FINITE-PRECISION ERROR MODELING USING AFFINE ARITHMETIC

Shervin Vakili, J. M. Pierre Langlois and Guy Bois

Department of Computer Engineering, École Polytechnique de Montréal, Montreal, Canada

ABSTRACT

This paper introduces a new approach for finite-precision error modeling based on affine arithmetic. The paper demonstrates that there is a common hazard in affine arithmetic-based error modeling methods described in the literature. The hazard is linked to early substitution of the signal terms that emerge in operations such as multiplication and division. The paper proposes postponed substitution combined with function maximization to address this problem. The paper also proposes a modification in the error propagation process to enhance the error modeling accuracy. An existing word length optimization method is reproduced to evaluate the efficiency of this modification. The results demonstrate that the proposed modification can improve the hardware area results by up to 7.0% at the expense of negligible complexity overhead.

Index Terms— Fixed-point arithmetic, error modeling, precision analysis, Affine Arithmetic

1. INTRODUCTION

Finite-precision error modeling is a key step in accuracy-aware fixed-point design. Error models are essential for word-length optimization methodologies, which have been the subject of a large body of research in the last decade. Accuracy and computational complexity are the main factors for the evaluation of an error modeling approach.

Many simulation-based and analytical techniques have been introduced in the literature for error modeling [1-3]. Interval arithmetic (IA) is a well-known analytical method which was originally invented to find the range of signals in a computational circuit [4]. The main drawback of IA is that it ignores the correlation among signals [3], [5]. Affine arithmetic (AA) is a preferable approach that addresses the correlation problem by taking signal interdependency into account. In AA, each signal is represented as a linear combination of certain primitive variables which stand in for sources of uncertainty. Fang *et al.* [3] and Lee *et al.* [6] introduced AA-based error propagation methods for quantization error modeling. Other works have tried to improve the AA-based method, but they typically fail to cover all features of the basic method [7].

This work was supported by the Fonds Québécois de la Recherche sur la Nature et les Technologies (FQRNT).

This paper illustrates a common hazard in existing AA-based error modeling approaches and proposes a solution to address it. The paper also suggests a modification of the propagation process which can effectively improve error model accuracy. Improvements are demonstrated and quantified using a set of widely used case studies.

Section 2 of this paper briefly reviews necessary affine arithmetic concepts. Section 3 describes existing error propagation models and their shortcomings. Section 4 presents our proposed solution. Section 5 gives experimental results and comparisons, and Section 6 concludes the paper.

2. AFFINE ARITHMETIC

In affine arithmetic, the estimated value $\hat{x}$ of a signal x is represented by the sum of a constant x_0 and a finite set of n uncertainty terms $x_i\varepsilon_i$, as follows:

$$\hat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 + \cdots + x_n\varepsilon_n, \quad \varepsilon_i \in [-1, 1].$$

Each ε_i element is an independent uncertainty factor of the total uncertainty of the signal. The estimated value of a signal x with specified range $[x_{min}, x_{max}]$ is represented in affine form by

$$\hat{x} = x_0 + x_1\varepsilon_1,$$

where

$$x_0 = \frac{x_{min} + x_{max}}{2}, \quad x_1 = \frac{x_{min} - x_{max}}{2}.$$

The affine form for addition-subtraction is calculated by adding-subtracting the affine expression of the inputs. Multiplication is more complex due to the emergence of non-affine terms in the result expression. The widely used solution is to replace these terms with an affine approximation that introduces a new uncertainty factor [3], [6], [8].

The range of each signal is calculated from its affine representation by finding the minimum and maximum values when the uncertainty factors are replaced by -1 or 1.

In error modeling with AA, the quantization errors must be added to the affine forms. The quantized value x_q of a signal x is represented in affine form as

$$x_q = x + E_{x_q}, \tag{1}$$

where

$$E_{x_q} = \begin{cases} 2^{-FB_{x_q}-1}.\varepsilon_x & Round\ to\ nearest \\ 2^{-FB_{x_q}}.\varepsilon_x & Truncation \end{cases}, \quad \varepsilon \in [-1, 1].$$

The E_{x_q} and FB_{x_q} terms correspond to the quantization error and fractional bit width of the quantized signal x_q, respectively. In Equation (1), we show two quantization approaches: truncation and round to nearest. With FB fractional bits, truncation and round to nearest cause a maximum error of 2^{-FB} and 2^{-FB-1}, respectively. To keep consistency with existing works, we use the round to nearest approach in the rest of this paper. Equation (1) shows a simplified version of the affine representation that is sufficient for subsequent calculations. A more detailed treatment can be found in [3].

Using the affine expression from (1), the quantization error from addition-subtraction is obtained as follows:

$$z_q = x_q \pm y_q = x \pm y + E_{x_q} \pm E_{y_q} + \delta$$
$$\Rightarrow E_{z_q} = E_{x_q} \pm E_{y_q} + \delta \quad (2)$$
$$\text{where } \delta = \begin{cases} 2^{-FB_{z_q}-1}\varepsilon_z, & FB_{z_q} < \max(FB_{x_q}, FB_{y_q}) \\ 0, & otherwise \end{cases}.$$

From (2), we see that the total error at the output of an addition is equal to sum of errors of its operands added to the quantization error of the output signal. The δ is nonzero only when the output has more fractional bits than both operands.

For multiplication, the error is:

$$E_{z_q} = xE_{y_q} + yE_{x_q} + E_{y_q}E_{x_q} + \delta \quad (3)$$
$$\text{where } \delta = \begin{cases} 2^{-FB_{z_q}-1}\varepsilon_z, & FB_{z_q} < FB_{x_q} + FB_{y_q} \\ 0, & otherwize \end{cases}.$$

In a commonly used conservative approximation [3], [6], the *x* and *y* terms in (3) are replaced by the maximum absolute values of the *x* and *y* inputs that can be inferred from the range values.

3. EXISTING ERROR PROPAGATION METHOD

To compute the finite-precision error of a computational flow which is composed of consecutive elementary operations, an error propagation procedure is required.

Fang *et al.* [3], Lee *et al.* [6] and Pu *et al.* [9] presented the widely accepted reference methods for AA-based error propagation. As a main contribution, this paper identifies a common hazard that may arise in all error modeling approaches used in these works. We illustrate this hazard by applying Lee's error modeling approach [6] on the example shown in Fig.1. The same problem emerges in Fang's and Pu's methods as well. Lee's method omits the conditional terms of the error equation by assuming there is always a quantization step after each operation. In other words, the number of output fractional bits of the operations is always assumed to be shorter than the maximum number of meaningful fractional bits that is produced by that operation. So, the error model for addition/subtraction becomes

$$E_{z_q} = E_{x_q} \pm E_{y_q} + 2^{-FB_{z_q}-1}\varepsilon_z.$$

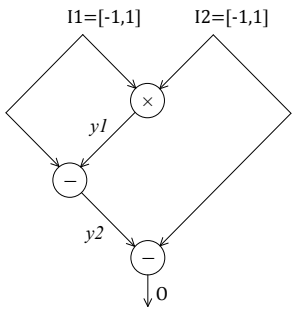

Fig. 1. Circuit that calculates $z = (I1 \times I2) - I1 - I2$. The input range values are shown in brackets.

For multiplication it is

$$E_{z_q} = xE_{y_q} + yE_{x_q} + E_{y_q}E_{x_q} + 2^{-FB_{z_q}-1}\varepsilon_z.$$

Accordingly, the error models of the signals of Fig. 1 are obtained as follows:

$$E_{I1} = 2^{-FB_{I_1}-1}\varepsilon_1$$
$$E_{I2} = 2^{-FB_{I_2}-1}\varepsilon_2$$
$$E_{y1} = (I1 \times E_{I2}) + (I2 \times E_{I1}) + E_{I1}E_{I2} + 2^{-FB_{y_1}-1}\varepsilon_3$$
$$E_{y2} = E_{y1} - E_{I1} + 2^{-FB_{y_2}-1}\varepsilon_4$$
$$E_O = E_{y2} - E_{I2} + 2^{-FB_O-1}\varepsilon_5.$$

The hazard arises in the substitution of the signal terms that emerge in the multiplication expressions. Common approaches, such as Lee's method, substitute these terms with the absolute value of the worst-case bound of the signals, regardless of the rest of the circuit. This has been claimed to be a conservative approximation. So, the $I1$ and $I2$ terms in the expression for E_{y1} above are substituted by the value 1.

Now, starting from the primary inputs, the error terms are substituted by their corresponding expression until the output error expression is obtained as follows:

$$E_O = E_{I1} + E_{I2} - E_{I1} - E_{I2} + E_{I1}E_{I2} + 2^{-FB_{y_1}-1}\varepsilon_3 + 2^{-FB_{y_2}-1}\varepsilon_4 + 2^{-FB_O-1}\varepsilon_5$$
$$= E_{I1}E_{I2} + 2^{-FB_{y_1}-1}\varepsilon_3 + 2^{-FB_{y_2}-1}\varepsilon_4 + 2^{-FB_O-1}\varepsilon_6. \quad (4)$$

Since the ε elements are in the range of -1 to 1, the upper bound of the error at the output z is

$$max(E_O) = 2^{-FB_{I_1}-FB_{I_2}-2} + 2^{-FB_{y_1}-1} + 2^{-FB_{y_2}-1} + 2^{-FB_O-1}.$$

The $E_{I1}E_{I2}$ term in (4) is small and is often disregarded [5],[3]. The incorrectness of the error expression in (4) can be easily shown by an example. If $I1 = I2 = 0$, the output error expression becomes

$$\acute{E}_O = -E_{I1} - E_{I2} + E_{I1}E_{I2} + 2^{-FB_{y_1}-1}\varepsilon_3 + 2^{-FB_{y_2}-1}\varepsilon_4 + 2^{-FB_O-1}\varepsilon_5,$$
$$\Rightarrow max(\acute{E}_O) = 2^{-FB_{I_1}-1} + 2^{-FB_{I_1}-1} + 2^{-FB_{I_1}-FB_{I_2}-2} + 2^{-FB_{y_1}-1} + 2^{-FB_{y_2}-1} + 2^{-FB_O-1},$$

which is larger than the value in (4) due to the contribution of the input quantization errors (E_{I1} and E_{I2}). In fact, the early substitution of the signal terms in (4) causes incorrect cancellation in later stages.

Cong *et al.* [5] used a more complex propagation method that does not generate the hazard. In this method, the signal terms are substituted with their corresponding affine representation instead of worst-case approximation. In the example of Fig. 1, the $I1$ and $I2$ terms in E_{y1} expression are substituted with $0 + \varepsilon_6$ and $0 + \varepsilon_7$, respectively.

$$E_{y1} = 2^{-FB_{I2}-1}\varepsilon_2\varepsilon_6 + 2^{-FB_{I1}-1}\varepsilon_1\varepsilon_7 + 2^{-FB_{y1}-1}\varepsilon_3.$$

The products $\varepsilon_2\varepsilon_6$ and $\varepsilon_1\varepsilon_7$ are replaced by new uncertainty factors as follows:

$$E_{y1} = 2^{-FB_{I2}-1}\varepsilon_8 + 2^{-FB_{I1}-1}\varepsilon_9 + 2^{-FB_{y1}-1}\varepsilon_3.$$

So, the output error expression is obtained as:

$$\begin{aligned} E_O &= 2^{-FB_{I2}-1}\varepsilon_8 + 2^{-FB_{I1}-1}\varepsilon_9 + 2^{-FB_{y1}-1}\varepsilon_3 - 2^{-FB_{I1}-1}\varepsilon_1 \\ &\quad -2^{-FB_{I2}-1}\varepsilon_2 + 2^{-FB_{y2}-1}\varepsilon_4 + 2^{-FB_O-1}\varepsilon_5 \\ \Rightarrow \max(E_O) &= 2^{-FB_{I2}} + 2^{-FB_{I1}} + 2^{-FB_{y1}-1} + 2^{-FB_{y2}-1} \\ &\quad +2^{-FB_O-1}. \end{aligned} \tag{5}$$

This is the correct error model for Fig. 1. The $E_{I1}E_{I2}$ term is neglected in Cong's paper [5]. Although this method can address the mentioned hazard, we found that it also faces a significant issue. Cong's method can occasionally fail to keep track of the correlation between new uncertainty factors. For example, applying this method on the circuits shown in Fig. 2, the following error expressions are obtained.

$$\begin{aligned} E_A &= 2^{-FB_A-1}\varepsilon_4 \\ E_B &= 2^{-FB_B-1}\varepsilon_5 \\ E_C &= 2^{-FB_B-1}\varepsilon_6 \\ E_{x1} &= (1+\varepsilon_1)2^{-FB_B-1}\varepsilon_5 + (1+\varepsilon_2)2^{-FB_A-1}\varepsilon_4 \\ &= 2^{-FB_B-1}\varepsilon_5 + 2^{-FB_B-1}\varepsilon_7 + 2^{-FB_A-1}\varepsilon_4 + 2^{-FB_A-1}\varepsilon_8 \\ E_{x2} &= (1+\varepsilon_2)2^{-FB_C-1}\varepsilon_6 + (1+\varepsilon_3)2^{-FB_B-1}\varepsilon_5 \\ &= 2^{-FB_B-1}\varepsilon_6 + 2^{-FB_B-1}\varepsilon_9 + 2^{-FB_C-1}\varepsilon_5 + 2^{-FB_C-1}\varepsilon_{10} \\ E_{x3} &= \big((1+\varepsilon_1)\times E_{x2}\big) + \big((2+2\varepsilon_{11})\times E_A\big) \\ &= (2^{-FB_B-1}\varepsilon_6 + 2^{-FB_B-1}\varepsilon_9 + 2^{-FB_C-1}\varepsilon_5 + \\ &\quad 2^{-FB_C-1}\varepsilon_{10} + 2^{-FB_B-1}\varepsilon_{12} + 2^{-FB_B-1}\varepsilon_{13} + \\ &\quad 2^{-FB_C-1}\varepsilon_{14} + 2^{-FB_C-1}\varepsilon_{14}) + (2^{-FB_A}\varepsilon_4 + \\ &\quad 2^{-FB_A}\varepsilon_{15}) \\ E_{x4} &= \big((2+2\varepsilon_{16})\times E_B\big) + \big((1+\varepsilon_3)\times E_{x1}\big) \\ &= (2^{-FB_B}\varepsilon_5 + 2^{-FB_B}\varepsilon_{17}) + (2^{-FB_B-1}\varepsilon_5 + \\ &\quad 2^{-FB_B-1}\varepsilon_7 + 2^{-FB_A-1}\varepsilon_4 + 2^{-FB_A-1}\varepsilon_8 + \\ &\quad 2^{-FB_B-1}\varepsilon_{18} + 2^{-FB_B-1}\varepsilon_{19} + 2^{-FB_A-1}\varepsilon_{20} + \\ &\quad 2^{-FB_A-1}\varepsilon_{21}) \\ E_z &= E_{x3} - E_{x4}. \end{aligned}$$

Note that in order to simplify calculations, we do not take the quantization error of intermediate signals ($e.g., x1$-$x4$) into account.

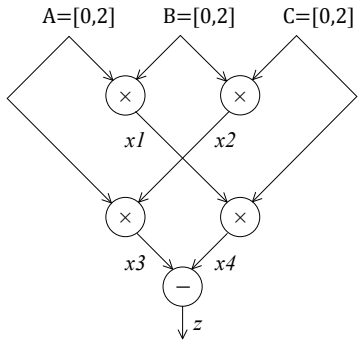

Fig. 2. Circuit that calculates $Z = (A \times B \times C) - (A \times B \times C)$.

Fig. 2 calculates $Z = (A \times B \times C) - (A \times B \times C) = 0$. In a correct propagation approach, the quantization errors of the input signals should be cancelled in the subtraction operation before reaching the output Z error expression. Hence, the correct output error expression is $E_z = 0$. However, Cong's method gives a different error expression due to the fact that this method fails to keep track of the correlation among new uncertainty factors. As illustrated by the example in Fig. 2, this issue can cause considerable overestimation of the error values.

Another significant challenge of Cong's method is the large number of uncertainty factors that it introduces in multiplication operations. This leads to high computational complexity of the propagation process.

4. PROPOSED APPROACH

We propose a modified error propagation approach with two improvements. The first improvement addresses the shortcomings of the existing methods. The second improvement involves the propagation of conditional terms which can enhance the accuracy.

4.1. Postponed substitution

An effective way to address the problems described in Section 3 is to postpone the signal term substitution until the last stage of the propagation process. Applying this modification to the circuit of Fig. 1, the $I1$ and $I2$ terms in $E_{y1}$are not substituted by a value until the output error E_O is obtained as:

$$\begin{aligned} E_O &= (I1-1)\times 2^{-FB_{I2}-1}\varepsilon_2 + (I2-1)\times 2^{-FB_{I1}-1}\varepsilon_1 \\ &\quad +2^{-FB_{y1}-1}\varepsilon_3 + 2^{-FB_{y2}-1}\varepsilon_4 + 2^{-FB_O-1}\varepsilon_5. \end{aligned}$$

Now, for a conservative evaluation, non-constant coefficients of the error terms are approximated to their maximum absolute values. For example, the $(I1-1)$ coefficient is approximated to $max(|I1-1|) = 2$ applying $I1 = -1$, while a value of 0 was calculated for this coefficient in Lee's method due to early substitution. The upper bound error expression

of (5) is eventually obtained for the circuit of Fig. 1 using our method. This case study shows that although early substitution of signal terms simplifies the error propagation process, it may lead to an underestimation of error bounds and consequently inaccuracy in precision analysis.

Applying this method on the circuit of Fig. 2 gives the correct result, i.e., $E_z = 0$. This shows that our method also addresses the overestimation issue of Cong's approach.

Solving the maximization function in the last stage forms the main complexity overhead of our approach. This overhead increases with the number of signal terms that participate in a non-constant coefficient.

4.2. Propagation of conditional terms

The second proposed improvement involves the conditional terms in error expressions. Existing methods assume that the quantization error is always introduced at the output of operations to eliminate conditional elements. Although this assumption simplifies the error propagation process, it may introduce unnecessary error elements that eventually lead to overestimation. Our second improvement proposes avoiding this assumption by propagating the conditional terms in the same way as other elements.

Applying this modification on the circuit of Fig. 1 gives the following upper bound error expression:

$$\max(E_O) = 2^{-FB_{I2}} + 2^{-FB_{I1}} + \delta_{y1} + \delta_{y2} + \delta_O$$

$$\text{where} \quad \delta_{y1} = \begin{cases} 2^{-FB_{y1}-1}, & FB_{y1} < FB_{I1} + FB_{I2} \\ 0, & otherwize \end{cases}$$

$$\delta_{y2} = \begin{cases} 2^{-FB_{y2}-1}, & FB_{y2} < max(FB_{y1}, FB_{I1}) \\ 0, & otherwize \end{cases}$$

$$\delta_O = \begin{cases} 2^{-FB_O-1}, & FB_O < max(FB_{y2}, FB_{I2}) \\ 0, & otherwize \end{cases}.$$

Experimental results in Section 5 demonstrate the effectiveness of this modification on word length optimization.

5. RESULTS AND COMPARISON

We developed a word length optimization system to evaluate the efficiency of the conditional term propagation. This system is a reproduction of the method proposed by Osborne *et al.* [8] and is implemented in MATLAB. The error models are given to the word length optimization process as input. Using a more accurate error model, the optimization process can potentially find shorter signal bit-lengths that still meet the requested output error bound. The shorter signals eventually lead to lower hardware cost during the hardware implementation. This section measures the hardware cost reduction and optimization time overhead obtained by using conditional term propagation in AA-based error modeling. For the experiments described in this section, we have employed five well-known case studies which include some commonly used transforms and operators in the signal and image processing domains.

Table 1. Comparing hardware area and optimization time with and without conditional term propagation

Case Study	# of Signals	Area (slices)			Opt. Time (s)	
		M1*	M2**	Imp (%)	M1	M2
Degree-4 Poly	13	1234	1172	5.3	5.1	5.2
B-Spline	15	719	691	4.1	3.5	3.5
RGB to YCrCb	19	558	537	3.9	5.7	5.9
2×2 Matrix Mult.	29	1939	1812	7.0	14.4	14.7
DCT 8×8	55	5178	4902	5.6	127.3	131.1

*M1: Without cond. term prop. **M2: With cond. term prop.

Lee *et al.* [6] described the case studies in detail. For hardware cost measurement, the case studies were modeled in VHDL. The polynomial approximation was implemented in general form, contrary to Lee *et al.* [6] who customized this approximation to a specific function. All experiments were performed on an Intel i7 3-GHz PC with 16 GB RAM. Designs were synthesized with Synplify 9.1 for a Xilinx Virtex 5 XC5VLX110 FPGA.

Table 1 illustrates the hardware area cost and optimization time for the five case studies. In these results, conditional term propagation saves hardware area by up to 7.0% at the expense of negligible complexity overhead. Achieved hardware savings are significant regarding the competitive results in previous works in this area. Fig. 3 compares the hardware savings obtained in various degrees of polynomial approximation. These results show a slight growth in hardware savings by increasing the application size. The requested output precision was fixed to 8-bits in all reported experiments.

6. CONCLUSION

We have demonstrated a common hazard in existing AA-based finite-precision error modeling methods using two counter examples. We have proposed postponed substitution approach to address this hazard. Furthermore, we have proposed conditional term propagation to enhance error modeling accuracy. The efficiency of our approach was evaluated through a set of case studies. The results show that the approach can yield significant hardware savings with negligible complexity overhead.

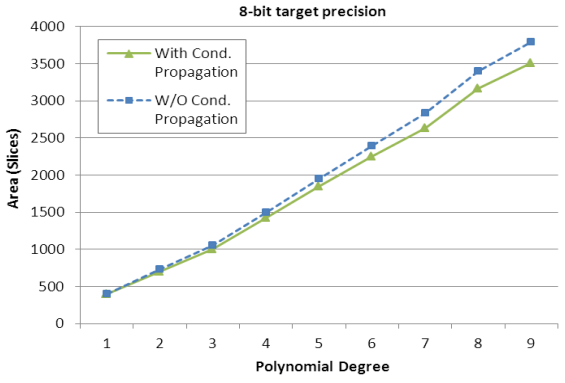

Fig. 3. Hardware area savings for various polynomial degrees.

7. REFERENCES

[1] L. Zhang, Y. Zhang, and W. Zhou, "Floating-point to fixed-point transformation using extreme value theory," in *IEEE/ACIS International Conference on Computer and Information Science*, 2009, pp. 271-276.

[2] D. Boland and G. A. Constantinides, "Bounding variable values and round-off effects using Handelman representations," *IEEE Trans. Comput.-Aided Des. Integr. Circuits and Syst.,* vol. 30, pp. 1691-1704, 2011.

[3] C. F. Fang, R. A. Rutenbar, and T. Chen, "Fast, accurate static analysis for fixed-point finite-precision effects in DSP designs," in *Proc. IEEE/ACM conf. Comput.-aided design (ICCAD'03)*, 2003, pp. 275-282.

[4] R. E. Moore and F. Bierbaum, *Methods and applications of interval analysis*: SIAM, Philadelphia, 1979.

[5] J. Cong*, et al.*, "Evaluation of static analysis techniques for fixed-point precision optimization," in *Proc. IEEE Symposium on Field Programmable Custom Computing Machines*, 2009, pp. 231-234.

[6] D. U. Lee, A. A. Gaffar, R. C. C. Cheung, O. Mencer, W. Luk, and G. A. Constantinides, "Accuracy-guaranteed bit-width optimization," *IEEE Trans. Comput.-Aided Des. Integr. Circuits and Syst.,* vol. 25, pp. 1990-2000, 2006.

[7] J. A. Lopez, C. Carreras, and O. Nieto-Taladriz, "Improved interval-based characterization of fixed-point LTI systems with feedback loops," *IEEE Trans. Comput.-Aided Des. Integr. Circuits and Syst.,* vol. 26, pp. 1923-1933, 2007.

[8] W. Osborne, R. Cheung, J. Coutinho, W. Luk, and O. Mencer, "Automatic accuracy-guaranteed bit-width optimization for fixed and floating-point systems," in *Proc. Conf. Field Programmable Logic and Applications*, 2007, pp. 617-620.

[9] Y. Pu and Y. Ha, "An automated, efficient and static bit-width optimization methodology towards maximum bit-width-to-error tradeoff with affine arithmetic model," in *Proc. Asia and South Pacific Design Automation Conference (ASPDAC)*, 2006, pp. 886-891.