Microprocessors and Microsystems 36 (2012) 324–332

Contents lists available at SciVerse ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

A dynamic non-uniform segmentation method for first-order polynomial function evaluation

Dongdong Chen, Seok-Bum Ko *

Department of Electrical and Computer Engineering, University of Saskatchewan, 57 Campus Drive, Saskatoon, SK, Canada S7N 5A9

ARTICLE INFO

Article history:
Available online 3 March 2012

Keywords:
Computer arithmetic
Elementary function evaluation
Linear approximation
Non-uniform segmentation
FPGA

ABSTRACT

This paper presents a new dynamic non-uniform segmentation method for the first-order polynomial function evaluation. The proposed method can evaluate the elementary functions (e.g. log, exp, sin, cos, tan, etc.) and combinations of these functions by an optimized linear approximation with the fewest non-uniform segments. Compared with the previous evaluation method based on the static bit-width analysis, the proposed method is mainly based on a dynamic bit-width analysis and capable of reducing the number of segments, which in turn can significantly reduce the memory size occupied in hardware. The proposed dynamic method can evaluate the function to satisfy accuracy by the linear approximation in which the input, coefficients, and intermediate values are rounded to fewer bit-width, which cannot be achieved by previous static non-uniform segmentation methods. The hardware performance evaluation results on FPGA show that the proposed method consumes about 66% fewer hardware resources, 56% less actual memory usage, and performs 32% shorter delay on average in comparison with the non-uniform segmentation method based on static bit-width analysis.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Elementary functions, such as logarithmic, exponential, trigonometric, reciprocal, and square root, and combinations of these functions (compound functions) are essential in digital signal processing, communication system, scientific computing and so on. The computation of the elementary arithmetic functions is expected to be quick and accurate. Muller [1] presents both software and hardware-oriented algorithms to compute elementary functions. The piecewise polynomial approximation algorithm is an attractive method because any elementary or compound function can be evaluated by a set of simple linear or higher-order polynomial approximations. Based on the polynomial approximation algorithm, the elementary function evaluation in hardware [2–9] has been realized on a field-programmable gate array (FPGA).

The piecewise linear approximation algorithm can evaluate elementary functions by a set of simple linear approximations, in particular, when the objective is to use these elementary functions to implement the high-speed and low-power applications. On the other hand, the high-order polynomial approximation can be applied to reduce the number of segments, and therefore the memory size can be reduced in hardware. However, by the high-order polynomial method, more multipliers and adders are required, which leads to a longer delay and a larger area in hardware. In order to achieve a required accuracy, the interval of the function can be split into a set of segments with the same size. Such an approach is called uniform segmentation method [2–7,9]. However, the shortcoming of this approach is that numerous segments make the size of the look-up table become too large to be actually practical. By way of contrast, a more effective approach is to determine the segment that has the largest size while maintaining the specified approximation accuracy. Such an approach in which segments have different width is called a non-uniform segmentation method [8,10,11].

Main challenges for designing a first-order polynomial structure based on the non-uniform segmentation method stem from the following three aspects. The first challenge is to determine the minimum number of the bit-width for internal signals in the fixed-point data path. The most commonly used approach for bit-width optimization is a dynamic method in which the bit-width of each signal is gradually adjusted to a point where all inputs meet the precision requirement [12–15]. In [16], a static bit-width optimization approach (MiniBit) is proposed, which is adopted in the static non-uniform segmentation methods [8] to compute the bit-width for each signal in mathematical manner. The second challenge is to select the best scheme to partition the interval of the function to the fewest number of segments, which can produce a minimum look-up table size for storing the coefficients and a relative simple segment index encoder (SIE). In [8], the domain of the function is partitioned at the point where the maximum absolute

* Corresponding author. Tel.: +1 306 966 5456.
E-mail addresses: doc220@mail.usask.ca (D. Chen), seokbum.ko@usask.ca (S.-B. Ko).

0141-9331/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.micpro.2012.02.016

error occurs. The third challenge is to compute the best-fit linear approximation with finite precision internal signals in each segment so that a faithful rounding can be guaranteed for accuracy.

In this paper, we propose a dynamic non-uniform segmentation method for the linear approximation function evaluation. The main advantages of this method are:

- it can reduce the uniform fractional bit-width (UFB) determined by MiniBit method by a dynamic bit-width analysis;
- it can limit the number of non-uniform segments to minimum by a binary search partition scheme (BSPS) [17];
- it can compute the best-fit optimized linear approximation in which internal signals are rounded to finite precision in each segment.

This paper is organized as follows: Section 2 presents the notations and analyzes the minimum maximum (minimax) linear approximation in one segment. In Section 3, we present the proposed dynamic non-uniform segmentation method for the linear approximation function evaluation. Section 4 gives the hardware implementation for function evaluation. In Section 5, we give experimental and comparison results. Section 6 gives conclusions.

2. Minimax polynomial approximation

2.1. Notations

In this paper, we deal with the linear approximation evaluation of the function $f(x)$ with its input and output in the binary fixed-point (BXP) format. The input value of x is a m-bit BXP number in the domain $[a, b]$; the function evaluation result is a n-bit BXP number. In order to achieve a specified accuracy, the interval of x is typically split into a set of subintervals, $[a_i, b_i]$, where $a \leqslant a_i < b_i \leqslant b$, and i is the segment index. According to [1], in each subinterval, there are many straight lines, defined as P_1 that can evaluate the function $f(x)$, and of which only $p^*(x)$ is the best-fit linear approximation, $c_{0i}x + c_{1i}$, for achieving the minimax absolute error:

$$\|f(x) - p^*(x)\|_\infty = \min_{p(x)\in P_1} \max_{a_i \leqslant x \leqslant b_i} |f(x) - p(x)| \tag{1}$$

With the piecewise linear approximation, errors are produced in three ways. The first one is the maximum linear approximation error, $|\varepsilon_a|$, resulted from the absolute value of the difference between the function $f(x)$ and its minimax linear approximation:

$$|\varepsilon_a| = \max_{a_i \leqslant x < b_i} |f(x) - (c_{0i} \times x + c_{1i})|, \tag{2}$$

where x, c_{0i}, c_{1i} and $D_i = c_{0i} \times x$ represent infinite precision inputs, coefficients and intermediate values respectively.

The second one is the absolute quantization error, $|\varepsilon_q|$, as shown in (3), produced by the finite precision of rounded inputs, x', coefficients, c'_{0i} and c'_{1i}, and intermediate values, D'_i, in the hardware implementation.

$$|\varepsilon_q| = \left|(c_{0i} \times x + c_{1i}) - \left(D'_i + c'_{1i}\right)\right| \tag{3}$$

The third one is the absolute final output rounding error, $|\varepsilon_r|$, whose maximum value is 0.5 unit in the last place (ulp). In order to obtain a n-bit accuracy, the following condition must be satisfied:

$$|\varepsilon_t| = |\varepsilon_a| + |\varepsilon_q| + |\varepsilon_r| \leqslant 2^{-n} \tag{4}$$

Table 1 summarizes the symbols and notations used in this paper.

2.2. Minimax error analysis in one segment

In each segment, the best-fit straight line can be found by Chebyshev theorem [1] which gives a characterization of the minimax approximations to a function.

Chebyshev Theorem: p^* is the minimax degree-n approximation to f on $[a_i, b_i]$, if and only if there are at least $n+2$ values, $a_i \leqslant x_0 < x_1 < \ldots < x_n < x_{n+1} \leqslant b_i$, such that:

$$p^*(x_i) - f(x_i) = (-1)^i[p^*(x_0) - f(x_0)] = \pm\|f - p^*\|_\infty \tag{5}$$

Based on Chebyshev theorem, there are at least three values, x_0, x_1 and x_2, where the minimax approximation error, ε_a, is kept balanceable and reached with alternate signs. The convexity of the function $f(x)$ implies that the differences between $f(x)$ and $p^*(x)$ at *starting* ($x_0 = ai$), *ending* ($x_2 = b_i$) and *tangent* ($f'(x_1) = c_{0i}$) points are equal, and represent the minimax error. Thus, we obtain:

$$\begin{cases} f(a_i) - (c_{0i} \times a_i + c_{1i}) = -\varepsilon_a \\ f(x_1) - (c_{0i} \times x_1 + c_{1i}) = \varepsilon_a \\ f(b_i) - (c_{0i} \times b_i + c_{1i}) = -\varepsilon_a \\ f(x_1)' - c_{0i} = 0 \end{cases} \tag{6}$$

According to (6), the coefficients c_{0i} and c_{1i}, the value x_1 and the minimax error ε_a are computed so that the best-fit minimax linear approximation in $[a_i, b_i]$ is determined. Since, the infinite precision input, coefficients and intermediate values have to be rounded to the finite precision x', c'_{0i}, c'_{1i} and D'_i in hardware. As a result, a quantization error, ε_q, is produced, and the best-fit linear approximation line obtained by Chebyshev theorem is moved to $p^*(x')$ as shown in (7), which is not a minimax linear approximation anymore, and the minimax approximation errors are not balanceable.

$$\begin{aligned} D_i &= c'_{0i} \times x' \\ p^*(x') &= D'_i + c'_{1i} \end{aligned} \tag{7}$$

In order to redetermine a new best-fit linear approximation, $p^*_r(x')$, with these rounded values of x', c'_{0i}, c'_{1i} and D'_i, we keep the value of c'_{0i} and adjust the value of c'_{1i} to c''_{1i} according to (8):

$$c''_{1i} = \frac{\max(f(x) - D'_i) + \min(f(x) - D'_i)}{2} \tag{8}$$

Thus, a new best-fit linear approximation $p^*_r(x')$ is obtained, which leads to reoccupy an optimized approximate error, ε'_a:

$$p^*_r(x') = D'_i + c''_{1i} \tag{9}$$

Algorithm 1. Determination of coefficients c'_{0i} and c''_{1i} in One Segment

Input: (1) Function, $f(x)$; (2) one segment, $[a_i, b_i]$; (3) precision of c'_{0i} in bits, q; (4) precision of c'_{1i} and $D'_i(x')$ in bits, p; (5) precision of x in $f(x)$, m.
Output: 1) Best-fit coefficients c'_{0i} and c''_{1i} ; 2) Minmax error, $|\varepsilon'_a|$

1: $[c_{0i}, c_{1i}] \leftarrow Chebyshev(f(x), [a_i, bi])$
2: $c'_{0i} \leftarrow round(c_{0i}, q)$
3: $c'_{1i} \leftarrow round(c_{1i}, p)$
4: $D'_i(x') = round(c'_{0i} \times x', p)$
5: $max \leftarrow \max_{\{x_i = a_i \text{ to } b_i \text{ steps } 2^{-m}\}} (f(x_i) - D'_i(round(x_i, q)))$
6: $min \leftarrow \min_{\{x_i = a_i \text{ to } b_i \text{ steps } 2^{-m}\}} (f(x_i) - D'_i(round(x_i, q)))$
7: $c''_{1i} \leftarrow \frac{max+min}{2}$
8: $|\varepsilon'_a| \leftarrow \frac{max-min}{2}$

Download English Version:

https://daneshyari.com/en/article/463099

Download Persian Version:

https://daneshyari.com/article/463099

Daneshyari.com