Journal of Advance Research in Science And Engineering
http://iphopen.org/index.php/se

Journal of Advance Research in Science and Engineering

https://iphopen.org/index.php/se

Online ISSN: 3050-8797 Print ISSN: 3050-9270

PUBLIC LIBRARY original article https://iphopen.org/ editor@iphopen.org

POWER MANAGEMENT IN EMBEDDED AI SYSTEMS: A MULTI-LAYERED APPROACH FOR EDGE COMPUTING APPLICATIONS

SENTHIL NATHAN THANGARAJ[*]
AMD, USA

*Corresponding Author: **Senthil Nathan Thangaraj**

Abstract
Power management is a major challenge for embedded AI systems at the network edge. These systems must run machine learning workloads under tight energy limits.

An effective solution needs a multi-layered approach. Key elements include the Power State Coordination Interface (PSCI), secure firmware, and Linux kernel features for runtime control. Core techniques such as dynamic frequency scaling, clock gating, suspend/resume, and memory or accelerator-specific optimizations further improve efficiency.

Environmental factors add to the challenge. Automotive and industrial systems must meet strict thermal limits. Battery-powered devices face even tighter energy budgets. Both require adaptive control strategies. From a software perspective, effective methods include specialized kernel drivers, standardized power APIs, and optimizations such as dynamic logic gating and real-time power monitoring. When combined, these enable power-efficient AI systems that maintain reliable performance while staying within thermal and energy boundaries.

Keywords: Edge Computing, Embedded AI Systems, Power Management, System-on-Chip Architectures, Thermal Constraints

DOI:-10.5281/zenodo.17678252 **Manu script # 371**

1. Introduction

AI at the network edge brings strict power management challenges for embedded systems. These platforms must run complex machine learning models under tight energy and thermal limits [1]. To handle this, they use heterogeneous SoCs that combine CPUs, GPUs, NPUs, and other accelerators. Each of these has a different power profile.

Conventional power management methods, built for general-purpose computing, often fall short for AI workloads. Edge AI tasks are bursty. They shift between idle periods and intense computation. This behavior requires careful coordination of power states to stay efficient. In battery-powered devices, runtime and reliability depend directly on power use [1].

The move to edge computing makes power efficiency even more critical. Processing data locally reduces latency. It enables real-time decisions and lowers reliance on the cloud [2]. Applications such as autonomous driving, industrial automation, and smart monitoring all need consistent performance within strict power and timing limits.

This paper explores a multi-layered approach to power management in embedded AI systems. We look at coordination across firmware, bootloaders, and operating systems. Standardized interfaces help synchronize power states across different components. We also study how dynamic resource allocation can balance performance, thermal limits, and energy efficiency. Finally, we present design principles for reliable and power-optimized edge AI systems.

2. Multi-Layered Power Management Architecture in AI SoCs

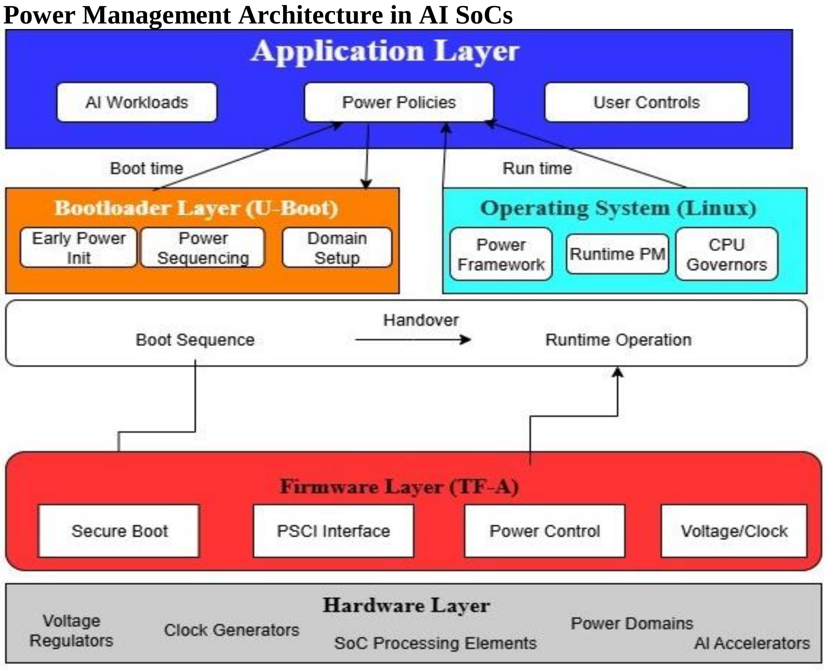

Fig 1: Multi-Layered Power Management Architecture [3, 4]

Firmware Layer

Modern AI SoCs manage power across several abstraction levels. At the lowest level, firmware provides direct control over hardware. Trusted Firmware (TF-A) creates a secure environment that regulates access to voltage regulators, clock generators, and power domain controllers [3]. This layer acts as an isolation barrier between critical power functions and higher-level software. The separation keeps the system stable during complex power state changes across CPUs, GPUs, and accelerators.

Specifications (PSCI)

The Power State Coordination Interface (PSCI) defines standard calls between firmware and the operating system. These calls hide SoC-specific details and let developers write portable software across different platforms [3]. PSCI supports CPU idle and suspend states, cluster-level power-down, and full system shutdown or reset. Although hardware-agnostic, vendors can extend it with workload-aware features for AI platforms.

Bootloader Frameworks
Bootloaders configure essential power settings during startup, before the OS takes control. They set voltage rails, program clocks, and isolate power domains needed by accelerators and memory [3]. Bootloaders also sequence power-on across domains and enforce safety checks to prevent damage from unstable supply conditions.

Operating System Integration
In heterogeneous AI SoCs, the Linux kernel manages runtime power across CPUs, GPUs, and NPUs [4]. Subsystems such as cpufreq, cpuidle, devfreq, and generic power domains (genpd) adjust performance points dynamically. Governors like *schedutil* raise or lower operating frequencies based on workload. The OPP and energy model frameworks give the kernel predictive control of voltage and frequency scaling.
Kernel power control also includes algorithms that anticipate future demand from workload patterns. Runtime PM subsystems manage individual components while respecting interdependencies defined in the device tree. This coordination ensures stable power transitions while allowing aggressive optimization during idle periods [4]. Device drivers then translate abstract requests into hardware register operations for accelerators, memory controllers, and peripherals.

Application Layer
At the top layer, applications and policies influence how power is managed. Workloads such as vision, speech, or sensor fusion create highly variable demand. Middleware or safety frameworks enforce minimum performance levels for critical tasks. User preferences, such as performance or power-saving modes, guide further allocation. Together, these factors connect real-world use cases with lower-level mechanisms to balance efficiency and reliability.

3. Core Power Management Technologies and Mechanisms

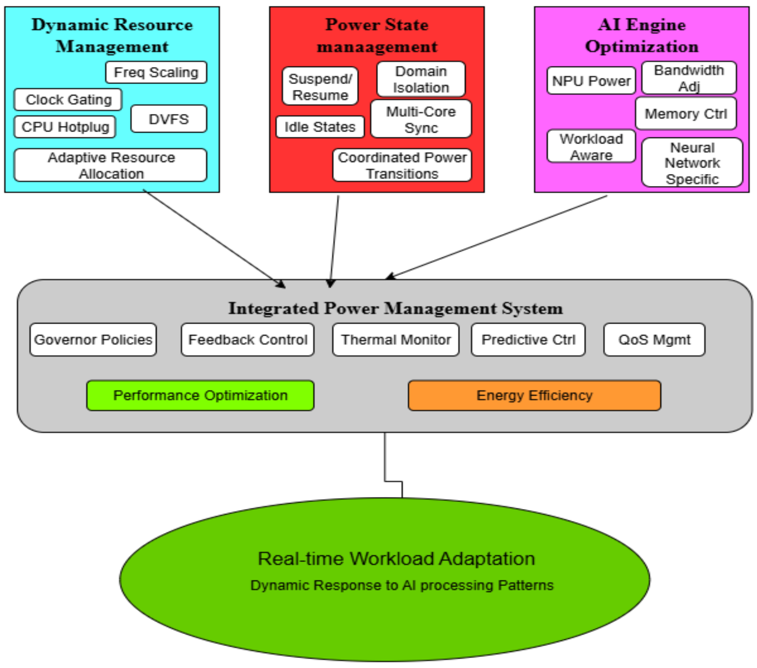

Fig 2: Core Power Management Technologies [5, 6]

Dynamic Resource Management
Modern AI SoCs use advanced methods to allocate resources based on workload and system state. CPU frequency scaling lets processors adjust operating speed to balance performance with power limits [5]. These decisions rely on models of performance and energy use to keep efficiency high across different workloads.

Dynamic voltage and frequency scaling (DVFS) changes both voltage and frequency across CPUs, GPUs, and NPUs. Feedback loops monitor thermal and performance margins to keep units efficient [5]. Clock gating adds another layer of savings by shutting off unused logic. This can happen at many levels, from individual gates to entire subsystems, depending on activity.

Power State Management
Fine-grained power states allow transitions across domains while keeping the system coherent. Suspend and resume operations save only essential state, letting non-critical blocks enter deep low-power modes [6]. Algorithms weigh how much state to keep against wake-up speed and energy trade-offs.
Multi-core systems need careful synchronization for power state changes. Caches, shared resources, and communication must remain consistent across cores [6]. Power domain isolation gives independent control over functional blocks. This prevents interference between active and inactive circuits and maintains signal integrity.

AI Engine-Specific Optimizations
AI workloads benefit from power methods tailored to neural processing. Memory controllers cut power by adapting to data movement patterns in inference and training. Techniques include bandwidth allocation, Temperature-Compensated Self-Refresh (TCSR), Fine-Granularity Refresh (FGR), per-bank refresh, and deep power-down modes [5]. Memory systems work with AI accelerators to anticipate access and adjust states proactively.

AI accelerators also use workload-aware power policies. These assign power budgets based on the needs of each neural network layer while meeting thermal limits [6]. Power allocation frameworks use detailed models of consumption to predict needs from network structure and input data.

4. Thermal and Power Constraints in Edge AI Applications

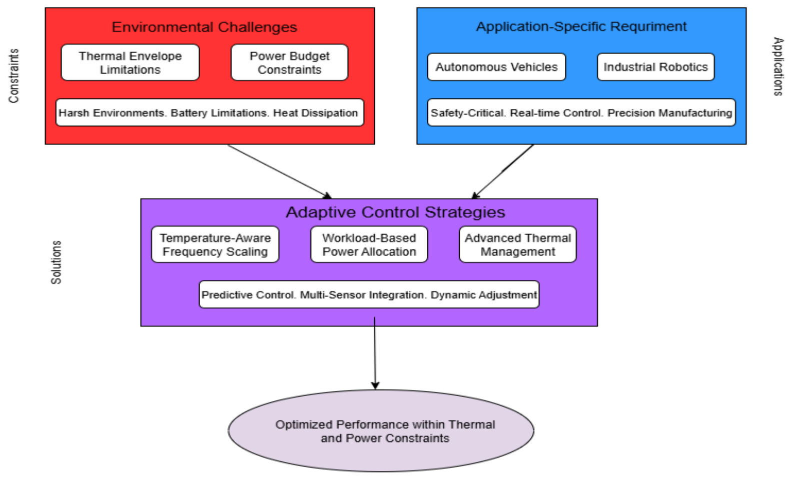
Constraints
Environmental Challenges
Thermal Envelope Limitations
Power Budget Constraints
Harsh Environments. Battery Limitations. Heat Dissipation
Application-Specific Requriment
Autonomous Vehicles
Industrial Robotics
Safety-Critical. Real-time Control. Precision Manufacturing
Applications
Solutions
Adaptive Control Strategies
Temperature-Aware Frequency Scaling
Workload-Based Power Allocation
Advanced Thermal Management
Predictive Control. Multi-Sensor Integration. Dynamic Adjustment
Optimized Performance within Thermal and Power Constraints

Fig 3: Thermal and Power Constraints in Edge Al [7, 8]

Environmental Challenges
Edge AI systems face tough environmental conditions that affect both power and thermal design. Automotive and industrial deployments often run across extreme temperatures, so they need advanced thermal management to stay stable [7]. The challenge is greater in enclosed spaces with little airflow or heat dissipation. In these cases, thermal throttling becomes critical to balance performance while keeping core AI functions alive.

Battery-powered edge devices face additional limits. Portable AI hardware must work within small energy reserves, which forces tradeoffs between computing power and runtime [7]. Power budgeting algorithms address this by giving priority to essential AI processing while cutting power to non-critical functions. Good power control also considers battery discharge behavior, thermal effects on the battery, and changing workload demands to keep efficiency high across scenarios.

Application-Specific Requirements
Autonomous vehicles bring unique challenges because their AI tasks are safety-critical. Systems for perception, decision-making, and control must keep running reliably despite temperature swings or heavy power demand [8]. The architecture must handle compute-heavy jobs like sensor fusion, object detection, and path planning in real time. At the same time, it has to stay thermally stable inside compact enclosures with little cooling.

Industrial robots also need tailored solutions. These systems mix real-time control with machine learning. They must meet strict timing for motion and safety while also adapting to tasks like vision, inspection, and optimization [8]. Power management here must prioritize critical control functions while adjusting power for AI inference to cut overall consumption.

Adaptive Control Strategies
Thermal-aware scaling uses sensors across the SoC to track temperature. Control logic then adjusts frequency and voltage to stay within safe thermal limits [7]. Predictive methods go further by anticipating when violations might happen and adjusting settings early to avoid performance loss.

Workload-aware power allocation considers both AI patterns and compute demand. It applies dynamic budgets that shift power toward active tasks while keeping within thermal and energy boundaries [8]. These adaptive strategies allow edge AI systems to maintain performance across different environments and workloads.

5. Case Studies and Evaluation of Power Management in Edge AI Systems

Power management challenges and strategies are best understood through real-world case studies. Autonomous vehicles, industrial robotics, and drones each demonstrate how layered power management operates under different constraints. Across all domains, the guiding principle is clear: safety-critical and mission-critical systems must always run, while non-essential workloads can be scaled back when resources are tight.

5.1 Autonomous Vehicles

In vehicles, safety drives every power decision. Systems for perception, decision-making, and control must operate reliably under fluctuating workloads and thermal limits.

Driving Scenario: Approaching a busy intersection
When a vehicle approaches a busy intersection, demand on its computing stack spikes. Multiple sensors capture high-resolution data, and the AI pipeline must process it in real time for safe navigation. Power management adapts across layers to ensure responsiveness:

- **Sensor Input Surge:** Cameras, LiDAR, and radar produce dense data streams. The perception pipeline sends workload hints through application-level QoS, such as frame deadlines.
- **Application Hints to Kernel:** QoS constraints map to system-level settings like minimum CPU frequency or memory bandwidth. The kernel's *pm_qos* interface enforces them to prepare accelerators before data arrives.
- **Driver Feedback:** As workloads increase, accelerator drivers report utilization metrics. If load is heavy, they signal the *devfreq* governor to raise operating points.
- **Kernel-Level Adjustments:** The kernel increases CPU and accelerator frequencies. The thermal framework caps non-critical tasks such as infotainment graphics to preserve power.
- **Firmware Enforcement:** Requests pass through firmware (SCMI/PSCI). Firmware checks them against safety rules and ensures accelerators stay above certified minimum levels. Voltage regulators adjust gradually to avoid instability.
- **Sustained High Load:** During the intersection, perception and planning CPUs run at elevated performance. Safety-critical tasks continue, while background tasks remain throttled.
- **Return to Cruise Mode:** After the vehicle clears the intersection, workloads ease. Governors scale accelerators down, gate idle domains, and lower CPU frequencies to save energy while keeping readiness.

This layered response, application hints, kernel governors, driver feedback, and firmware validation ensure safety and determinism. Critical perception and planning always get priority, while less important tasks scale back first.

5.2 Industrial Robotics

Industrial robots combine real-time control with machine learning workloads. Unlike vehicles, which focus mainly on navigation, robots must balance deterministic control loops with variable tasks such as vision-based inspection or predictive maintenance.

Power Challenges:

- **Mixed workloads:** Motion control needs strict timing, while AI-based inspection and optimization are bursty.
- **Harsh environments:** Heat, dust, and poor airflow limit cooling.
- **Safety-critical tasks:** Power fluctuations cannot disrupt safety sensors or motion control.

Management Strategies:
Robots keep control loops fixed at guaranteed frequencies while dynamically adjusting accelerators for vision and ML tasks. Power domain isolation ensures AI inference can be throttled without affecting safety-critical control. Predictive algorithms anticipate inspection cycles and allocate power in advance.

Outcomes:
Experiments with robotic arms show that isolating power domains cuts energy use by up to 20% without missing control deadlines. This demonstrates the importance of layered control across firmware, kernel, and applications.

5.3 Drones and UAVs

Drones operate under the strictest energy budgets of all. Every watt saved extends flight time, yet drones must still process high-resolution sensor data in real time to remain stable and complete missions.

Power Challenges:

- **Tight energy limits:** Small batteries restrict missions to minutes.
- **Real-time sensing:** Stabilization requires continuous low-latency feedback, while navigation demands constant inference.
- **Variable environments:** Outdoor wind, temperature, and mission load affect both computation and power.
- **Management Strategies:** Drones use lightweight scheduling frameworks that prioritize stabilization and navigation. Non-critical tasks such as image recording are throttled. Accelerators are power-gated when idle. Some systems offload heavy workloads to ground stations when reliable connectivity is available.

Outcomes:
Field tests show adaptive throttling extends flight time by 15–25% while maintaining navigation accuracy. Hybrid designs that combine onboard AI with selective cloud offloading perform even better but rely on strong communication links.

5.4 Comparative Insights

Looking across vehicles, robots, and drones highlights both shared principles and domain-specific differences:

- **Shared Principles:**
  - Layered coordination across firmware, OS, and applications is essential.
  - Safety- and mission-critical workloads always get priority.
  - Adaptive algorithms improve efficiency by anticipating workload demand.
- **Domain Differences:**
  - **Vehicles:** Large power budgets but strict thermal and safety limits.
  - **Robotics:** Mix deterministic real-time control with bursty AI inference.
  - **Drones:** Extreme energy constraints make every optimization count.

5.5 Lessons Learned

These case studies confirm that layered power management applies across domains, but strategies must be tailored to workload patterns and environmental conditions. Firmware ensures safe isolation, bootloaders establish stable startup, kernels perform runtime adjustments, and applications express workload priorities.
Future evaluation should include shared benchmarks that represent automotive, robotic, and drone workloads. Such testbeds would allow fair comparison of strategies, encourage standardization, and highlight best practices for industry adoption.

6. Future Directions and Challenges

As edge AI expands, power management must evolve beyond today's reactive methods. Future systems will require predictive control, stronger security, standardization, and adaptation to new workloads and energy sources.

AI-Driven Predictive Power Control

Current systems scale frequencies only after utilization rises, which can cause latency. Predictive power control uses AI models to forecast demand from workload history, sensor data, or thermal trends. Reinforcement learning and neural predictors can prepare accelerators in advance and reduce frequent switching. These models must remain lightweight and accurate under varied workloads.

Security and Reliability

Firmware and kernel power interfaces create new attack surfaces. Malicious code could disable cores or drain batteries. Future designs must treat power control as a security boundary by validating firmware calls, isolating untrusted applications, and enforcing cryptographic checks. Reliability also demands watchdogs, redundancy, and safe power defaults to prevent data corruption or unstable transitions.

Cross-Platform Standardization
SoCs expose power features differently, complicating software portability. Expanding PSCI/SCMI, improving Linux frameworks like cpufreq and genpd, and building cross-vendor APIs will reduce fragmentation. Standards must balance common interfaces with room for vendor-specific optimizations.

Evolving Workloads and Energy Sources
Workloads are shifting from vision-only inference to multi-modal tasks combining speech, vision, and sensor fusion. Each stresses power differently, requiring allocation by workload type as well as device. At the same time, edge devices in remote or mobile environments may rely on solar or energy harvesting. Algorithms must adapt not only to demand but also to fluctuating supply, scaling back or boosting performance as conditions allow.

Open Challenges
Key issues include lightweight prediction, thermal modeling, workload isolation, real-time validation, and fair benchmarking. Addressing these will require collaboration across hardware, OS, and applications, supported by both industry and academic research.

Conclusion
A multi-layered approach to power management addresses the major optimization challenges in modern edge AI systems. Power is no longer a secondary issue; it is a core design requirement that spans every stage, from firmware-level control to application-level strategies.

Coordination between application, firmware, bootloaders, and the operating system allows fine-grained power state management across CPUs, GPUs, and accelerators. This ensures real-time performance while adapting to workload shifts. Looking ahead, AI-driven prediction models show promise for anticipatory power control that adjusts before performance is affected.

Industry is also working toward cross-platform standards. These create unified interfaces that support different SoCs while still leaving room for vendor-specific optimizations. At the same time, new edge AI architectures are introducing demands that will require more advanced solutions for managing heterogeneous processing and dynamic workloads.

Engineers designing embedded AI systems should treat power management as a first-order constraint. Multi-layered designs that coordinate across hardware and software abstractions, while staying flexible for changing workloads and environments, will deliver the most efficient and reliable results.

References

1. Jonathan G Koomey et al., "Implications of Historical Trends in the Electrical Efficiency of Computing," ResearchGate, 2011. Available: https://www.researchgate.net/publication/224128141_Implications_of_Historical_Trends_in_the_Electrical_Efficiency_of_Computing
2. Pedro García López et al., "Edge-centric Computing: Vision and Challenges," ResearchGate, 2015. Available: https://www.researchgate.net/publication/282434420_Edge-centric_Computing
3. ARM Developer, "Arm Power State Coordination Interface Platform Design Document". Available: https://developer.arm.com/documentation/den0022/latest/
4. Robert Love, "Linux Kernel Development," Addison-Wesley, 2010. Available: https://www.doc-developpement-durable.org/file/Projets-informatiques/cours-&-manuels-informatiques/Linux/Linux%20Kernel%20Development,%203rd%20Edition.pdf
5. Dominik Brodowski, "CPUFreq - CPU frequency and voltage scaling code in the Linux(TM) kernel," Kernel. Available: https://docs.kernel.org/cpu-freq/index.html
6. Paul E. McKenney, "Is Parallel Programming Hard, And, If So, What Can You Do About It?" arXiv:1701.00854v6, 2023. Available: https://arxiv.org/abs/1701.00854
7. Massoud Pedram and Shahin Nazarian, "Thermal Modeling, Analysis, and Management in VLSI Circuits: Principles and Methods," IEEE, 2006. Available: https://weble.upc.edu/ifsin/Block5/paper_proc2.pdf
8. Parthasarathy Guturu and Bharat Bhargava, "Cyber-Physical Systems: A Confluence of Cutting Edge Technological Streams". Available: https://www.cs.purdue.edu/homes/bb/CPSReviewPaper.pdf
9. Jonathan Corbet et al., "Linux Device Drivers," lwn.net. Available: https://lwn.net/Kernel/LDD3/
10. Greg Kroah-Hartman, "Linux Kernel in a Nutshell," O'Reilly, 2006. Available: https://theswissbay.ch/pdf/Gentoomen%20Library/Operating%20Systems/Linux/O%27Reilly%20Linux%20Kernel%20in%20a%20Nutshell.pdf