Indonesian Journal of Electrical Engineering and Computer Science
Vol. 23, No. 3, September 2021, pp. 1708~1717
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v23.i3.pp1708-1717

Less memory and high accuracy logarithmic number system architecture for arithmetic operations

Siti Zarina Md Naziri, Rizalafande Che Ismail, Mohd Nazrin Md Isa, Razaidi Hussin
Faculty of Electronic Engineering Technology, Universiti Malaysia Perlis, Malaysia

Article Info

Article history:

Received May 15, 2021
Revised Jul 19, 2021
Accepted Jul 22, 2021

Keywords:

Accuracy
Co-transformation
Interpolation
Logarithmic number system
Memory consumption

ABSTRACT

Interpolation is another important procedure for logarithmic number system (LNS) addition and subtraction. As a medium of approximation, the interpolation procedure has an urgent need to be enhanced to increase the accuracy of the operation results. Previously, most of the interpolation procedures utilized the first degree interpolators with special error correction procedure which aim to eliminate additional embedded multiplications. However, the interpolation procedure for this research was elevated up to a second degree interpolation. Proper design process, investigation, and analysis were done for these interpolation configurations in positive region by standardizing the same co-transformation procedure, which is the extended range, second order co-transformation. Newton divided differences turned out to be the best interpolator for second degree implementation of LNS addition and subtraction, with the best-achieved BTFP rate of +0.4514 and reduction of memory consumption compared to the same arithmetic used in european logarithmic microprocessor (ELM) up to 51%.

Corresponding Author:

Siti Zarina Md Naziri
Faculty of Electronic Engineering Technology
Universiti Malaysia Perlis
Kampus Alam UniMAP, Pauh Putra, 02600 Arau, Perlis, Malaysia
Email: sitizarina@unimap.edu.my

1. INTRODUCTION

Since 1990s, various studies had been conducted on logarithmic number system (LNS) as the main operating number system in digital signal processor (DSP) to ensure its reliability as a substitution to floating point (FLP) arithmetic. The successful stories of logarithmic-based microprocessors started in the UK in 1995 by Coleman *et al*. in which they implemented some modifications later on to the LNS architecture in the European logarithmic microprocessor (ELM) [1], [2]. The ELM is a fabricated prototype logarithmic-based processor, which has a 32-bit co-transformation-based LNS arithmetic logic unit (ALU). This processor had demonstrated that LNS could be the best replacement to FLP with speed improvement, and had been assumed as a referred standard for related research on LNS designs. Besides ELM, another noteworthy LNS-based DSP processor is the gravity pipeline (GRAPE) by Makino and Taiji [3], which is an award-winning research for the computation of N-body gravitational forces among stars, and hidden markov modeling (HMM) for computing log-probabilities. However, the GRAPE supercomputer only utilized the LNS addition. The drawbacks of LNS subtraction preclude the wide spread of LNS usage in most applications.

Back to the arithmetic operations conducted in DSP, the LNS could simplify the highly-used arithmetic functions which are the FLP multiplication and division, by representing them in fixed-point (FXP) addition and subtraction respectively. Due to that, LNS had been extensively employed in arithmetic-

rich image processing areas, in which it is known as logarithmic image processing (LIP). The wide range of LIP implementation listed in [4]-[7] show the positive impact of logarithmic usage for image related implementation in DSP. Surprisingly, the operational speed for LNS could outperformed the FLP system for up to 50% [8]-[11]. Benefited the advantage of LNS, the blend of LNS and FLP or hybrid approach in [12]-[16] had increase the overall performance of the implementation. For a same purpose, there are also work in compromising residue number system (RNS) together with LNS [17], [18]. As the growing demand for LNS in applications, the urgency for the improvement of the LNS arithmetic had driven this work.

The strength of the LNS arithmetic is the computation for multiplication and division operation. However, the intricate representation for addition and subtraction operation in LNS suffer a lot of difficulty as it represented by a complex function in which known as non-linear function. Nonetheless, these drawbacks are alleviated and negotiable for low-precision applications as detailed in [19]. In describing the non-linear function, assume $Y = A \pm B$. Therefore, the logarithmic representation will be $\log_2 Y = \log_2(A \pm B)$. After derivation, the final equation is $\log_2 Y = \log_2 A + \log_2\left(1 \pm 2^{(\log_2 B - \log_2 A)}\right)$, with $\log_2\left(1 \pm 2^{(\log_2 B - \log_2 A)}\right)$ depicted the non-linear function. This function consumes enormous amount of ROMs for log value storage, thus increase the hardware and area requirements on the silicon. Hence, the challenge is to reduce the hardware cost while improving the speed of these operations in LNS at once.

To support the advancement of LNS arithmetic, various methods were discovered to enhance the quality of addition and subtraction operation of LNS as listed in [20] which include LUT and table partitioning method, and the most current method: the combination of co-transformation with interpolation. Yet, interpolation is found to be the finest scheme to be combined with other approach for logarithmic approximation. This finding evokes the idea for the new LNS architecture in this work.

In this paper, the key-most procedure in LNS which is the interpolation procedure is extensively elaborated in Section 2 with the best selection of interpolator for the new LNS. Section 3 exposes the process of evaluating and measuring the performance of the designed LNS system. The evaluation steps for each performance parameters in these processes which include memory utilization and accuracy are mentioned in detail. The results of the new LNS architecture are presented and discussed in Section 4. The final section concludes the main outcomes of the design.

2. THE ALGORITHM

2.1. Interpolation procedure in LNS

Interpolation is defined as a process of estimating values or other points using other data values at certain points [21]. This method implements closed-form representation of function as the basis for other numerical techniques, either numerical differentiation or integration. Essentially, the interpolated LUT is a method that works by associating the approximation function value from a single-hardware unit so-called as interpolating memory [22] together with the interpolation scheme for higher precision.

The usage of interpolators with higher degree can improve the accuracy of the interpolated results despite its disadvantages of utilizing a number of memory lookup tables and multipliers, especially in hardware implementation. However, each interpolator will differ in terms of higher degree segment representation albeit their linear representations are similar. Therefore, proper selection of interpolator could minimize the risk in producing results with higher precision.

Meanwhile, Lagrange interpolator that was used in recent LNS implementation as in [10] is not suitable to be implemented in higher degree form in hardware platform, as it has to recalculate the interpolation coefficients [23] with quite a number of multipliers. The labour of re-computing and high usage of multipliers may increase the area consumption and the speed of the overall design. Based on this reason, the Lagrange interpolator was omitted from this work, and the focus will be on other potential interpolators: Taylor and Newton Divided Difference interpolators. These interpolators are selected as they provide acceptable amount of multiplication unit that is needed to implement a higher degree portion of the interpolator, while able to utilize existing memory resources.

2.2. Newton divided difference interpolator

Newton divided difference interpolator is an interpolation technique used when the interval difference is irregular for all sequence of values. This polynomial interpolator is much preferable compared to lagrange as lagrange is not very competent and numerically unstable when there is addition of new points (Lagrange requires computing the polynomial again, from scratch) with the requirement of higher interpolation degree. Therefore, Newton is the choice of handling this type of data in interpolating these data incrementally. Taking the linear equation of newton divided difference (NDD) interpolator, the quadratic interpolatant of the same polynomial interpolator can be signified as follow:

$$f_2(r) = f_1(r) + a_2(r - r_0)(r - r_1) = a_0 + a_1(r - r_0) + a_2(r - r_0)(r - r_1) \quad (1)$$

Therefore, the second degree segment, $f_2(r_2)$ or y_2 will be denoted as:

$$f_2(r_2) = f_1(r) + a_2(r_2 - r_0)(r_2 - r_1) = a_0 + a_1(r_2 - r_0) + a_2(r_2 - r_0)(r_2 - r_1) = y_2 \quad (2)$$

Since then, the second degree coefficient, a_2 can be evaluated as:

$$a_2 = \frac{y_2 - y_1}{r_2 - r_1} - \frac{y_1 - y_0}{r_1 - r_0} \times \frac{1}{r_2 - r_1} = \frac{Df_1 - Df_0}{r_2 - r_1} \equiv D^2 f_0 \quad (3)$$

Using (3), any higher degree coefficient can be defined as:

$$a_N = \frac{D^{N-1} f_1 - D^{N-1} f_0}{r_N - r_1} \equiv D^N f_0 \quad (4)$$

Hence, (1) can be rewritten as:

$$f_2(r) = a_0 + Df_0(r - r_0) + D^2 f_0(r - r_0)(r - r_1) \quad (5)$$

With,

$$Df_0 = \frac{y_1 - y_0}{r_1 - r_0} \quad (6)$$

and

$$D^2 f_0 = \frac{y_2 - y_1}{r_2 - r_1} - \frac{y_1 - y_0}{r_1 - r_0} \times \frac{1}{r_2 - r_1} \text{ or } \frac{Df_1 - Df_0}{r_2 - r_1} \quad (7)$$

Each of the derivative table, Df_0 and $D^2 f_0$ are implemented as lookup tables. In another option, the second degree interpolator portion, $D^2 f_0$ may not require dedicated lookup table as it may be generated on-the-fly using the firstly created linear table, Df_0. However, this option will need extra addition and division operation for each generation of $D^2 f_0$ value. This action might sacrifice the area and the speed of the design on silicon upon the additional operational units used for the purpose. Therefore, the former option is selected for this work. For a fair comparison, the quadratic interpolatant for Taylor interpolator as been used in European logarithmic microprocessor (ELM) [2] was also evaluated in this work. The equation for Taylor interpolator up to a second degree can be represented as:

$$P(r) = f(r_0) + f'(r_0)(r - r_0) + \frac{f"(r_0)}{2}(r - r_0)^2 \quad (8)$$

or summarized as:

$$P_n(r) = \sum_{k=0}^{n} \frac{f^k(r_0)}{k!}(r - r_0)^k \quad (9)$$

with *n* as the degree of the polynomial. In the meantime, for $r = \log_2(2^r + 1)$, the first derivative is given as:

$$f'(r) = \frac{2^r}{2^r + 1} \quad (10)$$

and the second derivative as:

$$f"(r) = \frac{2^r}{(2^r + 1)^2} \quad (11)$$

Therefore, (8) can be rewritten as:

$$P_2(r) = f(r_0) + \frac{2^r}{2^r + 1}(r - r_0) + \frac{2^r}{2(2^r + 1)^2}(r - r_0)^2 \quad (12)$$

Based on this equation, each of the differential tables are generated and stored into lookup tables. For Taylor interpolator, LUT is the only method that can be used in approximating the final value of the LNS addition and subtraction.

3. RESEARCH METHOD

This work focussed on the improvements of the logarithmic arithmetic unit design in [11] by enhancing the two important procedures for the LNS addition and subtraction: interpolation and co-transformation. The new arrangement of the two procedures would be able to improve the addition and subtraction operation in LNS, which could represent a whole new logarithmic arithmetic unit architecture. The arrangements of these procedures can be represented by the function $F(r)$ in Figure 1, which illustrates the r region with dedicated procedures involved in positive region of r.

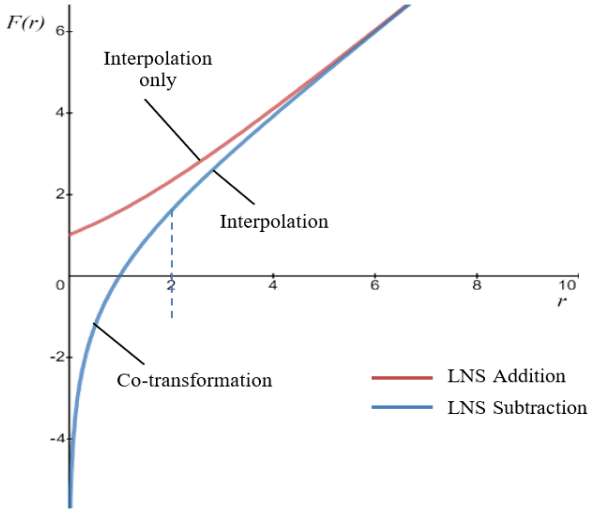

Figure 1. Interpolation and co-transformation procedure according to region for the new LNS addition and subtraction architecture design

The process of design validation requires specific simulation tools for each purpose. Therefore, the the addition and subtraction operations LNS design in this work had gone through the simulation process using specially modelled simulator programs with the control of appropriate data and results from the previous LNS design. The simulators were designed to perceive the best lookup table arrangements for the interpolator and co-transformation procedure in achieving the BTFP rate. The size and the number of lookup tables can be modified repeatedly and accordingly until it meets the design's expectation.

The LNS design simulator programs for both arithmetic operations were constructed in C language and the execution of the compilation process were done using Dev-C Compiler in Intel Core i7 processor. The execution of these programs will allow the measurement of the worst-case error for each lookup table combination. The best table size combinations for the interpolation procedure are retrieved from a number of compilations by the various table arrangements with the limit of accuracy within the FLP boundaries. These simulators also allowed the modification of variables as stated in Table 1. These tables which are labelled as F, D, and S are used in at most six segmented ranges which implied the concept of power-of-two partitions.

Figure 2 describes the development of the simulator programs which includes the main elements of the simulator. The process began with the design of the exponent and logarithm functions that were embedded and used throughout the simulations. Next was the process of describing the interpolator and co-transformation model. Note that the interpolator model covers the whole range of r for LNS addition and subtraction function, except for the subtraction function that allocates certain range for co-transformation region only. Therefore, the co-transformation model was specially designed to compute the difference, r that falls into the region (the region may range from $r = 0$ up to $r = 2$). For table computation purposes, a special module was also designed to generate the values for the lookup tables used in the interpolation and co-transformation functions.

Table 1. Simulation variables for the interpolator

Parameter	Description
F	Stored function value at r_n
D	Stored function of linear derivative at r_n
S	Stored function of second degree derivative at r_n
δ_n	Current value of $r - r_n$
δ_{n+1}	Current value of $r - r_{n+1}$
r	Current operand difference, in guarded format
r_n	Stored interpolation at point n
r_{n+1}	Stored interpolation at point $n + 1$

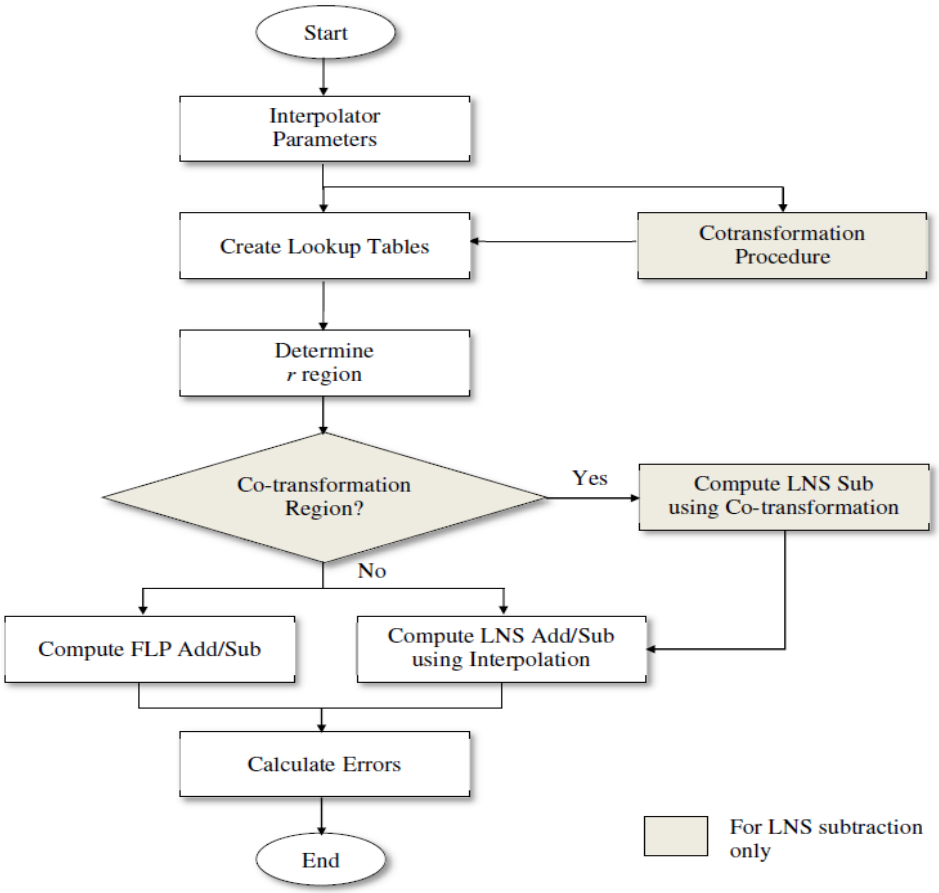

Figure 2. Simulator design flow for LNS addition and subtraction

Based on the region indicated by r, the next process will be the generation of the approximation results for the LNS addition and subtraction functions. At the same time, the exact result of r was computed through the FLP result module. This value will be compared with the approximated result in the next module. The comparison will expose the accuracy of the approximated result which indicates the precision of the developed LNS system.

4. RESULTS AND DISCUSSION

The analysis concentrates on the implementation of a second degree for various interpolators. Three interpolator tables are involved, which are F, D and S table with null involvement of the error correction scheme for the second degree interpolator-based design. As the reason pointed out in introduction section, only two interpolators were selected for further implementation testing which are Taylor and Newton divided difference interpolator. As the means to measure the accuracy of the LNS system and to compare it with the FLP system, the mathematical expressions as defined in [2] are adopted. The analysis deals with various lengths of guard bits. Table 2 conveys the accuracy of each design according to the interpolators and various number of guard bits.

Table 2. Maximum relative error for LNS addition and subtraction of second degree interpolators with extended co-transformation

Interpolator	F, D, S table size	Guard bits	Addition		Subtraction		Max error
			Rel err (hi)	Rel err (lo)	Rel err (hi)	Rel err (lo)	
Taylor	256	5	+0.4071	-0.3899	+0.4639	-0.5420	+0.5420
	256	6	+0.4171	-0.3682	+0.4439	-0.5225	+0.5225
Newton Divided Difference	256	3	+0.3276	-0.4757	+0.5166	-0.5566	+0.5566
	256	4	+0.3564	-0.4110	+0.4359	-0.4514	+0.4514
	256	5	+0.3673	-0.3786	+0.4093	-0.4261	+0.4261

In this analysis, the size of F, D and S table used for each interpolator are fixed to 256 words. Thus, the focal factor of this analysis is the size of guard bits. By varying the guard bits, it is found that the subtraction operation of LNS using Taylor interpolator could not achieve the FLP standard of 0.5 u.l.p. even with the usage of six guard bits. This is due to the fact that the approximation by Taylor is concentrated at a specific point which usually provide inaccurate approximations as the new point moves away from the particular point [24]. This condition even applies for higher degree polynomial. Thus, the situation limits the Taylor polynomial to only approximate numbers that is close to its initial point.

However, the same design was able to achieve the standard using Newton divided difference interpolator with smaller extent of guard bits, which in this case is of only four guard bits. The proposed design produced +0.4514 of maximum relative error, which is in the range of the better-than-floating-point (BTFP) rate. It should be noted that the relative error of addition operation using these interpolators is not the main issue in this work since the subtraction function controls the maximum error rate of these two complex LNS functions caused by co-transformation. The memory consumption analysis put more focus on the LNS subtraction design with Newton divided difference interpolator since it has provided the most optimum accuracy among other interpolators based on the implementation of second degree interpolator environment. Table 3 summarizes the storage required for memory tables for LNS addition and subtraction designs using first and second degree of the interpolator.

Table 3. Comparison of storage requirements for LNS addition and subtraction operation of various degree Newton divided difference interpolator with multiple range of co-transformation

Design	Co-transformation			Interpolation			Total
	Table	Organization	# Bits	Table	Organization	# Bits	
Standard Co-transformation + first degree NDD	F1	128 words	3,840	F, D, E Add	256 words × 6	96,256	226,560
	F2	256 words	7,936	F, D, E Sub	256 words × 5	82,688	
	F3	256 words	8,192	P	1,024 words	27,648	
	Subtotal	640 words	19,968	Subtotal		206,592	
Extended Co-transformation + first degree NDD	F1	256 words	7,680	F, D, S Add	256 words × 6	96,256	213,760
	F2	256 words	7,936	F, D, S Sub	256 words × 4	66,048	
	F3	256 words	8,192	P	1,024 words	27648	
	Subtotal	768 words	23,808	Subtotal		189,952	
Extended Co-transformation + second degree NDD	F1	256 words	7,680	F, D, S Add	256 words × 6	89,088	173,056
	F2	256 words	7,680	F, D, S Sub	256 words × 4	60,672	
	F3	256 words	7,936				
	Subtotal	768 words	23,296	Subtotal		149,760	

Note: NDD = Newton divided difference

Table 3 marks the impact of co-transformation range expansion up to $r = 2$ ($0 < r < 2$) that had greatly improved the bit storage utilization. Specifically, the extension of co-transformation range for the LNS subtraction function had reduced nearly 6% of the total memory required by a standard range of second order co-transformation ($0 < r < 1$) using the first degree Newton Divided Difference interpolator. The memory consumption was further reduced up to 24% by the usage of second degree of the same interpolator. The proposed design had also evaded the second degree table that caters the range from $16 < r < 32$ since the word of each address of this table is mostly near to 0. This action could reduce the usage of memory tables, and thus possibly condensed the overall silicon area for hardware design later. It also allows the interpolation process to be neglected for certain range as the range itself equals to the value of the LNS addition and the subtraction, $F(r)$, as declared in [25].

It should be highlighted that these statistics do not represent the actual hardware representation of the LNS addition and subtraction functions on a silicon since it is only an estimation of memory consumption in terms of bits by the LUTs used in interpolation and co-transformation. However, the slight reduction in memory consumption will benefit the overall hardware design by eliminating hardware logics that are related.

Beforehand, the accuracy analysis started with the implementation of LNS addition and subtraction functions in comparison with the first degree and second degree interpolations using newton divided difference interpolator. Among these two designs, the LNS addition and subtraction designs with a second degree newton divided difference interpolator had achieved the least maximum relative error that constitutes 8% more accuracy as compared to the design with the first degree interpolator. Figure 3 demonstrates that the employment of higher degree interpolators had greatly improved the accuracy of the LNS addition and subtraction results. However, the usage of a higher degree for interpolation might have an issue during hardware implementation in terms of an increase in the silicon area for the design as it requires additional logics to engage with additional multipliers.

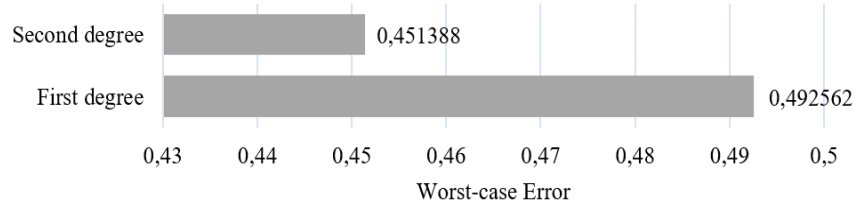

Figure 3. Accuracy comparison of LNS addition and subtraction using first and second degree Newton divided differences interpolation

Memory consumption for storage tables of the LNS addition and subtraction designs as portrayed in Figure 4 seems to have a decrement pattern with the usage of a second degree newton divided difference interpolator. The particular design was capable to reduce 19% of the total memory bits as compared to the design with a first degree interpolator. The usage of a second degree interpolator had slightly reduced the usage of memory bits in co-transformation as it consumed 23,296 bits as compared to the competitor with 23,808 bits. The elimination of P and E tables that are associated with the error correction portion in the first degree interpolator appears to result in a great reduction of memory bits allocated for interpolation tables. The interpolation procedure for the design with the second degree interpolator had accumulated 149,760 of memory bits while the other design consumed 189,952 bits for the table storage. To be precise, the former design only required 79% of memory bits as compared to the latter design.

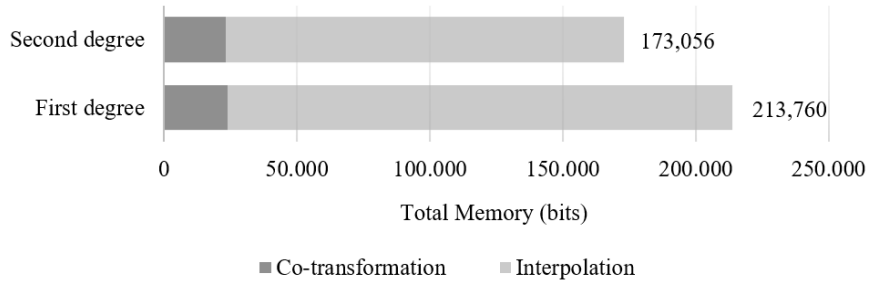

Figure 4. Storage consumption comparison using first and second degree Newton divided differences interpolation

The performance of the proposed work is compared with two leading LNS designs; Coleman's ELM [2] and Ismail and Coleman [10] in terms of accuracy. Both LNS arithmetic designs are selected as a benchmark for this research as these designs are constructed with similar steps, highlights, and domains: co-transformation and interpolation, and implemented using the same 0.18 µm technology. Table 4 shows the analysis of relative error by comparing the proposed design with the above-mentioned LNS designs. To note, these designs set the same restriction of equivalent FLP accuracy of 0.5 u.l.p. Based on the table, the newly designed LNS addition and subtraction operations offers the most accurate results as compared to the two other designs with similar configuration, while gaining the lowest relative error for the addition function.

Table 4. Error analysis of proposed and other LNS addition and subtraction designs with similar configuration and design technology

Error	Function	ELM [2]	Coleman & Ismail [10]	Proposed Design
Rel error (ns)	Add	+0.4544	+0.4527	+0.4110
	Sub	-0.4414	-0.4987	-0.4514
Max error	-	+0.4544	+0.4987	+0.4514

The usage of Newton divided difference interpolation up to a second degree had delivered more accurate results of addition and subtraction of LNS with the relative error of +0.4514 as reflected in Figure 5. The result offers 9% more accurate as compared to [10]. This work also beats the accuracy of special function (SF)-based LNS design [26] and even two FLP unit (FPU) designs [11]. This proves that the higher the degree of interpolator used, the better the accuracy achieved, as evidenced by previous equations. The statistics also demonstrates that LNS could offer better accuracy with achieving BTFP rate. Other than that, the usage of higher interpolator can be a substitution of the error correcting scheme, which is purposely designed to improve accuracy, as been integrated with the first degree interpolator as in [2] and [10]. However, the results may differ and may not applicable for each interpolator used.

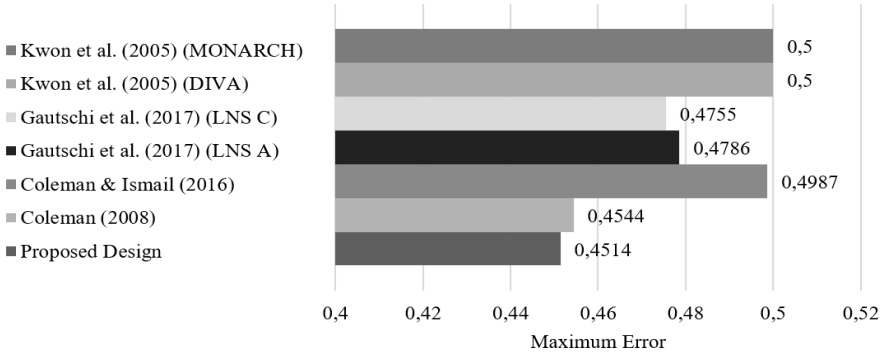

Figure 5. Accuracy comparison of proposed LNS design with other LNS designs and FLUs

On another aspect, the adoption of Newton divided difference interpolator up to the second degree in the new LNS design improves the memory consumption of the interpolation process of this work as it managed to achieve 149,760 bits for its F, D, and S table storage. Those tables constitute with the derivative (supporting interpolation degree) tables. This marks an overall storage reduction of 51% of [2] and 6% of [10] as illustrated in Figure 6. The expansion of co-transformation region up to $r = 2$, however, had caused a slight increment of co-transformation function storage of 9.6% (2,048 bits) as compared to the previous second order co-transformation as a result of a bigger size of co-transformation table that managed the extension range. In return, the table size associates with the interpolation process alone only contributes 86% of total memory bits, which is 2% lesser compared to [10]. Therefore, these outcomes prove that a storage-efficient LNS design could be realized with interpolators of a higher degree with some modifications on the algorithm structure (in this work, the second order co-transformation with range expansion) in order to gain the accuracy offered by the higher degree interpolation. This could reduce or even dismiss the initial assumption that a higher degree interpolation could increase the storage usage in total due to the fact that these degrees will consume extra lookup tables.

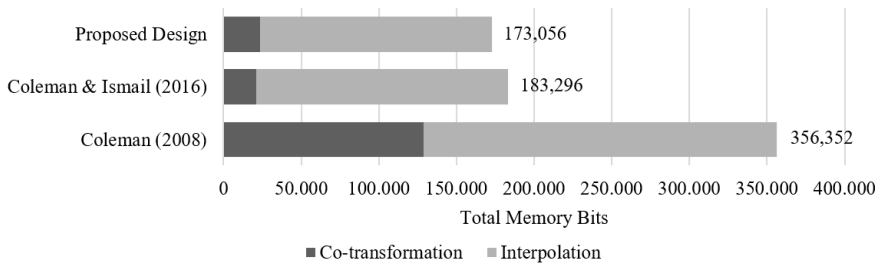

Figure 6. Storage consumption comparison of proposed and other LNS addition and subtraction design

5. CONCLUSION

The new implemented LNS design has shown the improvement of the extended range, second order co-transformation and interpolation function in the positive region of *r*. The new LNS design had been utilizing the second degree interpolation, which results in a great improvement in accuracy. The design increases the accuracy by the range of 2% to 9% as compared to the most recent and similar configuration and technology of LNS designs. Besides, the memory bit usage during interpolation process also able to be reduced even though the number is not too significant. In conclusion, the enhancement of interpolation process offers LNS design with better accuracy and managable memory consumption that can be benefited by the image processing applications, with LIP areas in particular.

REFERENCES

[1] J. N. Coleman, E. I. Chester, C. I. Softley, and J. Kadlec, "Arithmetic on the European logarithmic microprocessor," in *IEEE Transactions on Computers*, vol. 49, no. 7, pp. 702-715, July 2000, doi: 10.1109/12.863040.

[2] J. N. Coleman, *et al.*, "The European Logarithmic Microprocesor," in *IEEE Transactions on* Computers, vol. 57, no. 4, pp. 532-546, April 2008, doi: 10.1109/TC.2007.70791.

[3] J. Makino, M. Taiji, T. Ebisuzaki, and D. Sugimoto, "GRAPE-4: A Massively Parallel Special-Purpose Computer for Collisional N -Body Simulations," *Astrophys. J.*, vol. 480, no. 1, pp. 432-446, May 1997, doi: 10.1086/303972.

[4] S. Z. Md Naziri, R. C. Ismail, and A. Y. Md Shakaff, "The design revolution of logarithmic number system architecture," *2014 2nd International Conference on Electrical, Electronics and System Engineering (ICEESE)*, 2014, pp. 5-10, doi: 10.1109/ICEESE.2014.7154603.

[5] H. Gouinaud, Y. Gavet, J. Debayle, and J. Pinoli, "Color correction in the framework of Color Logarithmic Image Processing," *2011 7th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2011, pp. 129-133.

[6] M. Jourlin, J. Breugnot, F. Itthirad, M. Bouabdellah, and B. Closs, "Chapter 2-Logarithmic Image Processing for Color Images," in *Advances in Imaging and Electron Physics*, vol. 168, pp. 65-107, 2011, doi: 10.1016/B978-0-12-385983-9.00002-8.

[7] K. Mohammad, S. Agaian, and F. Hudson, "Implementation of Digital Electronic Arithmetics and its application in image processing," *Computers & Electrical Engineering*, vol. 36, no. 3, pp. 424-434, May 2010, doi: 10.1016/j.compeleceng.2009.10.002.

[8] M. G. Arnold and S. Collange, "A Real/Complex Logarithmic Number System ALU," *in IEEE Transactions on Computers*, vol. 60, no. 2, pp. 202-213, Feb. 2011, doi: 10.1109/TC.2010.154.

[9] R. C. Ismail and J. N. Coleman, "ROM-less LNS," *2011 IEEE 20th Symposium on Computer Arithmetic*, 2011, pp. 43-51, doi: 10.1109/ARITH.2011.15.

[10] J. N. Coleman and R. Che Ismail, "LNS with Co-Transformation Competes with Floating-Point," in *IEEE Transactions on Computers*, vol. 65, no. 1, pp. 136-146, 1 Jan. 2016, doi: 10.1109/TC.2015.2409059.

[11] Taek-Jun Kwon, J. Sondeen, and J. Draper, "Design Trade-Offs in Floating-Point Unit Implementation for Embedded and Processing-In-Memory Systems," in *2005 IEEE International Symposium on Circuits and Systems*, vol. 4, pp. 3331-3334, 2005, doi: 10.1109/ISCAS.2005.1465341.

[12] T. Stouraitis, "A hybrid floating-point/logarithmic number system digital signal processor," in *International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 1079-1082, 1989, doi: 10.1109/ICASSP.1989.266619.

[13] P. Lee, "An Evaluation of a Hybrid-Logarithmic Number System DCT/IDCT Algorithm," in *2005 IEEE International Symposium on Circuits and Systems*, vol. 5, no. 3, pp. 4863-4866, 2005, doi: 10.1109/ISCAS.2005.1465722.

[14] H. Zhang, H. J. Lee, and S. -B. Ko, "Efficient Fixed/Floating-Point Merged Mixed-Precision Multiply-Accumulate Unit for Deep Learning Processors," *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1-5, doi: 10.1109/ISCAS.2018.8351354.

[15] F. Lai, "A 10 ns hybrid number system data execution unit for digital signal processing systems," *in IEEE Journal of Solid-State Circuits*, vol. 26, no. 4, pp. 590-599, April 1991, doi: 10.1109/4.75060.

[16] C. Y. Sheng, R. C. Ismail, S. Z. M. Naziri, M. N. M. Isa, S. A. Z. Murad, and A. Harun, "Hybrid Floating Point/Logarithmic Number System Processor," in *IOP Conference Series: Materials Science and* Engineering, vol. 932, no. 1, p. 012059, 2020, doi: 10.1088/1757-899X/932/1/012059.

[17] M. G. Arnold, I. Kouretas, V. Paliouras, and A. Morgan, "One-Hot Residue Logarithmic Number Systems," *2019 IEEE 29th Int. Symp. Power Timing Model. Optim. Simulation, PATMOS 2019*, pp. 97-102, 2019, doi: 10.1109/PATMOS.2019.8862159.

[18] M. G. Arnold, V. Paliouras, and I. Kouretas, "Implementing the Residue Logarithmic Number System Using Interpolation and Cotransformation," in *IEEE Transactions on Computers*, vol. 69, no. 12, pp. 1719-1732, Dec. 2020, doi: 10.1109/TC.2019.2930514.

[19] B. Parhami, "Computing with logarithmic number system arithmetic: Implementation methods and performance benefits," *Comput. Electr. Eng.*, vol. 87, p. 106800, August 2020, doi: 10.1016/j.compeleceng.2020.106800.

[20] S. Z. M. Naziri, R. C. Ismail, and A. Y. M. Shakaff, "An Analysis of Interpolation Implementation for LNS Addition and Subtraction Function in Positive Region," *2016 International Conference on Computer and Communication Engineering (ICCCE)*, 2016, pp. 499-504, doi: 10.1109/ICCCE.2016.110.
[21] L. V. Fausett, ‘‘Applied Numerical Analysis Using MATLAB,’’ *2nd Editio. New Jersey: Pearson Education, Inc*., 2008.
[22] A. S. Noetzel, "An interpolating memory unit for function evaluation: analysis and design," in *IEEE Transactions on Computers*, vol. 38, no. 3, pp. 377-384, March 1989, doi: 10.1109/12.21124.
[23] C. Solares, E. W. Vieira, and R. Mínguez, “Functional Networks and the Lagrange Polynomial Interpolation,” *International Conference on Intelligent Data Engineering and Automated Learning. Springer*, Berlin, Heidelberg, vol. 4224, pp. 394-401, 2006, doi: 10.1007/11875581_48.
[24] L. Richard and J. Burden, ‘‘Douglas faires, Numerical Analysis,’’ *9th editio. Boston: Brooks/Cole, Cengage Learning*, 2011.
[25] S. Z. M. Naziri, R. C. Ismail, and A. Y. M. Shakaff, “Arithmetic addition and subtraction function of logarithmic number system in positive region: An investigation,” in *2015 IEEE Student Conference on Research and Development (SCOReD)*, Dec. 2015, pp. 447-450, doi: 10.1109/SCORED.2015.7449376.
[26] M. Gautschi, M. Schaffner, F. K. Gürkaynak, and L. Benini, "An Extended Shared Logarithmic Unit for Nonlinear Function Kernel Acceleration in a 65-nm CMOS Multicore Cluster," in *IEEE Journal of Solid-State Circuits*, vol. 52, no. 1, pp. 98-112, Jan. 2017, doi: 10.1109/JSSC.2016.2626272.

BIOGRAPHIES OF AUTHORS

Siti Zarina Md Naziri received the B.Eng (Hons) and M.Sc in Electrical and Electronics Engineering in 1999 and 2002, respectively from Universiti Sains Malaysia, while her Ph.D in Microelectronic Engineering from Universiti Malaysia Perlis in 2020. Presently, she is a senior lecturer in the Faculty of Electronic Engineering Technology, Universiti Malaysia Perlis and have been serving the University since 2006. Her research interest includes digital systems, VLSI and FPGA design, data encryption and steganography. She is a senior member of the Institute of Electrical and Electronics Engineers (IEEE) and IEEE Circuit and System (CAS), a member of Malaysian Solid State Science and Technology Society (MASS) and a graduate Engineer of the Board of Engineers Malaysia (BEM).

Rizalafande Che Ismail received the Ph.D degree from Newcastle University, United Kingdom, in Microelectronics System Design. He is now a Professor in the Faculty of Electronic Engineering Technology, Universiti Malaysia Perlis, and is engaged in designing low-power and high-speed architectures for digital systems. He is a senior member of the Institute of Electrical and Electronics Engineers (IEEE), a member of the British Computer Society (BCS), and certified Professional Engineer of the Board of Engineers Malaysia (BEM).

Mohd Nazrin Md Isa is a senior lecturer at the Faculty of Electronic Engineering Technology, Universiti Malaysia Perlis (UniMAP). He has joined the universiti as a lecturer since 2005. He received his B.Sc. degree in Electrical and Electronic Engineering with honours from Kolej Universiti Teknologi Tun Hussein Onn (KUiTTHO) in 2003, M.Sc. degree from Universiti Sains Malaysia in 2005 and the Ph.D. degree in Field Programmable Gate Array (FPGA) design from the University of Edinburgh, Scotland in 2013. He has authored more than 40 published technical papers in electronics and FPGA design. His current research activities include Internet of Things (IoT), FPGA and VLSI design with applications in bioinformatics, and image processing.

Razaidi Hussin received the Ph.D degree in Electronic and Electrical Engineering from University of Glasgow, UK in 2017 with a focus on oxide-reliability issues in complementary metal-oxide-semiconductor nanoscale devices. He joined Universiti Malaysia Perlis (or previously know as KUKUM) in 2002. He is currently a fulltime Senior Lecturer with the Faculty of Electronic Engineering Technology, Universiti Malaysia Perlis.