Al-Maaitah K, Qiqieh I, Soltan A, Yakovlev A. [Configurable-Accuracy Approximate Adder Design with Light-Weight Fast Convergence Error Recovery Circuit](#). In: *2017 IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AECT)*. 2017, Amman, Jordan: IEEE.

Copyright:

© 2018 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

DOI link to article:

<https://doi.org/10.1109/AECT.2017.8257753>

Date deposited:

13/03/2018

Configurable-Accuracy Approximate Adder Design with Light-Weight Fast Convergence Error Recovery Circuit

Khaled Al-Maaitah, Issa Qiqieh, Ahmed Soltan, Alex Yakovlev

School of Electrical and Electronic Engineering, Newcastle University, Newcastle upon Tyne, United Kingdom
{k.almaaitah, i.qiqieh1, ahmed.abd-el-aal, alex.yakovlev}@ncl.ac.uk

Abstract—Approximate computing has recently introduced a new era of low-power and high-speed circuit designs. Recent efforts in the domain of configurable-accuracy approximate designs have proposed substantial performance gains and energy savings by allowing performance-energy-accuracy trade-offs. In this paper, we propose a configurable-accuracy approximate adder with new light-weight error detection technique. This is followed by significance-driven error correction stages during run-time. The correction starts by recovering the higher magnitude errors at premier correction stages, which results in fast convergence and higher precision outputs. Compared to other equivalent approximate adders, the proposed design has drastically reduced the logic counts used for error detection process; hence, achieving lower overhead of silicon area and improving the energy-efficiency of the adder design with faster convergence to the exact results. A number of different bit-widths of the proposed adder (32-bit to 256-bit) are designed in Verilog and synthesized using Synopsys Design Compiler. Our post-synthesis experiments showed significant reductions of 12% and 10% for Dynamic and Leakage Power respectively, and 8% in the silicon area for the design with full correction stages. Moreover, the proposed adder with large bit-widths has reserved these reduction ratios while presenting better scalability overhead. Additionally, our low overhead proposed design has presented the chance to be improved in terms of increasing accuracy to reach 100% exact results as accurate conventional adder at the final correction stage.

I. INTRODUCTION

Approximate computing is a design paradigm that attempts to relax the strict accuracy requirements of conventional computing to achieve more performance and energy efficiency. The main idea of approximate computing is about enhancing design parameters such as execution speed and power consumption, meanwhile allowing computing errors to occur within acceptable frequency and magnitudes. The majority of modern applications from domains such as image and video processing, machine learning, computer vision, biomedical and Internet of things (IoT) applications, have some level of tolerance to imprecise operations. This error tolerance is due to applications' inherent resilience to low magnitude errors in their arithmetic operations. Different factors might account for the resilience of these applications, such as perceptual limitations of application users, noise and redundancy existence in the real-world input data. Thus, although approximation errors exist, a substantial portion of computations is still able to produce an output of sufficient and acceptable quality [1], [2], [3], [4], [5], [6], [7], [8].

In digital systems, arithmetic units like adders have been investigated remarkably in the context of approximate computing. A conventional adder design can be approximated by exploiting the fact that the typical length of the carry propagation chains is usually much shorter than the full width of the adder. This, in turn, allows the designers to break the carry chains of the adders via multiple smaller separated or overlapped sub-adders with lower computing overhead. As a result, an approximated adder built with short carry chains is able to effectively operate with more performance and improved energy efficiency while making acceptable outputs accuracy trade-off [5].

In general, the approximate designs aim to produce almost-correct results (i.e. low magnitude errors) and obtain power reductions and performance improvements in return. However, in some applications, avoiding severe quality degradation can be accomplished by augmenting an error recovery circuit. This circuit comprises error detection and correction parts, which are conditionally activated to overcome erroneous approximated outputs, thereby, preserving the desired quality. Various efforts suggested approximated adders with variable latency error recovery circuit, which utilizes additional clock cycle to operate the error recovery process for any detected errors [4], [5], [9], [10]. On the other hand, in contexts where the required accuracy level needs to be changed during execution time, the correction of erroneous results should be configurable to maximize the benefit of approximate operations. An example of accuracy-configurable approximate (ACA) adder has been proposed in [9], in which the addition operation is performed by overlapping sub-adders of fixed width, in addition to several controlled error correction stages [5], [7].

This idea of configurable-accuracy approximated adders had a remarkable attention by many other efforts like (GDA [6], GeAr [5], Accurus [7]). Nevertheless, these efforts have the common challenge of additional design overhead in terms of delay, power and area, and moreover, did not reach the 100% accuracy at the final correction stages.

This paper has three contributions regarding the configurable-accuracy proposed design, which can be summarized by the next points.

- 1) We propose an energy-efficient approximate adder with a new error detection technique, which leads to lower design overhead and more scalability for larger bit-width adder designs, yet, without any accuracy scarifying.

- 2) We propose correction stages with significance-driven structure to ensure correcting higher-magnitude errors early, and achieve fast convergence to the conventional exact adder outputs.
- 3) We demonstrate an extended version of the proposed design, which guarantees 100% accuracy at the final correction stage.

The rest of the paper is organized as follows. Section II presents the proposed design parts. A detailed numerical example of proposed addition and error correction is presented in section III. In section IV the experimental results and analysis are provided. Section V concludes the paper.

II. PROPOSED APPROXIMATE ADDER DESIGN

In this section, three parts of the proposed design have been presented. Part II-A shows the general architecture of the proposed approximated adder. The proposed error detection technique is introduced in part II-B, and finally, the significance-driven structure of the correction stages is placed at the third part II-C.

A. General Architecture

The proposed approximate adder exploits the idea that the critical (worst case) carry propagation through the whole adder width would take place on rare occasions (i.e. not often happened). Hence, the proposed approximate adder is divided into several sub-adders by cutting the length of the critical carry path. On the other hand, redundant parts of the addend inputs are overlapped at successive sub-adders. This redundant parts are used to predict a carry generation or short carry propagation within the new sub-adders. This leads to high probability of carry value handling, hence, increasing the outputs accuracy significantly. This general approach of the proposed approximate adder might be considered analogues to the efforts in [5], [7] and [9].

Fig. 1 shows the general implementation of the proposed approximate adder design, and the following points summarize its main parts.

- $2K$ is the bit-width of the sub-adder.
- K equals the half length of the carry-chain depth.
- Non-overlapped K -bits would result in final output SUM.
- The first sub-adder produces a $2k$ -bit result.
- The approximate adder consists of $M = ((N/k)-1)$ sub-modules, where N equals to the total bits number of the main adder.

B. Error Detection Circuit

The second part of the proposed design is the error detection and correction (EDC) circuit. This augmented circuit gives the approximated adder the ability to work gracefully in approximated and accurate modes during run-time. In the proposed design, the error will signal high when the overlapped (redundant) part of the sub-adder failed to handle a correct carry value which should be propagated from the previous sub-adder. For error detection between two sub-adders, the proposed error detection technique uses an XOR gate as shown

Fig. 1: General implementation for the proposed approximate adder [9].

in Fig. 2 to check the equality of the carry-out of the previous sub-adder and the carry-out of the redundant part of the current sub-adder. Hence, when an error is detected, the approximated SUM value has an error and needs to be corrected by adding '1' to its current value. This correction process guarantees the correct carry value propagation and can be accomplished by an incrementor circuit. The proposed technique decreases the logic complexity used for error detection between every two sub adders when compared to previous effort of ACA that used more logic to check values of larger number of bits (SUM bits of the redundant part AND the carry-in to this part in the previous sub-adder). Although this technique incurs a limited degradation of delay, significant enhancements of power consumption and area happened. Moreover, this technique is considered more beneficial in terms of scalability of larger adder widths.

Fig. 2: Proposed adder design Error Detection Technique.

C. Error Correction Circuit

The correction circuit basically uses an incrementor circuit that increments the erroneous SUM value by one. The bit width of the incrementor is equal to the width of the resulted approximated SUM bits of each sub-adder. On the other hand, the structure of the correction circuit is applied through several correction stages. This structure would give the ability to control the active correction stage independently while changing the accuracy level during the run-time; hence, achieving more power saving for low-level accuracy modes [9].

Fig. 3 shows the proposed significance-driven error correction circuit structure. This structure would guarantee that the resulted sum from the most significant sub-adders will be corrected first, and the final correction stage will correct the resulted sum from the least significant sub-adders. As

a result, the correction of high magnitude errors and fast convergence to the exact sum value will take place at the premier correction stages. This significance-driven correction scheme might be analogous to [7]; However, there was no remarkable modification for the error detection mechanism, in opposite to our proposed design that incurs lower design overhead and more scalability as will be shown in the results section.

On the other hand, In order to guarantee 100% accuracy of outputs at the final correction stage (highest level of accuracy), the carry-out values of the active correction stages should not be overlooked. Conversely, these values have to be considered and propagated to the successive most significant correction stages. As a result, the proposed design introduced a new extended version, which checks the correction stages carry-out values and their propagation in a way that achieves full accuracy at the final correction stage.

Fig. 3: Significance Driven Error Correction Stages (32-bit adder example) .

From Fig. 3, the main points of multi correction stages can be summarized as follows.

- 1) Stage (0) has the approximated sum result (without any correction).
- 2) Stages (1), (2) and (3) are the error correction stages.
- 3) The correction stage (incrementor) will result in the accurate SUM part (coloured green).
- 4) The output accuracy increases as the number of the active stages increases.
- 5) S0 is always correct as it uses carry-in = '0'.

III. NUMERICAL EXAMPLE

In this section, a (32-bit) binary numerical addition example is presented. The detailed example shows the main methodology of adder division, approximate addition and error detection and correction techniques of the proposed design versions versus ACA design.

Fig. 4: Proposed Design Numerical(32-bit) Addition Example.

TABLE I: Proposed Design versions vs ACA error detection.

Proposed Accurate Check	Proposed Design Check	ACA Check	Carry out	Redundant=1111	Carry [3]	Sub adder
-	-	-	1	-	0	1
0	0	0	1	No	1	2
0	0	0	1	No	1	3
1	1	1	0	Yes	0	4
1	0	0	0	Yes	0	5
1	0	0	0	Yes	0	6
1	0	0	0	Yes	0	7

Fig. 4 presents an example of (32-bit) addition of the proposed design. The main adder is divided into 7 sub-adders of (8-bit) width for each. Sub-adders except the first one have two parts, where the first least significant part has a redundant 4 bits from the previous sub-adder, and the most significant part has a new addend 4 bits which are used to result in the approximated sum. The carry-in to each sub-adder is truncated to '0'. The redundant bits at each sub-adder are used to predict the carry-in to the most significant part to approximate an almost correct sum result. Fig. 4, Part A presents the approximated sum of the proposed design, while Part B shows the exact (correct) sum that accomplished by a conventional adder. The two results A and B show the limited difference between the approximated and exact results, where a carry at A had to be propagated after the fourth sub-adder to the final carry-out that should be '1' instead of resulted approximated carry-out of '0'.

Table I shows the methodology of error detection of the proposed design versions (Proposed Design and Proposed_Accurate) and ACA design. Error detection simply means to check if there is a carry-out of the previous sub-adder of '1' must be propagated to the current sub-adder. As a result, if an error is detected, the approximated 4-bit sum value has to be incremented by '1' to reach the correct value. ACA design detects an error when the redundant 4-bit sum is equal to (1111), and the carry-out of the fourth bit of the previous sub-adder is equal to '1'. However, the proposed design error detection concerns about just checking the equality of two bits locations (the carry-out of the previous sub-adder and the carry-out of the fourth bit of the current sub-adder). The equality check will signal high if there is any error (carry propagation) detected. However, the Proposed_Accurate version introduces more accuracy by detecting the carry-out of the active correction stage to be propagated to the sum result of successive correction stages.

Sub-adder(7)	Sub-adder(6)	Sub-adder(5)	Sub-adder(4)	Sub-adder(3)	Sub-adder(2)	Sub-adder(1)
1111 1111 0000 0000	1111 1111 0000 0000	1111 1111 0000 0000	1111 1101 0000 0010	1101 1001 0010 1110	1001 1100 1110 1111	1100 1010 1111 0001
Carry Out 0	1111	1111	1111	1111	0000	1000
Proposed Accurate Active stage	Proposed Design Active stage	ACA Active stage	Error	Sub-adder		
0	0	0	No	1		
0	0	0	No	2		
0	0	0	No	3		
2	2	2	Yes	4		
2	0	0	Yes	5		
1	0	0	Yes	6		
1	0	0	Yes	7		

Fig. 5: Proposed Design versions vs ACA active error correction stages.

Fig. 6: 32-bit Proposed Design Versions vs ACA Dynamic Power (a), Area (b) and Delay (c) Comparison.

Fig. 5 shows a comparison between the proposed design versions and ACA design regarding the active correction stages when detecting an error. For ACA design, an error is detected at sub-adder number four, thus, regarding ACA correction stages ordering which starts correcting the errors of the least significant sum results, stage number 2 will be active to correct the error. The proposed design version detects the same error at the fourth sub-adder, and regarding the significance-driven architecture in Fig. 3, correction stage number 2 will be active to correct the error. On the other hand, for the Proposed_Accurate design version, all the errors of carry propagation starting from the fourth sub-adder to the final seventh sub-adder are detected. The Proposed_Accurate design version has the same significance-driven architecture of correction stages, thus, for the errors at the fourth and fifth sub-adders, correction stage number 2 will be active, and for the errors at the sixth and seventh sub-adders, correction stage number 1 will be active. As a result, all the errors will be corrected and guarantees 100% accuracy for the final sum result.

IV. EXPERIMENTAL RESULTS AND DESIGN TRADE-OFFS

This section has six parts; part IV-A describes the experiment setup methodology, part IV-B has the main design parameters such as power, area and delay evaluation of 32-bit adder example. The error analysis results comparisons are presented in part IV-C. The implementation of the proposed adder design with large bit widths is presented in part IV-D, and a further hardware metrics comparison regarding several design efforts is placed at part IV-E. Finally, an image (Gaussian blur) filter implementation using the proposed design is presented at part IV-F.

A. Experimental setup

To demonstrate the proposed approach, Verilog was used to apply different sizes of adders ranging from (32-bit to 256-bit). However, our main comparison for all correction stages used 32-bit adder with half carry chain (K) equals to (4-bit) for each sub-adder. These codes were synthesized and implemented using two different off-the-shelf tools; Firstly, Modelsim was used for compiling the Verilog codes and running the associated test benches. Secondly, Synopsys Design Compiler was utilized for synthesizing all sizes of the proposed adder versions when mapping the circuits to the UMC (Faraday 90nm) technology and evaluating for power, Delay and area. Matlab was used for implementing the (Gaussian blur) image filter test.

B. Design Parameters Evaluation

In order to make hardware evaluation, the proposed design was compared to the design effort in ACA [9]. The proposed design has two versions, where the first version was applied without considering the carry-out of correction stages, and the second version considers the carry-out of each active correction stage depending on the selected accuracy mode. For simplifying, the design version considering the correction stage carry-out was denoted by (Proposed_Accurate).

From Fig. 6, parts (a) and (b) can show that the proposed design behaves better than the Accuracy Configurable Adder (ACA)[9] in terms of design parameters such as dynamic Power and Area. The proposed design incurs smaller hardware area due to limiting the error detection logic gates counts when compared to the ACA error detection technique. Consequently, the decreased hardware would result in lower levels of power consumption.

For Delay values part (c) in Fig. 6, the proposed design shows a small enhancement of execution speed at the final correction stage, besides stable delay values through all correction stages. These values can show the independence characteristic of each sub-adder in which the critical path delay is the same for all segmented blocks. On the other hand, in the case of the Proposed_Accurate version (correction carry-out in concern), which reaches the full accuracy at the highest mode, despite the very limited degradation of correction stages delay values, it still behaves with higher speed regarding conventional adders like Ripple Carry Adder(RCA), and lower design overhead in terms of power consumption and area when compared to ACA for all stages.

Table II provides the average reduction ratios resulted from the proposed design when compared to ACA design. Obviously, significant improvements are introduced in terms of power (dynamic and leakage) and area values for all stages of the design with very limited negative ratio for delay.

TABLE II: Average Reduction Ratio Values of the Proposed design version compared to ACA design.

Parameter	Proposed Design vs ACA
Dynamic Power	7.3%
Leakage Power	5.2%
Area	4.6%
Delay	-1.9%

C. Error analysis

An error analysis was made to show the Relative Error Distance (RED) distribution of each design over different error distance values range. This error metric simply measures how far the significance of erroneous outputs of the

Fig. 7: 32-bit Adder Relative Error Distance Distribution Analysis, One correction stage (a), Two correction stages (b) and Three correction stages (c).

proposed design versions (Proposed and Proposed_Accurate) and the ACA [9] adder design when both compared to the exact outputs from a conventional exact adder. Despite the simplicity of this measurement, it would show the effect of the proposed design correction stages regarding the final quality of the outputs. The following equation shows the arithmetic expression of the RED value.

$$RED = \frac{|\text{Correct value} - \text{Approximated value}|}{\text{Correct value}} \quad (1)$$

For example, in case the RED value equals '0', the approximated output value is correct, and there is no difference between it and the exact value. However, if RED value equals '0.01', then approximated output value is not fully correct, and there is a difference between its value and the exact value by percentage = 1%.

Parts (a) and (b) of Fig. 7 show the error analysis of designs with one correction stage and two correction stages respectively. It can be noticed that our design versions (Proposed and Proposed_Accurate) have approximately the same error distribution as in ACA design. Nevertheless, they show more stability in terms of RED values as they start to be limited strictly between 100% and 99% of accuracy.

The highest level of accuracy when three correction stages need to be activated is presented by part (c) in Fig. 7. It is obviously shown that the behaviour of the proposed design version continues to show the same behaviour as ACA. However, the improved error detection mechanism of the Proposed_Accurate design version (by considering the carry propagation of the correction stages) shows the best result with 100% accurate outputs.

Fig. 8 presents the mean relative error distance(MRED) values comparison, which shows the mean value of errors within test space or input vectors at each correction stage [11]. When compared to ACA effort, the proposed design versions without correction overhead show the same mean error values. However, by using the first and second correction stages, the proposed versions' MRED values are drastically decreased showing better accuracy and consistency. This is due to the significance-driven structure of the correction stages which limited the final outputs to low magnitude values of errors early. For the final correction stage, our proposed design version provided MRED values similar to ACA, while the extended proposed version (Proposed_Accurate) presented no errors confirming 100% accuracy at the final correction stage outputs.

Fig. 8: 32-bit Adder designs MRED values comparison through different correction stages .

D. Large Bit Width Adders Evaluation

For design scalability checking, a further hardware evaluation was implemented for different adder designs (ACA and Proposed design version) with larger bit widths (64-bits, 128-bits and 256-bits). Hardware metrics evaluation for each design with its full correction stage architecture (i.e. using Three correction stages) is shown in Fig. 9. It can be noticed that the proposed design version keeps the positive reduction ratios in terms of power (dynamic and leakage) and area. However, it still shows the very limited delay degradation of the proposed design when larger bit-width adders are in use.

These reduction values of the large adder designs might clearly show the scalability advantage of the proposed design with acceptable design overhead.

Fig. 9: ACA vs Proposed Large Bit-Width Adder Designs Hardware Comparisons.

E. Further Hardware Comparison

Table III shows an extended comparison of the proposed design two versions (Proposed and Proposed_Accurate) regarding previous efforts. Obviously, the proposed design shows more advantages and enhancements regarding the design parameters such as power, delay and area. For both the power

TABLE III: Hardware metrics Comparison of Proposed Adders and previous efforts

Design	Technology	Adder Width	Max Correction Stages	Delay (ns)	Power (Dynamic)(μ W)	Power (Leakage)(μ W)	Area(μm^2)
ACA*	UMC 90 nm	32-bit	No correction	0.67	22.2436	2.4038	1001.952
ACA	UMC 90 nm	32-bit	Three	1	56.1614	4.1278	1855.728
Accurus [7]	UMC 90 nm	32-bit	Three	1.13	56.916	4.1796	1895.712
RCA	UMC 90 nm	32-bit	Accurate	2.22	18.1273	2.0263	702.816
Proposed*	UMC 90 nm	32-bit	No correction	0.67	22.2436	2.4038	1001.952
Proposed	UMC 90 nm	32-bit	Three	0.98	49.1171	3.726	1708.336
Proposed_Accurate	UMC 90 nm	32-bit	Three	1.93	49.9059	3.805	1774.976

NOTE: Designs without correction modes are marked with an asterisk (*).

and area, the proposed design versions have better values when compared to ACA and Accurus [7] designs. For the delay values, the Proposed design version shows a higher speed regarding ACA and Accurus designs. on the other hand, despite delay degradation of the Proposed_Accurate version, it still presents better speed than conventional Ripple Carry Adder (RCA).

F. Implementation Test

For implementation testing, Matlab was used to apply Gaussian blur image filter test to check the actual behaviour of (20-bits) Proposed_Accurate adder design version during multiple correction stages. The peak signal to noise ratio (PSNR) is used to measure the quality of the output images after applying Gaussian blur filter regarding the resulted image from the exact addition. From Figure 10, at Mode= 1 when just the most significant correction stage is in operation, PSNR has a value of (29.8 db) with acceptable quality and limited distortion of the output image. However, for Mode=2 where two correction stages are in action, a remarkable improvement was happened to the PSNR value by (42.6 db) with a very acceptable quality of output. Mode=3 with all three active correction stages, the output image has the optimum PSNR value the same as the resulted image of the conventional exact adder. Although the appearance of low PSNR value at the first correction stage, the proposed design might be considered as an attractive adder design for some application like the Biomedical applications, which generally interested in high speed, very low power and acceptable outputs quality.

V. CONCLUSION

In this paper, a configurable-accuracy approximated adder with new error detection technique was proposed. The new technique incurred lower design overhead with 12%, 10% and 8% reduction ratios for dynamic power, leakage power, and area respectively at the final correction stage (i.e. highest level of accuracy). Furthermore, the correction stages were structured in a significance-driven scheme, in which the first correction stage starts correcting the most significant SUM bit locations. This guarantees fast convergence to the exact output values at the premier correction stages. On the other hand, the proposed design was extended to a modified version, which considers the propagation of the correction stages' carry-out to successive most significant stages during active correction process. Remarkably, this (Proposed_Accurate) design version continued to show lower design overhead besides presenting the best results of accuracy consistency (i.e. between 99% and 100% accuracy levels), and 100% accurate results at the final stage.

Fig. 10: Gaussian blur Image filter test.

REFERENCES

- [1] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *2013 18th IEEE European Test Symposium (ETS)*, pp. 1–6, May 2013.
- [2] S. Venkataramani, S. T. Chakradhar, K. Roy, and A. Raghunathan, "Approximate computing and the quest for computing efficiency," in *Proceedings of the 52nd Annual Design Automation Conference, DAC '15*, (New York, NY, USA), pp. 120:1–120:6, ACM, 2015.
- [3] H. Jiang, J. Han, and F. Lombardi, "A comparative review and evaluation of approximate adders," in *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI, GLSVLSI '15*, (New York, NY, USA), pp. 343–348, ACM, 2015.
- [4] A. K. Verma, P. Brisk, and P. Ienne, "Variable latency speculative addition: A new paradigm for arithmetic circuit design," in *2008 Design, Automation and Test in Europe*, pp. 1250–1255, March 2008.
- [5] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "A low latency generic accuracy configurable adder," in *Proceedings of the 52nd Annual, DAC '15*, (New York, NY, USA), pp. 86:1–86:6, ACM, 2015.
- [6] R. Ye, T. Wang, F. Yuan, R. Kumar, and Q. Xu, "On reconfiguration-oriented approximate adder design and its application," in *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 48–54, Nov 2013.
- [7] V. Benara and S. Purini, "Accurus: A fast convergence technique for accuracy configurable approximate adder circuits," in *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 577–582, 2016.
- [8] L. Sekanina and Z. Vasiccek, "Approximate circuit design by means of evolvable hardware," in *2013 IEEE International Conference on Evolvable Systems (ICES)*, pp. 21–28, April 2013.
- [9] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *DAC Design Automation Conference 2012*, pp. 820–825, June 2012.
- [10] K. Du, P. Varman, and K. Mohanram, "High performance reliable variable latency carry select addition," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, pp. 1257–1262, March 2012.
- [11] I. Qiqieh, R. Shafik, G. Tarawneh, D. Sokolov, and A. Yakovlev, "Energy-efficient approximate multiplier design using bit significance-driven logic compression," in *Design, Automation Test in Europe Conference Exhibition (DATE), 2017*, pp. 7–12, March 2017.