MANTECH Publications
Journal of Research in VLSI Design, Tools and Technology
Volume 10, Issue 2, May-August, 2025

Approximate Computing Architectures for Low-Power VLSI

Rohit Choudhary
Research Scholar
Department of Electronics Engineering
Bhagwan Parshuram Institute of Technology, Solan, Himachal Pradesh
E-mail Id: *rohitchoudhary.ast@gmail.com*

ABSTRACT

The growing demand for energy-efficient electronic systems in domains such as mobile computing, Internet of Things (IoT), and artificial intelligence has led to the exploration of unconventional design methodologies. Approximate computing has emerged as a promising paradigm, trading off exactness for significant improvements in power efficiency, performance, and silicon area. This paper investigates the role of approximate computing architectures in low-power VLSI design. It highlights the theoretical foundation, techniques, and applications while providing an analysis of recent architectural innovations. Through comparative evaluations, approximate computing is shown to enable substantial energy savings, particularly in error-tolerant applications such as multimedia processing, data mining, and machine learning. The study concludes that approximate computing, when carefully applied, offers a practical pathway toward next-generation energy-efficient VLSI architectures.

KEYWORDS: *Approximate Computing, Low-Power VLSI, Energy-Efficient Design, Error-Tolerant Applications, Hardware Optimization*

INTRODUCTION

The demand for compact and energy-efficient VLSI systems has escalated in recent years due to the rapid expansion of mobile devices, embedded systems, and high-performance computing applications. Traditional design methodologies have largely focused on precision and correctness, but these approaches result in high power consumption and limited

scalability. Approximate computing introduces an alternative by accepting controlled inaccuracies in computation to reduce complexity and improve efficiency.

This paradigm is especially relevant for error-tolerant domains such as image processing, video encoding, machine learning, and IoT applications where minor inaccuracies have negligible impact on overall output quality. By leveraging approximations in arithmetic units, memory systems, and logic circuits, approximate computing can yield significant reductions in power consumption, delay, and silicon area.

APPROXIMATE COMPUTING PARADIGM

Approximate computing is founded on the principle that not all computations need to be exact for the end user to perceive acceptable results. Many modern applications, such as multimedia, machine learning, and data mining, are inherently tolerant to small inaccuracies. This observation has motivated a design philosophy where certain components of a system are intentionally simplified to save energy, area, and time at the cost of bounded computational error.

The paradigm can be explained in multiple dimensions:

1. Algorithm-Level Approximation

At the highest abstraction, algorithms can be redesigned to operate on fewer data points or use simplified mathematical models. For instance, in image compression, algorithms like JPEG already discard high-frequency information because the human eye is less sensitive to it. Similarly, approximate search algorithms in big data analytics skip certain comparisons, thereby reducing computational burden without severely affecting accuracy.

2. Architecture-Level Approximation

At the architectural level, approximation techniques target critical computational blocks. For example, approximate adders and multipliers reduce complexity in arithmetic logic units (ALUs). Architectures may also use *precision scaling*, where computations are carried out with fewer bits. This level of approximation directly impacts the energy consumed by digital processors and accelerators.

3. Circuit-Level Approximation

On the lowest level, transistor networks and gate designs are modified to reduce switching activity and critical path delay. Logic simplification, gate pruning, or voltage overscaling are commonly used. These methods achieve significant power savings with minimal degradation in functional accuracy.

4. Cross-Layer Approximation

Advanced designs often combine approximation across multiple levels. For instance, an approximate multiplier at the hardware level may be paired with an error-resilient algorithm, ensuring that the overall system quality remains within acceptable bounds.

Approximate computing shifts the design paradigm from "always correct" to "good enough," enabling dramatic improvements in power-performance trade-offs for modern VLSI systems.

APPROXIMATE ARCHITECTURES IN VLSI

Approximate computing manifests in several architectural strategies that influence both micro-level (logic gates, adders, multipliers) and macro-level (processing cores, memory systems) design.

Approximate Arithmetic Units

- **Approximate Adders**:

Conventional adders propagate carries across all bit positions, increasing delay and power. Approximate adders truncate carry propagation or replace it with simplified logic in higher-order bits. While this introduces minor numerical errors, it yields significant energy savings. Popular designs include carry cutback adders, error-tolerant adders, *and* inexact speculative adders.

- **Approximate Multipliers**:

Multipliers are power-hungry due to their reliance on partial product generation. Approximate multipliers omit certain partial products, compress terms approximately, or truncate operands. For example, *broken-array multipliers* skip selected cells, reducing hardware complexity by up to 40% while maintaining acceptable error margins for multimedia workloads.

Approximate Memory Systems

Memory systems consume a large share of system power, particularly in data-intensive tasks. Approximation in memory is achieved by:

- Lowering DRAM refresh rates since not all data is equally critical.
- Using lossy compression schemes to reduce storage and communication overhead.
- Designing approximate SRAMs that tolerate occasional bit flips without affecting user experience, especially in image and video storage.

Approximate Logic Circuits

Logic circuits are simplified by reducing the number of gates in a Boolean function. Gate-level approximation methods selectively replace critical gates with smaller equivalents or even remove redundant paths. These designs minimize transistor count and switching energy, directly contributing to low-power VLSI.

System-Level Approximations

At the system level, approximation can be applied holistically:

- **Approximate accelerators** for machine learning and signal processing tasks.
- **Dynamic precision scaling**, where the processor adjusts the accuracy of computations based on workload demands.
- **Hybrid systems** combining accurate and approximate cores, enabling adaptive energy-quality trade-offs.

Table 1: Comparative Analysis of Approximate and Accurate Units

Component	Accurate Design	Approximate Design	Improvement
32-bit Adder	High power, full precision	Truncated carry adder	~35% power saving
16x16 Multiplier	High complexity	Partial product skipping	~40% area reduction
SRAM Memory Block	High refresh rate	Reduced refresh cycles	~30% energy saving
Video Processing Core	High fidelity	Approximate DCT core	~45% power saving

APPLICATIONS OF APPROXIMATE COMPUTING

The real strength of approximate computing lies in its suitability for a wide range of error-tolerant applications:

1. Multimedia Processing

Human sensory systems (vision and hearing) are inherently tolerant to small distortions. Video coding standards such as MPEG or H.264 already rely on lossy compression. Approximate hardware for Discrete Cosine Transform (DCT), motion estimation, and filtering achieves up to 50% power savings with negligible perceptual loss in quality.

2. Machine Learning and Artificial Intelligence

Neural networks and deep learning systems operate effectively with noisy or quantized inputs. Approximate multipliers and adders can accelerate matrix multiplications and convolution operations with reduced energy, enabling efficient inference on mobile and embedded devices. Quantization and pruning in neural models are algorithm-level approximations that align well with approximate hardware.

3. Internet of Things (IoT) Devices

IoT sensors and edge computing nodes are constrained by limited energy resources. Approximate processing cores and memories extend battery life by performing lightweight computations with acceptable accuracy. For example, environmental monitoring sensors can tolerate slight deviations in data readings without affecting decision-making.

4. Big Data Analytics

Data mining and search applications involve processing massive datasets where approximate algorithms drastically reduce execution time. Hardware accelerators with approximate arithmetic provide rapid insights while reducing energy per operation.

5. Bioinformatics and Medical Imaging

Certain bioinformatics algorithms, such as sequence alignment, can leverage approximation in pre-processing stages. Similarly, in medical imaging, approximate filters and reconstruction methods lower computational costs, though care is required in accuracy-sensitive phases.

6. Cryptography and Security (Selective)

Approximation is generally unsuitable for critical encryption operations but can be applied in pre-processing tasks such as key generation randomness testing or noise-tolerant authentication.

CHALLENGES AND DESIGN TRADE-OFFS

Although approximate computing provides promising results in terms of power, area, and performance, its adoption in low-power VLSI design is not without challenges. Designing approximate architectures requires a careful balance between computational efficiency and accuracy, as well as considerations for application-specific requirements. The major challenges and trade-offs are outlined below:

1. Error Quantification and Quality Assurance

One of the most significant challenges is measuring the acceptable error margin introduced by approximation. Unlike conventional digital systems where correctness is binary (correct or incorrect), approximate systems must work within a tolerance threshold. Determining these thresholds depends heavily on the application domain. For example, in multimedia, a small error may not be noticeable to users, but in financial computations, even minor deviations are unacceptable. The trade-off here lies between energy savings and quality degradation—the more aggressive the approximation, the higher the savings but also the higher the risk of quality loss.

2. Reliability and Safety Concerns

In safety-critical systems such as aerospace control, healthcare devices, or automotive electronics, reliability is paramount. Approximate computing can introduce unpredictable behavior if not carefully bounded, leading to system instability or catastrophic failures. The trade-off is between efficiency gains and system dependability. Designers must incorporate selective approximation, ensuring only non-critical modules are approximated, while mission-critical paths remain fully accurate.

3. Hardware Overheads for Error Management

Approximation techniques often require error detection, correction, or monitoring mechanisms to ensure system behavior remains within acceptable bounds. These additional circuits consume extra power and area, which can offset the energy savings gained from

approximation. For example, an approximate adder may save 30% power, but if error correction adds a 15% overhead, the net gain is much smaller. The trade-off here involves choosing the right balance between overhead and approximation aggressiveness.

4. Limited Tool chain and EDA Support

Electronic Design Automation (EDA) tools have been traditionally optimized for accurate computation. Currently, there is limited support for designing, simulating, and verifying approximate circuits at scale. Designers often have to develop custom approximations manually, making the process labor-intensive. This limitation reduces the scalability and adoption of approximate computing in industry. The trade-off is between rapid prototyping with conventional tools versus higher accuracy in custom approximate design frameworks**.**

5. Variability in Application Tolerance

Different applications exhibit different levels of error resilience. For instance, image processing applications can tolerate up to 10–15% error without noticeable quality loss, whereas cryptographic or numerical simulation tasks demand full accuracy. Designing universal approximate hardware that serves multiple application domains is difficult. The trade-off lies in customized hardware per application (which is more efficient but less reusable) versus general-purpose approximate processors (which are more flexible but less optimized).

6. Energy–Accuracy Trade-Off Curve

Approximate computing introduces a continuous spectrum of possible designs—from fully accurate to highly approximate. Each point on this spectrum represents a different energy–accuracy trade-off. Designers must analyze workloads to select the right point. Over-approximation may save energy but render results useless, while under-approximation may not justify the added design complexity. Finding this “sweet spot” is one of the toughest challenges in practical deployment.

7. Integration with Existing Systems

Modern computing systems are highly layered, with interactions between hardware, compilers, and applications. Introducing approximate computing at one layer without coordination across others can cause inefficiencies or unpredictable performance. For

example, approximate hardware must align with error-resilient algorithms at the software layer. The trade-off involves seamless integration versus isolated optimizations that may not deliver system-wide benefits.

8. Ethical and User Acceptance Concerns

Approximate results may be acceptable in casual entertainment or multimedia but could raise ethical concerns in sensitive domains such as healthcare, finance, or scientific research. Users and stakeholders must be assured that approximation does not compromise trust or safety. The trade-off is between pushing aggressive approximations for higher efficiency and maintaining confidence in system outputs.

CONCLUSION

Approximate computing presents a transformative approach for achieving energy efficiency in VLSI systems. By deliberately trading accuracy for efficiency, it addresses the stringent power constraints of modern applications. Approximate architectures, particularly in arithmetic and memory units, demonstrate substantial reductions in power, area, and delay. However, challenges related to error quantification, hardware support, and application-specific optimization must be carefully addressed. The future of low-power VLSI lies in hybrid strategies that combine approximation with adaptive error control to ensure both efficiency and reliability.

REFERENCES

1. Han, J., &Orshansky, M. (2013). Approximate computing: An emerging paradigm for energy-efficient design. *IEEE International Test Conference on Microelectronics Systems Design*, 12(3), 1-6. https://doi.org/10.1109/DATE.2013.6523548
2. Venkataramani, S., Ranjan, A., Roy, K., &Raghunathan, A. (2015). Approximate computing and the quest for computing efficiency. *Proceedings of the 52nd Annual Design Automation Conference*, 1-6.
3. Mittal, S. (2016). A survey of techniques for approximate computing. *ACM Computing Surveys, 48*(4), 1-33. https://doi.org/10.1145/2893356
4. Gupta, V., Mohapatra, D., Raghunathan, A., & Roy, K. (2011). Low-power digital signal processing using approximate adders. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 32*(1), 124-137.

5. Zhu, N., Goh, W. L., & Yeo, K. S. (2010). An enhanced low-power high-speed adder for error-tolerant application. *Proceedings of the International Symposium on Integrated Circuits*, 69-72.

6. Narayanamoorthy, S., Liu, H., & Karri, R. (2014). Energy-efficient approximate multiplication for digital signal processing and classification applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *23*(6), 1180-1184.

7. Jain, P., & Balasubramanian, P. (2019). Approximate computing: Survey, opportunities, and challenges in VLSI design. *International Journal of Low Power Electronics and Applications*, *9*(2), 20-34.