Single-Cycle MIPS Processor based on Configurable Approximate Adder

Amir E. Oghostinos[1], Kareem Moussa[1], Amr Elnaggar[1], Alaa AbdAlRhman[2], and Ahmed Soltan[1,2]
[1]School of Engineering and Applied Sciences, Nile University, Giza 12677, Egypt
[2]Nanoelectronics Integrated Systems Center (NISC), Nile University, Giza, Egypt

***Abstract*—Enhancing computer architecture performance is a significant concern for architecture designers and users. This paper presents a novel approach to computer architecture design by using an approximate adder with configurable accuracy in a single-cycle MIPS processor as a study case. Using approximate adders decreased the delay on the expense of the design area. Using approximate computing with the MIPS processor, the timing performance has been improved by 253.4% compared to the lookahead adder. It has been implemented and tested using System-Verilog.**

***Index Terms*—Approximate computing, Computer architecture, Single-cycle MIPS processor, Adder, Performance**

I. Introduction

Approximate computing is an emerging design technique that trades off the correctness of conventional computations with modern computing systems' performance and energy efficiency. Approximate computing aims to improve design parameters metrics such as execution speed and cost while allowing computing errors to occur within an acceptable frequency and magnitudes [1]. As a result, this would increase the performance, decrease the energy consumed by systems and increase the density of systems in one chip [2]–[4]. Approximate computing can be done at different levels, such as electronic circuit design (transistor level), arithmetic operation approximation (adders, multipliers), and system requirement [5]. Approximate computing can either minimize the delay by logic reduction or voltage scaling. Probabilistic CMOS can reduce energy consumption by most higher voltages to high critical circuits and most significant bits to ensure an acceptable accuracy while carefully reducing the supply voltages for the least significant bits that have a more negligible impact on the final result's accuracy. A probabilistic adder is designed based on a conventional adder with various supply voltages depending on the importance of bits [6]. However, this technique has a very high implementation cost due to the complex control for the supply voltages. Therefore, most approximate arithmetic circuits are designed as simple schemes to control and implement based on the logic reduction method. In general, approximate computation components have more negligible latency and low accuracy. Therefore, it can not be acceptable in any accurate applications. For example, image processing and video streaming applications accept tolerance in calculations based on the purpose of the processing [5]. However, the accuracy required in images used for diagnosis differs from the accuracy of images used for vision aid systems used by visually impaired people.

Hence, Approximate computing found its way into different applications such as image processing and video compression. In these applications, the main focus was on the processing speed, and performance optimization for timing [7]. Furthermore, approximate computing introduces a new concept of configurable accuracy computing [3]. In configurable accuracy, the computation accuracy is related to the timing constraints and the available power budget. By using configurable accuracy computing, we can implement battery-controlled wearable devices. Hence, the available power budget controls the accuracy of calculations and system performance.

Approximate computing has been used mainly to perform particular functions in digital systems. Especially in the image processing applications and video streaming to minimize the processing time and guarantee live streaming with high frames per second rates. However, approximate computing can potentially impact using general-purpose architectures such as single-cycle, multi-cycle, and pipeline architectures because its design is not applicable in high accuracy applications, as mentioned. Indeed, Approximate computing can be applied to both adders and multipliers [8]. However, this work illustrates only the impact of using adders as an example.

This work studies approximate adders in general processor purposes with MIPS single-cycle architecture as an exemplar of approximate adders in general-purpose processors. The modified architectures of MIPS can be used in power-sensitive applications as wearable biomedical devices [9]. This architecture enhances addition performance by leveraging approximate adders, which perform summation faster than the common adders. However, because the most common implementations use full adders, they cause an unnecessarily long critical path. Using a configurable approximate adder in this work led to shortening the critical path and having the shortest delay possible while maintaining up to 100% adding accuracy.

This paper is organized as follows: a literature review of adders is introduced in Section II. The methodology of using the approximate adders is illustrated in Section III. Finally, the results and discussion are presented in Sections IV, V.

II. Approximate Adders

In [2], a delay comparison of six adders: Ripple Carry, Conditional Sum, Conditional detect Carry-Lookahead, Carry-Skip, Carry-Select. Ripple Carry adder is implemented using n (number of bits) one-bit full adders. Hence, the ripple-carry adder has the longest path for the carry propagation, which causes a worst-case delay of 51.84 ns. However, it is the simplest and cheapest adder design. On the other hand, the conditional-sum adder is based on having two groups of output. The first one assumes the input carry is 0, while the other group assumes that the input carry is 1. Then, the output carry is predicted based on the input carry/pattern. Hence, the worst delay for the conditional sum is 85% better than the ripple carry adder for the same technology [2].

The completion detection conditional sum adder is a modified version of the conditional-sum adder. It detects the sum generation, which decreases the worst delay to 7.92 ns. The Carry-Lookahead improves the Ripple-Carry adder by using the inputs to calculate the carries at the beginning, which reduces the propagation time to 24.96 ns. The Carry-Skip skips the carry if the carry of the following stages is not needed as it is equal to the first stage. As a result, it has the worst delay of 26.37 ns. Carry-Select adder, each stage produces two outputs. One of the outputs is for the input carry "1", and the other is for the input carry "0". According to the actual input carry, the correct output is selected. As a result, the Carry-Select adder's worst delay of 24.75 ns. [2], [10], [11].

Indeed, Carry-Skip, Conditional-Sum, and Conditional-Detect adders can be considered an early stage of approximate adding. However, these adders always target 100% accuracy. Hence, they did not make any use of the prediction scheme. On the other hand, approximate designs aim to generate almost correct results (i.e., the error is acceptable within a range based on the application) to reduce the calculation latency. 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, preserving the desired quality. Moreover, some Approximate computing techniques introduce the idea of configurable accuracy. In these techniques, the accuracy is correlated to energy consumption [12], [13], which leads to a new concept of energy-based configurable processors. In various efforts suggested, approximate adders with variable latency error recovery circuit utilize an additional clock cycle to operate the error recovery process for any detected errors [11], [3]. 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.

By adding accurate configurable computing to general-purpose architectures, new instructions will be added to use the proper accuracy based on the available power. However, these new instructions can add complexity to the assembly and, hence, the programming process.

III. Proposed Design

In this work, approximate adders are used with single-cycle architectures to illustrate the impact of approximate computing on architecture performance. The single-cycle architecture consists of three adders. The adder inside the ALU is replaced with an accuracy configurable approximate adder, as depicted in fig.1. The other two adders are related to the program counter, and both have to give a very accurate result. Hence, using traditional adders for these two adders is better considering the needed accuracy. The assembly of the processor is modified to two more addition instructions: ADDC1 and ADDC2, where C stands for configurable and the number following the instruction refers to the level of accuracy. The single-cycle MIPS processor was Designed using System-Verilog hardware description and verification language, a superset of the Verilog language extended with verification features.

The MIPS processor in fig. 1 is theoretically divided into five stages: Fetch, Decode, Execute, Memory, and Writeback. Each stage contains one costly main operation. Signals from the control path control every main block in these stages. These signals dictate operations such as whether the register should be written read or any other operation performed by the ALU.

A. Single-Cycle Processor Design

The MIPS processor in fig. 1 is theoretically divided into five stages: Fetch, Decode, Execute, Memory, and Writeback. Each stage contains one costly main operation. For that, signals from the control path control every main block in these stages. These signals dictate operations such as whether the register should be written read or any other operation performed by the ALU.

B. Adder Design

The approximate adder that is designed and explained in [3] and illustrated in fig. 2 is used instead of a full adder for the add operations to reduce the total delay of the add operation, as the add operations have a critical path length equal to the length of a single operand (32-bit in this case). The approximate adder is divided into four smaller sub-adders, where carry is predicted for each sub-adder instead of being propagated from the previous sub-adder; thus, the critical path is one-fourth the size of the operand. Then, a correction stage in fig. 3 is employed to get variable accuracy up to 100% for the output result. For cases where errors are not tolerated, such as the load byte instruction where the memory address must be exact, fully accurate addition is used. In other cases where error can be tolerated, the accuracy can be adjusted during processor operation to decrease the delay, leading to an overall increase in throughput. Other adders in the processor used for incrementing program counter and calculating PC branch are using typical ripple-carry adders as their delay is not of concern as they are not on the critical path of the processor.

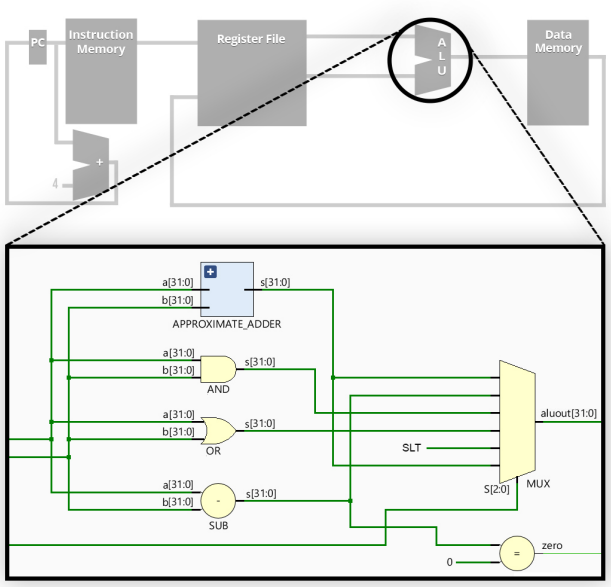

Fig. 1: Proposed Single-Cycle MIPS Processor emphasizing the ALU design

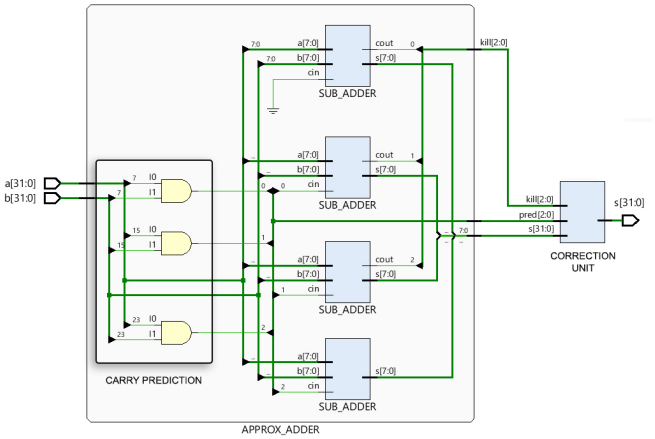

Fig. 2: The approximate adder used in the ALU

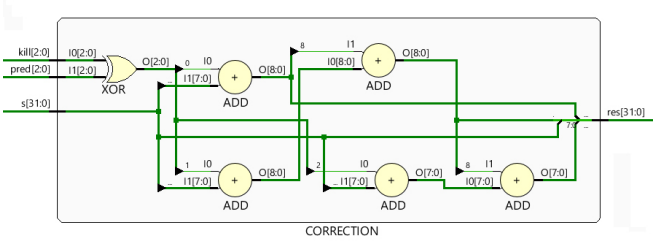

Fig. 3: Full (3-stage) Correction Unit

IV. Results

To test the implementation, a simple program that carries out several addition operations is loaded into the instruction memory, and the final value is observed as output from the ALU. Several accuracy variations were tested: Approximate adder with 3-stages (full) correction in fig. 5, 1-stage of correction in fig. 4, and without correction stage.

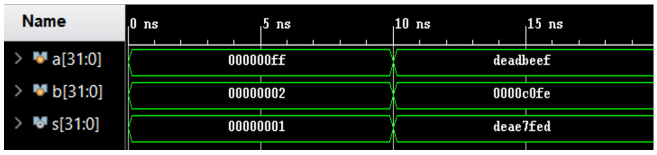

Fig. 4: Addition with 1-stage Correction

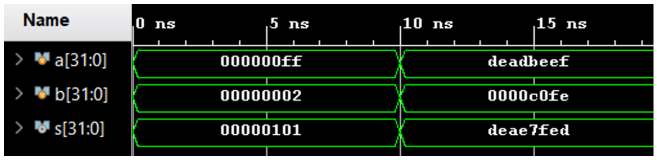

Fig. 5: Addition with 3-stage (full) Correction

TABLE I

Adders	Maximum Delay (ns)
Ripple Adder	51.84
Carry-Lookahead	24.96
Carry-Skip	26.37
Carry-Select	24.75
Approximate Adder with 3-stage correction (Full accuracy)	9.85
Approximate Adder with 1-stage correction	9.23
Approximate Adder without correction	6.28

The delay of the addition operation is $9.85\ ns$, using the approximate adder adjusted to full accuracy, which is shorter than ripple-carry and typical adders with higher delays, as shown in Table I.

V. Conclusion

This paper briefly discussed different approximate adders architectures and compared their delays. Then, a configurable-accuracy processor is introduced using a configurable approximate adder in a single-cycle MIPS processor. Finally, two new assembly instructions are introduced to configure the processor through software with two different accuracy levels. The approximate adder has a maximum delay of 6.28 ns when configured to minimum accuracy.

References

[1] S. Mittal, "A survey of techniques for approximate computing," *ACM Computing Surveys (CSUR)*, vol. 48, no. 4, pp. 1–33, 2016.

[2] M. A. Franklin and T. Pan, "Performance comparison of asynchronous adders," in *Proceedings of 1994 IEEE Symposium on Advanced Research in Asynchronous Circuits and Systems*. IEEE, 1994, pp. 117–125.

[3] K. Al-Maaitah, G. Tarawneh, A. Soltan, I. Qiqieh, and A. Yakovlev, "Approximate adder segmentation technique and significance-driven error correction," in *2017 27th International Symposium on Power and Timing Modeling, Optimization and Simulation (PATMOS)*. IEEE, 2017, pp. 1–6.

[4] A. AbdAlRahman, A. Soltan, and A. G. Radwan, "An optimized implementation of gl fractional-order," in *2021 IEEE International Midwest Symposium on Circuits and Systems (MWSCAS)*. IEEE, 2021, pp. 669–672.

[5] G. Rodrigues, F. Lima Kastensmidt, and A. Bosio, "Survey on approximate computing and its intrinsic fault tolerance," *Electronics*, vol. 9, no. 4, p. 557, 2020.

[6] K. Palem and A. Lingamneni, "Ten years of building broken chips: The physics and engineering of inexact computing," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 12, no. 2s, pp. 1–23, 2013.

[7] Y. Zhang, X. Yang, L. Wu, J. Lu, K. Sha, A. Gajjar, and H. He, "Exploring slice-energy saving on an video processing fpga platform with approximate computing," in *Proceedings of the 2018 2nd International Conference on Algorithms, Computing and Systems*, 2018, pp. 138–143.

[8] S. Vahdat, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Tosam: An energy-efficient truncation-and rounding-based scalable approximate multiplier," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 27, no. 5, pp. 1161–1173, 2019.

[9] A. Soltan, J. M. Barrett, P. Maaskant, N. Armstrong, W. Al-Atabany, L. Chaudet, M. Neil, E. Sernagor, and P. Degenaar, "A head mounted device stimulator for optogenetic retinal prosthesis," *Journal of neural engineering*, vol. 15, no. 6, p. 065002, 2018.

[10] S. M. Nowick, K. Y. Yun, P. A. Beerel, and A. E. Dooply, "Speculative completion for the design of high-performance asynchronous dynamic adders," in *Proceedings Third International Symposium on Advanced Research in Asynchronous Circuits and Systems*. IEEE, 1997, pp. 210–223.

[11] V. Dhandapani, "An efficient architecture for carry select adder," *World Journal of Engineering*, 2017.

[12] D. Ma, R. Thapa, X. Wang, X. Jiao, and C. Hao, "Workload-aware approximate computing configuration," in *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2021, pp. 920–925.

[13] E. Farahmand, A. Mahani, M. A. Hanif, and M. Shafique, "High performance and optimal configuration of accurate heterogeneous block-based approximate adder," *arXiv preprint arXiv:2106.08800*, 2021.