IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Approximate Multiplier by Partial Product Preforation

Miss. Sasikala. V.P[1], Mr. R. Dharmalingam[2]

ME VLSI Department, Maharaja Institute of Technology, Coimbatore[1]

ME VLSI, Head of the Department, Maharaja Institute of Technology, Coimbatore[2]

Abstract: Approximate computing appear as a promising solution to reduce their power dissipation. Such applications process large redundant data sets or noisy input data derived from the real world, do not have a golden result, perform statistical/probabilistic computations, and/or demand human interaction, thus their exactness is relaxed due to limited human perception. Approximate computing can be applied at both software and hardware levels. Hardware-level approximation mainly targets arithmetic units, such as adders and multipliers, widely used in portable devices to implement multimedia algorithms, e.g., image and video processing. Partial product generation, we introduce the partial product preforation method for creating approximate multipliers. Inspired from, we omit the generation of some partial products, thus reducing the number of partial products that have to be accumulated, we decrease the area, power, and depth of the accumulation tree.

Keywords: Approximate arithmetic circuits, approximate computing, approximate multiplier, error analysis, low power, Integrated synthesis environment.

I. INTRODUCTION

Approximate computing has emerged as a potential solution for the design of energy-efficient digital systems. Uses such as multimedia, identification and data mining are inherently error-tolerant and do not require a perfect accuracy in calculation. For these uses, approximate circuits may play an important role as a promising alternative for decreasing area, power and delay in digital systems that can tolerate some loss of accuracy, thereby achieving better performance in energy efficiency. As one of the key components in arithmetic circuits, adders have been extensively studied for approximate implementation New methodologies to model, analyze and evaluate the approximate adders have been discussed. However, there has been relatively less effort in the architecture of approximate multipliers. A multiplier usually consists of 3 stages: partial product execution, partial product accumulation and a carry propagation adder (CPA) at the final stage. Considers using approximate adders to generate the radix-8 Booth encoding 3x with error reduction. In approximate partial products are computed using inaccurate 2 × 2 multiplier blocks, while exact adders are used in an adder tree to accumulate the approximate partial products. Briefly discusses the use of approximate speculative adders for the final stage addition in a multiplier. The error tolerant multiplier (ETM) of is based on the truncation of a multiplier into an accurate multiplication part for MSBs and a non-amplification part for LSBs. In this paper, a novel approximate multiplier design is proposed using a simple, yet fast approximate adder. This newly architectured adder can process data in parallel by cutting the carry breeding chain (and thus, introducing an error). It has a critical path delay that is even shorter than a conventional one-bit full adder. Albeit concurring a high error rate, this adder simultaneously computes the sum and generates an error signal; this feature is worked to reduce the error in the final result of the multiplier. In the proposed approximate multiplier, a simple tree of the relative adders is used for partial product accumulation and the error signals are used to refund the error for obtaining a better accuracy. Compared to the traditional (exact) Wallace and Dadda trees, the proposed multiplier has a significantly shorter critical path as well as a reduced circuit complexity.

II.INTRODUCTION ABOUT MULTIPLIERS

Multipliers play an important role in today's digital signal processing and various other applications. With advances in technology, many developers have tried and are trying to design multipliers which offer either of the following architecture targets – high speed, low power consumption, regularity of layout and hence less area or even aggregate of them in one multiplier thus making them suitable for various high speed, low power and compact VLSI implementation. The common compounding method is "add and shift" algorithm. In parallel multipliers number of partial products to be additive is the main parameter that determines the performance of the multiplier. To reduce the number of partial products to be additional, Modified Booth algorithm is one of the most popular algorithms. To obtain speed improvements Wallace Tree algorithm can be application wise is to reduce the number of sequential adding

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

stages. Further by aggregating both Modified Booth algorithm and Wallace Tree technique we can see advantage of both algorithms in one multiplier. However with improved parallelism, the amount of shifts between the partial products and intermediate sums to be addition will increase which may result in decreased speed, increase in silicon area due to irregularity of structure and also increased power decayed due to increase in interconnect resulting from complex routing. On the other hand "serial-parallel" multipliers compromise speed to obtain better performance for area and power consumption. The selection of a parallel or serial multiplier actually builds on the nature of application. In this lecture we introduce the multiplication algorithms and architecture and analyze them in terms of speed, area, power and combination of these metrics.

III. ARRAY MULTIPLIER

The configuration of an array multiplier is shown in Figure. There is a one-to-one topological correspondence between this hardware structure The execution of **N** partial products requires **N x M** two-bit AND gates most of the places of the multiplier is ardent to the adding of the N partial products, which requires **N** -1 M-bit adders. The shifting of the partial products for their proper adjustment is performed by simple routing and does not require any logic. The overall structure can easily be compressinto a rectangle, resulting in a very efficient layout. Due to the array organization, determining the propagation stoppage of this circuit is not straightforward. Consider the implementation of the limited sum adders are implemented as ripple-carry structures. Operation optimizationrequires that the critical timing way is to be identified first. This turns out to be nontrivial. In fact, a huge number of paths of almost identical length can be identified. A closer look at those critical ways yields an approximate expression for the propagation delay.

$$t_{mult} = [(M-1) + (N-2)]t_{carry} + (N-1)t_{sum} + t_{and}$$

where $\mathbf{t_{carry}}$is the propagation stoppage between input and output carry, $\mathbf{t_{sum}}$is the delay between the input carry and sum bit of the full adder, and $\mathbf{t_{and}}$is the stoppage of the AND gate. Since all critical paths have the same length, speeding up just one of them—for instance, by restoration one adder by a faster one such as a carry-select adder—does not make much feel from a design standpoint. AH critical ways have to be attacked at the same time. From the above equation, it can be deduct that the minimization of $\mathbf{t_{mult}}$wanted the minimization of both $\mathbf{t_{carry}}$.

IV.DADDA MULTIPLIER

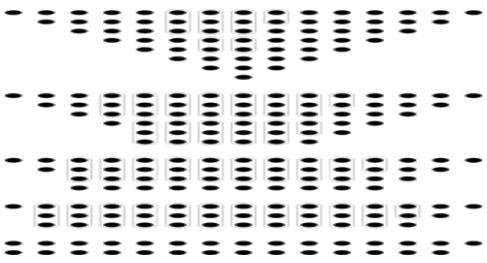
The **Dadda multiplier** is a hardware multiplier architecture invented by computer scientist Luigi Dadda in 1965. It is similar to the Wallace multiplier, but it is slightly higher (for all operand sizes) and requires fewer gates (for all but the smallest operand sizes). In fact, Dadda and Wallace multipliers have the same 3procedures consists:

1. Multiply (logical AND) each bit of one of the altercation, by each bit of the other, yielding n^2results. Depending on position of the multiplied bits, the wires ferry different weights, for example wire of bit result of $a_2 b_3$is 32.
2. Reduce the number of partial multiplications to two layers of full and half adders.
3. Merge the wires in 2 numbers, and add them with a conventional adder.

However, dissimilar Wallace multipliers that reducts as much as possible on each layer, Dadda multipliers do as few contractions as possible. Because of this, Dadda amplifiers have a less expensive reduction phase, but the numbers may be a few bits longer, thus wanted slightly bigger adders.To achieve this, the structure of the second step is governed by slightly more complicated rules than in the Wallace tree. As in the Wallace tree, a new layer is added if any weight is carried by three or more wires. The reducing rules for the Dadda tree, however, are as follows:

- Take any three wires with the similar weights and input them into a full adder. The reaction will be an output wire of the same weight and an output wire with a larger weight for each three input wires.
- If there are two wires of the same weight left, and the modern number of output wires with that weight is balanced to 2 (modulo 3), input them into a half adder. Otherwise, pass them through to the next layer.
- If there is just one wire larboard, connect it to the next layer.

DADDA TREE

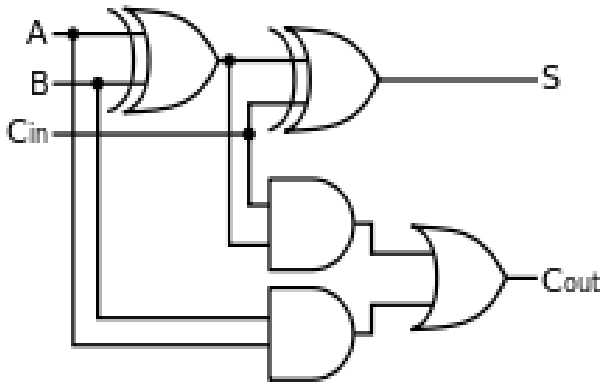


IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

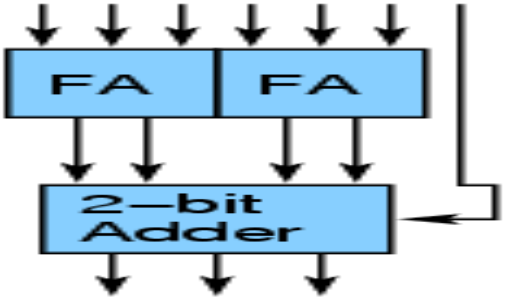
EXAMPLE OF DADDA REDUCTION ON 8X8 MULTIPLIER



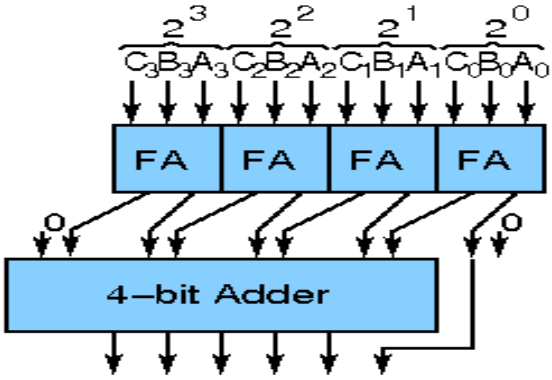
VI. DESIGN OF WALLACE TREE ADDERS

There are many cases where it is desired to add more than two numbers together. The direct way of adding together m numbers (all n bits wide) is to add the first two, then add that sum to the next, and so on. This wanteds a total of m − 1 additions, for a total gate delay of O(m lg n) (over bearing lookahead carry adders). Instead, a tree of adders can be formed, taking only O(lg m · lg n) gate delays.

A Wallace tree adder auditioned together n bits to produce a sum of log2n bits. The design of a Wallace tree adder to add seven bits (W7) is adorned below:



Wallace multiplier is an efficient parallel multiplier. In the current Wallace tree multiplier, the first step is to form unfinished product array (of N2 bits). In the second step, groups of three nearest rows each, is collected.
An adder tree to add three 4-bit numbers is shown below:



Each group of three rows is redacted by using full adders and half adders. Full adders are used in each column where there are three bits whereas half adders are applicationwise in each column where there are two bits. Any single bit in a column is passed to the next level in the same column without processing. This reduction procedure is repeated in each successive stage until only two rows balance. In the final step, the balance two rows are added using a carry propagating adder.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

A Reduced Complexity Wallace Multiplier Reduction Technique

Waters et al. presented reduced complexity Wallace multiplier reduction approach. It is a modification to the second phase reduction method used in the conventional Wallace multipliers, in which number of the half adders is greatly reduction. In the first phase, the partial product array is formed and it is transferred in the form of an inverted pyramid array. An inverted pyramid array is formed when the bits in the left half of the partial multiplication array is shifted in the upward direction.This array is divided into group of three rows every full adders are used in each column. Half adders are used only when the number of reduction stages of the modified Wallace multiplier is overflowed that of the conventional Wallace multiplier.

Both multipliers yield same measurements in the terms of delay and have same number of the reduction stages, but the modified Wallace multiplier has the advantage of reduction complexity as number of half adders is 80% less than the conventional Wallace multiplier in the second phase. However due to reducing in number of half adders, the total gate count in modified Wallace reducing is always less than that of the ordinary Wallace reduction. The number of full adders is somewhat increased between 1-5 for 8-64 bit modified Wallace multiplier.

The Wallace tree has three steps:

- Multiply (that is - AND) each bit of one of the arguments, by every bit of the other, passive results. Depending on position of the multiplied bits, the wires carry different weights.
- Reductioning the number of partial products to two by layers of full and half adders.
- Group the wires in two numbers, and add them with aordinary adder.

As long as there are three or more wires with the same weight add following flows:

- Take any 3 wires with the same weights¬ and input them into a full adder. The conclude will be an output wire of the same weight and an output wire with a higher weight for each three input wires.
- If there are 2 wires of the similar weight¬ left, input them into a half adder.
- If there is just one wire left, connect it to¬ the next layer.

V.PROPOSED APPROACHES-SQUARE-ROOT CARRY SELECT ADDER:

The square-root carry select adder is build up by equalizing the delay through two carry chains and the block multiplexer signal from previous stage. It is also termed as non-linear carry select adder. The previouschanges maded SQRT CSLA uses Binary to Excess-1 Converter (BEC) instead of RCA with Cin=1 in the regular CSLA to obtain lower delay with slightly increase in area. The basic idea of the proposed architecture is that which replaces the BEC logic by same Boolean Logic. The proposed architecture generates a duplicate sum and carry-out signal by using NOT and OR gate and exclusive value with the help of multiplexer. The multiplexer is used to select the correct output coordinating to its previously carry-out signal

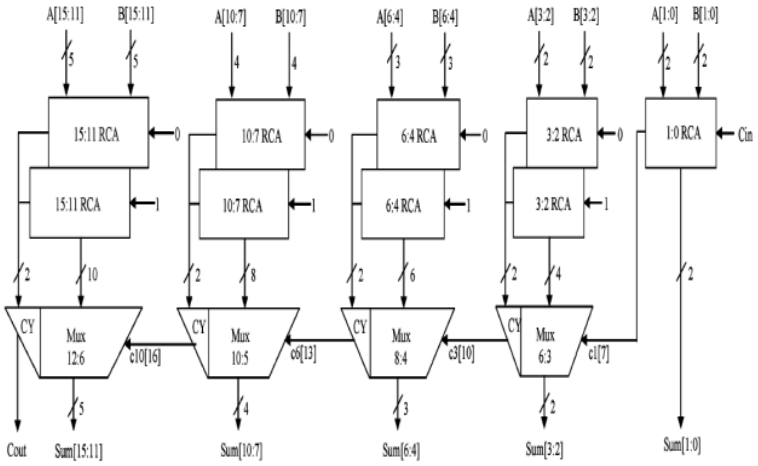
The modified 16-bit SQRT CSLA using BEC is shown in Fig. 4. The architecture is again divided into five groups with different sizes of Ripple carry adder and BEC. The groupie, group III , group IV and group Vof 16-bit SQRT CSLA are shown. The parallel Ripple carry adder with Cin=1 is recovered with BEC. One input to the multiplexer goes from the RCA with Cin=0 and other input from BEC. Comparing the individual groups of both daily and changed SQRT CSLA, it is clear that the BEC structure reduces delay.

In proposed architecture, an area-able to carry select adder by sharing the common Boolean logic term to remove the duplicated adder cells in the ordinary carry select adder is shown in this way, it saves many transistor counts andobtains a low power.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017



Through searching the truth table of a single bit full adder, to find out the output of summation signal as carry-in signal is logic '0' is the converse signal of itself as carry-in signal is logic '1'. By sharing the common Boolean logic term in addition generation, a proposed carry select adder design is illustrated in figure.

VI.CONCLUSION

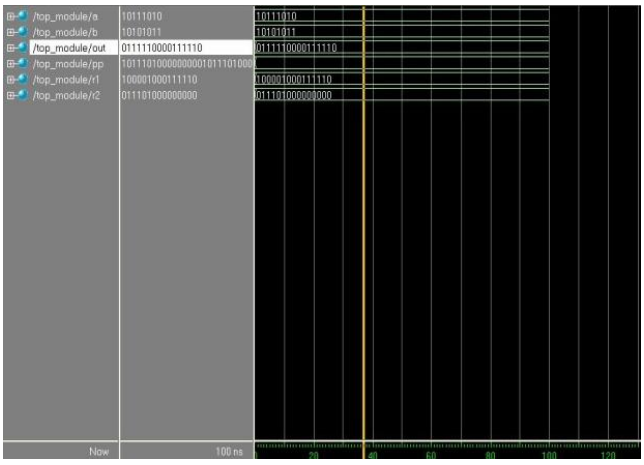
Power, delay and area are the combining factors in VLSI design that limits the performance of any circuit. This work presents a simple technique to reduce the area, delay and power of a multiplier using CSLA architecture. Several popular and well-known schemes, with the objective of improving the speed of the parallel multiplier, have been developed in past. This proposed modified Wallace and Dadda multipliers are introduced a very important iterative realization of parallel multiplier. This advantage becomes more pronounced for multipliers of bigger than 16 bits. In this way, the transistor calculation of proposed.Therefore Wallace and Dadda multipliers are reduced having less area and low power which makes it simple and capable for VLSI hardware implementations.

VII. SIMULATION RESULTS AND ITS DESCRIPTION

This work has been developed using Xilinx tool it shows the comparison between the various adders likeordinary CSLA, Modified CSLA, regular SQRT CSLA, Modified SQRT CSLA and exposed SQRT CSLA for 8-bit,16-bit, 32-bit and 64-bit. The parameters on which they arecorrelated are area, delay and power. Fig. 10 depicts that theproposed SQRT CSLA has less number of gates and sinceits aless area. Fig. shows the adder circuit for delaycomparison. The results compared in Fig. 12 shows that thepower consumption of proposed SQRT CSLA is reduced. Itis clear that power, area and delay of proposed SQRT CSLAfor 8-bit, 16-bit, 32-bit and 64-bit is reduced as compared toother adders.

Array multiplier Accurate and Approximate Results

Accurate Results

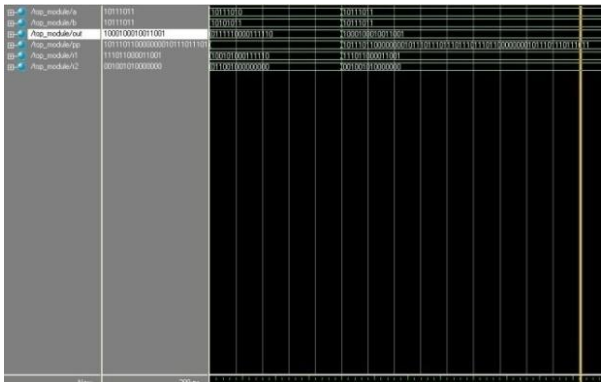


IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

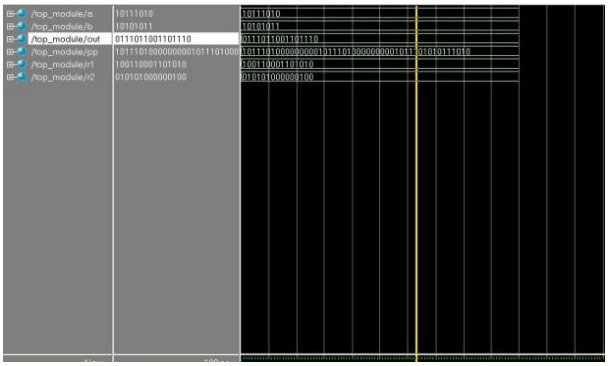
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Approximate Results

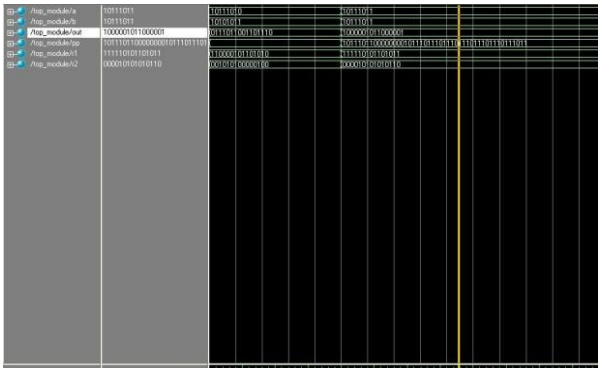


Dadda Multiplier Modified Accurate and Approximate Results

AccurateResults

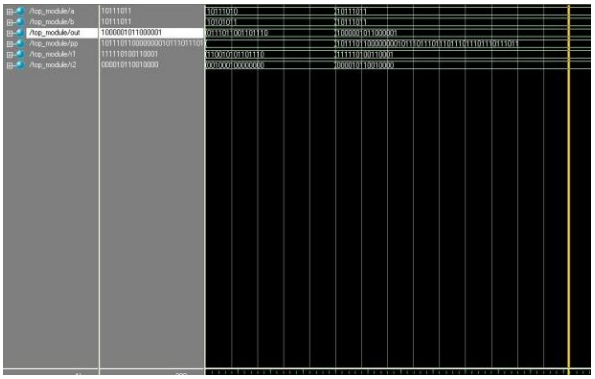


Approximate Results



Wallace Modification Accurate and Approximate Results

Accurate Results



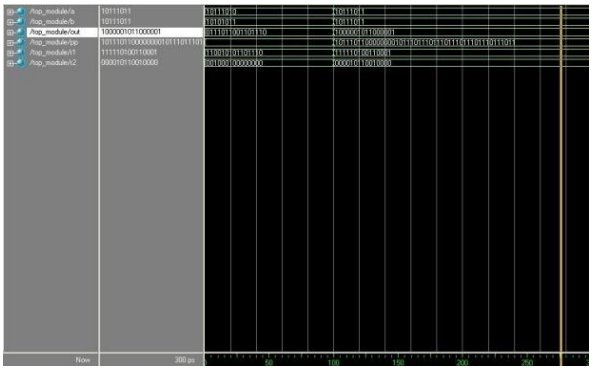
DOI10.17148/IJARCCE.2017.6577

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 6, Issue 5, May 2017

Approximate Results



REFERENCES

[1] V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Analysisand characterization of inherent application resilience for approximatecomputing," in Proc. 50th ACM/EDAC/IEEE Design Autom. Conf.,May/Jun. 2013, pp. 1–9.
[2] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, "MACACO:Modeling and analysis of circuits for approximate computing," in Proc.IEEE/ACM Int. Conf. Comput.-Aided Design, Nov. 2011, pp. 667–673.
[3] S. T. Chakradhar and A. Raghunathan, "Best-effort computing:Re-thinking parallel software and hardware," in Proc. 47th ACM/IEEEDesign Autom. Conf., Jun. 2010, pp. 865–870.
[4] E. J. King and E. E. Swartzlander, "Data-dependent truncation scheme for parallel multipliers," in Proc. Conf. Rec. 31st Asilomar Conf. Signals,Syst. Comput., Nov. 1998, pp. 1178–1182.
[5] M. J. Schulte and E. E. Swartzlander, "Truncated multiplication with correction constant," in Proc. 6th VLSI Signal Process., Oct. 1993,pp. 388–396.
[6] Y. Liu, T. Zhang, and K. K. Parhi, "Computation error analysis indigital signal processing systems with overscaled supply voltage," IEEETrans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 4, pp. 517–526, Apr. 2010.
[7] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy,"IMPACT: Imprecise adders for low-power approximate computing," inProc. Int. Symp. Low Power Electron. Design, Aug. 2011, pp. 409–414.
[8] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in Proc. 24th Annu. Conf. VLSI Design, Jan. 2011, pp. 346–351.
[9] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design andanalysis of approximate compressors for multiplication," IEEE Trans.Comput., vol. 64, no. 4, pp. 984–994, Apr. 2015.
[10] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong, "Design oflow-power high-speed truncation-error-tolerant adder and its applicationin digital signal processing," IEEE Trans. Very Large Scale Integr. (VLSI)Syst., vol. 18, no. 8, pp. 1225–1229, Aug. 2010.
[11] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in Proc. Design,Autom. Test Eur., Mar. 2008, pp. 1250–1255.
[12] S. Banescu, F. de Dinechin, B. Pasca, and R. Tudoran, "Multipliers forfloating-point double precision and beyond on FPGAs," ACM SIGARCHComput. Archit. News, vol. 38, no. 4, pp. 73–79, Jan. 2011.
[13] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performanceapproximate multiplier with configurable partial error recovery," in Proc.Conf. Design, Autom. Test Eur., Mar. 2014, Art. no. 95.
[14] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard,"Managing performance vs. accuracy trade-offs with loop perforation,"in Proc. 19th ACM SIGSOFT Symp., 13th Eur. Conf. Found. Softw.Eng. (ESEC/FSE), Sep. 2011, pp. 124–134.
[15] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," IEEE Trans. Comput., vol. 62,no. 9, pp. 1760–1771, Jun. 2012.
[16] A. Lingamneni, C. Enz, K. Palem, and C. Piguet, "Synthesizingparsimonious inexact circuits through probabilistic design techniques,"ACM Trans. Embedded Comput. Syst., vol. 12, no. 2S, May 2013, Art. no. 93.
[17] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N. S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," IEEE Trans. Very Large ScaleIntegr. (VLSI) Syst., vol. 23, no. 6, pp. 1180–1184, Jun. 2015.
[18] S. Hashemi, R. I. Bahar, and S. Reda, "DRUM: A dynamic rangeunbiased multiplier for approximate applications," in Proc. IEEE/ACMInt. Conf. Comput.-Aided Design, Nov. 2015, pp. 418–425.
[19] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs.New York, NY, USA: Oxford Univ. Press, 2000.
[20] C. Lee, M. Potkonjak, and W. H. Mangione-Smith, "MediaBench:A tool for evaluating and synthesizing multimedia and communicationssystems," in Proc. 13th Annu. IEEE/ACM Int. Symp. Microarchitecture,Dec. 1997, pp. 330–335.
[21] D. Zuras and W. H. McAllister, "Balanced delay trees and combinatorial division in VLSI," IEEE J. Solid-State Circuits, vol. 21, no. 5,pp. 814–819, Oct. 1986.
[22] L. Dadda, "Some schemes for parallel multipliers," Alta Frequenza, vol. 34, no. 5, pp. 349–356, Mar. 1965.
[23] B. Jose and D. Radhakrishnan, "Delay optimized redundant binaryadders," in Proc. 13th IEEE Int. Conf. Electron., Circuits Syst. (ICECS),Dec. 2006, pp. 514–517.