

# 几种常见近似加法器的比较

赵磊

成都信息工程大学工程实践中心, 四川 成都

Email: zhaolei@cuit.edu.cn

收稿日期: 2021年7月1日; 录用日期: 2021年7月31日; 发布日期: 2021年8月10日

## 摘要

加法器作为重要的算术模块, 在决定运算速度和功耗方面起着关键作用。对运算速度和效率的需求以及一些应用的容错特性促进了近似加法器的发展。传统加法器一般采用精确加法运算, 电路面积、功耗均较大。近年出现了一种新兴的电路设计方法——近似加法计算, 通过简化电路适当降低计算精度, 最终实现面积、功耗、延时与精度的折中。本文比较了目前国内外主流的近似加法器设计, 并在误差和电路特性方面进行了比较评估。仿真结果表明, LOA由于完全利用逻辑或门进行低位运算, 面积最小, 功耗也最小, 但未考虑精度的问题, 错误率最高; ETAIL和ACA面积比LOA稍大, 功耗也相应增加, 并且设计时考虑了精度, 使得错误率降低; ACA在延时方面优势最突出; SCSA配置了窗口加法器, 面积与功耗更大, 这也使其精度得到了更大的提升。

## 关键词

近似加法器, 容错, 进位跳跃, Verilog HDL语言

# Comparison of Several Common Approximate Adders

Lei Zhao

Electronic Experimental Center, Chengdu University of Information Technology, Chengdu Sichuan

Email: zhaolei@cuit.edu.cn

Received: Jul. 1<sup>st</sup>, 2021; accepted: Jul. 31<sup>st</sup>, 2021; published: Aug. 10<sup>th</sup>, 2021

## Abstract

As an important arithmetic module, the adder plays a key role in determining the operation speed and power consumption. The demand for computing speed and efficiency and the fault tolerance

文章引用: 赵磊. 几种常见近似加法器的比较[J]. 电路与系统, 2021, 10(3): 15-23.

DOI: 10.12677/ojcs.2021.103003

of some applications have promoted the development of approximate adders. Traditional adders generally use precise addition operations, and the circuit area and power consumption are relatively large. In recent years, a new circuit design method—approximate addition calculation, has appeared. By simplifying the circuit, the calculation accuracy is appropriately reduced, and finally the area, power consumption, delay and accuracy are compromised. This article compares the current mainstream approximate adder designs at home and abroad, and compares and evaluates the errors and circuit characteristics. The simulation results show that the LOA has the smallest area and the lowest power consumption due to the full use of logic or gates for low-bit operations, but the accuracy is not considered, and the error rate is the highest; ETAIL and ACA have a slightly larger area than LOA, and the power consumption increases accordingly, and accuracy is considered in the design, which reduces the error rate; ACA has the most prominent advantage in terms of delay; SCSA is equipped with a window adder, which has a larger area and power consumption, which also improves its accuracy.

## Keywords

Approximate Adder, Fault Tolerant, Carry Jump, Verilog HDL Language

Copyright © 2021 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

## 1. 引言

虽然在计算领域一直存在计算误差,但多媒体、无线通信、识别和数据挖掘等应用程序可以允许一些误差的发生[1] [2] [3]。由于人类感知的局限性,这些错误在图像、音频和视频处理等应用程序中并没有明显的区别[4]。在数字信号处理系统中,来自外界的输入都是带有噪声的,因此计算结果的精度有限。许多应用程序都是基于统计计算或概率计算的,如分类和识别算法[5]。由于这些应用程序的性质,计算中的微小错误不会导致显著的性能下降[6]。因此,近似计算适用于许多能够允许一定精度损失[7] [8]。

随着 CMOS 的物理尺寸缩小到几十纳米,提高电路性能和提高功率已经越来越困难[9]。近似计算被提倡作为一种节省面积和功耗的新方法,并在精度的有限损失下提高性能[10] [11]。虽然有计算误差,但诸如多媒体(图像、音频和视频)处理、无线通信、识别和数据挖掘等应用程序对某些误差是可以接受的[12] [13] [14] [15]。由于这些应用程序的统计概率性质,计算中的微小误差不会导致性能明显下降。近似计算作为一种以合理的速度和功耗实现复杂计算的潜在技术,从电路到编程语言都得到了积极的探索[16] [17]。在电路设计中,加法器和乘数器一直是一个焦点,因为它们在决定许多计算密集型应用的性能和功耗方面发挥着关键作用[18]。过去对近似计算的研究已经从电路设计扩展到编程语言。Kelly 等的研究中,提出了一种近似平方电路,这是一种新的逻辑合成方法,以减少给定误差率阈值下合成电路的面积[19]。在 Chippa 等人的研究中,提出了一种提出了一种可扩展的设计方法来实现高效硬件,用于容错应用[20]。还有学者提出了基于笛卡尔遗传规划的近似数字电路设计的自动设计过程[21] [22]。Sampson 等开发了 EnerJ 语言,这是对 Java 的一个扩展,此语言支持用于低功耗计算的近似数据类型[23]。目前各种计算和内存架构已被提出,用于支持近似计算应用程序[24] [25]。近年来,一些学者采用蒙特卡罗模拟来获取数据以进行分析,使用误差率(ER)、误差距离(ED)和平均误差来评估近似设计的误差特征,提出了评价近似加法器的设计指标和分析方法[26]。具体为评价与硬件相关的数据,包括关键路径延迟、电路面积和功

耗, 以及包括功率延迟产品(PDP)和区域延迟产品(ADP)在内的复合指标, 来评估这些设计的电路特性。

本文对精确加法器和几种近似加法器的逻辑设计、电路原理等进行了比较和评估。近似计算在多媒体处理、模式识别、机器学习和数据挖掘等应用中对误差的可容错性有很大区别, 本文对近似计算的巨大的能效和性能改进的潜力进行说明和比较, 为更好地在不同场景使用近似加法器提供参考。

## 2. 精确加法器介绍

一位二进制全加器(Full Adder, FA)是加法器的基本组成单元, 能将本位的两个二进制数和邻低位来的进位数进行相加[27]。假设  $A, B$  分别为两个多位二进制数,  $A_i$  与  $B_i$  分别表示两个数的其中某一位, 他们作为 FA 的两输入,  $C_{i-1}$  表示前一位的进位,  $S_i$  表示本位和,  $C_i$  表示进位输出。那么 FA 的输出可用下列公式计算:

$$S_i = A_i \oplus B_i \oplus C_{i-1} \quad (1)$$

$$C_i = (A_i \oplus B_i)C_{i-1} + A_i B_i \quad (2)$$

如果用  $G_i$  和  $P_i$  表示每一位的进位产生信号和进位传递信号, 他们可用式表示:

$$G_i = A \cdot B \quad (3)$$

$$P_i = A + B \quad (4)$$

这样,  $S_i$  和  $C_i$  可由  $G_i$  和  $P_i$  表示:

$$S_i = P_i \oplus C_{i-1} \quad (5)$$

$$C_{o,i} = G_i + P_i \cdot C_{i-1} \quad (6)$$

由二进制全加器 FA 可以构成精确加法器。通常有两种形式: 串行进位加法(Ripple Carry Adder, RCA)和超前进位加法(Carry Look-ahead Adder, CLA)。串行进位加法 RCA 的原理是: 通过将  $N$  个 FA 串联来进行  $N$  位的加法运算。低位的进位输出 CO 依次加到相邻高位的进位输入端 CI。假设  $N = 4$ , 则四位 RCA 电路结构如图 1 所示:

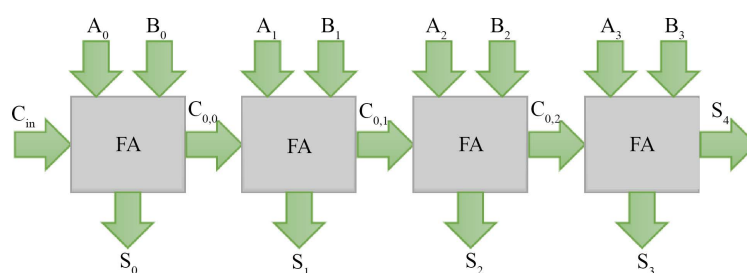


Figure 1. Circuit structure diagram of Ripple Carry Adder (RCA)  
图 1. RCA 电路结构

从图 1 中可看出, RCA 的特点是电路简单, 但由于高位必须等到低位运算完成才能进行, 电路产生的延时很大, 运算速度较慢。超前进位加法 CLA 的原理是: 提前计算每一位的进位输出。将由式(5)、(6)得到的每一位的进位输出带入下一位的进位表达式里并以此类推, 可迭代出每一位的进位输出, 只与第一位的进位信号和各个加数相关, 以此消除对前一级进位信号的依赖, 减小电路的最大延时, 提高运算速度。一个四位超前进位加法器的原理如图 2 所示:

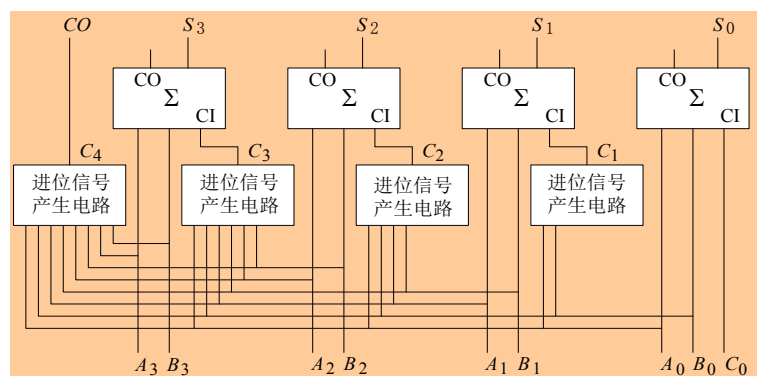


Figure 2. Schematic diagram of Carry Look-Ahead Adder (CLA)  
图 2. 超前进位加法器原理图

由图 2 可看出, 由于高位的进位信号产生跟前面各位的加数都相关, CLA 的面积会很大, 位数达到一定值以后, CLA 的面积会增加到严重影响运算性能的程度。为提高加法器性能, 获得速度与面积的平衡, 可采用近似计算的方法来设计新型加法器。近似计算是一种新兴的电路设计方法, 其计算结果是带有误差的近似值, 这是通过牺牲一定精度但不影响总体结果的方式。电路结构简化, 从而提高电路性能, 降低电路功耗, 节约电路面积。

### 3. 不同近似加法器比较

#### 3.1. 低位或门加法器 LOA (Low-Part-OR Adder, LOA)

这种加法器利用或门来计算低位和, 得到的是近似值。同时利用精确加法器来计算高位和。为提高计算精度, 低位最高两位通过与门产生进位信号并传递给高位精确部分[28]。由于这种方法低位计算结果为近似值, 得到的结果错误率很高。并且低位的位宽越大, 错误率越高。假设两个输入信号 IN1 和 IN2 低位位宽均为  $n$ , 高位位宽  $h$ , 低位位宽  $l$ , 输出为 OUT, 那么 LOA 电路结构如图 3 所示:

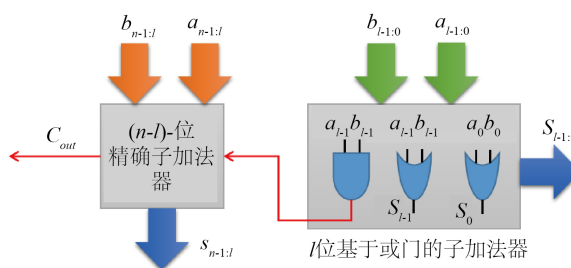


Figure 3. Circuit diagram of Low-Part-OR Adder (LOA)  
图 3. LOA 电路结构图

LOA 的关键路径为图中的  $C_{in}$  进位, 记为  $O$ , 其值为  $\log(n-1)$ 。

#### 3.2. 容错加法器(Error-Tolerant Adder, ETA)

这种加法器同样是高位精确计算, 低位近似处理。最初的 ETA 是利用修改后的逻辑异或门来计算低位和, 用精确加法器计算高位和。这种算法同样错误率较高。后来人们提出改进方法, 利用分块的思想, 将整个电路结构分成若干个子加法器模块, 这样可以将整条进位传输路径截断成较短路径, 减少电路的延时与动态功耗。改进的 ETAlI 由进位发生器、加数和发生器组成, 低位的进位发生器产生的进位信号

传递到相邻高位的加数和发生器。ETAII 的结构原理图如图 4 所示。

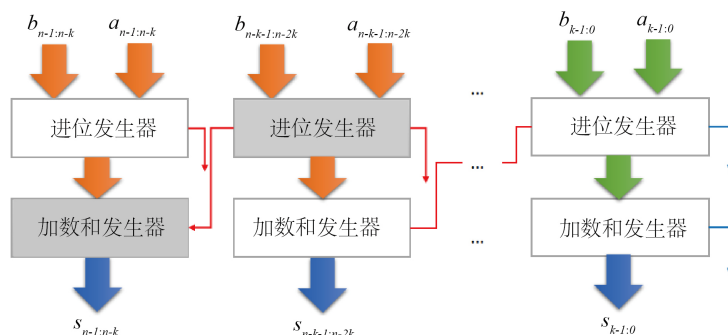


Figure 4. Circuit diagram of Error-Tolerant Adder II (ETAII)  
图 4. ETAII 电路结构图

由图 4 可看出, ETAII 电路内部存在进位预测, 比单纯的分模块计算加数和更加准确, 但同时电路也比较复杂, 并且由于进位路径较长, 延迟较大。它的延迟  $O = \log(2k)$ , 其中  $k$  为每个子加法器的位宽。

### 3.3. 精度可配置加法器(Accuracy Configurable Adder, ACA)

这种加法器可以通过改变电路结构来配置精度, 因此可以实现精度与性能和功耗之间的平衡[29]。以 16 位加法为例, ACA 的结构示意图如图 5 所示:

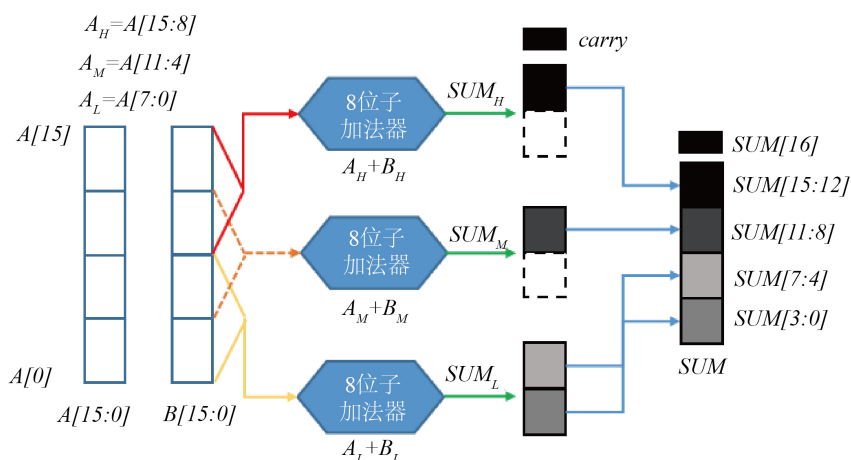


Figure 5. Circuit diagram of Accuracy Configurable Adder (ACA)  
图 5. ACA 电路图

根据图 5, 在 16 位 ACA 中, 由三个 8 位子加法器模块分别产生部分求和的结果。子模块内部高四位计算结果, 低四位计算进位输出。引入中间加法器(AM + BM)以提高精度, 若没有中间加法器, 当第八位进位为 1 时计算会出现错误。对于随机输入模式, 错误率为 50.1%, 而引入中间加法器后, 随机输入模式的计算错误率降低到 5.5%。将 ACA 扩展至更一般的情况, 当输入数据位宽为  $N$ , 子模块位宽为  $2k$  时, ACA 的实现框图如图 6 所示。

根据图 6,  $k$  值增大时, 电路延时增大, ACA 的精度提高。而  $k$  值减小时, 电路延时也随之减小, 但 ACA 的计算错误率会上升。它的延迟  $O = \log(2k)$ ,  $k$  为每个子加法器的位宽。

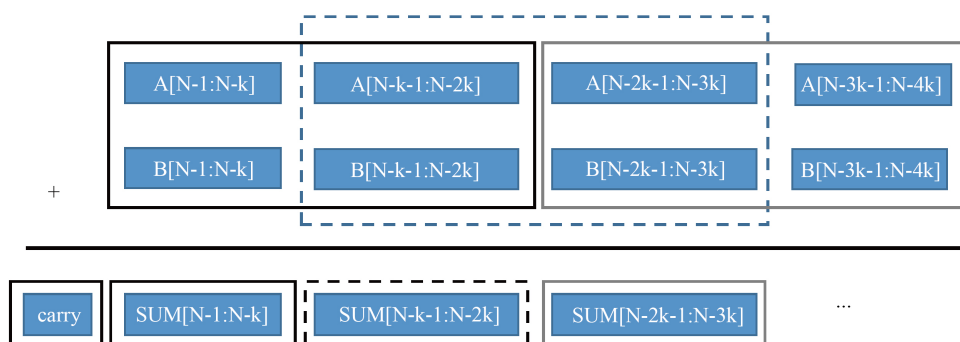


Figure 6. Schematic diagram of Accuracy Configurable Adder (ACA)

图 6. ACA 结构图

### 3.4. 进位预测选择加法器(Speculative Carry Select Addition, SCSA)

SCSA 将加法器分为多个子模块，每个子模块通过配置窗口加法器来进行结果的预测以及进位的选择[30]。窗口加法器首先通过两个子加法器提前计算进位分别为“1”和“0”时的结果，然后根据上一级的进位输出进行选择。SCSA 的窗口加法器电路结构如图 7 所示：

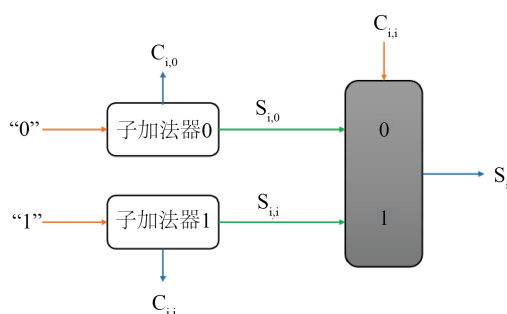


Figure 7. Schematic diagram of Speculative Carry Select Addition (SCSA)

图 7. 进位预测选择加法器电路结构图

窗口加法器的配置提升了 SCSA 的计算正确率，但电路面积和功耗的开销也以此而增大。它的路径延迟由两部分组成，一部分是子加法器的延迟  $O = \log(k)$ ，另一部分是由数据选择器产生的延迟。

### 3.5. 进位跳跃加法器(Carry-Skip Adder, CSA)

CSA 也利用了模块化的思想，通过子模块来产生进位与部分和[31]。此外，CSA 利用了一种进位跳跃机制，其原理图如图 8 所示。

根据图 8，CSA 的进位跳跃机制具体为：在计算第  $(i+1)$  个子模块的进位输入信号时，如果第  $(i)$  个子模块的进位传递信号为“1”，那么将第  $(i-1)$  个子模块的进位输出作为该进位输入的结果，否则将第  $(i)$  个子模块的进位输出作为该进位输入的结果。该近似加法器与 SCSA 相似，通过配置额外的结构降低错误率，减少延时，但电路面积与功耗也因此而增加。它的关键路径延迟  $O = \log(2k)$ 。

## 4. 不同近似加法器评估比较

为评估电路性能，将所提出的近似加法器用 Verilog HDL 语言实现，并利用 28 纳米的标准单元库在 Design Compiler 下进行了电路综合。对所有设计都采用相同的工艺、电压和温度，采用相同的优化选项。





增加。同时由于这两种设计考虑了精度，使得错误率大大降低。ACA 在延时方面优势最突出，因为其进位路径短，且相对于 ETAI 来说进位链更少。SCSA 配置了窗口加法器，面积与功耗更大，这也使其精度得到了更大的提升。CSA 配置了额外的进位模块，面积是这几种加法器里面最大的，甚至超过 CLA。同时精度也是最高的，已经非常接近精确加法器。

## 参考文献

- [1] Han, J. and Orshansky, M. (2013) Approximate Computing: An Emerging Paradigm for Energy-Efficient Design. 2013 18th IEEE European Test Symposium (ETS), Avignon, 27-30 May 2013, 1-6. <https://doi.org/10.1109/ETS.2013.6569370>
- [2] Xu, Q., Mytkowicz, T. and Kim, N.S. (2015) Approximate Computing: A Survey. *IEEE Design & Test*, **33**, 8-22. <https://doi.org/10.1109/MDAT.2015.2505723>
- [3] Venkataramani, S., Chakradhar, S.T., Roy, K., et al. (2015) Approximate Computing and the Quest for Computing Efficiency. 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC), San Francisco, 7-11 June 2015, 1-6. <https://doi.org/10.1145/2744769.2751163>
- [4] Agrawal, A., Choi, J., Gopalakrishnan, K., et al. (2016) Approximate Computing: Challenges and Opportunities. 2016 IEEE International Conference on Rebooting Computing (ICRC), San Diego, 17-19 October 2016, 1-8. <https://doi.org/10.1109/ICRC.2016.7738674>
- [5] Khudia, D.S., Zamirai, B., Samadi, M., et al. (2015) Rumba: An Online Quality Management System for Approximate Computing. *Proceedings of the 42nd Annual International Symposium on Computer Architecture*, Portland, 13-17 June 2015, 554-566. <https://doi.org/10.1145/2749469.2750371>
- [6] Chippa, V.K., Venkataramani, S., Chakradhar, S.T., et al. (2013) Approximate Computing: An Integrated Hardware Approach. 2013 Asilomar Conference on Signals, Systems and Computers, Pacific Grove, 3-6 November 2013, 111-117. <https://doi.org/10.1109/ACSSC.2013.6810241>
- [7] Mohapatra, D., Chippa, V.K., Raghunathan, A., et al. (2011) Design of Voltage-Scalable Meta-Functions for Approximate Computing. 2011 Design, Automation & Test in Europe IEEE, Grenoble, 14-18 March 2011, 1-6. <https://doi.org/10.1109/DATE.2011.5763154>
- [8] Liu, W., Lombardi, F. and Shulte, M. (2020) A Retrospective and Prospective View of Approximate Computing [Point of View]. *Proceedings of the IEEE*, **108**, 394-399. <https://doi.org/10.1109/JPROC.2020.2975695>
- [9] Yazdanbakhsh, A., Mahajan, D., Esmailzadeh, H., et al. (2016) AxBench: A Multiplatform Benchmark Suite for Approximate Computing. *IEEE Design & Test*, **34**, 60-68. <https://doi.org/10.1109/MDAT.2016.2630270>
- [10] Zhang, Q., Yuan, F., Ye, R., et al. (2014) Approxit: An Approximate Computing Framework for Iterative Methods. *Proceedings of the 51st Annual Design Automation Conference*, San Francisco, 2-5 June 2014, 1-6. <https://doi.org/10.1145/2593069.2593092>
- [11] Vassiliadis, V., Riehme, J., Deussen, J., et al. (2016) Towards Automatic Significance Analysis for Approximate Computing. 2016 IEEE/ACM International Symposium on Code Generation and Optimization (CGO), Barcelona, 12-18 March 2016, 182-193. <https://doi.org/10.1145/2854038.2854058>
- [12] Venkataramani, S., Ranjan, A., Roy, K., et al. (2014) AxNN: Energy-Efficient Neuromorphic Systems Using Approximate Computing. 2014 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), La Jolla, 11-13 August 2014, 27-32. <https://doi.org/10.1145/2627369.2627613>
- [13] Pashaeifar, M., Kamal, M., Afzali-Kusha, A., et al. (2018) Approximate Reverse Carry Propagate Adder for Energy-Efficient DSP Applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, **26**, 2530-2541. <https://doi.org/10.1109/TVLSI.2018.2859939>
- [14] Soares, L.B., Bampi, S. and Costa, E. (2015) Approximate Adder Synthesis for Area- and Energy-Efficient FIR Filters in CMOS VLSI. 2015 IEEE 13th International New Circuits and Systems Conference (NEWCAS), Grenoble, 7-10 June 2015, 1-4. <https://doi.org/10.1109/NEWCAS.2015.7182095>
- [15] Ebrahimi-Azandaryani, F., Akbari, O., Kamal, M., et al. (2019) Block-Based Carry Speculative Approximate Adder for Energy-Efficient Applications. *IEEE Transactions on Circuits and Systems II: Express Briefs*, **67**, 137-141. <https://doi.org/10.1109/TCSII.2019.2901060>
- [16] Lee, J., Seo, H., Kim, Y., et al. (2020) Approximate Adder Design with Simplified Lower-Part Approximation. *IEICE Electronics Express*, **17**, Article ID: 20200218. <https://doi.org/10.1587/elex.17.20200218>
- [17] Ban, T., Wang, B. and Naviner, L. (2018) Design, Synthesis and Application of a Novel Approximate Adder. 2018 IEEE 61st International Midwest Symposium on Circuits and Systems (MWSCAS), Windsor, 5-8 August 2018,



- 488-491. <https://doi.org/10.1109/MWSCAS.2018.8624023>
- [18] Kim, Y., Zhang, Y. and Li, P. (2013) An Energy Efficient Approximate Adder with Carry Skip for Error Resilient Neuromorphic VLSI Systems. 2013 *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Jose, 18-21 November 2013, 130-137. <https://doi.org/10.1109/ICCAD.2013.6691108>
- [19] Temel, M., Slobodova, A. and Hunt, W.A. (2020) Automated and Scalable Verification of Integer Multipliers. In: *International Conference on Computer Aided Verification*, Springer, Cham, 485-507. [https://doi.org/10.1007/978-3-030-53288-8\\_23](https://doi.org/10.1007/978-3-030-53288-8_23)
- [20] Chippa, V.K., Mohapatra, D., Raghunathan, A., *et al.* (2010) Scalable Effort Hardware Design: Exploiting Algorithmic Resilience for Energy Efficiency. *Design Automation Conference IEEE*, Anaheim, 13-18 June 2010, 555-560. <https://doi.org/10.1145/1837274.1837411>
- [21] Vasicek, Z. and Sekanina, L. (2014) Evolutionary Approach to Approximate Digital Circuits Design. *IEEE Transactions on Evolutionary Computation*, **19**, 432-444. <https://doi.org/10.1109/TEVC.2014.2336175>
- [22] Mrazek, V., Sarwar, S.S., Sekanina, L., *et al.* (2016) Design of Power-Efficient Approximate Multipliers for Approximate Artificial Neural Networks. *Proceedings of the 35th International Conference on Computer-Aided Design*, Austin, 7-10 November 2016, 1-7. <https://doi.org/10.1145/2966986.2967021>
- [23] Sampson, A., Dietl, W., Fortuna, E., *et al.* (2011) EnerJ: Approximate Data Types for Safe and General Low-Power Computation. *ACM SIGPLAN Notices*, **46**, 164-174. <https://doi.org/10.1145/1993316.1993518>
- [24] Nguyen, D.T., Kim, H., Lee, H.J., *et al.* (2018) An Approximate Memory Architecture for a Reduction of Refresh Power Consumption in Deep Learning Applications. 2018 *IEEE International Symposium on Circuits and Systems (ISCAS)*, Florence, 27-30 May 2018, 1-5. <https://doi.org/10.1109/ISCAS.2018.8351021>
- [25] Chen, Y., Yang, X., Qiao, F., *et al.* (2016) A Multi-Accuracy-Level Approximate Memory Architecture Based on Data Significance Analysis. 2016 *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Pittsburgh, 11-13 July 2016, 385-390. <https://doi.org/10.1109/ISVLSI.2016.84>
- [26] Swendsen, R.H. (1993) Modern Methods of Analyzing Monte Carlo Computer Simulations. *Physica A: Statistical Mechanics and Its Applications*, **194**, 53-62. [https://doi.org/10.1016/0378-4371\(93\)90339-6](https://doi.org/10.1016/0378-4371(93)90339-6)
- [27] Srivastava, A. and Venkatapathy, K. (1996) Design and Implementation of a Low Power Ternary Full Adder. *VLSI Design*, **4**, 75-81. <https://doi.org/10.1155/1996/94696>
- [28] Yao, T., Gao, D. and Fan, X. (2012) Three-Operand Floating-Point Adder. *IEEE International Conference on Computer & Information Technology*, Chengdu, 27-29 October 2012, 192-196.
- [29] Shafique, M., Ahmad, W., Hafiz, R., *et al.* (2015) A Low Latency Generic Accuracy Configurable Adder. 2015 *52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, San Francisco, 7-11 June 2015, 1-6. <https://doi.org/10.1145/2744769.2744778>
- [30] Kai, D., Varman, P. and Mohanram, K. (2012) High Performance Reliable Variable Latency Carry Select Addition. 2012 *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, 12-16 March 2012, 1257-1262. <https://doi.org/10.1109/DATE.2012.6176685>
- [31] Chirca, K., Schulte, M., Glossner, J., *et al.* (2004) A Static Low-Power, High-Performance 32-bit Carry Skip Adder. *Euromicro Symposium on Digital System Design*, Rennes, 31 August-3 September 2004, 615-619. <https://doi.org/10.1109/DSD.2004.1333335>
- [32] Jiang, H., Liu, C., Liu, L., *et al.* (2017) A Review, Classification, and Comparative Evaluation of Approximate Arithmetic Circuits. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, **13**, 1-34. <https://doi.org/10.1145/3094124>