VLSI Implementations of Low-Power Leading-One Detector Circuits

Khalid H. Abed

Department of Computer Engineering
Jackson State University
Jackson, MS. 39217 U.S.A.
khalid.abed@jsums.edu

Raymond E. Siferd

Department of Electrical Engineering
Wright State University
Dayton, Ohio 45435 U.S.A.
rsiferd@cs.wright.edu

ABSTRACT

This paper presents two approaches to design leading-one detector (LOD) circuits, which locate the leading-one position in a binary word. The first approach provides fast leading-one detectors (LODs), and the second emphasizes the design of novel hardware-efficient and low-power LODs. Each approach is used to obtain 0.6 µm CMOS VLSI implementations of 16-, 32-, and 64-bit LOD circuits. Simulations of the 16-, 32-, and 64-bit fast LODs run at 310, 265, and 215 MHz, respectively. The 16-, 32-, and 64-bit low-power LODs consume 14.85, 44.70, and 61.67 milliwatts, respectively while operating at V_{DD} equal to 5 volts and their maximum speed.

I. INTRODUCTION

Leading-one detector (LOD) circuits are used in areas, such as floating-point (FLP) arithmetic and binary logarithms. Speed and high performance are usually considered as primary design goals of FLP processors. The use of a fast LOD sub-circuit in a FLP normalization circuit is crucial to achieve such design objectives. On the other hand, binary logarithms have been used in recent years to implement hardware-efficient and low-power VLSI circuits. The LOD is used in logarithmic converters to find the leading-one position, which is used to determine the integer portion and the fractional part of the binary logarithm. Hardware-efficient and low-power leading-one detectors (LODs) are necessary to implement low-power logarithmic converters.

Schmookler *et al.* [1] reviewed some leading one/zero anticipators and predictors, which are used for FLP normalization. Schmookler provided a brief description of leading-zero detectors (LZDs) described by Oklobdzija in [2], [3], and [4]. Oklobdzija's LZDs generate a binary word, which is equal to the leading-one position but does not generate a decoded binary word. Anticipators are presented by Schmookler *et al.* [5], and by Suzuki *et al.* [6], and leading-one predictors are described by Bruguera *et al.* in [7] and [8]. Anticipators and predictors are not suitable for binary logarithms applications because they are hardware intensive, but they use a leading one/zero sub-circuit.

LODs used in binary logarithms should be simple unlike the complex LODs used in FLP arithmetic. To generate binary logarithms, Mitchell [9], Combet *et al.* [10], Frangakis [11], and Hoefflinger [12] used shifting and counting to find the leading-one position. This method is very slow and requires a number of clock cycles that depends on the leading-one position. SanGregory *et al.* [13] used a bit-by-bit serial evaluating circuits to compute the binary logarithm. This LOD evaluates an *N*-bit word serially starting from the most-significant bit (MSB) to least-significant bit (LSB). SanGregory used a look-ahead technique to locate the leading-one for large values of *N* in a reasonable time. However, this technique fails to help for small values of *N*, such as *N* equal to 1 because it activates the critical path in these serial circuits, and this limits the maximum operating frequency.

Existing LOD techniques are either very slow or hardware intensive and dissipate a lot of power. Neither shifting-and-counting method nor bit-by-bit serial evaluation circuits can be used to implement fast, hardware-efficient, and low-power LODs. At this time, it seems that there is no effective technique to handle the problem of locating a leading-one with fast, hardware-efficient, and low-power LOD circuits. This creates the need to explore new approaches, such as parallel evaluating circuits rather than serial or a combination of both.

In this paper, we first design the proposed 16-, 32-, and 64-bit fast LOD circuits, and provide the speed, area, and power results for these three LODs. Then, we propose a novel approach to design fast, hardware-efficient and low-power LODs. We use this approach to implement 16-, 32-, and 64-bit LOD circuits, and provide their resulting maximum operating frequency, size, and power dissipation. The structure of the paper is as follows. In section II, we implement fast LODs and in section III, we propose and implement novel LOD circuits. In section VI, we provide some conclusions.

0-4244-0169-0/06/$20.00 © 2006 IEEE.

II. IMPLEMENTATIONS OF FAST LODs

The approach we use to design LOD circuits is based on dividing the input binary word into groups, which evaluate in parallel and independently of each other, but evaluation within each group is performed serially. These groups may not be equal in size, but the division should be made to maximize the speed of the circuit and to simplify the full custom implementations by using similar LOD blocks. The division by groups in this approach eliminates the dependency among all input bits unlike other schemes. Input bits dependency is reduced from the total number of input bits to almost twice the maximum number of input bits in a single group.

A. A 4-bit LOD circuit

The 4-bit LOD circuit, shown in Figure 1, is a serial circuit that evaluates from the MSB d_3 to the LSB d_0. We select a 4-bit LOD because it can be easily replicated, and it provides less overall gate delay compared to other *N*-bit LODs. The circuit generates a 4-bit active-high decoded binary word, and the high bit corresponds to the leading-one. The worst-case delay of the 4-bit LOD circuit corresponds to the 4-bit input binary word "0001".

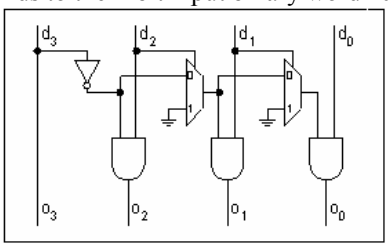

Figure 1. A 4-bit LOD circuit.

B. A 16-bit LOD circuit

The 16-bit LOD circuit, shown in Figure 2, is a serial/parallel circuit, which is designed based on dividing the 16 bits into four groups of four bits. Each group corresponds to a 4-bit LOD circuit; therefore, the first stage consists of four 4-bit LODs, which generate four groups of 4-bit active-high decoded binary words. We refer to the active-high decoded binary group that has the leading-one as the leading-group. The second stage 4-bit LOD produces a 4-bit control word, which has one high bit that identifies the leading-group. This word controls four select lines to four groups of 4-bit MUXes. The high bit of the control word is used to permit four MUXes to pass the leading-group. The remaining three low bits of the control word force the remaining twelve MUXes to pass zeros to the output. The circuit generates a 16-bit active-high decoded binary word that has the leading-one. The worst-case delay of this circuit corresponds to the 16-bit input binary word that is expressed in hexadecimal as "0001".

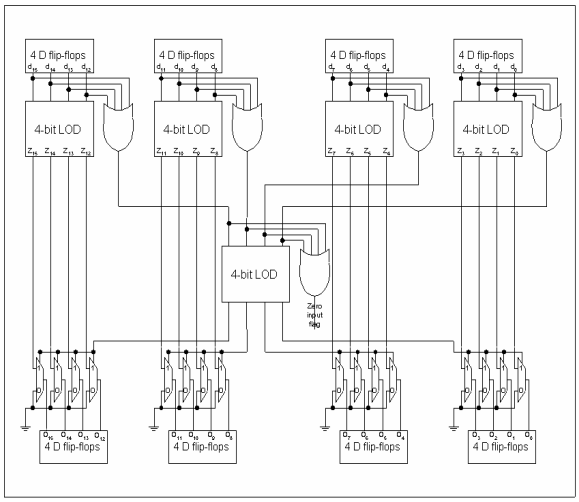

Figure 2. A 16-bit LOD circuit.

C. A 32-bit LOD circuit

The 32-bit LOD circuit is designed based on dividing the 32 bits into 8 groups of 4 bits. The first stage has eight 4-bit LODs, and the second stage has two 4-bit LODs to determine the leading-group of the bits d_0 to d_{15} and the leading-group of d_{16} to d_{31}. Then, the third stage 2-bit LOD generates a 2-bit control word, which determines whether the leading-one is in the bits d_0 to d_{15} or in the bits d_{16} to d_{31}. The 2-bit word controls two groups of 4-bit MUXes where each MUXes group passes the second stage 4-bit LOD output if its control bit is a "1", and passes "0000" if its control bit is a "0". The output, generated by each of the second stage 4-bit LODs and passed through the 4-bit MUXes, controls four select lines to four groups of 4-bit MUXes. Only one select line is high to allow 4-bit MUXes to pass the 4-bit leading-group. The circuit generates a 32-bit active-high decoded word that has the leading-one. The worst-case delay of the circuit corresponds to the 32-bit hexadecimal word "00000001".

In general, we designed the 32-bit LOD using two 16-bit LODs with a 2-bit LOD circuit, which generates a 2-bit control word that determines whether the leading-one is located in the first 16 bits or the second 16 bits. This word also controls two groups of 4-bit MUXes where each MUXes group passes the second stage 4-bit LOD output if its control bit is a "1", and passes "0000" if its control bit is a "0". The output of each 4-bit MUXes controls the select lines of four 4-bit output MUXes in each of the two 16-bit LODs.

D. A 64-bit LOD circuit

In general, the 64-bit LOD is similar to the 32-bit LOD circuit. The 64-bit LOD, shown in Figure 3, is designed using four 16-bit LODs, and the third stage is designed by a 4-bit LOD instead of the 2-bit LOD used in the 32-bit LOD. The third stage LOD generates a 4-bit control word that determines which of the four 16-bit groups has the leading-one. This 4-bit word also controls four groups of 4-bit MUXes where each MUXes group passes the second stage 4-bit LOD output if its control bit is a "1", and passes "0000" if its control bit is a "0". The output of each of the four groups of 4-bit MUXes controls the select lines of four 4-bit output MUXes in each of the four 16-bit LODs. The high bit of the third stage LOD identifies the 16 bits that has the leading one, and the high bit of the second stage LOD of these 16 bits identifies the 4-bit leading-group.

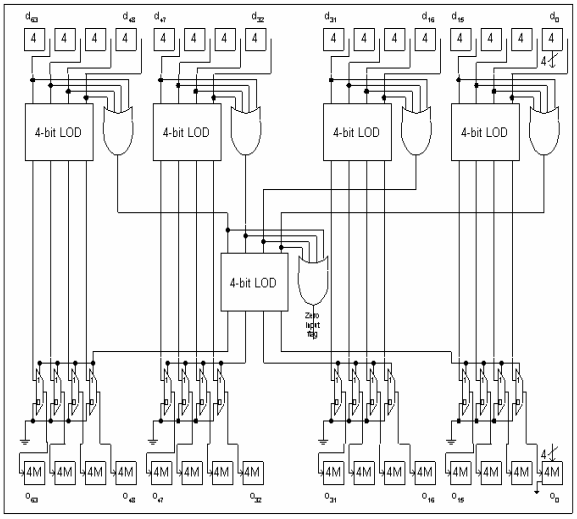

Figure 3. A 64-bit LOD circuit.

E. Simulations and implementations results

The 16-, 32-, and 64-bit LOD circuits are implemented using 0.6 µm CMOS technology. The layouts of these LODs are shown together in Figure 4. Even though these LODs were designed with input and output registers as shown in Figure 2, the results given in Table 1 represent these circuits without registers. The 16-bit LOD requires 305λ x 1250λ of chip area, and the 32 and 64-bit LODs require two and four times of chip area, respectively compared to the 16-bit LOD. The LODs operate at V_{DD} equal to 5 volts and generate the leading-one position in a single clock cycle. Simulations based on extracted netlists from each separate layout, shown in Figure 4, indicate the 16-, 32-, and 64-bit LOD circuits run at a maximum clock frequency of 310, 265, and 215 MHz, respectively. While operating at V_{DD} equal to 5 volts and the maximum frequency, the 16-, 32-, and 64-bit LODs dissipate 35.61, 63.34, and 103.15 milliwatts, respectively.

Figure 4. Layouts of the 16-, 32-, and 64-bit LODs.

LOD circuit	Layout size in λ	Maximum speed	Power consumed
16-bit	305 x 1250	310 MHz	35.61 mw
32-bit	305 x 2500	265 MHz	63.34 mw
64-bit	305 x 5000	215 MHz	103.15 mw

Table 1. Results for the fast LODs.

This approach solved the problem of finding the leading-one with speed, but it requires a lot of hardware. These LODs are suitable for the use in high-performance circuits, such as these used to design FLP normalization circuits.

III. LOW-POWER LOD CIRCUITS

In the previous approach, the first stage consists of many 4-bit LODs, where each 4-bit LOD is used to evaluate each 4-bit of the input data word. Therefore, the first stage requires *N*/4 4-bit LODs, and this large number of LODs improves the speed, but it increases the hardware used and the power consumed. This problem must be eliminated to implement low-power and hardware-efficient LODs. In this approach, we use a single 4-bit LOD circuit to perform the operation of the first stage, instead of *N*/4 4-bit LODs. We reduce the evaluation to a 4-bit only rather than the evaluation of all bits in the input data word; as a result, we design and implement the following LODs with much less hardware requirements and power consumptions.

A. Implementation of a 16-bit LOD circuit

The serial 4-bit LOD circuit, shown in Figure 1, is used to design other *N*-bit LOD circuits. The 16-bit LOD circuit, shown in Figure 5 (a), is designed based on dividing the 16 bits into 4 groups of 4 bits. Four 4-bit OR gates are used to determine whether the group has at least a "1" or not. The 4-bit output

of the OR gates is used by a 4-bit LOD, which produces a 4-bit control word that determines the leading-group. This control word is used as the control signals of the 4x4 transmission-gate array shown in Figure 5 (b). The purpose of the transmission-gate array is to select the 4-bit group of the input data that has the leading-one. The high bit of the 4-bit control word activates the column, which corresponds and passes the 4-bit input data that will produce the leading-group. For example, the signal s_0 controls the first column, which passes the four LSBs if s_0 is "1". Then, the 4-bit output of the switches is evaluated by another 4-bit LOD, which generates the leading-group. The 4x4 MUXes array, shown in Figure 5 (c), is controlled by the 4-bit control word. Each control bit controls a 4-bit MUXes row. If the control bit is "1", the four MUXes row passes the corresponding leading-group, but each of the other three rows passes "0000" to the output because their control bits are all zeros. The 16-bit LOD circuit generates a 16-bit active-high decoded binary word that has the leading-one.

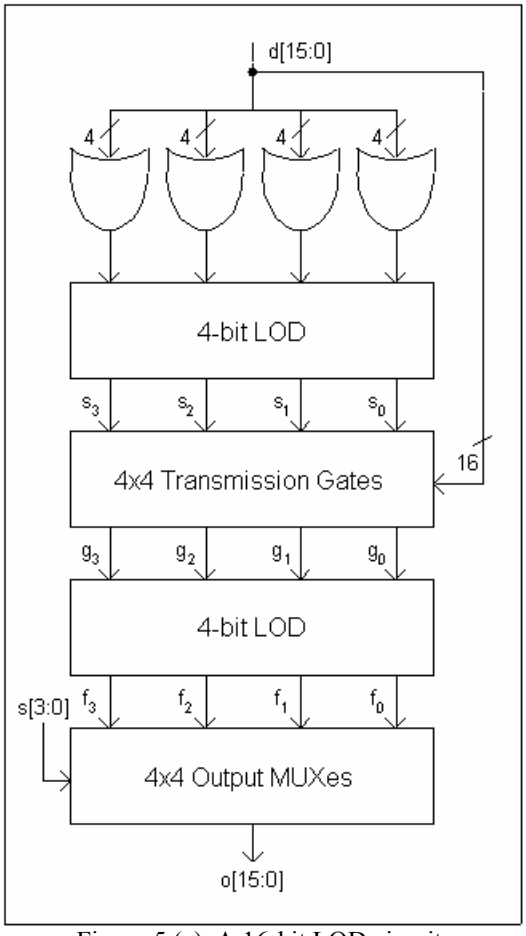

Figure 5 (a). A 16-bit LOD circuit.

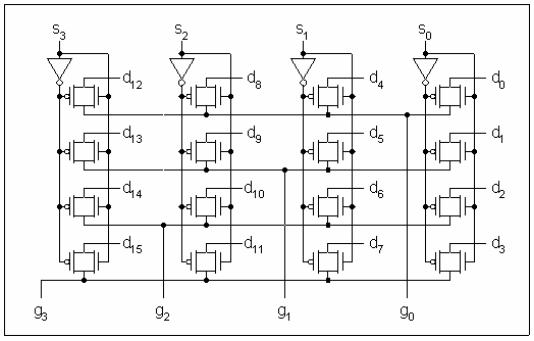

Figure 5 (b). A 4x4 transmission-gate array circuit.

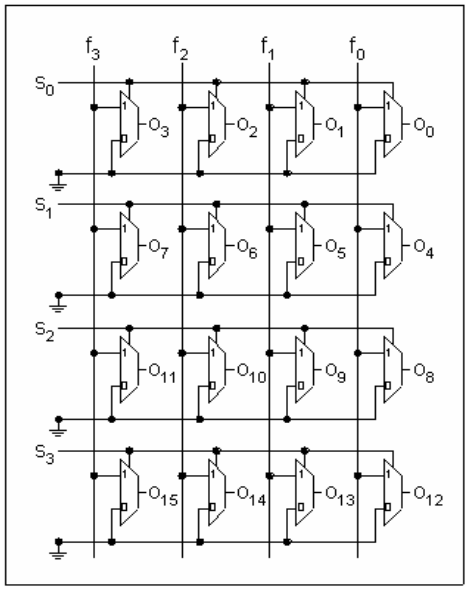

Figure 5 (c). A 4x4 MUXes array circuit.

B. Implementation of a 32-bit LOD circuit

The 32-bit LOD circuit is designed based on the use of the 16-bit LOD circuit shown in Figure 5 (a). As a result, we divide the 32 bits into 2 groups of 16 bits. Five 4-bit OR gates are used to determine if the group of bits d_0 to d_{15} or the group of d_{16} to d_{31} has at least a "1". A 2-bit LOD produces a 2-bit control word, which determines whether the leading-one is in the bits d_0 to d_{15} or in the bits d_{16} to d_{31}. The control word is used as the control signals of a 16x2 transmission-gate array. The high bit of the 2-bit control word activates the column, which corresponds and passes the 16-bit input data that has the leading-one. Then, the 16-bit output of the switches is evaluated by a 16-bit LOD, which generates the 16-bit leading-group. The 2-bit word controls a 2x16 MUXes array, and each bit controls a 16-bit MUXes row. If a control bit is "1", the sixteen MUXes row passes the corresponding 16-bit leading-group, but the other row passes sixteen zeros to the output because its control bit is "0".

C. Implementation of a 64-bit LOD circuit

The 4- and 16-bit LOD circuits, shown in Figure 1 and Figure 5 (a), are used to design the 64-bit LOD circuit. The 64-bit LOD circuit, shown in Figure 6, is designed based on dividing the 64 bits into 4 groups of 16 bits. Five 4-bit OR gates are used to determine whether the group has at least a "1" or not. The 4-bit output of the OR gates is used by a 4-bit LOD, which produces a 4-bit control word that determines the leading-group. This control word is used as the control signals of a 16x4 transmission-gate array. The purpose of the transmission-gate array is to select the 16-bit group of the input data that has the leading-one. The high bit of the 4-bit control word activates the column, which corresponds and passes the 16-bit input data that has leading-one. Then, the 16-bit output of the switches is evaluated by a 16-bit LOD, which generates the leading-group. The 4x16 MUXes array is controlled by the 4-bit control word. Each control bit controls a 16-bit MUXes row. If the control bit is "1", the sixteen MUXes row passes the corresponding leading-group, but each of the other three rows passes sixteen zeros to the output because their control bits are all zeros. The 64-bit LOD generates a 64-bit decoded word that has the leading-one.

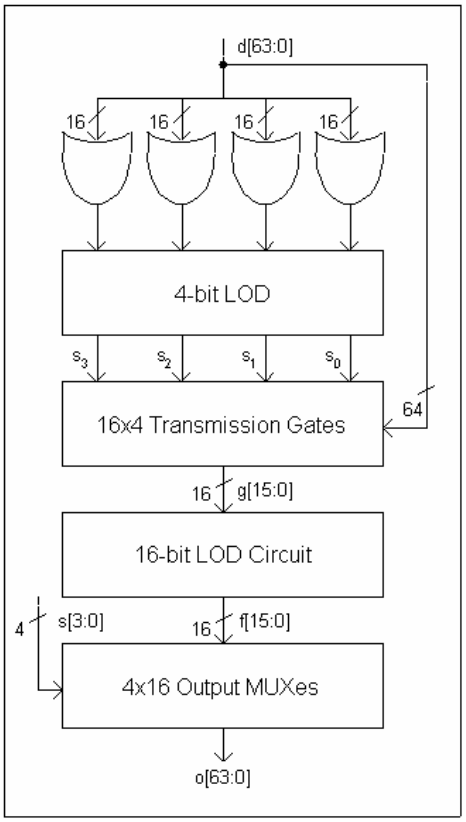

Figure 6. A 64-bit LOD circuit.

D. Simulations and implementations results

The 16-, 32-, and 64-bit LOD circuits are implemented using 0.6 µm CMOS technology. The layouts of these LODs are shown together in Figure 7. These LODs were designed without input and output registers as shown in Figure 5 (a) and Figure 6. The 16-bit LOD requires 305λ x 1250λ of chip area, and the 32 and 64-bit LODs require two and four times as much, respectively, of chip area compared to the 16-bit LOD. The LODs generate the leading-one position in a single clock cycle. Simulations based on extracted netlists from each separate layout, shown in Figure 7, indicate the 16-, 32-, and 64-bit LOD circuits run at a maximum clock frequency of 300, 200, and 180 MHz, respectively. While operating at V_{DD} equal to 5 volts and the maximum frequency, the 16-, 32-, and 64-bit LOD circuits dissipate 14.85, 44.70, and 61.67 milliwatts, respectively. The LODs designed using this approach more hardware-efficient and dissipate less power compared to these LODs designed using the first approach.

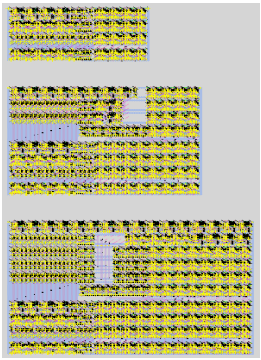

Figure 7. The 16-, 32-, and 64-bit LODs.

LOD circuit	Layout size in λ	Maximum speed	Power consumed
16-bit	340 x 543	300 MHz	14.85 mw
32-bit	686 x 745	200 MHz	44.70 mw
64-bit	846 x 945	180 MHz	61.67 mw

Table 2. Results for the low-power LODs.

Hardware-efficiency and low-power dissipation are usually the design objectives of LOD circuits, which are required for binary logarithms applications. The LODs designed using this approach have some advantages, such as hardware-efficiency, low-power dissipation, and good speed. This approach is suitable for low-power circuits, such as these used to design logarithmic converters.

IV. CONCLUSIONS

CMOS VLSI implementations of two types of leading-one detector circuits have been presented. The approaches have been used to design fast, hardware-efficient, and low-power LODs. The 16-, 32-, and 64-bit LODs, designed using the first approach, have solved the problem of locating the leading-one position in binary words while maintaining high speed. The 16-, 32-, and 64-bit LODs of the second approach have less hardware and dissipate less power than these LODs designed with the first method. If the binary word is required instead of the decoded binary word, a small size of N x $\log_2$ (N) ROM can be used with an LOD to generate the binary word that corresponds to the leading-one position. The resulting LOD/ROM circuit will also perform well in terms of speed and power consumed.

REFERENCES

[1] M. Schmookler and K. Nowka, "Leading Zero Anticipator and Detection: A Comparison of Methods," Proceedings of the 15th Symposium on Computer Arithmetic, June 2001.

[2] V. Oklobdzija, "An Implementation Algorithm and Design of a Novel Leading Zero Detector Circuit," 26th IEEE Asilomar Conference on Signals, Systems, and Computers, pp. 391-395, 1992.

[3] V. Oklobdzija, "An Algorithmic and Novel Design of a Leading Zero Detector Circuit: Comparison with Logic Synthesis," IEEE Transactions on VLSI Systems, pp. 124-128 1993.

[4] V. Oklobdzija, "Comments on Leading-Zero Anticipatory Logic for High-Speed Floating Point Addition," IEEE Journal of Solid-State Circuits, pp. 292-293, 1997.

[5] M. Schmookler and D. Mikan, "Two-State Leading Zero/One Anticipator (LZA)," US Patent#5493520, Feb. 1996.

[6] H. Suzuki, H. Morinaka, H. Makino, Y. Nakase, K. Mashiko, and T. Sumi, "Leading-Zero Anticipatory Logic for High-Speed Floating Point Addition," IEEE Journal of Solid-State Circuits, pp. 1157-1164, 1996.

[7] J. Bruguera and T. Lang, "Leading-One Prediction Scheme for Latency Improvement in Single Datapath Floating-Point Adders," Proceedings International Conference on Computer Design, pp. 298-305, October 1998.

[8] J. Bruguera and T. Lang, "Leading-One Prediction with Concurrent Position Correction," IEEE Transactions on Computers, pp. 298-305, October 1999.

[9] John N. Mitchell, Jr., "Computer multiplication and division using binary logarithms," IRE Transactions on Electronic Computers, pp. 512-517, August 1962.

[10] M. Combet, H. Zonneveld, and L. Verbeek, "Computation of the base two logarithm of binary numbers," IEEE Transactions on Electronic Computers, pp. 863-867, Dec. 1965.

[11] Frangakis, G. P., "A new binary logarithm based computing system," Proceedings IEE, pp. 169-173, September 1983.

[12] B. Hoefflinger, "Efficient VLSI digital logarithmic CODECS," Electronics Letters, pp. 1132-1134, June 1991.

[13] Samuel L. SanGregory, Raymond E. Siferd, Charles Brother, and David Gallagher, "A fast, low-power logarithm approximation with CMOS VLSI implementation," IEEE Midwest Symposium on Circuits and Systems, August 1999.