# A generalized architecture for efficient multi-segment logarithm approximation

Dimitris Bariamis, Dimitris Maroulis, Dimitris K. Iakovidis

Department of Informatics and Telecommunications
University of Athens
rtsimage@di.uoa.gr

## Abstract

*This paper presents a novel hardware architecture for the approximation of the base-2 logarithm of integers at a high accuracy and with low resource requirements. The generalized piecewise linear approximation approach allows the utilization of a very large number of linear segments for the approximation of the logarithm function, whereas state of the art architectures are limited to a maximum of six segments. In the proposed implementation, up to 1024 linear segments are employed. The exploitation of the capabilities of modern FPGAs, such as large BlockRAMs and fast multipliers allow a high frequency potential combined with low resource requirements, resulting in a pipelined implementation that only requires up to 329 slices, one BlockRAM and one multiplier. The selected number of segments affects the approximation error, which is lower than the comparable architectures by up to five orders of magnitude. The properties of the proposed architecture render it suitable for use as a generic component in systems that require fast, accurate and resource-efficient logarithm calculation.*

## 1. Introduction

The logarithm function is used in a multitude of applications in the signal and image processing, telecommunications, biomedical and industrial domains. Several of these applications require a large number of logarithm calculations per second in order to achieve their performance goals. Moreover, the logarithm calculations have to be performed accurately. For example, the pre-processing of microarray data entails the logarithmic normalization of thousands of data samples for each microarray image. Inaccurate computations of the logarithm in such an application can lead to erroneous medical decisions. Another example is any real-time pattern recognition system such as [2], where inaccurate computations of features based on the logarithm can lead to a deteriorated recognition performance.

Hardware architectures that have been proposed for the approximation of the base-2 logarithm include implementations of several families of algorithms, including power series [3] and polynomial methods [4], high-radix algorithms [5], the CORDIC (Coordinate Rotation Digital Computer) algorithm [6] and piecewise linear approximation methods [7-10]. The iterative CORDIC algorithm has commonly been used for logarithm approximation; however implementing it as a pipelined circuit in an area-efficient way on FPGA remains a challenge. A vast number of implementations based on table lookup and polynomial approximation are examined in [11], but possible optimizations based on the available FPGA resources are not taken into account. Alternatively, many hardware architectures aiming at crude, however fast approximation of the logarithm, implement the piecewise linear approximation approach based on Mitchell's method [7]. According to this approach the logarithm function is approximated by a small number of consecutive linear segments. In [2] and [8] this method was implemented by using only two segments; in [9] four segments were used; whereas in [10] a VLSI architecture that uses two, three and six segments was proposed.

Inspired by Mitchell's method, and motivated by the need of combining both computational efficiency and approximation accuracy, we propose a novel architecture for the approximation of the base-2 logarithm in a fast and area-efficient way, exploiting the capabilities of modern FPGAs. The proposed architecture implements a generalized piecewise linear approximation of the logarithm function that allows for a large number of linear approximation segments, providing up to five orders of magnitude lower approximation error than the other piecewise methods. The achieved approximation accuracy depends on the number of segments used, which affects the size of a ROM that is used for storing the parameters that control the computation. The implementation of the ROM using a single FPGA BlockRAM minimizes both the slice and the BlockRAM requirements, resulting in

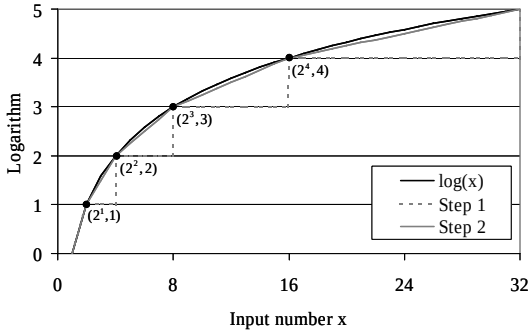


**Figure 1. Steps 1 and 2 of logarithm approximation**

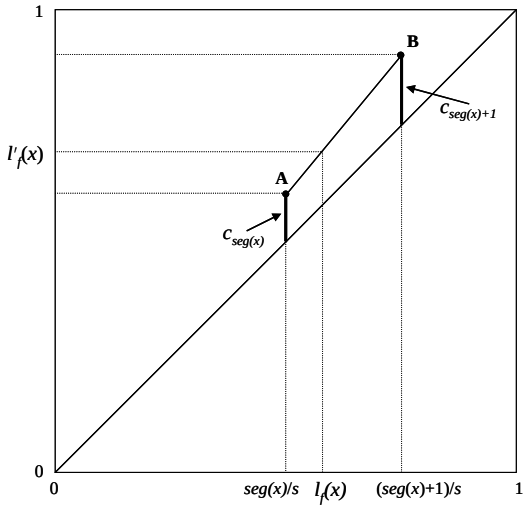


**Figure 2. $l_f^{'}$ as a function of $l_f$**

a small circuit that occupies less than 0.5% of the total FPGA area.

The rest of this paper is organized in three sections. Section 2 describes the logarithm approximation approach and the FPGA implementation of the proposed architecture. The results of the experiments conducted are presented in Section 3. The conclusions of this study are summarized in Section 4.

## 2. Logarithm Estimation via Piecewise Linear Approximation

A piecewise linear approximation approach that involves three steps is considered. It involves the calculation of the integral part of the logarithm, the linear approximation of the fractional part of the logarithm by one segment and the piecewise linear approximation of the fractional part by several segments. The piecewise linear approximation of the third step is performed by splitting the linear approximation produced in the second step into *s* consecutive linear segments. The steps are described in detail below:

### 2.1. Step 1

The integral part of the logarithm, $l_i(x) = \lfloor \log_2(x) \rfloor$, is determined by the position of the Most Significant Bit (MSB) of the input number *x*, where

$$2^n \le x < 2^{n+1} \Rightarrow l_i(x) = n \quad (1)$$

### 2.2. Step 2

The fractional part of the logarithm, i.e. $l_f = \log_2(x) - l_i(x)$, is estimated by linear approximation between the points $(2^n, n)$ and $(2^{n+1}, n+1)$, $n$=1,2,…, of the function $\log_2(x)$ resulting in Eq. 2. The approximated fractional part is

$$l_f(x) = \frac{x - 2^{l_i(x)}}{2^{l_i(x)+1} - 2^{l_i(x)}} = \frac{x}{2^{l_i(x)}} - 1 \quad (2)$$

### 2.3. Step 3

In the third step, a new approximation $l_f'(x)$ of the fractional part of the logarithm is derived as a function of $l_f(x)$. It is obtained by piecewise linear approximation between the points $(2^n, n)$ and $(2^{n+1}, n+1)$ using *s* linear segments of equal length.

The fractional part $l_f(x)$ determines the segment *seg*(*x*) to which *x* belongs, according to Eq. 3.

$$seg(x) = \lfloor s \cdot l_f(x) \rfloor \quad (3)$$

As illustrated in Fig. 2, for an input number *x*, the endpoints *A* and *B* of the segment *seg*(*x*) are elevated by $c_{seg(x)}$ and $c_{seg(x)+1}$ and their coordinates are derived by Eqs. 4 and 5 respectively. The equation that defines the linear segment *AB* is simplified to the equivalent Eq. 6 producing the equation used for the calculation of $l_f^{'}(x)$.

$$A \equiv \left( \frac{seg(x)}{s}, \frac{seg(x)}{s} + c_{seg(x)} \right) \quad (4)$$

$$B \equiv \left( \frac{seg(x)+1}{s}, \frac{seg(x)+1}{s} + c_{seg(x)+1} \right) \quad (5)$$

$$l_f^{'}(x) = l_f(x) + c_{seg(x)} + \left(c_{seg(x)+1} - c_{seg(x)}\right) \cdot \left(s \cdot l_f(x) - seg(x)\right) \quad (6)$$

The optimal parameters $c_i$ can be determined by minimizing the approximation error *E* (Eq. 7), which represents the sum of the relative differences between the actual and approximated values of the logarithm.

$$E = \sum_x \left| \frac{\left(l_i(x) + l_f'(x)\right) - \log_2(x)}{\log_2(x)} \right| \quad (7)$$

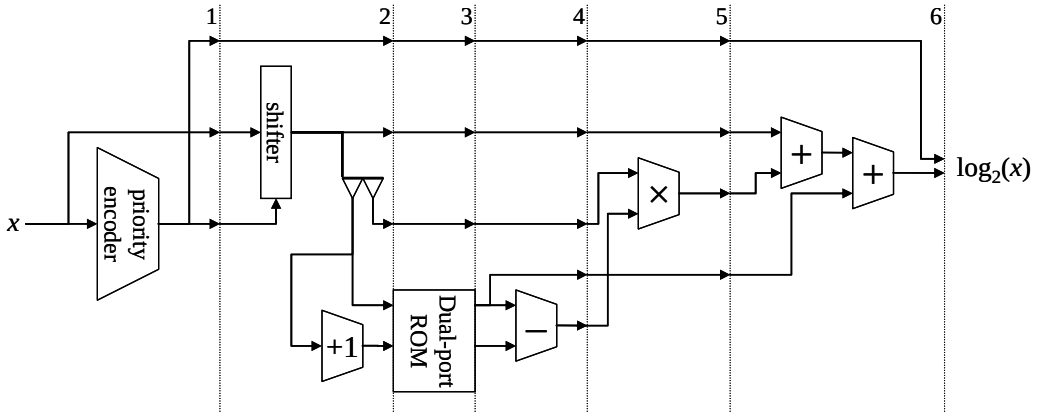


**Figure 3. The implementation of the logarithm approximation architecture**

### 2.4. Implementation

The proposed architecture is implemented as a fully pipelined circuit, in order to achieve a throughput of one result per clock cycle. It consists of 6 pipeline stages and one dual-ported ROM. The ROM stores the parameters $c_i$ in a $b$-bit wide fixed point representation and is implemented on a single FPGA BlockRAM. Figure 3 illustrates the structure of the pipelined circuit. In the first pipeline stage, a priority encoder is used to locate the MSB of the input number $x$. The output of the priority encoder is the integral part $l_i(x)$ of $\log_2(x)$. In the second stage a shifter is used to isolate the fractional part $l_f(x)$. It is worth noting that the priority encoder and shifter used in the first two stages only consume a limited amount of FPGA resources due to their small width.

For the calculation of $l_f'(x)$, we have regarded $s$ as a power of two. Thus, the quantities $seg(x)$ and $s \cdot l_f(x) - seg(x)$, needed for the calculation of Eq. 6, can be calculated from the fixed-point representation of $l_f(x)$ by separating its bits into two parts, without performing any arithmetic or logic operations.

In order to proceed with the calculation of $l_f'(x)$, $seg(x)$ and $s \cdot l_f(x) - seg(x)$ are calculated at the end of the second stage. The third stage involves two concurrent ROM lookups at addresses $seg(x)$ and $seg(x)+1$, retrieving the parameters $c_{seg(x)}$ and $c_{seg(x)+1}$. In the fourth stage, $c_{seg(x)}$ is subtracted from $c_{seg(x)+1}$ and in the fifth stage the result is multiplied by $s \cdot l_f - seg(x)$ using one of the on-chip multipliers. In the last stage, the result of the fifth stage is added to the sum of $l_f(x)$ and $c_{seg(x)}$ to produce $l_f'(x)$, which is concatenated to $l_i(x)$ in order to produce the final $\log_2(x)$ approximation.

## 3. Results

Experiments were conducted to evaluate the performance of the proposed architecture. The results are accompanied with comparisons with state of the art architectures implementing piecewise linear approaches for logarithm approximation.

The proposed architecture for the approximation of the base-2 logarithm of 16-bit integers was implemented on a Xilinx Virtex-5 FPGA (XC5VLX110T-3) which features 69120 slice flip-flops and LUTs, 64 DSP48E multiplier blocks and 148 36kbit BlockRAMs. The number of linear segments $s$ and the width $b$ of the parameters $c_i$ are adjusted so that the ROM resides on a single 36kbit BlockRAM and the multiplier is implemented on a single 25×18 bit multiplier of a DSP48E block. Therefore $s$ ranges from 2 to 1024, whereas $b$ ranges from 2 to 24 bits. The logarithm approximation output is a fixed-point value that utilizes 4 bits for the representation of the integral part, whereas the fractional part occupies up to 32 bits.

In order to evaluate the proposed architecture compared to the other piecewise linear approximation approaches, the obtained approximation error $E$ was calculated for each method, as shown in Table 1. The method proposed by Hall et al. [9] surpasses the other approaches and achieves $E=1.65 \cdot 10^{-4}$. These piecewise linear approximation approaches [8-10] aim to provide a fast approximation of the logarithm that requires few hardware resources. Due to their design considerations, they employ hardwired logic in order to increase the accuracy of [7].

In contrast to the aforementioned methods, the proposed architecture yields considerable gains in accuracy by efficiently using the available FPGA resources, instead of relying on hardwired error-correcting logic. The comparison of the proposed architecture to the implementation of [9] on the same

**Table 1:**
**Error *E* of piecewise linear approximation methods**

| Method | Segments | Error *E* |
|---|---|---|
| Mitchell et al. [7] | 1 | 3.99E-03 |
| SanGregory et al. [8] | 2 | 1.89E-03 |
| Hall et al. [9] | 4 | 1.65E-04 |
| Abed et al. [10] | 2 | 1.11E-03 |
| | 3 | 6.75E-04 |
| | 6 | 2.28E-04 |
| Proposed architecture | 4 | 1.37E-04 |
| | 64 | 4.01E-07 |
| | 1024 | 1.83E-09 |

**Table 2:**
**Comparison of the proposed architecture to Hall et al. [9]**

| | Hall et al. [9] | Proposed architecture | |
|---|---|---|---|
| Error *E* | 1.65E-04 | 1.37E-04 | 1.83E-09 |
| Segments *s* | 4 | 4 | 1024 |
| Bits *b* | - | 8 | 24 |
| Slices | 180 | 254 | 329 |
| Frequency | 280MHz | 318MHz | 288MHz |
| BlockRAMs | 0 | 1 | 1 |
| Multipliers | 1 | 1 | 1 |

FPGA is presented in Table 2 and reveals that the error $E$ can be decreased by five orders of magnitude ($E$=1.83·10$^{-9}$) when using the proposed architecture, by utilizing one BlockRAM and only moderately increasing the occupied slices. It is important to note that the 329 slices required by the proposed architecture represent less than 0.5% of the total FPGA area. In order to reach similar accuracy using more complex methods, such as polynomial [4] or CORDIC [6], significantly more FPGA resources would be required. The floating point logarithm approximation architecture presented in [4] achieves a lowest approximation error in the order of $E$=10$^{-8}$, which is higher than that of the proposed architecture, while requiring 6 times more multipliers when synthesized on the same FPGA. Also, the CORDIC architecture, which produces fixed point output, occupies nearly 8 times more slices than the proposed architecture for $E$ in the order of 10$^{-9}$.

## 4. Conclusions

We presented an FPGA-based architecture for fast and area-efficient approximation of the base-2 logarithm on FPGA devices. The novel features of this architecture can be summarized in the following:

- It implements a piecewise linear approximation of the logarithm function using a large number of linear segments, in order to achieve up to five orders of magnitude higher accuracy than the comparable methods.
- In contrast to the state of the art implementations, it is designed to exploit the available FPGA resources, such as BlockRAMs and multipliers, in order to provide low FPGA area utilization and high frequency.

The first feature allows the proposed architecture to be used in a variety of FPGA designs as a generic component that provides highly accurate logarithm calculation. The second feature enables a significant reduction in FPGA slices by using only one BlockRAM for the implementation of the dual-ported ROM and one FPGA multiplier, also allowing the architecture to reach a high frequency potential, thus rendering the proposed architecture suitable for high-throughput applications.

Other logarithm approximation architectures implementing piecewise linear approximation [8-10] use up to a maximum of 6 segments, whereas they cannot be reconfigured to further increase the number of segments, as the number of segments is hardwired. The proposed architecture overcomes this limitation, consequently increasing the accuracy, while retaining the low FPGA area requirements and the frequency advantages of these methods. Thus, it can be embedded into any FPGA-based application that requires fast and accurate logarithm approximation for throughput-sensitive and real-time applications.

## Acknowledgement

This work was realized under the framework of the Reinforcement Program of Human Research Manpower (“PENED 2003” – 03ED324), co-funded 25% by the General Secretariat for Research and Technology, Greece, and 75% by the European Social Fund.

## References

[1] D.M. Rocke and B. Durbin, “Approximate variance-stabilizing transformations for gene-expression microarray data”, Bioinformatics, Vol. 19 no. 8, pages 966–972, 2003

[2] D. Bariamis, D.K. Iakovidis, D. Maroulis, “Dedicated hardware for real-time computation of second-order statistical features for high resolution images”, Lect. Notes in Comp. Science, Volume 4179 LNCS, Pages 67-77, 2006

[3] D. M. Mandelbaum, and S. G. Mandelbaum, “A Fast, Efficient Parallel-Acting Method of Generating Functions

Defined by Power Series, Including Logarithm, Exponential, and Sine, Cosine," IEEE Trans. on Parallel and Distributed Systems, vol. 7, no. 1, pp. 33-45, Jan. 1996.

[4] J. Detrey, F. de Dinechin, "Parametrized floating-point logarithm and exponential functions for FPGA", Micropr. and Microsys., Vol. 31, Issue 8, pp. 537-545, Dec. 2007.

[5] J.-A. Pineiro, M.D. Ercegovac, J. D. Bruguerax, "High–Radix Logarithm with Selection by Rounding", Proceedings of the IEEE Int. Conf. on Application-Specific Systems, Architectures, and Processors, pp. 101-110, July 2002

[6] J. E. Volder, "The CORDIC Trigonometric Computing Technique", IRE Trans. Electronic Comp., 8:330-334, 1959.

[7] J.N. Mitchell Jr., "Computer Multiplication and Division Using Binary Logarithms," IRE Trans. Electronic Comp., 11:512-517, 1962.

[8] S.L. SanGregory, R.E. Siferd, C. Brother, and D. Gallagher, "A Fast, Low-Power Logarithm Approximation with CMOS VLSI Implementation," Proc. IEEE Midwest Symp. Circ. and Syst., Aug. 1999.

[9] E.L. Hall, D.D. Lynch, and S.J. Dwyer III, "Generation of Products and Quotients Using Approximate Binary Logarithms for Digital Filtering Applications," IEEE Trans. Comp., vol. 19, pp. 97-105, Feb. 1970.

[10] K. H. Abed and R. E. Siferd "CMOS VLSI Implementation of a Low-Power Logarithmic Converter," IEEE Trans. Comp., vol. 52, pp. 1421-1433, Dec 2003.

[11] D. Lee, A. Abdul Gaffar, O. Mencer and W. Luk, "Optimizing hardware function evaluation", IEEE Trans. Comp., vol. 54, no. 12, pp. 1520-1531, Dec 2005.