Special Issue for International Conference of Advanced Materials Engineering and Technology (ICAMET 2013), 28-29 November 2013, Bandung Indonesia

Advances in Environmental Biology, 7(12) October Special Issue 2013, Pages: 3601-3606

A Comparative Analysis between Logarithmic Number System and Floating-point ALU

[1]R.C. Ismail, [2]J.N. Coleman, [1]N. Norzahiyah, [1]Z. Sauli

[1]School of Microelectronic Engineering, University Malaysia Perlis, Pauh Putra Campus, 02600 Arau, Perlis, Malaysia.
[2]School of Electrical & Electronic Engineering, Newcastle University, NE1 7RU United Kingdom.

ARTICLE INFO

Article history:
Received 11 September 2013
Received in revised form 21 November 2013
Accepted 25 November 2013
Available online 5 December 2013

Key words:
Logarithmic Number System and Floating-point ALU

ABSTRACT

The ease and accuracy of executing the multiplication and division operations by using either fixed-point addition or subtraction is what makes the logarithmic number system an attractive option. However, its main drawback is the complexity of performing addition and subtraction operations. In this paper, an analysis is carried out to compare the current state-of-the-art logarithmic number system and floating-point. The results predominantly show that the logarithmic number system offers better speed, accuracy and power-efficiency than floating-point. A new proposal for logarithmic addition and subtraction using a co-transformation method is presented, with the aim of improving its performance further.

INTRODUCTION

Digital signal processing (DSP) is a vital element for many electronic applications such as multimedia, digital filtering and household appliances. Most of the DSP algorithms need to be computed in real-time and require a wide dynamic range of numbers. The implementation of fixed point (FXP) DSP has a major limitation because of restricted accuracy which is the result of finite word-length effects. Floating point (FLP) DSP therefore has become an alternative to overcome this precision restriction in FXP architectures. However, FLP is not the only solution. Lately, many DSP researchers have been involved in proposing an efficient processor based on the logarithmic number system (LNS) which can improve the speed and accuracy of DSP architectures [1-6].

LNS provides major advantages over FXP and FLP in terms of speed and accuracy when it comes to computing multiplication and division operations. This is because of the similarity of their architectures with FXP addition and subtraction to perform the functions as shown in equation (1) and (2). The main obstacle within the LNS is executing addition and subtraction, due to the complexity of their architectures as described in equation (3) and (4).

$$\log_2(x \times y) = i + j \tag{1}$$

$$\log_2(x \div y) = i - j \tag{2}$$

$$\log_2(x + y) = i + \log_2(1 + 2^{j-i}) \tag{3}$$

$$\log_2(x - y) = i + \log_2(1 - 2^{j-i}) \tag{4}$$

where;

$i = \log_2 x$; $j = \log_2 y$

Several publications have proposed techniques to improve this trade-off, and as a result, the LNS able to operate at better performance than FXP or FLP [1,7]. Although the speed and accuracy of the LNS have been intensively discussed, less work has been done in considering the impact of LNS on power dissipation, and

Corresponding Author: R.C. Ismail, School of Microelectronic Engineering, University Malaysia Perlis, Pauh Putra Campus, 02600 Arau, Perlis, Malaysia.
E-mail: rizalafande@unimap.edu.my

specifically power-efficiency. An attempt by Paliouras and Stouraitis [15] has shown that the adoption of LNS can significantly reduce total power dissipation by reducing the bit precision. However, no previously published papers have critically explored the effect of power consumption in terms of power-efficiency in 32-bit word-lengths.

Given this trade-off, the aim of this work is to perform an analysis between 32-bit FLP and LNS in terms of speed, area, power consumption, power-efficiency and accuracy of addition, subtraction and multiplication operations based on an ASIC implementation. The results will then be used to propose a new LNS architecture which can significantly improve its performance.

The paper is organised into three major parts. In Section 2, a brief description of previously proposed LNS architectures is presented. A benchmark analysis between FLP [8] and LNS [1] is presented in Section 3. In Section 4, an idea of the newly proposed LNS technique is explained.

Related Work:

Arithmetic Algorithms:

The first and simplest LNS architecture for addition and subtraction was introduced in 1975 [8], which was a direct implementation of (3) and (4) using lookup tables (LUT) covering all possible values of $F(r = j - i)$. Due to the fact that the size of the LUT will increase exponentially when word-length increases, the proposed architecture was restricted to only a 12-bit device and would have been impractical for more than 20 bits. In 1991, Lewis [6] suggested a design for LNS addition and subtraction by implementing an interpolation scheme with the Taylor series, at a limit of 28 bits. This design used a first-order of Taylor series, which requires two LUTs, a multiplier and an adder to compute the logarithmic addition and subtraction functions. The use of FXP multiplication has a much higher cost, both in terms of latency and area, and dissipates more power. Therefore, with the increases in word sizes, the design will become inefficient and more complicated.

There is an alternative technique for function evaluation to replace the interpolation scheme. By using bipartite tables [9], only LUTs and adders are used, thus eliminating the multiplier in the interpolation scheme. However, the difficulty of implementing this technique to interpolate the subtraction function at r close to zero as shown in Fig. 1, will end up with larger table sizes. Apparently, about 2031K bits are required to estimate the log function at 24-bit word-lengths.

In 1999, Coleman [7] proposed a co-transformation method. This technique was introduced to cater for the problematic issue of interpolating the subtraction function at r close to zero. The technique transformed a subtraction with i and r, where $r > -1$ into an equivalent subtraction with $i2$ and $r2$, where $r2 < -1$. Computation of the latter subtraction then used a novel interpolation technique with an error correction algorithm. With accuracy and speed better than FLP, it is proof that this technique is a well-suited approach to be implemented for LNS architecture. Using the same concept as described by Coleman, Arnold in [11] substantially improved this co-transformation method by changing the ROM contents and the way that the interpolation unit was computed. Hence, slightly better accuracy than Coleman's method was achieved.

Units for Measurement:

In making a selection for an arithmetic unit for a particular application, several criteria must be considered. The speed is usually considered to be the worst case delay from input to the output. The area of each circuit is based on the total gates or cells.

In order to account for both power and performance, or so called power-efficiency, metrics such as BIPS/W, BIPS^2/W and BIPS^3/W are used [16-17]. The most popular metric is BIPS/W, which is equivalent to the reciprocal of mW/MIPS, which indicates on average how much energy a device needs to execute an instruction, i.e. its energy-efficiency. However, this metric cannot differentiate between two devices which have relatively different speed and performance. Hence, in order to measure the compromise between energy efficiency and performance, BIPS^2/W is used. Subsequently, others have raised the issue that neither of these metrics considers the effect of a reduction in voltage, since $power \approx C \cdot V^2 \cdot f$. Therefore BIPS^3/W is proposed as another important metric to measure energy efficiency when voltage scaling is taken into account. As a result, BIPS^3/W is independent of the supply voltage and this metric is widely used nowadays to compare devices using different supply voltages and fabrication technologies [18].

Benchmark Analysis:

Design Implementation:

We synthesised an ASIC implementation of one the freely downloadable FLP libraries [12]. It is IEEE 754 compliant, that is, it considers all exceptional cases. For LNS, the technique presented in [1] is used, which also covers all exceptional cases. The comparisons will focus on 32-bit (single-precision format) word-lengths based on a flowthrough (non-pipelined) architecture. Small modifications are made in the FLP libraries to exclude pipeline registers as initially they are in pipeline mode. The libraries are synthesised based on the Synopsys Design Compiler (DC) tool using the High Speed (HS) Faraday 0.13μm CMOS technology with the power supply set to 1.2V. Each of the arithmetic units has been exhaustively tested for functionality, worst case delay, total cell area, average power consumption and power-efficiency. In ensuring all libraries have been fully optimised, the Synopsys DC Ultra option is used which eventually can provide an automated flow that encapsulates data-path and timing optimisation, thus significantly improving the quality of the results. Default switching activity with random pattern numbers embedded in the DC tool is employed to estimate the total power dissipation in all libraries.

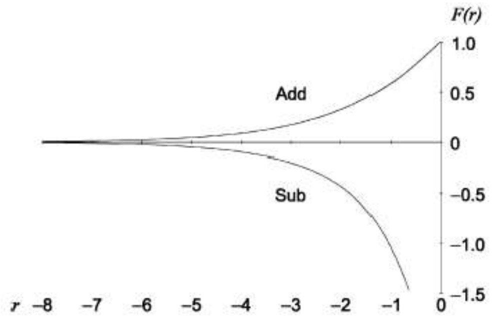
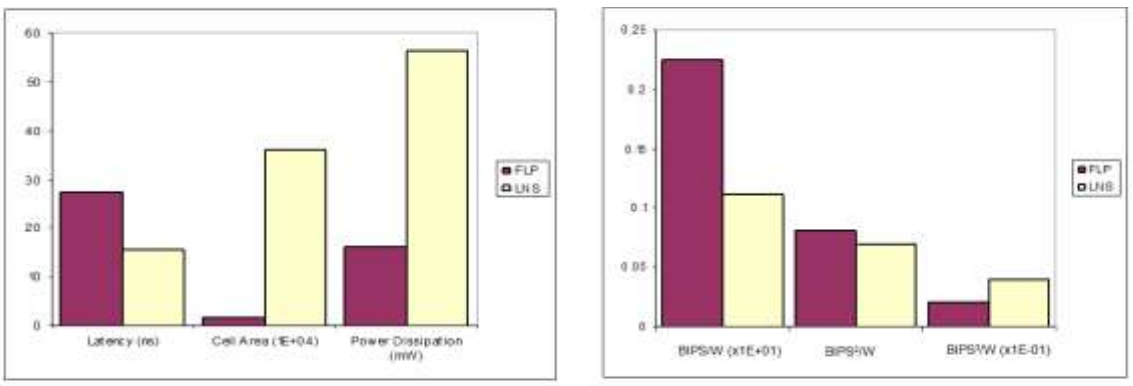


Fig. 1: LNS add and sub functions [1].

Latency, Area and Power:

As shown in Fig. 2(a), LNS addition and subtraction outperform FLP in terms of speed, a result that is comparable with the published results in [1]. In contrast, LNS tends to consume more power because of the use of multipliers during interpolation, which contribute nearly 50% of the total power dissipation in the system, on top of 25% dissipated from LUT access. From the point of view of power efficiency, the two results tend to cancel out; as seen in Fig. 2(b), one or the other appears more efficient depending on which metric is used.

In arithmetic operations, the multiplier is the most area hungry component and has always been a limiting factor in terms of speed [13]. In the LNS, the multiplier has the same format as the FXP adder, therefore it is able to achieve remarkable speed and the most optimum power-efficiency when compared with FLP, as shown in Fig. 3. In this analysis, LNS multiplier is based on a high performance FXP adder described in [14].

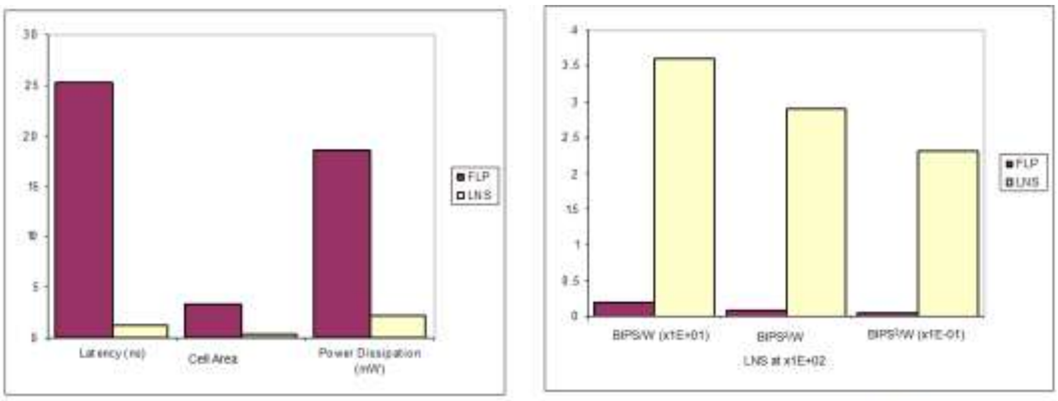


(a) Latency, area and power. (b) Power efficiency.

Fig. 2: Analysis of 32-bit add/sub unit.

Accuracy:

Results of accuracy evaluation are taken from [1], as the same architecture has been used for the analysis in Section 3.1. The accuracy analysis is not needed for LNS multiplication because it always returns exact results. As described in Table 1, the LNS add/sub upholds $e_{\text{max rel arith}}$ within the FLP limit of 0.5. Furthermore, values of $|e|_{\text{av rel arith}}$ are also equivalent to FLP even though a small bias does exist.



(a) Latency, area and power.

(b) Power efficiency.

Fig. 3: Analysis of 32-bit multiplier unit.

Case Study:

The Multiply-Accumulate (MAC) unit is amongst the most important operations in DSP applications, thus it is of interest to execute this operator for case study purposes. As expected, LNS has the lowest delay to propagate the inputs to the outputs as presented in fig. 4(a). Again, the LNS implementation has the larger area, and dissipates slightly more power. However, the reduced delay more than compensates for the larger power requirement and, as shown in Fig. 4(b), the LNS implementation offers a substantially better power efficiency regardless of the choice of metric.

Discussion:

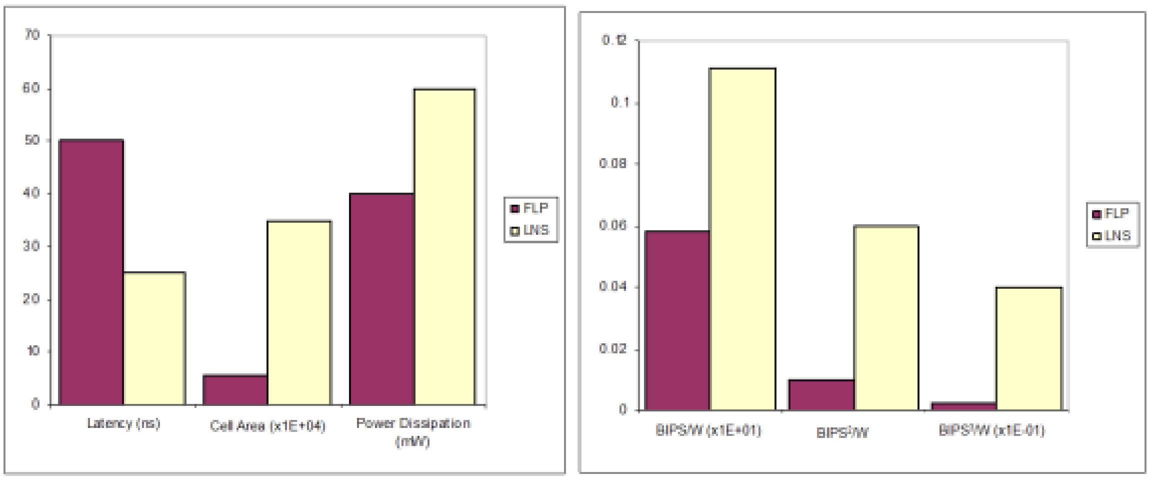
Evidently the LNS has achieved a great deal over FLP in terms of speed and accuracy. In terms of power efficiency, the results for addition and subtraction operations are ambivalent, but for the MAC operation the LNS already appears to deliver a dramatic improvement. Clearly it is worthwhile to pursue the goal of further reductions in LNS power dissipation, while doing nothing to the detriment of its speed. Reductions in area might also be sought. It has to be noted that the use of multipliers during the LNS interpolation can be costly, either in area or power, thus by excluding this unit one can potentially enhance LNS performance.

Proposed Technique:

As previously mentioned, the implementation of the multipliers in LNS whilst performing interpolation can significantly increase the total power and delay of the system. Hence, a new approach is under investigation which can bypass the interpolation process. In order to achieve a higher speed and lower power LNS, the proposed technique is a development of the range shifter concept illustrated in [7]. Based on Fig. 1, the range shifter will be recursively used at each interval of *r* at powers of two, i.e. -1..0, -2..-1, -4..-2, -8..-4 etc. Whichever range the initial subtraction falls into, the respective range shifter will transform *i* and *r* into two new values, *i2* and *r2*, where *r2* is in a range at least one power of 2 less than that of *r*. The same process will then be repeated continuously until it comes to $r2 \approx -27$. At this end point, the non-linear function reaches zero, thus giving equations (3) and (4) approximately equal to the updated value of *i2*. Clearly the achievement of the final result is an interpolation free process. Although the proposed technique is expected to involve a large size of LUT, on-going improvements are being carried out to gradually reduce them.

Table 1: Error analysis for 32-bit add/sub unit [1].

	Operation	$e_{max\ rel\ arith}$	$e_{min\ rel\ arith}$	$e_{av\ rel\ arith}$	$\lvert e\rvert_{av\ rel\ arith}$
LNS	Add	0.4544	-0.4047	0.0457	0.1777
	Sub	0.4414	-0.4952	0.0455	0.1776
FLP	Add	0.5	-0.5	0	0.1733
	Sub	0.5	-0.5	0	0.1733



(a) Latency, area and power. (b) Power efficiency.

Fig. 4: Analysis of 32-bit MAC unit.

Conclusion:

Hitherto, LNS has been shown to be superior to FLP in terms of speed and accuracy at 32-bit word-lengths, but little previous work has measured its effectiveness in terms of power-efficiency. The analysis presented here demonstrates that the LNS is capable of achieving substantially better power-efficiency than FLP, not only in multiplication, but also in the crucial MAC operation. A proposed improvement has been outlined, which by excluding the interpolation scheme is expected to deliver a higher performance still.

REFERENCES

[1] Coleman, J.N., E.I. Chester, C.I. Softley and J. Kadlec, 2000. "Arithmetic on the European Logarithmic Microprocessor," *IEEE Transaction on Computers,* pp: 702-715.

[2] Coleman, J.N., C.I. Softley, J. Kadlec, R. Matousek, M. Tichy, Z. Pohl, A. Hermanek and N.F. Benschop, 2008. "The European Logarithmic Microprocesor " *IEEE Transaction on Computers,* pp: 532-546.

[3] Taylor, F.J., R. Gill, J. Joseph and J. Radke, 1988. "A 20 Bit Logarithmic Number System Processor ", *IEEE Transaction on Computers,* pp: 190-200.

[4] Arnold, M.G., 2001. "A Pipelined LNS ALU ", *IEEE Computer Society Workshop on VLSI,* pp: 155-161.

[5] Jesus, G., G.A. Mark, B. Leonidas and V.K. Mayuresh, 2004. "LNS Architectures for Embedded Model Predictive Control Processors," *ACM International Conference on Compilers, Architecture, and Synthesis for Embedded Systems,* pp: 79-84.

[6] Lewis, D.M., 1990. "An Architecture for Addition and Subtraction of Long Word Length Numbers in the Logarithmic Number System," *IEEE Transaction on Computers,* pp: 1325-1336.

[7] Coleman, J.N. and E.I. Chester, 1999. "A 32-Bit Logarithmic Arithmetic Unit and Its Performance Compared to Floating-Point," *IEEE Symposium on Computer Arithmetic,* pp: 142-151.

[8] Earl E. Swartzlander and A.G. Alexopoulus, 1975. "The Sign/Logarithm Number System," *IEEE Transactions on Computers,* pp: 1238-1242.

[9] Schulte, M.J. and J.E. Stine, 1997. "Symmetric Bipartite Tables for Accurate Function Approximation," *Proc. 13th Symp. Computer Arithmetic*, pp: 175-183.

[10] Arnold, M.G., Thomas A. Bailey, John R. Cowles and Mark D. Winkel, 1998. "Arithmetic Co-Transformations in the Real and Complex Logarithmic Number Systems," *IEEE Transactions on Computers,* pp: 777-786.

[11] Arnold, M.G., 2002. "Improved Co-Transformation for LNS Subtraction," IEEE International Symposium on Circuits and Systems, pp: 752-755.

[12] Jidan Al-Eryani, 2008. "32-bit Floating-Point Unit (FPU100)," Opencores.org (2006), retrieved from http://www.opencores.org/projects.cgi/web/fpu100/overview, Accessed on the 12th June.

[13] Mahant-Shetti, S.S., P.T. Balsara and C. Lemonds, 1999. "High Performance Low Power Array Multiplier Using Temporal Tiling," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* 7: 121-124.

[14] Ruiz, G.A. and M. Granda, 2004. "An Area-Efficient Static CMOS Carry-Select Adder Based on a Compact Carry-Look-Ahead Unit," *Microelectronics Journal,* 35: 939-944.

[15] Paliouras, V. and T. Stouraitis, 2001. "Low-Power Properties of the Logarithmic Number System," *IEEE Symposium on Computer Arithmetic,* pp: 229-236.

[16] Srinivasan, V., D. Brooks, M. Gschwind, P. Bose, V. Zyuban, P.N. Strenski and P.G. Emma, 2002. "Optimizing Pipelines for Power and Performance," *Proceedings of the IEEE International Symposium on Microarchitecture,* pp: 333-344.
[17] Brooks, D.M., *et al*., 2000. "Power-Aware Microarchitecture: Design and Modeling Challenges for Next-Generation Microprocessors," *IEEE Micro.,* 20: 26-44.
[18] Grochowski, E. and M. Annavaram, 2006. "Energy per Instruction Trends in Intel Microprocessors," *Technology@Intel Magazine,* pp: 1-8.