MANTECH Publications

Journal of Embedded Systems & Its Applications

Volume 5 Issue 1, January-June 2020

Dynamic Voltage Scaling & Power Management Algorithms in Embedded and Computing Systems

Ramesh Sharma[1], N. Pradeep Kumar[2], Farhan Ali[3]

Professor, Associate Professor

Department of Embedded Systems

Acharya N.G. Ranga Agricultural University, Hyderabad

Rameshsharma27@rediffmail.com[1], n121pradeepk@yahoo.com[2], farhanaaa2@gmail.com[3]

Abstract

Power consumption has become a critical design concern in modern embedded systems, mobile devices, IoT nodes, and high-performance processors. With battery operated devices and dense computing platforms, efficient power management techniques are required to extend operational lifetime and reduce thermal stress. Dynamic Voltage Scaling (DVS) is one of the most effective techniques used to reduce dynamic power by adjusting supply voltage and frequency according to workload requirement. This paper presents a detailed review of Dynamic Voltage Scaling and associated power management algorithms used in real-time systems, embedded processors, and multicore architectures. Various DVS algorithms such as static, dynamic, predictive, and feedback-based methods are discussed. The paper also covers task scheduling strategies, hardware support, challenges, and real-time constraints involved in implementing DVS. Comparative analysis and practical considerations for implementing DVS in embedded firmware are also provided.

Keywords: *Dynamic Voltage Scaling, Power Management, Embedded Systems, Energy Efficiency, Scheduling Algorithms, Low Power Design, DVFS, Real-Time Systems*

1. INTRODUCTION

Power management is a major concern in present generation computing devices. From wearable devices to large data centers, minimizing power usage without affecting performance is an important objective. In CMOS circuits, dynamic power consumption is given by:

P=C×V2×fP = C \times V^2 \times fP=C×V2×f

where CCC is switching capacitance, VVV is supply voltage and fff is operating frequency.

From the equation, it is clear that voltage has quadratic effect on power. Hence reducing voltage gives significant energy savings. Dynamic Voltage Scaling (DVS) and Dynamic Voltage and Frequency Scaling (DVFS) techniques exploit this property by adjusting voltage and frequency based on system load.

Initially developed for mobile processors, DVS is now widely used in embedded systems, real-time controllers, and multicore processors. However, improper scaling may violate timing constraints and affect system reliability. Therefore, intelligent power management algorithms are required.

2. BASICS OF DYNAMIC VOLTAGE SCALING

Dynamic Voltage Scaling (DVS) is a runtime power optimization technique in which the processor supply voltage is adjusted according to computational demand. Since voltage and frequency are closely related in digital circuits, reducing voltage generally requires reducing the clock frequency to maintain stable operation. By exploiting this relationship, significant energy savings can be achieved during periods of low workload.

In CMOS-based processors, dynamic power consumption depends heavily on voltage and frequency. When the processor is not fully utilized, running it at maximum voltage and speed wastes energy. DVS allows the system to “slow down” intelligently when full performance is not required, thereby extending battery life and reducing heat generation.

DVS is commonly implemented in microcontrollers, embedded processors, mobile CPUs, and even high-performance servers. It works transparently during runtime without stopping program execution, making it highly suitable for real-time and embedded applications.

2.1 Working Principle

The working principle of DVS is based on matching the processor performance with the workload demand.

- **When workload is low → lower frequency → reduce voltage**

  If the CPU has fewer tasks or is waiting for events (I/O, interrupts, sensor data), it does not need to operate at maximum speed. The clock frequency is reduced first, and then the supply voltage is lowered accordingly. This leads to quadratic reduction in power.

- **When workload increases → increase frequency → raise voltage**

When computational demand rises, the system increases the clock frequency to meet timing requirements. To support higher frequency safely, the voltage is also increased.

This adjustment happens dynamically with the help of:

- Performance monitoring units (PMU)
- Operating system scheduler or firmware
- On-chip voltage regulators and PLL circuits

Importantly, these transitions occur **without shutting down the processor**. The system continues execution while voltage and frequency levels are modified in small controlled steps.

Key idea:

Run the processor *only as fast as necessary*, not as fast as possible.

2.2 DVS vs DVFS

Although often used interchangeably, DVS and DVFS are slightly different in concept.

Feature	**DVS**	**DVFS**
Voltage control	Yes	Yes
Frequency control	Indirect	Direct
Energy saving	High	Very High
Complexity	Medium	High

3. NEED FOR POWER MANAGEMENT ALGORITHMS

While Dynamic Voltage Scaling (DVS) and Dynamic Voltage and Frequency Scaling (DVFS) provide the hardware capability to adjust voltage and frequency, simply lowering the voltage without a strategy is **not sufficient** for efficient power management. Modern embedded systems, mobile processors, and IoT devices have diverse workloads, real-time constraints, and performance requirements. To extract meaningful energy savings while maintaining correct system behavior, **intelligent algorithms** are required to make runtime decisions about when and how to scale voltage and frequency.

3.1 Limitations of Simple Voltage Reduction

1. **Task Execution Requirements Vary:**
   Not all tasks are equally critical. Some can tolerate delays, while others must meet strict deadlines. A naive approach that uniformly lowers voltage may delay critical tasks and violate real-time constraints.
2. **Non-linear Energy-Delay Tradeoffs:**
   Reducing voltage lowers energy consumption but also reduces performance. Excessive reduction can prolong task execution, which may **increase total energy** due to static leakage power. Without careful control, energy savings may be minimal or even negative.
3. **System Heterogeneity:**
   Embedded systems often include multiple processing cores, peripherals, and sensors. Voltage scaling in one domain may require coordination with others, or else the system may become unstable.
4. **Workload Dynamism:**
   Workload fluctuates over time. Simple voltage reduction cannot adapt to sudden increases in computation demand. Dynamic and intelligent algorithms are needed to respond to workload variations effectively.

3.2 Key Questions Addressed by Power Management Algorithms

To efficiently manage power, algorithms must make decisions along multiple dimensions:

1. **Which task can tolerate delay?**
   - Tasks are analyzed based on priority, deadlines, and slack time.
   - Non-critical or background tasks can be executed at lower frequencies to save energy.
2. **When to scale voltage?**
   - Voltage adjustments incur **transition overheads**.
   - Scaling too frequently can waste energy and increase latency. Algorithms decide the **optimal timing** for scaling based on workload prediction and system state.
3. **How much to scale?**
   - Scaling must balance energy savings with performance.
   - Algorithms determine **safe voltage-frequency pairs** for each workload level, ensuring stability and timing correctness.
4. **How to maintain real-time deadlines?**
   - In real-time and embedded systems, missing a deadline can cause system failure.

- Algorithms combine scheduling techniques with voltage scaling to guarantee that **all timing constraints are met** while reducing energy.

3.3 Role of Power Management Algorithms

Power management algorithms act as the **intelligent control layer** between hardware capabilities and application workloads. They analyze the system state, predict future demand, and dynamically adjust voltage and frequency. Effective algorithms aim to:

- Maximize **energy efficiency**
- Minimize **performance loss**
- Ensure **system reliability**
- Adapt to **dynamic workloads** and environmental conditions

Example: In an IoT sensor node, the CPU may be scaled down during idle periods but scaled up immediately when sensor data arrives. A power management algorithm ensures that scaling decisions are made optimally, considering both energy consumption and real-time responsiveness.

4. CLASSIFICATION OF DVS ALGORITHMS

Dynamic Voltage Scaling (DVS) algorithms determine **how, when, and by how much** the processor's voltage and frequency should be adjusted to save energy while meeting performance or timing requirements. Depending on the design strategy, workload predictability, and system requirements, DVS algorithms can be broadly classified into **four main categories**: Static, Dynamic (Online), Predictive, and Feedback-Based. Each type has its own advantages, limitations, and suitable applications.

4.1 Static DVS

Definition:

Static DVS sets voltage levels **at design time**, based on estimated workloads and performance requirements. Once programmed, the voltage schedule remains fixed during runtime.

Working Principle:

- Predefined voltage and frequency levels are assigned to tasks or system states.
- No real-time monitoring or decision-making is required.

Example:

In a microcontroller used for a sensor application where the data acquisition rate is constant, voltage can be statically reduced to the minimum required to meet the periodic task deadlines.

Advantages:

- **Simple implementation:** No complex runtime decision-making is needed.
- **No runtime overhead:** Energy consumption for monitoring and scaling is eliminated.
- **Predictable behavior:** Easier to verify in real-time systems.

Disadvantages:

- **Not adaptive:** Cannot respond to sudden workload variations.
- **Poor for dynamic workloads:** Leads to wasted energy during idle periods or insufficient performance during spikes.

Applications:

- Fixed-rate embedded systems, simple IoT devices, or systems with highly predictable workloads.

4.2 Dynamic (Online) DVS

Definition:

Dynamic DVS algorithms **adjust voltage and frequency at runtime** based on current workload and system state.

Working Principle:

- Monitor CPU utilization, task execution, or system events.
- Dynamically scale voltage and frequency to match workload demand.

Example:

A mobile processor running multiple applications may lower voltage when only background tasks are running and increase it when a game or video is executed.

Advantages:

- **Adaptive:** Can handle variable workloads efficiently.
- **Better energy savings:** Reduces energy consumption during idle or low-activity periods.

Disadvantages:

- **Requires monitoring hardware/software:** PMU or OS support is needed for runtime assessment.

- **Transition overhead:** Frequent voltage/frequency switching can introduce delays or energy loss if not optimized.

Applications:

- Smartphones, tablets, general-purpose embedded systems, and modern microcontrollers.

4.3 Predictive DVS

Definition:

Predictive DVS algorithms **anticipate future workload** using historical data, task profiles, or probabilistic models and adjust voltage and frequency proactively.

Working Principle:

- Analyze task execution patterns or system logs.
- Predict future CPU demand.
- Preemptively scale voltage and frequency to match expected workload.

Example:

In a video streaming application, the algorithm predicts the next high-computation frame decoding requirement and increases CPU frequency slightly before the frame arrives.

Advantages:

- **Energy efficient:** Minimizes unnecessary energy use by anticipating demand.
- **Reduces performance lag:** Avoids delays caused by reactive scaling.

Disadvantages:

- **Prediction errors:** Incorrect workload prediction can lead to missed deadlines or energy wastage.
- **Higher complexity:** Requires storage for history and computation of predictions.

Applications:

- Multimedia processors, advanced mobile SoCs, and IoT devices with recurring workload patterns.

4.4 Feedback-Based DVS

Definition:

Feedback-based DVS algorithms use **real-time feedback loops** to adjust voltage and frequency based on system performance metrics like CPU utilization, temperature, or task completion rates.

Working Principle:

- Monitor key parameters (CPU load, temperature, deadline miss ratio).
- Adjust voltage/frequency iteratively to maintain optimal energy-performance tradeoff.

Example:

A server CPU monitors temperature; if thermal limits are approached, voltage and frequency are reduced to prevent overheating. When the temperature drops, the CPU speed is restored.

Advantages:

- **Robust and adaptive:** Can correct deviations from expected behavior.
- **Maintains reliability:** Prevents overheating or missed deadlines.

Disadvantages:

- **Requires continuous monitoring:** Some overhead in power and computation.
- **Stability issues:** Improper control can cause oscillations in voltage or frequency.

Applications:

- High-performance processors, data centers, temperature-sensitive embedded systems.

5. DVS IN REAL-TIME SYSTEMS

Real-time systems must complete tasks before deadlines. DVS must not violate timing constraints.

5.1 Slack Time Utilization

If a task finishes early, extra time (slack) is used to run next tasks at lower frequency.

5.2 Earliest Deadline First (EDF) with DVS

EDF scheduler combined with DVS adjusts frequency according to task deadlines.

5.3 Rate Monotonic Scheduling (RMS) with DVS

Tasks with fixed priorities use voltage scaling based on utilization.

6. TASK SCHEDULING AND DVS

Scheduling Method	DVS Approach	Energy Efficiency	Complexity
EDF	Slack reclamation	High	Medium
RMS	Utilization based	Medium	Low
Priority based	Load monitoring	Medium	Medium

Scheduling Method	DVS Approach	Energy Efficiency	Complexity
Predictive	History based	Very High	High

7. HARDWARE SUPPORT FOR DVS

Modern processors include:

- Multiple voltage domains
- On-chip voltage regulators
- PLL for frequency control
- Power management unit (PMU)

Examples: ARM Cortex series, Intel SpeedStep, AMD PowerNow.

8. DVS IN MULTICORE PROCESSORS

Multicore systems can scale voltage:

- Per core basis
- Cluster basis
- Global basis

Per-core scaling provides best energy savings but increases design complexity.

9. ENERGY-AWARE ALGORITHMS

9.1 Utilization Based Scaling

Voltage proportional to CPU utilization.

9.2 Temperature Aware Scaling

Reduces voltage when chip temperature rises.

9.3 Workload Characterization

Different tasks have different power profiles. Algorithms classify tasks.

10. MATHEMATICAL MODEL OF DVS

Energy consumption for a task:

E=∑i=1nC×Vi2×fi×tiE = \sum_{i=1}^{n} C \times V_i^2 \times f_i \times t_iE=i=1∑nC×Vi2×fi×ti

Optimization objective is to minimize EEE while meeting deadline DDD.

11. CHALLENGES IN IMPLEMENTING DVS

1. Voltage transition delay
2. Stability issues at low voltage
3. Real-time deadline violation
4. Hardware complexity
5. Memory access latency variation
6. Peripheral timing mismatch

12. POWER MANAGEMENT IN EMBEDDED FIRMWARE

Firmware designers must:

- Use sleep modes
- Schedule tasks properly
- Avoid busy waiting
- Use interrupt driven design
- Integrate OS power hooks

13. CASE STUDY: DVS IN IOT SENSOR NODE

A sensor node spends most time in idle state.

Mode	Voltage	Frequency	Power
Active sensing	3.3V	48 MHz	High
Processing	2.4V	16 MHz	Medium
Idle	1.8V	4 MHz	Low
Sleep	1.2V	32 kHz	Very Low

Significant battery improvement is observed.

14. Comparison of Power Management Techniques

Technique	Energy Saving	Cost	Complexity
Clock Gating	Medium	Low	Low
Power Gating	High	High	High

Technique	Energy Saving	Cost	Complexity
DVS/DVFS	Very High	Medium	Medium
Sleep Modes	High	Low	Low

15. INTEGRATION WITH OPERATING SYSTEMS

Operating systems like:

- FreeRTOS
- Embedded Linux
- VxWorks

provide APIs for frequency and power scaling.

16. FUTURE TRENDS

- AI based power prediction
- Machine learning assisted DVS
- Fine grained voltage islands
- Energy harvesting systems with adaptive DVS

17. FIGURE: CONCEPTUAL DVS OPERATION

18. DISCUSSION

Dynamic Voltage Scaling is not only hardware feature but algorithmic problem. Proper integration of scheduling, prediction, and hardware control gives best result. Poorly designed algorithm may even increase energy due to frequent transitions.

19. CONCLUSION

Dynamic Voltage Scaling and power management algorithms play very important role in modern embedded and computing systems. By intelligently adjusting voltage and frequency based on workload, significant power reduction is achieved without affecting system performance. Different algorithms such as static, dynamic, predictive and feedback based

approaches offer different tradeoffs. Real-time systems require careful slack management to avoid deadline miss. With support from modern processors and operating systems, DVS has become practical and effective technique. Future systems will combine AI and adaptive learning for more efficient power management.

REFERENCES

1. T. Ishihara and H. Yasuura, "Voltage scheduling problem for dynamically variable voltage processors," *ISLPED*, 1998.
2. P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded systems," *SOSP*, 2001.
3. F. Yao, A. Demers, and S. Shenker, "A scheduling model for reduced CPU energy," *FOCS*, 1995.
4. S. Herbert and D. Marculescu, "Analysis of dynamic voltage/frequency scaling in chip multiprocessors," *ISLPED*, 2007.
5. J. Luo and N. K. Jha, "Power-conscious joint scheduling of periodic tasks," *DAC*, 2000.
6. R. Jejurikar and R. Gupta, "Dynamic voltage scaling for system-wide energy minimization," *ISLPED*, 2004.
7. ARM Ltd., "ARM Cortex Processor Power Management Guide," Technical Report.
8. Intel Corporation, "Enhanced Intel SpeedStep Technology," White Paper.
9. A. Sinha and A. Chandrakasan, "Dynamic voltage scheduling using adaptive filtering," *ICCAD*, 2001.
10. W. Kim et al., "System level analysis of fast dynamic voltage scaling," *ISLPED*, 2008.