Review

Low-Power Techniques for FPGA and ASIC Design: A Comprehensive Survey

Raj Parikh *

Posted Date: 25 March 2025

doi: 10.20944/preprints202503.1756.v1

Keywords: low-power design; FPGA and ASIC optimization; power gating and scaling; energy-efficient computing

Review

Low-Power Techniques for FPGA and ASIC Design: A Comprehensive Survey

Raj Parikh

Independent Researcher; rparikh356@gmail.com

Abstract: Power efficiency has emerged as a significant constraint in the design of modern digital systems, requiring a holistic approach that includes architectural, register-transfer level, and physical implementation stages. This survey comprehensively reviews low-power design techniques for FPGA and ASIC technologies developed in the last five years. It addresses high-level synthesis optimizations, RTL power-aware methodologies, and dynamic power management techniques, including clock gating, power gating, and voltage scaling. The backend activities discussed in the paper are power-driven placement, multi-threshold and multi-voltage design, leakage minimization, and robust power grid architectures. The quantitative trends in junction scaling and process technology, such as FinFET and GAA transistors, are elaborated along with emerging paradigms of chipset-based integration and machine learning-driven design automation. It also addresses application-specific low-power techniques for application domains such as IoT, AI accelerators, and high-performance computing. The paper ends with directions toward adaptive, context-aware systems that minimize real-time power consumption (given the workload and changing conditions). This work presents a coherent reference for designers, researchers, and engineers to comprehend cutting-edge low-power design methodologies through a unified consolidation of the scientific and industrial body of knowledge.

Keywords: low-power design; FPGA and ASIC optimization; power gating and scaling; energy-efficient computing

Introduction

Recently, the immense growth of battery-enhanced mobile devices, portable electronics, and green computing has created the challenge of designing high-performance and energy-efficient integrated circuits. As this landscape evolves, low-power design has become a significant consideration across FPGA and ASIC design flows. Static power, including leakage, gate, FDSOI, and concurrent logic—directly affects battery life, thermal reliability, and overall system cost, especially as technologies are ushered into deep submicron regimes [1], [2].

Dynamic Power and Leakage Power in VLSI Modern CMOS-based VLSI circuits consume the power mainly in two forms: 1) Dynamic power, which is consumed during the load capacitances charging and discharging during transistor switching; 2) Static (leakage) power, which appears due to sub-threshold currents when transistors are not switching. However, as we drive towards dynamic power optimization, aggressive scaling of the device dimensions and threshold voltages has dramatically raised the supply-side leakage currents, making static power a vital bottleneck for performance at advanced nodes (7nm FinFETs and beyond) [3], [4].

Over the last decade, various design strategies and methodologies have been proposed and thoroughly explored to address these power issues. They encompass architectural and RTL-level strategies such as clock gating, power gating, operand isolation, glitch minimization, and high-level synthesis (HLS)-guided optimizations, all designed to minimize spurious switching activity and maximize the dispatch of data path resources [5], [6], [7]. HLS tools allow designers to experiment with tradeoffs between performance, area, and power much earlier in the abstraction process,

 doi:10.20944/preprints202503.1756.v1

automating low-power aware transformations that have generally been done by hand and are error-prone [8].

At the back end, power-aware techniques such as multi-threshold CMOS (MTCMOS), multi-VDD domains, dynamic voltage and frequency scaling (DVFS), power-aware placement and routing optimizations are of utmost importance for achieving power-efficiency at physical layout level [9], [10]. In particular, redundant switching is a glitch source and can account for as much as 40% of dynamic power in some designs [11]. Hence, these strategies work exceptionally well at reducing glitch power.

Studies show that process-aware and variation-tolerant designs have become even more critical in advanced nanoscale technologies, where variations in power performance and glitch susceptibility are more prominent due to parasitic components and manufacturing non-idealities [1], [4]. In addition, many domain-specific investigations of low-power MAC units, phase-frequency detectors, and linear feedback shift registers (LFSRs) illustrated the extent to which application-aware microarchitectural decisions can yield substantial power savings while maintaining computational throughput [12], [13], [14].

These works show that no single power optimization technique is enough. However, aggressive power reductions can only be attained by a holistic, cross-layer effort (from algorithmic modeling, architectural analysis, RTL design, and physical design spanning multiple processes). Automated EDA tools, UPF/CPF-based power intent modeling, and power-aware verification workflows have all become essential for realizing low-power, sustainable VLSI systems in real-life deployments [15].

Frontend optimizations

1. **Optimizations on High-Level Synthesis (HLS)**

   High-Level Synthesis (HLS) maps high-level language descriptions down to RTL implementations, enabling exploring design trade-offs at a higher level of abstraction. HLS has key low-power benefits, as highlighted below:

   A. **Design Space Exploration:** Early simulation of power within different architectural options (e.g., loop unrolling, pipelining) before RTL commitment. Power estimation can inform performance vs. switching activity trade-offs [1], [2]. However, it can perform multiple operations per cycle as per the HLS due to resource sharing and time-multiplexing. This incurs latency at the expense of lower capacitance and toggling [3]. The scheduler already has power-aware heuristics (lower activity ops won't be executed together). This minimizes peak switching activity and balances the power consumption [4].

   B. **Glitch Approach:** Glitching power is the usage of power in the logic drain in the combinational logic due to the unbalanced paths. Some HLS tools reorders operations to produce stable intermediate signals [5].

   C. **Clock Gating Insertion:** HLS synthesizers can insert clock gates around FSMs and data paths. This prevents unused modules from toggling without designer input [4].

Examples of HLS-based low-power circuits in practice are PLL Phase-Frequency Detectors utilizing AVLS and LECTOR, achieving reduced leakage and active power consumption [6], as well as tailor-made low-power Multiply-Accumulate (MAC) units designed for DSP scenarios [7].

2. **Power-Aware RTL Design Methodologies**

   RTL power-aware design is essential because the synthesized gate-level netlist supports aggressive power gating, clock gating, and voltage scaling.

 doi:10.20944/preprints202503.1756.v1

A. **Power Intent Definition (UPF/CPF):** These standards at RTL specify power domains, retention logic, level shifters, and shutdown conditions. It achieves functional simulation of power behaviors such as sleep, retention, and isolation [8]. Finite state machines (FSMs) using gray coding or one-hot encoding are often used to decrease the number of transitions compared to binary encoding, significantly where only one state changes at a time [9]. RTL can gate logic that fetches data paths that are not used often with enable signals. For example, a multiply unit only utilized during initialization can be gated with conditional execution so that it does not inadvertently toggle [10].

B. **Block Gating at RTL:** Heterogeneous clock gating groups the flip flops sharing the same enable and replaces them in the clock path with gated versions. This meta-level operates at an abstraction where synthesizers can identify and apply using standard library cells [11].

C. **Reducing activity in unused logic:** Unused combinational logic must be tied to the constant (0 or 1) or defined as "don't care" (X) to avoid unpredictable switching [12].

D. **Signal balance:** Unequal delay paths can cause glitches.

E. **RTL pipelining:** Paths are balanced so that transitions happen simultaneously to reduce intermediate power spikes [13].

3. **Switching & Glitch Power Reduction through Logic Restructuring**

Then, after RTL is synthesized, several logic restructuring methods are used to cut down dynamic and static power:

A. **Basics of Balanced Path Decomposition:** At the inputs of gates, the delay is adjusted so that signals have the same time of arrival (TOA), minimizing subtle transitions or glitches. In unbalanced logic, glitch activity can account for 20–40% of dynamic power [4], [5].

B. **Operand Isolation:** Combinational blocks (e.g., multiplier) create downstream logic that may not always be used. Isolation gates are placed to prevent the inputs from toggling unless needed [14].

C. **Another example—is CMOS Gates**: CMOS dynamic power depends on the order of the input toggle—signal toggle rates that influence the switching energy [15].

D. **Duplication of Logic:** High-fanout nets can be broken down into isolated duplicated drivers, decreasing capacitive loads per driver and thus switching power [3].

E. **Gate Merging:** Allowing multiple gates to be combined into a single, more complicated one to minimize the number of internal transitions. It serves to simplify logic and avoid toggling paths [15].

These techniques are equally crucial in FPGAs, where the reordering of LUT inputs and flip–flop staging attempts to align signal arrivals and reduce glitches. Using this don't-care-based optimization approach, dramatic reductions in switching activity have been demonstrated in [5].

4. **Low-Power Optimizations that are functional unit–specific**

 doi:10.20944/preprints202503.1756.v1

Besides global techniques, researchers have proposed power optimization within individual functional blocks:

A. **At the MAC units:** Use gated accumulation registers, isolate operands, and enable gating on the clock. Partial reconfiguration can also unload unused MACs in FPGA implementations [7].
B. **Linear Feedback Shift Registers (LFSRs):** The optimization of taps and XOR/XNOR functions reducing switching plays an essential role in minimizing the LFSR power [16].
C. **Phase Frequency Detectors (PFDs):** Low-power phase frequency detectors are also used to reduce potential leakage during the idle states of PLL feedback loops by using LECTOR or stacking-based logic [6].

5. Early Power Model and Predictive Estimation

Power modeling at the HLS or RTL stage can help avoid late-stage redesigns and budget overruns:

A. **Activity-Based Estimators:** compute the toggle rates based on the simulation or probabilistic models.
B. **Analytical Models:** These models use the input statistics and signal probabilities to obtain power estimates without simulating the full RTL [17].

Correlating with post-synthesis estimates within 10–20% in many studies [1], [4], such early modeling supports design space exploration.

6. Clock Gating

One of the most common techniques for reducing dynamic power is clock gating. Usually, it turns off the clock signal to idle circuit blocks to prevent switching activity.
Everything in Synchronous Systems is clocked. This adds logic (AND/OR gates or embedded gated flip-flops) to prevent the clock from reaching logic when it is not being used [1].

A. **Levels of Gating: Acceptable level:** Individual registers or small data paths gate if they are enabled.
B. **Coarse-grained:** Entire functional blocks, such as multipliers, memory controllers, or peripherals, are disabled when not needed [2].
C. **Synthesis-Aided Gating:** Nowadays, most RTL synthesis tools can infer clock gating automatically and will optimize their timing and area if standard enable signals are found, which allows designers to alleviate some workload [3].
D. **Design Challenges:** If not balanced properly, you'll introduce clock skew. Gated clocks complicate static timing analysis. It needs cautious gating control logic, especially in vector operations or pipelined units [5]

 doi:10.20944/preprints202503.1756.v1

In large ASICs and SoCs, clock gating alone can give about 10–30% dynamic power savings [6].

7. **Power Gating**

Power gating addresses static (leakage) power, which is non-negligible in nanometer CMOS technologies.

Power off the idle circuit partitions via insertion of high-Vth, sleep transistors between the logic and power rails. Such leakage paths are absent when the device is turned off [7].

A. **Implementation Styles:**

Fine-grained power gating – small blocks are connected to their sleep transistor with high savings but increased area.

Coarse-grained power gating: In this case, complete subsystems like CPU cores, DMA engines, or DSPs are switched off together using common sleep transistors [8].

B. **State Retention & Isolation:**

State retention flip-flops maintain the logic state for power-down cycles.

Isolation cells block floating or undefined values from propagating to always-on logic [9].

Control strategy: The power-management units coordinate the "save → power-off → restore" cycle and often stage the wake-up process to limit inrush currents that can corrupt adjacent logic [10].

C. **In ASICs vs. FPGAs:** ASICs use physical sleep transistors for fine-grained power gating. Unlike processors, entry-level solutions (FPGAs) provide only limited power gating at the granularity of bank/block or using partial reconfiguration methods [11]. Power gating can minimize power leakage by 90–95% for blocks in an inactive state, which is advantageous for standby and mobile applications [12].

8. **Techniques for Voltage Scaling (Multi-VDD and DVFS)**

A. Multi-VDD: The multi-VDD design approach divides the chip into voltage islands, with each island powered at a different constant voltage according to performance requirements [4].

Example of designs: 1.0 V CPU core, 0.8 V peripherals, 0.6 V for always-on blocks.

B. Power Savings: Dynamic power scales quadratically with supply voltage **($P \propto V^2$)** but achieving significant savings with relatively small reductions in voltage [13].

C. Level Shifters: For signals crossing from a lower-voltage domain to a higher one. Usually located at domain boundaries and inserted via synthesis or floor planning tools [14].

D. Design Complexity: Requires level shifters, which adds area/power overhead. This necessitates multi-rail power delivery networks and attenuated isolation [15].

E. SoC Usage: Commonly used in mobile and multimedia. SoCs with different performance profiles among blocks.

 doi:10.20944/preprints202503.1756.v1

9. **Dynamic Voltage and Frequency Scaling (DVFS)**

Dynamic Voltage and Frequency Scaling (DVFS) extends voltage scaling to dynamically adjusting voltage and frequency at runtime.

Operation: Heavy load → increase V, max and f, max, light load → decrease both. That helps prevent some from running at peak power when they don't need to. **Power Scaling: Since ($P \propto V^2f$), [17])** in the reduction of dynamic power by reducing both voltage and frequency together [17,16].

A. **P-States:** DVFS employs performance states, known as "Turbo," "Nominal," and "Eco" modes. For that are predefined voltage-frequency pairs.

B. **Implementation Needs:** On-chip regulators or external PMICs that can scale based on control signals. Programmable clock generators (e.g., PLLs or DPLLs) for frequency steering with voltage. Control firmware keeps track of workload and makes transitions [17].

C. **Voltage Settling:** To avoid timing errors, the voltage must stabilize before increasing frequency. Ramp times and response latency limit the speed of DVFS adaptation.

D. **FPGA-specific:** Most FPGAs are not native DVFS enhanced for logic fabrics, though they may enable it for SoC processing subsystems or IO blocks. Some FPGAs use adaptive body biasing or power islands to simulate DVFS behavior [18]. **In variable-load systems such as CPUs, smartphones, and edge AI accelerators, DVFS provides 20–40% average power reduction.**

Backend Techniques

1. **Placement and Routing at Low-Power**

In the physical design flow, Placement and routing (P&R) directly affect dynamic and leakage power at the post-layout phase. Reducing interconnecting length, minimizing switching capacitance, optimizing clock distribution, etc., are essential techniques.

A. **Activity-Driven Placement:** Heuristics aware of toggle rates in modern placement tools are used to minimize the capacitance of high-activity nets. Wires are shorter and switched less frequently, so interconnect capacitance saves dynamic power [1]. For instance, low-activity nets can be deprioritized at the cost of increased wire length by optimizing hot nets [2].

B. **Merge of a Multi-Bit Flip-Flop (MBFF)** MBFFs pack multiple flops into the same physical cell and share the clock driver across the flop cells. This: Reduces clock buffer count.

Unbuffered Minimization and Routing → Minimizing clock net capacitance

It saves power and area with timing that is not affected.

MBFF-based optimizations have shown up to 20–30% power reduction on the clock tree [3]. MBFF insertion is typically performed post-placement based on the physical proximity of single-bit flops.

 doi:10.20944/preprints202503.1756.v1

C. **Placement Along with Gate Sizing and Vt Swapping:** opportunistically shrink over-provisioned gates and swap any non-critical ones with higher threshold voltage (HVT) variants for lower leakage.

2. **Clustering based on IR Drop and Activity**

   High-activity cells will take larger switching currents and, if remote from power straps, can create localized IR drops. To avoid droop-induced delay and wasted power from timing slacks or retries [5], these cells can be placed close to robust power taps or decoupling capacitors.

3. **Retiming and Pipeline Insertion**

   Inserting pipeline registers minimizes the length of combinational paths and localizes switching in long interconnects, especially in ASICs and FPGAs. This helps not only to cut glitch power but also to lower capacitive coupling across long wires [6].

4. **Crosstalk and Coupling-Aware Routing**

   Routing engines add shield wires, increase routing track widths, or use guard bands between high-speed nets to avoid toggling on the victim nets. While this is primarily a measure to improve signal integrity, it directly reduces dynamic power due to crosstalk [7].

5. **Clock Routing Optimization**

   Clock tree/mesh are designed in: Early clock splitting, Minimal skew buffering, Distributed gating (clock-gate cells next to loads)

   Thus, high power efficient distribution of the clock is guaranteed. FPGA toolchains emulate these gains with regional clock routing and power-aware packing [8].

6. **Multi Threshold-Voltage (Multi-Vt) Annihilation:** Multi-Vt design (combination of different levels of threshold voltage in cells) helps to balance leakage vs. performance:

   LVT (Low-Vt): Fast, leaky

   HVT(High-Vt): Slow, high-leakage

   SVT (Standard): Mid-point

7. **MTCMOS Strategy:** Tools typically hold LVTs on critical paths for a final leakage recovery pass and switch non-timing critical cells to SVT or HVT types [9].

   Leakage Reduction Impact: HVT cells leak 5–10× less than LVT cells [10]. According to research [18], 5–15% (or more) of LVT cells can completely dominate leakage if not controlled. The thermal and leakage profiles also vary across clusters [11].

8. **Advanced Process Support:** Such options exist in 7nm and below via metal gate work function engineering, allowing fine-grained leakage control [12].

Power Grid Design and IR Drop Management

A. **IR Drop and Power Efficiency: Voltage drops in the grid: Logic Slower → More Timing Margin → Increased Dynamic Power**

   Causes short circuit power during transitions

   Overvoltage of triggered voltage from AVS, increasing global power [13]

 doi:10.20944/preprints202503.1756.v1

It is expected to keep VDD drop to less than 5%.

Designers mitigate this with: Wider metal straps, Strategic C4 bump placement, Decoupling capacitors are distributed [14]

B. **EM-Aware Routing**

Note: High current densities are prone to electromigration (EM). Tuning inrush currents from gated blocks, for example, requires routing power grids with redundant vias, wide wires, and staged power-ups [15].

C. **Current-Aware Floor planning**

For example, high-switching units (e.g., GPU cores) are positioned nearest to the voltage sources. Power-up blocks are distributed spatially to minimize clustered power-ups [16].

Leakage reduction and Stand-by modes

Multi-Vt and power gating are not the only static power suppression techniques.

A. **Reverse Body Bias (RBB):** Both FDSOI and triple-well bulk processes offer RBB, which raises Vt dynamically during idle modes to minimize leakage. Per-process-dependent effectiveness is further reduced in FinFETs [17].

B. **Transistor Stacking:** Intermediate node biasing makes two or more off transistors in a series leak less than one alone. Libraries use this in gate-level topologies (i.e., NAND/NOR) or dedicated sleep cells [18].

C. **SRAM Light-Sleep and Retention:** Memory is a leakage hotspot. Solutions include: WL/BL biasing in light sleep mode, Low power state retention SRAMs, Unused memory macros power-down control [18]

Process Variation Effect on Power

Vt, oxide thickness, and length affect leakage and dynamic power.

A. **Leakage Sensitivity:** This leads to exponential variations in leakage over the die with varying Vt changes. Guard banding is used for guard banding in statistical corners such as "FF slow-leaky" [17].

B. **Adaptive Compensation: To minimize worst-case overheads:** Adaptive Voltage Scaling (AVS) adjusts supply voltage at a per-chip granularity. Dynamic post-silicon tuning methods [15] tune bias or clock dynamically

Temperature and Age Differences

Leakage doubles approximately every 10 degrees Celsius [15]. Hot spots result in static power increments for the corresponding aggressor circuit, and aging (NBTI/PBTI) increases Vt, which reduces the path speed and, thus, affects the dynamic power balance [12].

Industry Trends and Future Directions

1. **Software-Controlled Power Management**

Software-managed power control is the new way for all modern SoCs — especially mobile and data center SoCs. Understanding and implementing software-level runtime architectures, where proper OS, firmware, or hypervisors dynamically handle DVFS, clock/power gating, and per-core states. With UPF and CPF gaining broad design flow

 doi:10.20944/preprints202503.1756.v1

adoption, formal verification, power-aware simulation, and emulation are used to verify complex power-management interactions [6, 15]. Architectural designs feature more independent power domains, so architects, RTL designers, and software developers must work together closely.

The introduction of FinFETs (16nm/7nm) and now GAA transistors (3nm) [10, 12] have improved electrostatic control and dramatically reduced leakage. But FinFETs came with higher gate capacitance and less voltage scaling headroom. Because they are variable, this H endpoint region is not used well (optimal 0.4–0.5V). Consequently, multiple VT flavors will be available across the SQ level and standard cell track heights to trade speed with foundry power. With GAA nanosheets, designers have yet another option for leakage vs. performance control. These device innovations are also matched with increasing power density, and thus, thermal-aware design, dark silicon, and fine-grained throttling are becoming more relevant than ever [14, 22].

2. **Heterogeneous Integration and Chiplets**

As Moore's Law stops, the industry moves to chipsets packaged in 2.5D (interposer) or 3D. These allow mixing process technologies (i.e., pairing a low-power IO die with a performance core die). 3D stacking lowers interconnect power but makes dissipating heat and delivering power vertically more complex [16]. Internal regulators now enable per-chipset DVFS, reducing conversion losses and improving response time. Power management chipsets may also be included in future systems to control the supply and monitor locally.

3. **Automating Machine Learning and Explorative Data Analysis**

New AI-enhanced design tools are hitting the market to search for vast parameter spaces for power reduction. ML algorithms now aid in:

Identifying the best DVFS partitions.

Tuning of multi-Vt thresholds and placement settings [21].

One proposed approach uses reinforcement learning to optimize the voltage island creation process. Power-aware clustering also helps these neural networks in placement legalization. Although new, ML-powered CAD can help leach a low-power design more quickly.

This is a more general-purpose technique that, if done appropriately, is widely applicable.

4. **Different domains drive innovation**

Sensor and IoT are centered on below-threshold functionality and harvesting energy. Analog/mixed-signal blocks are duty-cycled for very low-power standby. Energy per inference is excellent, but AI/ML accelerators go further. Examples are approximate computing (e.g., low-precision MACs), power gating of unused units, and custom data paths [8, 18]. Microcontrollers now widely enable near-threshold execution for background tasks that can quickly transition to nominal mode for performance bursts.

 doi:10.20944/preprints202503.1756.v1

In summary, there is much more progress in low-power design beyond architectural and transistor-level techniques. And software, machine learning, packaging, and emerging devices are increasingly joining the fray. The newer systems will be adaptive and context-aware; they'll dynamically observe workload, temperature, and silicon characteristics to tune power usage on the fly. The need for energy-efficient digital design will only escalate as data centers get more significant, the Internet of Things flourishes, and edge computing expands. The new wave of design automation needs to put forth intelligent and self-optimizing flows so a balance between algorithmic expertise, architectural features, and physical domain expertise can work together to reduce energy per operation without impacting performance.

Conclusions

Power efficiency has become a first-order constraint in digital design, driving a Panoply of techniques ranging from high-level architectural decisions to transistor-level optimizations. These low-power, low-level optimizations for FPGAs and ASICs alike have significantly been consolidated in the past five years. At the front end, strategies such as power-aware high-level synthesis, RTL coding, and dynamic power management (clock gating, power gating, and DVFS) enable designers to reduce unwanted switching and customize a design's activity for the workload. With backend strategies like power-driven placement, multi-Vt/multi-VDD, and strong power distribution, the physical implementation is also tuned to remove unnecessary power/activity in the driven silicon. A comparative analysis of our various techniques makes it clear that a combination of them is required since no one method can achieve the aggressive power targets required by 21st-century applications on its own. Instead, designers combine these techniques to strike against power on multiple fronts: lowering the work performed, work per operation, and leakage when idle [1–3, 5, 6].

As the recent literature and industry-level survey suggests, the tendency of low-power design to become more automated and sophisticated seems clear. Formal power intent standards (UPF/CPF) [9] that provide a typical representation of power intent have been adopted in practice, and power as a design concern (in all aspects of design, including verification) has achieved significant maturity [15]. The growth of multi-domain, software-managed power schemes in large SoCs has driven the evolution of tools and methodologies to manage the complexity. On the one hand, the relentless evolution of semiconductor technology brings challenges and opportunities: new devices (FinFETs/GAA transistors) and 3D integration (e.g., die stacking) require an update of some topologies adopted to implement power strategies; however, they also open up new knobs to turn (e.g., wider Vt range [10], faster DVFS transitions with the use of integrated voltage regulators [12,20,22]), etc.

The shift for FPGAs, traditionally a flexible but power-inefficient solution, shows the gap between traditional and specialized solutions narrowing — modern FPGAs on new nodes provide architectural low-power options and a richer software ecosystem for proper power handling [2, 8, 16]. This makes FPGAs a practical choice even in power-oriented use cases (such as mobile units and data center accelerators), algorithmic optimizations are incorporated to take advantage of the reconfigurability to establish coupled low-power data paths. ASIC design, however, remains at the edge with fine-grained control over the individual transistors. Cutting-edge-state ASICs may use per-clock-cycle power gating, adaptive body biasing for dynamic leakage trimming, or machine-learning-based controllers for workload prediction and proactive voltage scaling [14, 17].

In the future, routes could depend on considerably more flexible and context-conscious power control. More sophisticated on-chip monitors (for workload, temperature, and process) will also continue to enable feedback into power control loops that can optimize power in real-time. Another avenue is designed for ultra-low-power standby: as the IoT kicks in, devices that draw nanowatts in sleep and then wake briefly, sometimes milliseconds, to do a job are going to be essential. This may also trigger the development of non-conventional design methods such as power gating with non-volatile state retention [16], energy-harvesting friendly circuits [17], and extreme voltage scaling (sub-threshold operation) [18]. In supercomputing, we seek to sustain Moore's Law exponential

 doi:10.20944/preprints202503.1756.v1

improvement in machine compute rate under stringent "power walls" [12, 13] (with likely more specialized accelerators and exotic "cooling"/power supply designs (liquid cooling with localized voltage conversion, etc.) [14].

So overall, low-power design is a complex issue that needs a cross-stage, cross-domain harnessing of techniques. These techniques have been refined over the past five years, thriving in leading products from smartphones to FPGAs in cloud servers, and both academic investigations and industry best practices enrich the breadth of knowledge. Ordered-of-magnitude improvements in energy efficiency are accessible via high-level optimizations, RTL instantiation with profound sagacity, and state-of-the-art backend implementations. The value of low-power design will only increase as we move into an epoch in which energy efficiency and environment-friendly computing are of the utmost concern. This survey highlighted the significant approaches and their compromises. It will serve the readers as a basis for understanding how modern FPGAs and ASICs satisfy the stringent power constraints of today's workloads. Further, HW/SW throw is going to keep the downward pressure on power consumption over the power and electricity budgets without compromising performance.

References

1. Vaithianathan, M., Patil, M., Ng, S. F., & Udkar, S. (2024). Low-power FPGA design techniques for next-generation mobile devices. *ESP International Journal of Advancements in Computational Technology, 2*(2), 82–93.
2. Dai, S., & Campbell, K. (2015). High-level synthesis for low-power design. *IPSJ Transactions on System LSI Design Methodology, 8*, 12–25.
3. Mansuri, N., Vakhare, V., & Shah, K. (2019, June 26). Low power implementation techniques for ASIC physical design. *EDN Network*. https://www.edn.com/low-power-implementation-techniques-for-asic-physical-design/
4. Synopsys, Inc. (n.d.). What is glitch power? *Synopsys Low Power Glossary*. Retrieved January 2023, from https://www.synopsys.com/glossary/what-is-glitch-power.html
5. Dillinger, T. (2020, June 26). Multi-Vt device offerings for advanced process nodes. *SemiWiki*. https://semiwiki.com/semiconductor-services/ic-implementation/285912-multi-vt-device-offerings-for-advanced-process-nodes/
6. Foster, H. (2021, January 27). Part 11: The 2020 Wilson Research Group functional verification study – Low power trends. *Verification Horizons Blog (Siemens EDA)*. https://blogs.sw.siemens.com/verificationhorizons/2021/01/27/part-11-the-2020-wilson-research-group-functional-verification-study-low-power-trends/
7. Chen, W., Wang, Y., Xu, J., & Pedram, M. (2020). Performance comparisons between 7-nm FinFET and conventional bulk CMOS standard cell libraries. *IEEE Transactions on Circuits and Systems II: Express Briefs, 67*(12), 3562–3566. https://doi.org/10.1109/TCSII.2020.2986084
8. Maxfield, M. (2006, September 30). Reducing power with an advanced multi-Vdd methodology. *EE Times*. https://www.eetimes.com/reducing-power-with-an-advanced-multi-vdd-methodology/
9. Semiconductor Engineering. (2020). Dynamic voltage and frequency scaling (DVFS). *Semiconductor Engineering Knowledge Center*. https://semiengineering.com/knowledge_centers/processes/power/dynamic-voltage-and-frequency-scaling-dvfs/
10. Chatti, K., Rima, A., & Lahiani, F. (2022). Dynamic voltage and frequency scaling and duty-cycling for ultra low-power wireless sensor nodes. *Electronics, 11*(24), 4071. https://doi.org/10.3390/electronics11244071
11. Garima Thakur, Shruti Jain, & Harsh Sohal (2022). Current issues and emerging techniques for VLSI testing – A review. *Measurement: Sensors, 24*, 100497. https://doi.org/10.1016/j.measen.2022.100497
12. Premananda, B. S., & Sreedhar, S. (2022). Low-power phase frequency detector using hybrid AVLS and LECTOR techniques for low-power PLL. *Advances in Electrical and Electronic Engineering, 20*(3), 294–301. https://doi.org/10.15598/aeee.v20i3.4593
13. Srinivas, L., Chandrakala, & Kumar, G. K. (2021). A review on low power area-efficient architecture for linear feedback shift registers. *Dogo Rangsang Research Journal, 11*(1), 328–330.

 doi:10.20944/preprints202503.1756.v1

14. Haq, S. U., & Sharma, V. K. (n.d.). Challenges in low power VLSI design: A review. *Proceedings of the 5th International Conference on Electronics, Communication and Aerospace Technology (ICECA 2021)*. IEEE. https://doi.org/10.1109/ICECA52323.2021.9676055
15. Varadharajan, S. K., & Nallasamy, V. (2017, March). Low power VLSI circuits design strategies and methodologies: A literature review. *Proceedings of the IEEE Conference on Emerging Devices and Smart Systems (ICEDSS)*. https://doi.org/10.1109/ICEDSS.2017.8073702
16. Raut, K. J., Chitre, A. V., Deshmukh, M. S., & Magar, K. (2021). Low power VLSI design techniques: A review. *Journal of University of Shanghai for Science and Technology, 23*(11), 172–180.
17. Malipatil, S. (2017). Review and analysis of glitch reduction for low power VLSI circuits. *International Journal for Research in Applied Science & Engineering Technology (IJRASET), 5*(11), 1386–1389.
18. Sahu, A. K., Samanth, R., Mendez, T., & Nayak, G. S. (2021, July). VLSI design techniques for low power MAC unit: A review. *AIP Conference Proceedings, 2358*, 050013. https://doi.org/10.1063/5.0057909