = \frac{\sum_{A=\widehat{A}_{\min}}^{\widehat{A}_{\max}} \epsilon_{\text{rss}}(A)}{\widehat{A}_{\max} - \widehat{A}_{\min} + 1}, \tag{5}$$

where $\widehat{A}_{\min}$ and $\widehat{A}_{\max}$ define the range of representable numbers.

Due to definitions (2) and (5), the average relative step size for the fixed-point case, is

$$\epsilon_{\text{ave,lin}} = \frac{1}{2^n - 1} \sum_{i=1}^{2^n - 1} \frac{1}{i} = \frac{\psi(2^n) + \gamma}{2^n - 1}, \tag{6}$$

where γ is the Euler gamma constant, lin refers to a linear system, and function ψ is defined through

$$\psi(x) = \frac{d}{dx} \ln \Gamma(x), \tag{7}$$

where $\Gamma(x)$ is the Euler gamma function.

In the case of the LNS, as $\epsilon_{\text{rel,LNS}}$ is constant over the range, due to (4), it occurs that

$$\epsilon_{\text{ave,LNS}} = b^{2^{-l}} - 1. \tag{8}$$

In the following, the maximum number representable in each number system is computed and utilized to compare the ranges of the representations. The maximum number representable by an n-bit linear integer is $2^n - 1$; therefore the corresponding upper bound is

$$A_{\max}^{\text{lin}} = 2^n - 1. \tag{9}$$

The maximum number representable by (k, l, b)-LNS encoding (1), is

$$A_{\max}^{\text{LNS}} = b^{2^k - 2^{-l}}. \tag{10}$$

Therefore, according to the equivalence criteria posed earlier, an LNS is equivalent to an n-bit linear fixed-point representation, when the following restrictions are simultaneously satisfied:

$$A_{\max}^{\text{LNS}} \geq A_{\max}^{\text{lin}} \tag{11}$$

n	$b = 1.5$			$b = 2$			$b = 2.5$		
	k	l	n_{eq}	k	l	n_{eq}	k	l	n_{eq}
5	4	2	10	3	3	8	2	3	6
6	4	3	10	3	4	8	2	4	11
7	4	4	10	3	5	8	3	5	11
8	4	5	10	4	5	8	3	6	11
9	4	5	10	4	6	16	3	7	11
10	5	6	19	4	7	16	3	7	11
11	5	7	19	4	8	16	4	8	22
12	5	8	19	4	9	16	4	9	22
13	5	9	19	4	10	16	4	10	22
14	5	10	19	4	11	16	4	11	22
15	5	11	19	4	12	16	4	12	22

Table I: Correspondence of n, k, l, and n_{eq} for various b.

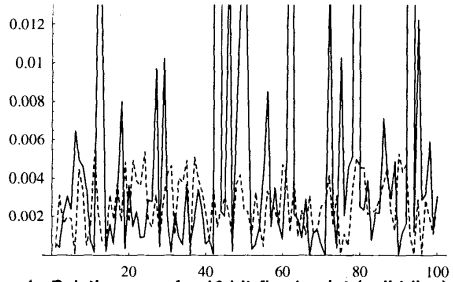


Figure 1: Relative error for 10-bit fixed-point (solid line) and $l = 5$-bit LNS (dashed line), for a 100-point Gaussian signal. Fixed-point errors in larger than 0.013, are truncated.

$$\epsilon_{\text{ave,LNS}} \leq \epsilon_{\text{ave,lin}}. \tag{12}$$

Hence, from (6) and (8)–(10), it follows that

$$b^{2^k - 2^{-l}} \geq 2^n - 1 \tag{13}$$

$$b^{2^{-l}} - 1 \leq \frac{\psi(2^n) + \gamma}{2^n - 1}, \tag{14}$$

which, when solved for minimal values of k and l, give

$$l = \left\lceil -\log_2 \log_b \left(1 + \frac{\psi(2^n) + \gamma}{2^n - 1}\right) \right\rceil \tag{15}$$

$$k = \left\lceil \log_2 \left(\log_b (2^n - 1) + 2^{-l} \right) \right\rceil. \tag{16}$$

The above analysis is summarized in the following theorem.

Theorem 1 *A (k, l, b)-LNS covers a range at least as long as an n-bit fixed-point system with an average representational error equal or smaller to that of the fixed-point system, when l and k are given by (15) and (16), respectively.*

Values of k and l that correspond to various values of n for various values of b, can be seen in Table I. In case that the input signal is normalized in the interval $(-1, +1)$, the fixed-point length n, corresponds to a fractional step 2^{-n}. The LNS integral word length can be obtained by solving

$$-(2^k - 2^{-l}) \leq \log_b 2^{-n} \tag{17}$$

for minimal k, which is equivalent to the range specification (11). The relative representational error, ϵ_{rel}, of a number of real value A encoded in a number system with a representation of value $\widehat{A}$ is defined as

$$\epsilon_{\text{rel}} = \frac{|A - \widehat{A}|}{A}. \tag{18}$$

Notice that $A \neq \widehat{A}$ due to the finite length of the digital words. Fig. 1 depicts the absolute values of relative errors in a 100-point random Gaussian sequence for 10-bit linear and $l = 5$-bit LNS,

n	σ	k	l	n	σ	k	l
8	40	4	4	16	11000	5	12
	85	4	5		22000	5	13

Table II: Values of l and k experimentally derived for $b = 1.4$, to provide a particular SNR.

computed by Theorem 1. It can be seen that LNS performance is better.

While the word lengths k and l computed via (15) and (16) meet the posed equivalence specifications (11) and (12), LNS offers a larger range than the equivalent fixed-point representation. Let n_{eq} denote the word length of a fixed-point system which can cover the range offered by an LNS defined through (15) and (16), or, equivalently, let n_{eq} be the smallest integer which satisfies

$$2^{n_{\text{eq}}} - 1 \geq b^{2^k - 2^{-l}}. \tag{19}$$

From (19) it follows that

$$n_{\text{eq}} = \left\lceil (2^k - 2^{-l}) \log_2 b \right\rceil. \tag{20}$$

It should be stressed that, when $n_{\text{eq}} \geq n$, the precision of the particular fixed-point system is better than that of the LNS derived by (15) and (16). Equation (20) reveals that the particular LNS, while meeting the precision of an n-bit linear representation, in fact, covers the range provided by an n_{eq}-bit linear system.

Koren [13] compares LNS to a floating-point representation using the Average Representation Relative Error (ARRE). However, this approach is not applicable for comparison to a fixed-point representation, as fixed-point ARRE does not converge, due to the discontinuity at zero.

Another possible LNS-to-linear equivalence measure, common in signal processing applications [8], is the signal-to-noise ratio, SNR, defined as

$$\text{SNR} = 10 \log_{10} \frac{P_{\text{signal}}}{P_{\text{noise}}}, \tag{21}$$

where P_{signal} and P_{noise} are the power of the signal and the representation truncation noise. Table II shows those (k, l, b)-LNS representations that are experimentally found to offer better SNR than the corresponding n-bit fixed-point system, for a 1000-point Gaussian sample of standard deviation σ. It is noted that the word lengths k and l depend on the standard deviation σ. Table II demonstrates that for a particular n, computed as $n = \lceil \log_2(3\sigma) \rceil$, the value of l varies by a bit, depending on σ of the input.

3 Power Dissipation and LNS Encoding

3.1 Uniform Input

Let $p_{0\to1}(i)$ be the bit assertion probabilities, i.e., the probability of the ith bit transition from 0 to 1. Assuming that data are temporaly independent, it holds that

$$p_{0\to1}(i) = p_0(i)p_1(i) = \Big(1 - p_1(i)\Big)p_1(i), \tag{22}$$

where $p_0(i)$ and $p_1(i)$ are the probabilities of the ith bit being 0 or 1, respectively. Due to the assumption of uniform data distribution, it holds that $p_0(i) = p_1(i) = 1/2$, which, due to (22), gives $p_{0\to1}(i) = 1/4$.Therefore, all bits in the linear fixed-point representation exhibit a constant probability $p_{0\to1}(i)$, $i = 0, 1, \ldots, n-1$.

However, the probabilities of bit assertions in LNS words, $p^{\text{LNS}}_{0\to1}(i)$, are not constant, in contrast to the probabilities $p_{0\to1}(i)$; they depend on the significance of the ith bit. To evaluate the probabilities $p^{\text{LNS}}_{0\to1}(i)$, the following experiment is performed. For all possible values of X in a n-bit system, the corresponding $\lfloor \log_b X \rfloor$ values in a (k, l, b)-LNS format are derived and probabilities $p_1(i)$ for each bit are computed. Then, $p^{\text{LNS}}_{0\to1}(i)$ is obtained by (22).

The actual assertion probabilities for the bits in an LNS word, $p^{\text{LNS}}_{0\to1}(i)$, are depicted in Fig. 2. It can be seen that $p^{\text{LNS}}_{0\to1}(i)$ for the more significant bits is substantially lower than $p^{\text{LNS}}_{0\to1}(i)$ for the less significant bits. Also, it can be seen that $p^{\text{LNS}}_{0\to1}(i)$ depends on b. This behavior, which is due to the inherent data compression property of the logarithm function, reduces the average activity in the entire word. Average activity savings percentage, S_{ave} is

$$S_{\text{ave}} = \left(1 - \frac{4}{n} \sum_{i=0}^{k+l-1} p^{\text{LNS}}_{0\to1}(i)\right) 100\%, \tag{23}$$

where $p^{\text{FXP}}_{0\to1}(i) = 1/4$ for $i = 0, 1, \ldots, n-1$, n denotes the length of the fixed-point system, and the word lengths k and l are computed via Theorem 1. Savings percentage S_{ave} is demonstrated in Fig. 3(a) for various values of n and b, and it is found to be more than 15% in certain cases.

However, as implied by the definition (20) of n_{eq}, the linear system, which provides a range exceeding that of an LNS defined via Theorem 1, requires n_{eq} bits. If the reduced (in this case) precision of (k, l, b)-LNS compared to n_{eq}-bit fixed-point system, is acceptable for a particular application, S'_{ave} can be used to describe the relative efficiency of LNS, instead of (23), where

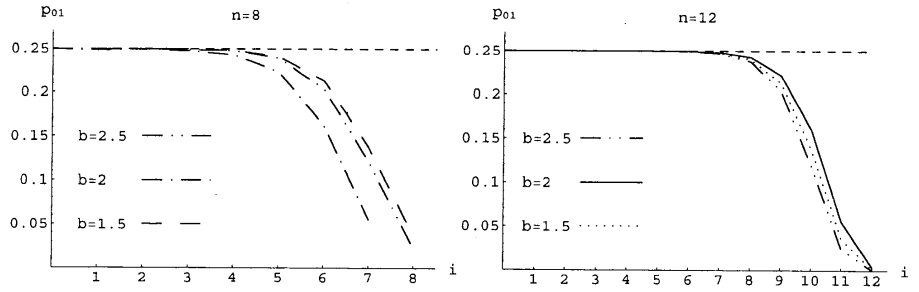
$$S'_{\text{ave}} = \left(1 - \frac{4}{n_{\text{eq}}} \sum_{i=0}^{k+l-1} p^{\text{LNS}}_{0\to1}(i)\right) 100\%. \tag{24}$$

Savings percentage S'_{ave} is demonstrated in Fig. 3(b) for various values of n and b. Savings are found to exceed 50% in some cases. Notice that Fig. 3 reveals that, for a particular word length n, the proper selection of logarithm base b can significantly affect the average activity. Therefore, the choice of b is important in designing a low-power LNS-based system. Notice that truncation has been used in the conversion to LNS.

3.2 Correlated Gaussian Input

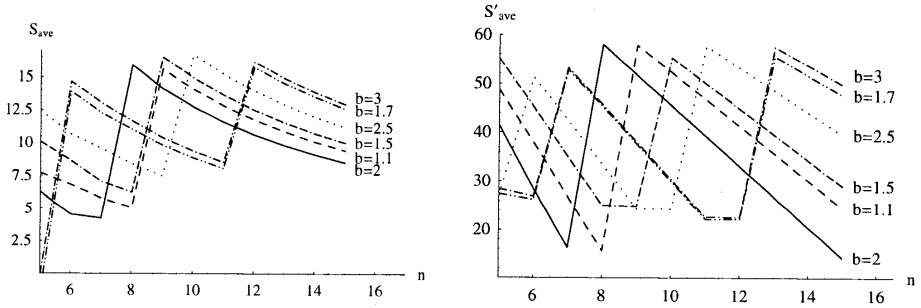
To demonstrate the impact of the compression property of the logarithm on the switching activity of the signal, initially, the distribution of the difference of consecutive logarithmic images of linear samples, taken from a correlated Gaussian process, is studied. This approach does not attempt to derive bit activity from word-level statistics. However, it demonstrates the compression properties of the logarithm, which qualitatively explains the observed reduced activity of the most-significant part of an LNS word. Subsequently, an experimental evaluation of the bit activity is offered.

Assume that two successive inputs to an LNS-based system are the LNS representations of two random variables, x and y, which follow a bi-variate joint Gaussian distribution with zero mean, $\mu = \mu_x = \mu_y = 0$, equal standard deviation, $\sigma = \sigma_x = \sigma_y$,



(a) (b)

Figure 2: Activities against bit significance i, in an LNS word, for $n = 8$ (a) and $n = 12$ (b) and various values of the base b. The horizontal dashed line is the activity of the corresponding n-bit fixed-point system.



(a) (b)

Figure 3: Percentage of average activity reduction due to the use of LNS, compared to n-bit (a) and to n_{eq}-bit (b) linear fixed-point system, for various bases b of the logarithm.

and a correlation factor ρ. The corresponding probability-density function $f(x, y; \rho, \sigma)$ is given by

$$f(x, y; \rho, \sigma) = \frac{\exp(-\frac{x^2+y^2-2xy\rho}{2(1-\rho^2)\sigma^2})}{2\pi\sigma^2\sqrt{1-\rho^2}}, \quad (25)$$

where ρ, $-1 < \rho < 1$, is the correlation factor and σ is the common standard deviation. Landman and Rabaey exploit this particular distribution in the derivation of the Dual Bit Type method for modeling activity [14].

The objective of the analysis which follows is to derive the distribution of the logarithms of the consecutive input data modeled by the couple (x, y) to be used in the derivation of the probability-density function of the random variable $\log_b|x| - \log_b|y|$. As an intermediate step to the derivation of the distribution followed by the logarithmic data, the distribution of the absolute value of a Gaussian random variable with probability-density function $f_{\text{abs}}(x, y; \rho, \sigma)$ is required:

$$f_{\text{abs}}(x, y; \rho, \sigma) = f(x, y; \rho, \sigma) + f(-x, y; \rho, \sigma) + f(x, -y; \rho, \sigma) + f(-x, -y; \rho, \sigma), \quad (26)$$

where each term of the sum corresponds to a quadrant in the xy-plane. By algebraic manipulations, (26) gives

$$f_{\text{abs}}(x, y; \rho, \sigma) = \frac{1 + \exp(\frac{-2xy\rho}{(1-\rho^2)\sigma^2})}{\exp(\frac{x^2+y^2-2xy\rho}{2(1-\rho^2)\sigma^2})\,\pi\sqrt{1-\rho^2}\,\sigma^2}, \quad (27)$$

when $x, y \geq 0$ and $f_{\text{abs}}(x, y; \rho, \sigma) = 0$, otherwise. The probability-density function $f_{\log}(w, v; \rho, \sigma)$ of the distribution of the random variables w and v, obtained as

$$w = \log_b|x| \quad (28)$$

$$v = \log_b|y|, \quad (29)$$

is computed from (27) as

$$f_{\log}(w, v; \rho, \sigma, b) = \frac{f_{\text{abs}}(x_1, y_1; \rho, \sigma)}{|J(x_1, y_1)|}, \quad (30)$$

where $x_1 = b^w$, $y_1 = b^v$, and $J(x, y)$ denotes the Jacobian of the transformations (28) and (29) [15], given as

$$J(x, y) = \begin{vmatrix} \frac{\partial w}{\partial x} & \frac{\partial w}{\partial y} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} \end{vmatrix} = \frac{1}{xy\log^2 b}. \quad (31)$$

Finally, from (27)–(31), it follows that

$$f_{\log}(w, v; \rho, \sigma, b) = \frac{b^{w+v}\left(1 + \exp(\frac{-2b^{w+v}\rho}{(1-\rho^2)\sigma^2})\right)\log^2 b}{\exp(\frac{b^{2w}+b^{2v}-2b^{w+v}\rho}{2(1-\rho^2)\sigma^2})\,\pi\sqrt{1-\rho^2}\,\sigma^2}. \quad (32)$$

The probability-density function $f_{\log,\text{diff}}(z_{\log}; \rho, b)$ of the random variable $z_{\log} = \log_b|x| - \log_b|y|$ is obtained by integrating (32) [15], as

$$f_{\log,\text{diff}}(z_{\log}; \rho, b) = \int_{-\infty}^{+\infty} f_{\log}(x, x - z_{\log}; \rho, \sigma, b)dx, \quad (33)$$

$$= \frac{2b^z\left(1 + b^{2z}\right)\sqrt{1-\rho^2}\log b}{\pi\left(1 + b^{4z} + b^{2z}\left(2 - 4\rho^2\right)\right)}. \quad (34)$$

Equation (34) reveals that $f_{\log,\text{diff}}(z; \rho, b)$ does not depend on σ. For comparison, the probability-density function of the difference, z, of two Gaussian random variables x and y, i.e., $z = x - y$, is obtained by integrating (25), resembling the

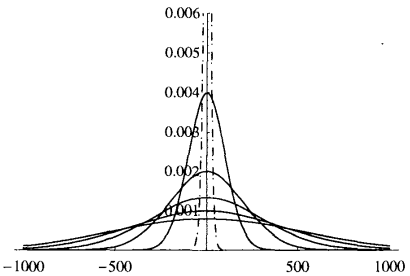


Figure 4: As σ increases, $f_{\text{linear,diff}}(z;\rho,\sigma)$ becomes wider, while $f_{\log,\text{diff}}$ (shown as dashed line) is not altered.

derivation of (34) from (33) to give

$$f_{\text{linear, diff}}(z;\rho,\sigma) = \frac{\exp\left(\frac{z^2}{4(-1+\rho)\sigma^2}\right)}{2\sigma\sqrt{\pi}\sqrt{1-\rho}}. \quad (35)$$

The probability-density functions defined by (34) and (35) are plotted in Fig. 4, where it can be seen that, as σ increases, $f_{\text{linear,diff}}$ becomes wider, while $f_{\log,\text{diff}}$ is not affected. From a bit-activity point of view, this behavior indicates that reduced bit activity should be expected in the most significant part of an LNS word. However, there exists an implicit dependence between the standard deviation σ of the input Gaussian distribution and the distribution of the actual logarithmic differences. The implicit dependence emerges through the word lengths required and the representation equivalence assumptions. The actual values occurring in the system, comprise an l-bit fractional part, and they can be modeled by multiplying the logarithmic difference values, $z_{\log}$, by 2^l. In case the input word length n is defined as $n = \lceil \log_2(3\sigma) \rceil$, and the fractional word length l depends on n as defined by Theorem 1, the distribution of the logarithmic differences, considering the l-bit fractional part, also depends on σ. To model the effect of the number of fractional bits, l, a new random variable $q = 2^l z_{\log}$ is formed, with a probability-density function given by

$$f_{\log,\text{diff,bit}}(q;b,\rho,l) = \frac{1}{2^l} f_{\log,\text{diff}}\left(\frac{q}{2^l};\rho,b\right) \quad (36)$$

$$= \frac{2^{1-l}\, b^{\frac{q}{2^l}} \left(1 + b^{2^{1-l}q}\right) \sqrt{1-\rho^2}\, \log b}{\pi \left(1 + b^{2\cdot 2^{-l}q} + b^{2^{1-l}q}\,(2-4\rho^2)\right)}. \quad (37)$$

The probability-density functions $f_{\log,\text{diff,bit}}(q;b,\rho,l)$ and $f_{\text{linear,diff}}(z;\rho,\sigma)$ are plotted in Fig. 5, where the areas of probability $\alpha = 0.9$ are indicated. It is shown that the logarithmic difference q generally spans a more narrow interval than the linear difference; in other words, the logarithmic samples are generally "closer;" hence less activity in the most-significant bits of the logarithmic word is anticipated. The experiments described in the following paragraphs justify this theoretical conclusion.

Activity measurements are performed by generating a random sample, which follows a distribution with density function $f(x,y;\rho,\sigma)$ of (25) and subsequently applying it to an LNS encoder. The LNS representation is composed of the sign bit and the value of the logarithm in two's complement format. The integral and the fractional word lengths k and l, are derived as dictated by Theorem 1. The bit assertion probability per bit

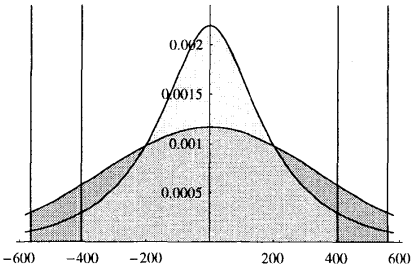


Figure 5: The probability-density functions $f_{\text{linear,diff}}(z;\rho,\sigma)$ and $f_{\log,\text{diff,bit}}(q;b,\rho,l)$. The regions of probability $\alpha = 0.9$ are marked for both distributions (dark fill for $f_{\text{linear,diff}}(z;\rho,\sigma)$, light fill for $f_{\log,\text{diff,bit}}(q;b,\rho,l)$), and it is shown that the logarithm case is narrower.

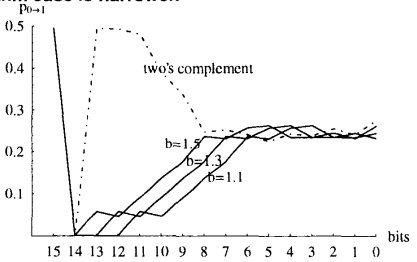
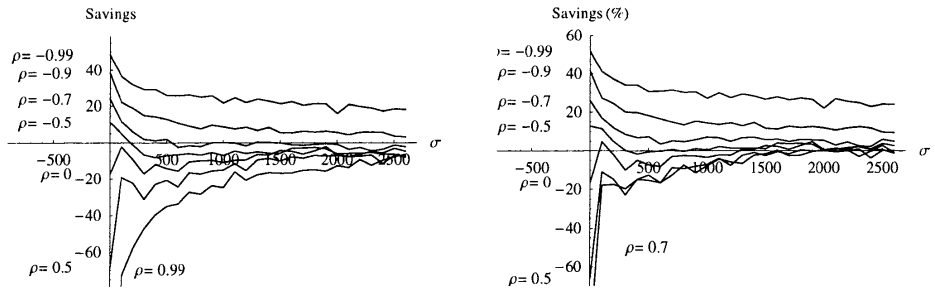


Figure 6: Probability $p_{0\to1}$ of a 0 to 1 transition per bit position, for two's complement encoding (dotted line) and LNS encoding for various values of b. The input is an anti-correlated ($\rho = -0.99$) zero-mean Gaussian signal.

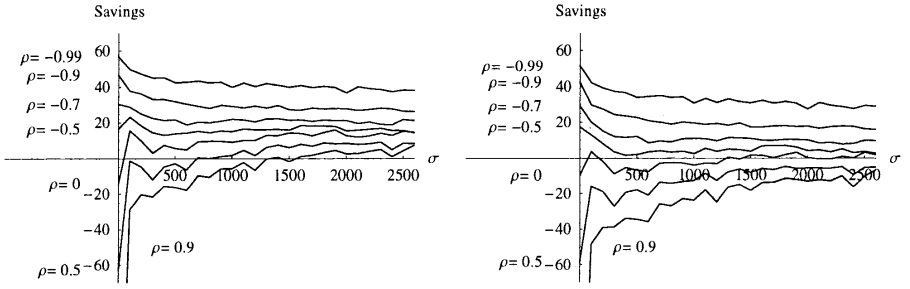
position $p_{0\to1}$, for two's complement encoding and for base-b LNS encoding are depicted in Fig. 6. Fig. 6 shows the signal activity per bit position, for a zero-mean Gaussian anti-correlated ($\rho = -0.99$) input signal, with $\sigma = 1500$, and word length $n = 14$. The applicability of the Dual Bit Type method [14] to model the activity depicted in Fig. 6 is apparent. The bit assertion activity, assuming the same signal encoded in LNS and various bases b, is also shown in Fig. 6. Fig. 6 reveals that significant activity reduction is possible despite the fact that the sign-bit activity is identical in both cases, and the word length of the LNS representation is slightly increased.

Activity savings for $n = 14$, $b = 2$, and various values of the correlation factor ρ, versus the standard deviation of the input distribution σ, are depicted in Fig. 7(a). Fig. 7(a) reveals that for very anti-correlated signals, significant savings in activity are achieved. However, the situation is reversed for correlated signals. The experiment is repeated for a $b = 1.5$ LNS and the results are depicted in Fig. 7(b). It can be seen that the signal activity is reduced, when compared to the base $b = 2$ case. The experiment is repeated for 16-bit data, as shown in Fig. 8. The impact of the logarithm base b selection on bit assertion activity, is depicted in Fig. 9. Fig. 9 reveals that for base values approaching unity, the savings percentage increases. Therefore, there is an optimal value for b, which minimizes signal activity. Finally, by relaxing the error specification and comparing the activity of an LNS signal to the activity of an n_{eq}-bit two's complement signal, as defined by (20), the savings depicted in Fig. 10 are achieved.



(a) (b)

Figure 7: Signal activity savings for $n = 14$, versus standard deviation σ, for various values of correlation ρ and two values of the base b, $b = 2$ in (a) and $b = 1.5$ in (b).



(a): $b = 1.1$ (b): $b = 2$

Figure 8: Savings plots for zero-mean Gaussian input pair, data word length $n = 16$, versus standard deviation σ for several degrees of correlation ρ, and various logarithmic bases b, $b = 1.1$ (a) and $b = 2$ (b).

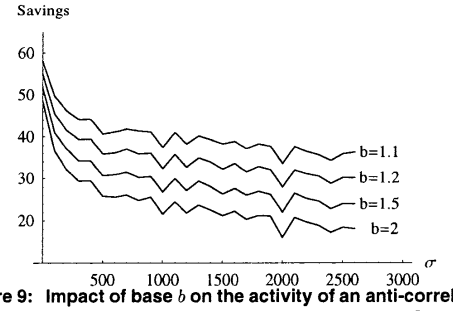


Figure 9: Impact of base b on the activity of an anti-correlated signal, for which $\rho = -0.99$ versus σ. A zero-mean Gaussian input is assumed, of word length $n = 14$.

bits	Adder	Multiplier	Divider	Multiplier times down	Division times down
16	90	573	3757	6.37	41.74
32	182	3874	7293	21.29	40.07

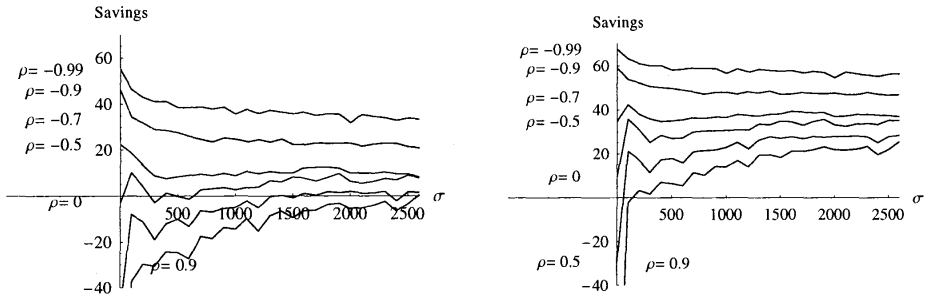
Table III: Average number of transitions [16], exploited to compute the reduction due to LNS. A Wallace/Dadda multiplier is assumed, and a SRT Radix-4 divider.

4 Power Dissipation and LNS Hardware

Let x and y be the (k, l, b)-LNS images of the linear quantities X and Y. The LNS transforms the basic arithmetic operations. Specifically, n-bit multiplication and division are reduced to $(k + l)$-bit addition and subtraction, respectively, while the computation of roots and powers is reduced to simple division and multiplication by a constant, respectively. It is noted that Theorem 1 is used to derive l and k. The n_{eq} model is not adopted, since it does not guarantee that the equivalence criteria are met. In addition, since $n_{eq} > n$, an n_{eq}-bit system is more complex than an n-bit one, hence the use of Theorem 1 favors the linear representation.

Reduction in multiplication complexity is quantified in the following. Assume that an n-bit carry-save array multiplier, which has a complexity of $n^2 - n$ 1-bit full adders (FAs), is replaced by an n-bit adder, which, assuming $k + l = n$, has a complexity of n FAs, for a ripple-carry implementation [13]. Therefore, the adoption of LNS reduces area complexity by a factor r_{C_L}, $r_{C_L} = (n^2 - n)/n = n - 1$,which grows linearly with word length n.

To clearly demonstrate the performance improvement for multiplication/division on power dissipation, the average number of logic transitions reported for ripple-carry addition, Wallace/Dadda multiplication, and radix-4 SRT division reported by Callaway and Swartzlander [16] are exploited. The average numbers of transitions are shown in Table III, for various word lengths n. Due to the reduction of multiplication and division to addition/subtraction, the average transition count drops several times per operation, as shown in Table III. However, addition and subtraction are complicated in LNS, since they require a table look-up operation for the evaluation of the function $\log_b(1 \pm b^{y-x})$. A table look-up operation requires a ROM of $2^n \times n$ bits, a size which can inhibit LNS utilization for large values of n.



(a): $b = 2$ (b): $b = 1.1$

Figure 10: LNS activity savings in comparison to an n_{eq}-bit system, for $n = 14$ and $b = 2$ (a) and $b = 1.1$ (b).

n (bits)	LNS (mW/MHz) Base b	Addition	Subtraction	Average	Fixed-point Multiplication
8	1.5	0.99	0.99	0.99	2.02
	2	0.90	0.89	0.89	
	2.5	0.98	0.98	0.98	
10	1.5	1.14	1.16	1.15	2.64
	2	1.18	1.18	1.18	
	2.5	1.09	1.11	1.10	
12	1.5	1.49	1.52	1.50	3.35
	2	1.58	1.58	1.58	
	2.5	1.46	1.49	1.48	
14	1.5	2.38	2.46	2.42	4.13
	2	2.62	2.61	2.62	
	2.5	2.26	2.36	2.31	

Table IV: Estimation of power dissipated in LNS subtraction-addition, in comparison to $n \times n$-bit fixed-point multiplication, for various fixed-point word lengths, n, and bases b.

Due to the importance of multiplication-addition operations in DSP applications, it is of interest to compare the cost of their linear and LNS implementation. From the hardware architecture viewpoint, the LNS multiplier is both functionally and structurally identical to a fixed-point adder. Hence, the problem of comparing the amount of power dissipated by a combined multiplication-addition operation, is transformed to comparing the power dissipation figures of the LNS adder/subtractor and a corresponding fixed-point multiplier.

The power dissipation of an LNS addition/subtraction is

$$P_{\text{tot}} = P_{\text{comp}} + P_{\text{address}} + P_{\text{LUT}} + P_{\text{add}}, \qquad (38)$$

where P_{comp} is the power dissipated at an initial comparison to determine the larger of the two operands, P_{address} is the power dissipated for address generation, P_{LUT} is the power dissipated for Look-Up Table (LUT) access, and P_{add} is the power dissipated for the final addition. As the number of LUTs increases, and correspondingly the term P_{LUT} decreases, the address generation circuitry dissipates more power, i.e., P_{address} increases. A quantitative power dissipation comparison is offered in Table IV. The power dissipation data in Table IV refer to a 0.7μm CMOS technology, operating at a supply voltage of 5V. The power dissipation figures are obtained by adding the power dissipation terms of (38). Each term corresponds to an adder/subtractor or a LUT. The address generation consists of several parallel subtractions, to provide addressing to a particular LUT. The dissipation of the building blocks is taken from the technology datasheets [17]. The estimation procedure reveals that an average of 70% of power is dissipated for LUT access, 20% for address generation, and 10% for for initial comparison and the post-LUT addition. Notice that the LNS adder/subtractor is not implemented using a single LUT, but, instead using an assortment of small independent LUTs, as depicted in Fig. 11. A digest of techniques for the design of low-power ROMs is offered by de Angel and Swartzlander [18]. The breakdown of the LUT required to store the addition/subtraction function values is motivated by the shape of the addition and the subtraction function. As the value of the argument increases, fewer bits are required to represent the stored value. This observation is particularly important for LNS subtraction, which is very steep near zero, thus requiring a larger word length. At the cost of introducing hardware complexity overhead for the generation of the addresses to the small LUTs, the architecture of Fig. 11 is derived, in which each LUT corresponds to a region of the approximation interval where the function values are of equal word length. To further reduce memory size, it is noted that the most significant bit can be omitted, since it is always asserted. This approach, based on LUT breakdown, is similar to the memory compression scheme employed by Taylor *et al.* [5].

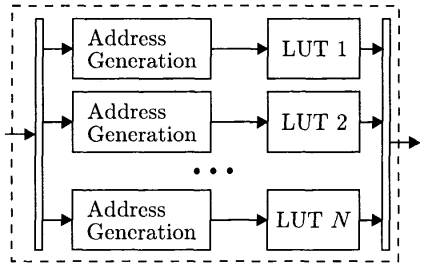


Figure 11: The organization of an LNS adder/subtractor memory, composed of N small LUTs.

Finally, complexity overhead is imposed by the linear-to-logarithmic and logarithmic-to-linear conversion. However, as the number of operations grows, the conversion overhead remains constant; therefore its contribution to the overall power budget can be compensated by a sufficient number of operations. Assume that the processing of an input sample requires N multiplication-

n	$P_{\text{for}} = P_{\text{inv}}$ (mW/MHz)	$N_{\min}$
8	1.53	3
10	2.42	4
12	5.70	7

Table V: Values of $N_{\min}$ for various linear word lengths n.

additions. Let the power dissipation of the linear, P_{lin}, and logarithmic system, $P_{\log}$, be given by

$$P_{\text{lin}} = N(P_{\text{m}}^{\text{lin}} + P_{\text{add}}^{\text{lin}}) \quad (39)$$

$$P_{\log} = P_{\text{for}} + N(P_{\text{m}}^{\log} + P_{\text{add}}^{\log}) + P_{\text{inv}}, \quad (40)$$

where P_{for} and P_{inv} is the power dissipated for forward and inverse conversion, and $P_{\text{m}}^{\text{lin}}$, $P_{\text{add}}^{\text{lin}}$, $P_{\text{m}}^{\log}$, and $P_{\text{add}}^{\log}$ denote the power dissipated for linear multiplication, linear addition, LNS multiplication, and LNS addition, respectively. The number $N_{\min}$ of multiplication/addition operations required to compensate the conversion overhead, can be obtained from (39) and (40) as

$$N\left(P_{\text{m}}^{\text{lin}} + P_{\text{add}}^{\text{lin}}\right) > P_{\text{for}} + N\left(P_{\text{m}}^{\log} + P_{\text{add}}^{\log}\right) + P_{\text{inv}} \Rightarrow \quad (41)$$

$$N_{\min} = \left\lceil \frac{P_{\text{for}} + P_{\text{inv}}}{P_{\text{m}}^{\text{lin}} + P_{\text{add}}^{\text{lin}} - P_{\text{m}}^{\log} - P_{\text{add}}^{\log}} \right\rceil. \quad (42)$$

Due to the structural identification of linear addition and logarithmic multiplication, $P_{\text{add}}^{\text{lin}} = P_{\text{m}}^{\log}$, (42) is reduced to

$$N_{\min} = \left\lceil (P_{\text{for}} + P_{\text{inv}}) \Big/ \left(P_{\text{m}}^{\text{lin}} - P_{\text{add}}^{\log}\right) \right\rceil. \quad (43)$$

Example values of $N_{\min}$ for various word lengths are shown in Table V, assuming $P_{\text{m}}^{\text{lin}}$ and $P_{\text{add}}^{\log}$ values taken from Table IV. A straightforward implementation of the converters is assumed, as a ROM LUT. Table V demonstrates that an LNS system becomes preferable after a moderate number of multiply-add operations.

5 Conclusions

The impact of LNS onto power dissipation of a processing system has been investigated. The discussion is based on proposed conditions of equivalence between LNS and fixed-point systems. It has been shown that LNS can lead to significant average bit activity reduction for linear input data of both uniform and correlated Gaussian distribution, in certain cases, such as anti-correlated Gaussian data. It has been found that the efficiency of the LNS representation is dominated by the choice of word lengths k and l, and—the often neglected—logarithm base b. Furthermore the impact of LNS onto the power dissipated in various arithmetic operations has been discussed from a hardware architecture viewpoint, in comparison to the fixed-point case. It is revealed that, given a sufficiently large computational complexity, power savings are achieved even when conversion overhead is taken into consideration.

Acknowledgment

The contribution of the referees, and in particular of referee B, is gratefully acknowledged.

References

[1] A. P. Chandrakasan and R. W. Brodersen, *Low power digital CMOS design*. Boston: Kluwer Academic Publishers, 1995.

[2] B. Parhami, *Computer arithmetic: Algorithms and hardware designs*. New York: Oxford University Press, 2000.

[3] S. Ramprasad, N. R. Shanbhag, and I. N. Hajj, "Analytical estimation of signal transition activity from word-level statistics," *IEEE Transactions on Computer-Aided Design*, vol. 16, pp. 718–733, July 1997.

[4] D. M. Lewis, "114 MFLOPS Logarithmic Number System arithmetic unit for DSP applications," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 1547 –1553, Dec. 1995.

[5] F. Taylor, R. Gill, J. Joseph, and J. Radke, "A 20 bit Logarithmic Number System processor," *IEEE Transactions on Computers*, vol. 37, pp. 190–199, Feb. 1988.

[6] M. G. Arnold, T. A. Bailey, J. R. Cowles, and J. J. Cupal, "Redundant logarithmic arithmetic," *IEEE Transaction on Computers*, vol. 39, pp. 1077–1086, Aug. 1990.

[7] M. G. Arnold, T. A. Bailey, J. R. Cowles, and M. D. Winkel, "Arithmetic co-transformations in the real and complex Logarithmic Number Systems," *IEEE Transactions on Computers*, vol. 47, pp. 777–786, July 1998.

[8] T. J. Sullivan, *Estimating the power consumption of custom CMOS digital signal processing integrated circuits for both the uniform and logarithmic number systems*. PhD thesis, Washington University, 1993.

[9] R. E. Morley, Jr., G. L. Engel, T. J. Sullivan, and S. M. Natarajan, "VLSI based design of a battery-operated digital hearing aid," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2512–2515, 1988.

[10] J. R. Sacha and M. J. Irwin, "The Logarithmic Number System for strength reduction in adaptive filtering," in *Proceedings of International Symposium on Low-Power Electronics and Design, (ISLPED'98)*, (Monterey, CA), pp. 256–261, 1998.

[11] E. Swartzlander and A. Alexopoulos, "The sign/logarithm number system," *IEEE Transactions on Computers*, vol. 24, pp. 1238–1242, Dec. 1975.

[12] V. Paliouras and T. Stouraitis, "Signal activity and power consumption reduction using the logarithmic number system," in *Proc. of ISCAS 2001*, 2001. to appear.

[13] I. Koren, *Computer Arithmetic Algorithms*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[14] P. E. Landman and J. M. Rabaey, "Architectural power analysis: The Dual Bit Type method," *IEEE Transactions on VLSI Systems*, vol. 3, pp. 173–187, June 1995.

[15] A. Papoulis, *Probability, random variables, and stochastic processes*. McGraw-Hill, third ed., 1991.

[16] T. K. Callaway and E. E. Swartzlander, Jr., "Low power arithmetic components," in *Low power design methodologies* (J. M. Rabaey and M. Pedram, eds.), Kluwer Academic Publishers, 1996.

[17] ES2, *ES2 ECPD07 library databook*. Rousset, France: European Silicon Structures, 1992.

[18] E. de Angel and E. E. Swartzlander, Jr., "Survey of low power techniques for ROMs," in *Proc. of 1997 International Symposium on Low Power Electronics and Design*, (Monterey, CA, USA), pp. 7–11, 1997.