Approximate Leading One Detector Design for a Hardware-Efficient Mitchell Multiplier

Shyama Gandhi
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
smgandhi@ualberta.ca

Mohammad Saeed Ansari
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
ansari2@ualberta.ca

Bruce F. Cockburn
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
cockburn@ualberta.ca

Jie Han
Dept. of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
jhan8@ualberta.ca

***Abstract*—We propose two approximate leading one detector (LOD) designs and an approximate adder (for summing two logarithms) that can be used to improve the hardware efficiency of the Mitchell logarithmic multiplier. The first LOD design uses a single fixed value to approximate the *'d'* least significant bits (LSBs). For *d*=16 this design reduces the hardware cost by 19.91% compared to the conventional 32-bit Mitchell multiplier and by 15.19% when compared to a recent design in the literature. Our design is smaller by 32.33% and more energy-efficient by 56.77% with respect to a conventional Mitchell design. The second design partitions the *'d'* bits into smaller fields and increases the accuracy by using a multiplexing scheme that selects a closer approximation to the actual input value. This design reduces the hardware cost by 17.98% compared to the original Mitchell multiplier and by 13.15% when compared to the other recent design. Our design is smaller by 29.17% and more energy-efficient by 56.18% with respect to the conventional Mitchell design. In the approximate adder, the *'m'* least significant bits are set to a fixed bias of alternating ones and zeros. The optimal values of *'d'* and *'m'* are chosen to preserve the full accuracy of the conventional Mitchell multiplier while reducing the hardware cost. The new designs produce increased signed errors for inputs less than or equal to 2^{16} but for larger numbers the accuracy is equal to that of the conventional Mitchell multiplier. The approximation affects only the 2^{16}-1 smallest input values out of 2^{32}-1. The new approximate multipliers are suitable for applications where approximation errors affecting the least significant digits can be tolerated.**

***Index Terms*—leading one detector, approximate adder, Mitchell logarithmic multiplier, approximate arithmetic**

I. Introduction

Multiplication is a frequent operation in many computing systems that accounts for significant hardware cost and power consumption. Approximate arithmetic can reduce the hardware cost and can be competitive for applications where an inaccurate result is sufficient. Approximate circuits are used in error-tolerant applications such as digital signal processing (DSP), machine learning and in several problems in scientific computing [1].

When multiplication speed is a more important factor than great accuracy, the logarithmic number system can be preferable over linear arithmetic because it eliminates the generation of partial products. In some DSP applications, the logarithmic system has been used successfully despite some loss in accuracy for arithmetic operations [2].

Mitchell proposed a logarithmic multiplier [2] that computes the binary logarithm of the two input operands using a leading one detector (LOD) and then computes the antilogarithm of their sum. The LOD plays a significant role in logarithmic multiplier design [3][4]. We studied 32-bit LOD designs using 4-bit LOD circuits and we found that the approximation of the least significant LODs can significantly reduce the hardware cost while preserving the accuracy of the conventional Mitchell multiplier. An approximate 16-bit logarithmic multiplier using the so-called set-one-adder [1] that overestimates the approximate product was also studied, but found to be uncompetitive for 32 bits.

In this paper we propose two approximate LOD designs. Design I approximates the four least significant 4-bit LODs with a constant bias, and the sixteen least significant bits (LSBs) of the adder are approximated with alternating ones and zeros. The optimal number of bits to be approximated was obtained using error metrics generated from numerical experiments using large random samples of the input space while preserving the accuracy of the conventional Mitchell multiplier. Design II approximates the four least significant 4-bit LODs to one of four constant biases, which is selected using the output from a simple *OR* operation on the input bits of the four least significant LODs.

The rest of the paper is organized as follows: Section II reviews the conventional approximate Mitchell multiplier and 4-bit LOD designs. Section III describes the two proposed approximate multiplier designs. Section IV presents simulation results, accuracy analyses and hardware costs for the proposed designs and compares them with another recent design in the literature. The main conclusions are given in Section V.

II. Background

A. The Conventional Mitchell Multiplier

Let *A* and *B* be the two 8-bit inputs to a Mitchell multiplier, expressed as follows [1]:

$$A = 2^{k_1}(1 + m_1), 0 \leq m_1 < 1 \quad (1)$$

$$B = 2^{k_2}(1 + m_2), 0 \leq m_2 < 1 \quad (2)$$

where the leading one position for both the inputs is denoted by k_1 and k_2, respectively. The mantissas m_1 and m_2 for the approximate logarithm approximations are the bits to the right side of the leading one for both inputs. The Mitchell algorithm approximates the actual base-2 logarithm of the input N as $log_2(N)=k+m$ instead of the actual value $log_2(N)=k+log_2(1+m)$. The two approximate logarithms for inputs A and B are then given by [1]:

$$log_2(A) = k_1 + m_1 \quad (3)$$

$$log_2(B) = k_2 + m_2 \quad (4)$$

These logarithms are then added as follows [1]:

$$log_2(A \times B) = k_1 + k_2 + m_1 + m_2 \quad (5)$$

The output carry from the addition of the mantissa fields is the carry-in for the addition of k_1 and k_2. The approximated product will be obtained by taking the anti-logarithm as follows: Set the Most Significant Bit (MSB) to 1, append the mantissa m of $k\&m$ (where & denotes concatenation), and then zero the remaining least significant bits.

Fig. 1 shows an example of an 8-bit Mitchell multiplier with inputs 45 and 147. First, the values of k_1, k_2, m_1 and m_2 are obtained. The approximate logarithms for the inputs are then k_1 & m_2 and k_2 & m_2. Adding these logarithms produces $k\&m$ as shown in (5). The approximated product is then '1' at bit position 12, followed by m = "1000111" and then five zeros.

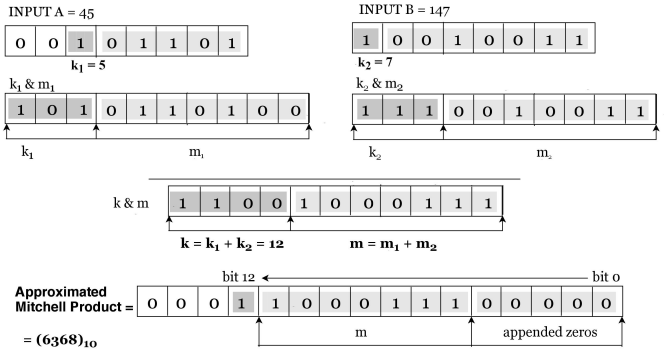

Fig. 1: Example of Mitchell Approximate Multiplication

B. Conventional 4-bit Leading One Detector Designs

Wider LODs, such as 16-bit and 32-bit LODs, can be composed from 4-bit LOD slices [3]. The 32-bit LOD in the proposed design is implemented using eight copies of the 4-bit LOD slice shown in Fig. 2. The 4-bit LOD produces a 4-bit binary output with one bit set to '1' at the position of the leading one and the remaining bits cleared to '0'.

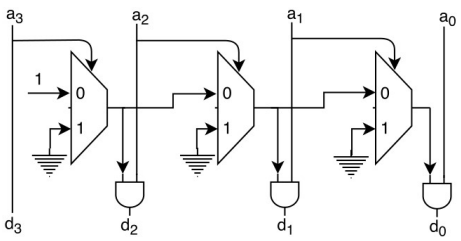

Fig. 2: 4-bit Leading one detector [3]

III. Proposed Design

The two significant components of an approximate logarithmic multiplier are the LOD and the adder that sums the logarithms of the inputs. In this section we propose new approximation techniques for both of these circuits.

The exact 32-bit LOD is implemented using 4-bit LOD circuits, as shown in Fig. 3. The 32-bit input is partitioned into 8 fields of 4-bits each that are evaluated by the first stage of eight 4-bit LODs operating in parallel. The two 4-bit LODs in the second stage determine the leading one in the two 16-bit fields d_{31} to d_{16} and d_{15} to d_0. The 2-bit LOD in the third stage determines whether the leading one is in d_{31} to d_{16}, or in d_{15} to d_0. The output of this 2-bit LOD ("00","01" or "10") provides a select bit for the two groups of 4-bit multiplexers. The output of the 4-bit LOD in the second stage is passed through if the select line is '1' or else "0000" will be the output of multiplexer. The combined 8-bit output from the multiplexers forms the select inputs for the eight groups of 4-bit multiplexers in the last stage. Only one bit from the 8-bit select word will be '1' and the multiplexer will pass the 4-bit decoded word obtained from the first-stage 4-bit LOD. The final 32-bit output has only one bit set to '1' (at the leading 1) with the remaining bits cleared to '0'.

A. Approximate LOD Design I

This first design approximates the 16 least significant bits with a single fixed bias. If the input to the LOD is greater than the 2^{16}, there will be no error in the approximated output product. Otherwise, inputs of less than 2^{16} will lead to small signed output errors. Simulation was performed to determine the maximum number of bits that can be approximated in the LOD without causing any loss of accuracy in the final Mitchell product. Based on the accuracy evaluation results presented in the next section, we found that the four least significant 4-bit LODs in Fig. 3 can be safely approximated with a constant hexadecimal bias of x"0400", as shown in Fig. 4. The value x"0400" = 1024, is half of 2^{11} = 2048. With this bias sometimes the obtained approximated product will be higher and sometimes lower than the Mitchell approximated product. Since the four least significant 4-bit LODs in the first stage have been approximated, we can approximate the least significant 4-bit LOD in the second stage shown as "LOD A" in Fig. 3.

TABLE I: Bias Selection for Approximate LOD Design II

OR4	OR3	OR2	OR1	Bias for LOD approximation
1	X	X	X	x"4000"
0	1	X	X	x"0400"
0	0	1	X	x"0040"
0	0	0	1	x"0004"

B. Approximated LOD Design II

In Design I, fixing the 16 LSBs to a constant value for some input pairs can cause a relatively large error. The technique

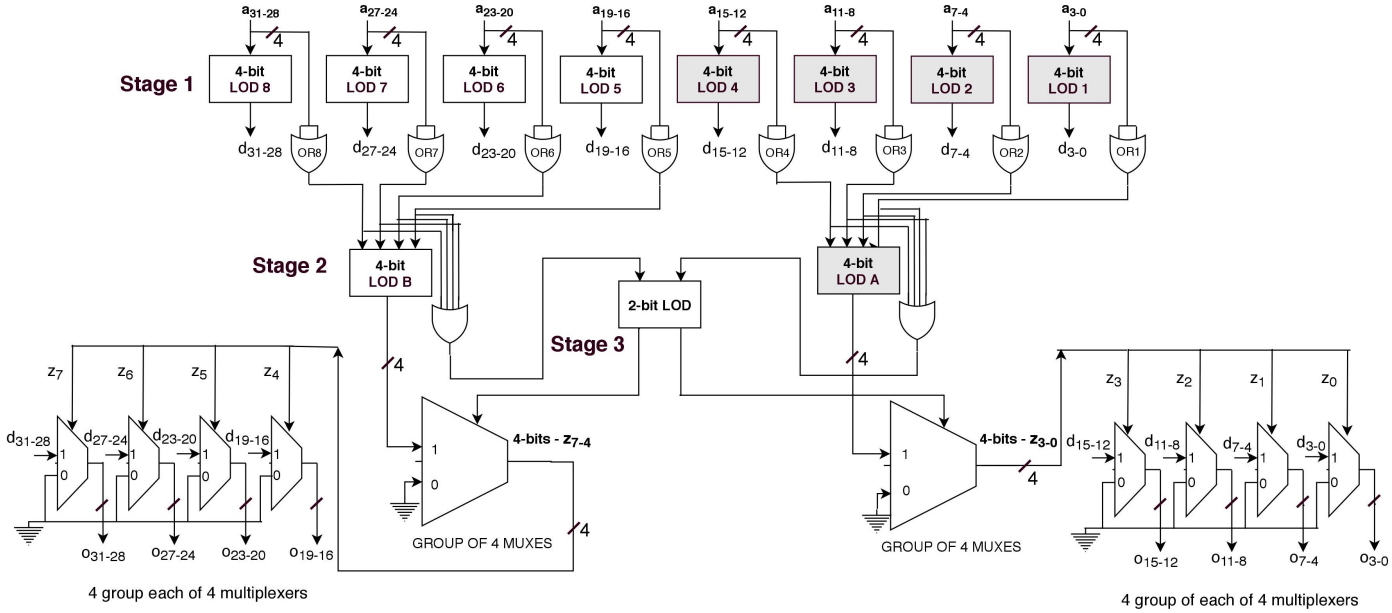

Fig. 3: 32-bit Leading one detector [3]

proposed in this subsection is not as hardware-efficient as Design I but it is more accurate. Design II approximates the four least significant LODs to one of the four constant biases, x"4000", x"0400", x"0040" or x"0004" based on the input bits to these four LODs. The biases were chosen based on the design which is a priority encoder implemented using the *OR* gates, as shown in Fig. 5. Table 1 shows how the bias for the LOD approximation is selected using a multiplexer.

The adder that sums the logarithms also forms a significant part of the Mitchell multiplier and it can be approximated to further reduce the hardware cost. The approximation is done by biasing the optimum number of bits with alternating ones and zeros. Using computer simulation we determined that the 16 least significant bits can be approximated without any loss of accuracy in the product. By choosing an alternating sequence of ones and zeros, the Mitchell multiplier will sometime overestimate and sometime underestimate the true product resulting in roughly symmetrical error distribution [7].

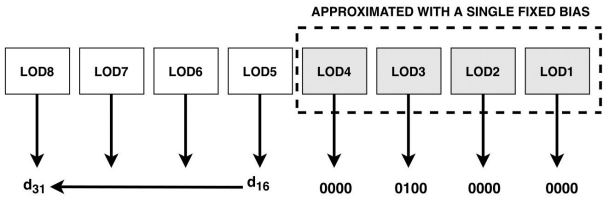

Fig. 4: Approximation of the LOD using a single fixed bias

IV. Simulation Results

This section provides simulation results for four 32-bit logarithmic multiplier designs. The accuracy evaluations were obtained from *MATLAB* models of the approximate logarithmic multipliers. We evaluated the accuracy using the Mean

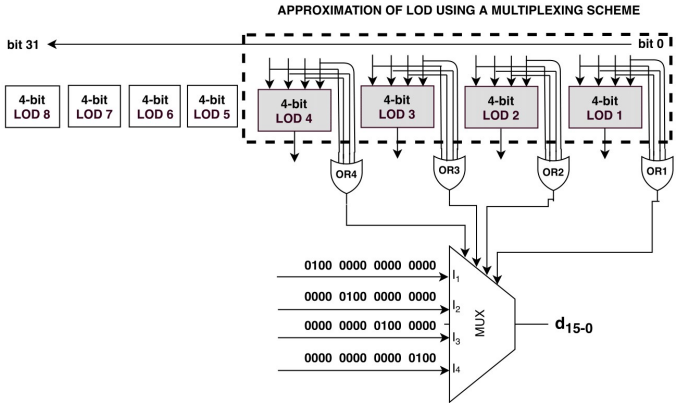

Fig. 5: Approximation of LOD using multiplexing scheme

Relative Error distance (MRED) for 32-bit multiplier designs with 10^4, 10^5, 10^6 and 10^7 randomly chosen input positive integer pairs. The MRED metric gives the average relative error distance between the approximate product and the exact product. From the simulation results, it was found that the value of MRED = 0.0385 does not change after 10^6 iterations for all four designs. With this MRED value, the four least significant LODs can be approximated in the first stage as shown in Fig. 3. For the adder, the 16 least significant bits can be approximated with same MRED for both Designs I and II.

Fig. 6 plots the MRED for 16-bit versions of the two proposed multipliers that were derived from the 32-bit designs. Note how the MRED varies with parameters t_1 and t_2, where t_1 is the number of least significant bits in LODs to be approximated and t_2 is the number of least significant bits approximated for the adder. As t_1 and t_2 increase, the MRED increases. The optimum value of these parameters is chosen

to match the MRED accuracy of the Mitchell multiplier.

TABLE II: Hardware Utilization and Performance Parameters

Prototypes	LUTs	Area $(nm)^2$	Delay (ns)	PDP (fJ)
Conventional Mitchell [2]	467	2097.93	3.93	2925.10
ALM-SOA [1]	441	1540.28	2.93	1332.85
Proposed Design I	**374**	**1419.51**	2.77	**1264.50**
Proposed Design II	**383**	**1485.77**	2.78	**1281.58**

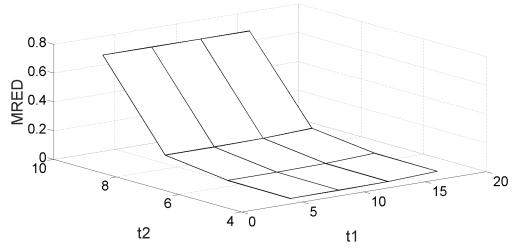

Fig. 6: Surface plot for 16-bit Multiplier designs

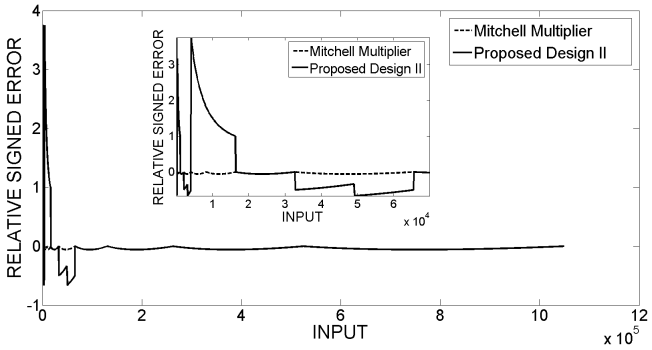

Fig. 7: Relative Signed Error in Logarithmic Approximations

Figs. 7 and 8 plot the relative signed error, [(N-2^{logN})/N], and absolute error, [|N-2^{logN}|], for the Mitchell, Design I and Design II approximate logarithm functions. As shown in the insets, Designs I and II have larger approximation errors than Mitchell's design for inputs up to 2^{16}, but the errors are identical for all three designs for inputs greater than 2^{16}. In Fig. 7, for smaller numbers the errors are double-sided and there is a chance that they will cancel each other out in the application calculation. One possible solution is to, when necessary, scale up the numbers and match the exact Mitchell error. As shown in Fig. 8, for numbers less than 2^{16}, the absolute error obtained due to our new approximation is less than the worst-case error for all the designs.

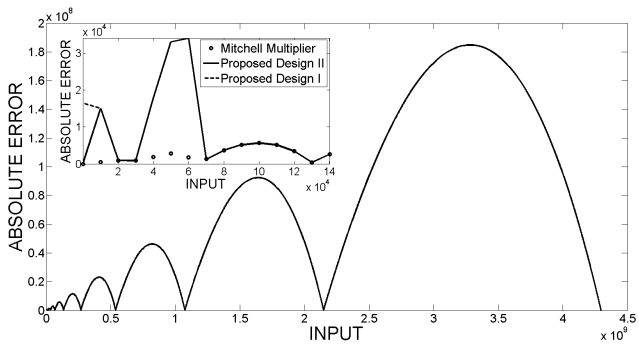

Fig. 8: Absolute Error in Logarithmic Approximations

Approximation of four least significant LODs in the first stage and one least significant LOD in the second stage together with 16 LSBs approximated for the adder yields the Look Up Table (LUT) utilizations shown in Table II. ALM-SOA requires 5.56% less hardware compared to the conventional Mitchell design. Our Design I and II show reductions in resources by 19.91% and 17.98%, respectively, with respect to the conventional Mitchell design. The hardware utilization for all four designs was obtained when they were implemented for a Xilinx xc7z010clg400-1 FPGA using Vivado. The results for Area, Delay and Power Delay Product (PDP) were obtained using Synopsys Design Compiler (DC) for ST Micro's CMOS 28-nm process. Table II shows that the Proposed Designs I and II are smaller by 32.33% and 29.17% respectively, with respect to conventional Mitchell design. Proposed Designs I and II are 56.77% and 56.18% more energy-efficient than the conventional Mitchell design.

V. Conclusion

Two designs that closely approximate the logarithmic Mitchell multiplier have been proposed. Both designs approximate the LOD and the adder. Design I approximates the four least significant 4-bit LODs with a single fixed bias. Design II approximates the four least significant 4-bit LODs with a bias chosen using a priority encoder controlled multiplexer. The approximation technique adopted for the adder is same for both designs, where the 16 LSBs are approximated with alternating ones and zeros. Designs I and II were compared to a recent design [1] in the literature and our designs have smaller resource utilizations by 15.19% and 13.15%, respectively. Compared with the original Mitchell multiplier, the new designs achieve hardware reductions of 19.91% and 17.98%, respectively. In terms of area and PDP, our designs are smaller and more energy-efficient than the conventional Mitchell design and ALM-SOA. For inputs up to 2^{16}, Designs I and II do indeed have larger errors than Mitchell's design but for many random pairs drawn over the full 2^{32} input space, the mean relative errors for all four multipliers are the same.

References

[1] Weiqiang Liu , Jiahua Xu, Danye Wang, Chenghua Wang, Paolo Montuschi and Fabrizio Lombardi, "Design and Evaluation of Approximate Logarithmic Multipliers for Low Power Error-Tolerant Applications", *IEEE Transactions on Circuits and SystemsI: Regular Papers*, Vol. 65, No. 9, pp. 2856-2868, September 2018.

[2] J. N. Mitchell, "Computer multiplication and division using binary logarithms", *IRE Trans. Electronic Computers*, pp. 512-517, Aug. 1962.

[3] Khalid H. Abed, Raymond E. Siferd, "CMOS VLSI Implementation of Low-Power Logarithmic Converter", *IEEE Transactions on Computers*, Vol. 52, No. 11, pp. 1421-1433, November 2003.

[4] K Kunaraj, Dr. R. Seshasayanan, "Leading one detectors and leading one position detectors - an evolutionary design methodology," *Can. J. Elect. Computer Eng.*, vol. 36, No. 3, pp. 103-110, Summer 2013.

[5] V. Mahalingam and N. Ranganathan, "An Efficient and Accurate Logarithmic Multiplier based on Operand Decomposition", *IEEE Proceedings of the 19th IEEE International Conference on VLSI Design*, 6 pp., 2006.

[6] Dr. Jie Han, Michael Orshansky, "Approximate Computing: An Emerging Paradigm For Energy-Efficient Design", 1*8th IEEE European Test Symposium (ETS)*, pp. 1-6, 2013.

[7] Mohammad Saeed Ansari, Bruce F. Cockburn, Jie Han, "A Hardware-Efficient Logarithmic Multiplier with Improved Precision", to appear in *Design, Automation and Test in Europe*, 8 pp., 2019.