# Optimization of Energy Consumption in Battery-Operated Real-Time Systems*

Swathi Konduru[1], Ashikahmed Bhuiyan[2], and Md Amiruzzaman[2,†]

[1] Nano Analytics, Pennsylvania, USA
Swathi.konduru@nano-analytics.com
[2] West Chester University of Pennsylvania, West Chester, Pennsylvania, USA
{ABhuiyan, MAmiruzzaman}@wcupa.edu

**Abstract**

Real-time applications emphasize worst-case scenarios due to their precise timing demands. However, many critical embedded systems applications have other requirements, such as energy consumption. The results of this study can help in e-business infrastructure; such as payment gateways, online transactions, and data driven logistics. Especially this would be beneficial in energy saving and improving operations costs for embedded systems in e-business. This study presents a model for work scheduling that minimizes energy usage by implementing a suitable scheduling algorithm for dynamic power management (DPM) and dynamic voltage scaling (DVS) and evaluating power management approaches' effectiveness.

Applying DVS leads to an 84.03% reduction in energy consumption while using DPM reduces the total energy consumed by devices by 66.76%. Combining DPM and DVS yields similar energy savings to DVS alone (83.22%) but with the potential for more significant energy savings due to external device involvement. While this combination can lead to longer execution times, a significant reduction in energy consumption makes it a worthwhile trade-off for energy-efficient computing.

**Keywords:** Real Time Systems, Embedded Systems, Energy Optimization.

## 1 Introduction

When it comes to embedded systems, it is crucial to consider energy consumption. Real-time energy systems increasingly consume electricity, and optimizing their consumption will result in significant savings and environmentally friendly benefits, as evidenced by recent statistics [14]. The demand for data center services has grown steadily in recent years, with a projected growth rate of 4% to 5% annually, according to a report from the International Energy Agency IEA. A corresponding increase in energy consumption is expected due to this growth, with data centers projected to account for 3.2% of global electricity consumption by 2025 [8].

Energy optimization also plays a pivotal role in ensuring the efficiency, reliability, and sustainability of commercial computing infrastructures. Many e-business platforms rely on large-scale distributed systems, data centers, and edge devices—such as retail point-of-sale terminals, payment gateways, and IoT sensors—that operate under real-time constraints. Reducing the energy footprint of these components can significantly lower operational costs, improve service uptime, and align with corporate sustainability goals [8]. In particular, energy-efficient scheduling mechanisms like DVS and DPM are crucial for maintaining performance while minimizing

*Proceedings of The 2025 IFIP WG 8.4 International Symposium on E-Business Information Systems Evolution (EBISION 2025), Article No. 24, December 16-18, 2025, Sapporo, Japan. 

†Corresponding author

energy waste in systems that continuously process financial transactions, customer data, and logistics operations.

Moreover, as E-Business ecosystems increasingly integrate edge computing and mobile commerce, the energy behavior of embedded devices directly affects user experience, system responsiveness, and overall cost-effectiveness. Hence, studying and improving energy-aware scheduling algorithms is not only technically beneficial but also strategically relevant for business continuity and resource management within digital enterprises.

Optimized energy consumption is essential for battery-operated real-time systems' efficient and reliable functioning to ensure that they operate as long as possible. It is especially relevant for applications where the system is situated in a distant or hard-to-reach location, and, in some cases, it may not be feasible or costly to replace batteries regularly. According to a study by the Department of Energy, wireless sensor networks consume up to 40% of energy for data transmission. The energy consumption of remote wireless network equipment can be reduced by optimizing protocols and techniques and improving performance in these devices [11]. By reducing the need to replace batteries and the energy costs associated with charging batteries, energy consumption can also lead to cost savings. In addition, energy savings can contribute to mitigating concerns about the environmental impact associated with battery storage.

Dynamic Power Management (DPM) and Dynamic Voltage and Frequency Scaling (DVFS) are known for energy optimization. During task execution, the DPM policy reduces processor power consumption by utilizing idle intervals. Suppose that the idle interval equals (or is greater than) a certain threshold, that is, the break-even time [7, 4]. In that case, the processor is switched to a low-power sleep mode, thus reducing its power consumption. DVFS is to adjust the voltage and frequency of the processor dynamically based on the current workload. When the workload is high, the voltage and frequency are increased to provide the necessary processing power. When the workload is low, the voltage and frequency are decreased to save energy. This approach [2, 3, 5]enables the processor to operate at the minimum possible voltage and frequency while meeting performance requirements.

The combination of DPM and DVFS techniques is often used to optimize energy consumption in real-time systems [6]. This combination can substantially reduce energy consumption but maintains the required performance level (and timing requirement). The DPM will adjust the system's power consumption based on workload, and the DVFS will adapt processor voltage and frequency to match this workload. The system can achieve even more significant energy savings by using this combination. For example, DPM can be used to put the system into a low power state during periods of low workload, while DVFS reduces the voltage and frequency of the processor to reduce the power consumption further. In a period of high workload, DPM can bring the system from a low power state, and DVFS increases processor voltage and frequency to provide sufficient processing power. The combination of DPM and DVFS has been widely adopted in a wide range of devices, including cell phones, computers, and servers. It's shown to deliver substantial power savings while maintaining the required level of performance [6].

This work focuses on three case studies that use DPM, DVS, and their integration, along with the Least Slack Time (LST) scheduling algorithm, to optimize energy consumption in battery-operated real-time embedded systems. Specifically, we have made the following contributions:

- We have analyzed DVS, DPM, and the integration of DVS and DPM techniques to evaluate their performance in optimizing energy consumption in battery-operated real-time systems using Python simulation.

- We explored different scheduling algorithms and chose the least Slack Time algorithm compatible with all three case studies. The challenge was to find one algorithm that

would work with all three case studies because it is not ideal to compare the performance of these case studies if we were to use different scheduling algorithms for each case study.

## 2 Literature Review

Some of the existing works in optimizing energy consumption are reviewed in this section. In [13], Zhou Tang et al. proposed an energy-efficient DVFS-enabled workflow task scheduling algorithm (DEWTS) that is based on Dynamic Voltage and Frequency Scaling (DVFS) to reduce the total power consumption. DEWTS is used in processors that are DVFS enabled and applies to most data centers' scheduling systems. The proposed approach is shown to have reduced power consumption by up to 46.5% compared to other state-of-the-art models.
In [12], Tajana Simunic et al. proposed a system combining DVS and DPM techniques, thus enabling more power savings. DVS algorithm was proposed based on change point detection that is used to detect the change in arrival or decoding rates and the frequency setting policy, which uses the change point detection to set the frequency and voltage of the processor. This approach was tested on MPEG video and MP3 audio algorithms on a portable "SmartBadge" device. Then, this approach is combined with the DPM algorithm and shows a factor of three savings in energy.

In [1], Ijaz Ali et al. proposed a dynamic power-aware scheduling scheme for mixed-criticality tasks on unicore processors. To reduce the overall energy, energy level in high-criticality mode was reduced first to tackle the difficulty in minimizing power trade-off in Hi-criticality mode. When there is an idle period between high critical job execution, the processor will be switched to low critical mode. In terms of energy reduction in mixed-criticality, the proposed scheme is shown to be effective and superior to static methods. In [9], Ridha Mehalaine et al. proposed a flexible scheduling scheme to minimize energy consumption using Dynamic Voltage Scaling (DVS) technique on a multiprocessor system with uncertain data. The proposed approach also reuses the time savings that show the difference between the worst-case execution time (WCET) and the actual execution time (AET). Simulations have been performed using MATLAB to evaluate the model and confirm its reliability for reducing energy consumption in period-independent tasks.

In [10], Sheikh et al. proposed some changes to the existing models by incorporating the change in execution time and energy consumption brought about by the number of cache partitions assigned to a task for a three-dimensional energy optimization problem that includes minimizing the core-, cache- and system level energy consumption. According to the results, optimizing cache energy plays a significant role in reducing overall energy consumption. Some of the limitations of these existing works are that these approaches are specified to a particular device or processor or tasks which limits their applicability to other scenarios or applications. The complexity and the lack of readability limit the practicality of these approaches. Also, there are a lot of approaches and methods involved in energy optimization which makes it hard to choose a suitable approach that suits the application.

## 3 Task Definition

The tasks are periodic in nature, in which the deadline line of the task is equal to its period, and the first instance of the periodic task is released at time 0. The tasks are defined by task name, deadline, Worst Case Execution Time (WCET), Actual Execution Time (AET), which is less than WCET, and priority assigned to it.

### 3.1 Task scheduling algorithm

A task scheduling algorithm based on the Least Slack Time (LST) policy and task priority as the tiebreaker. It starts by setting the current time [11]. The algorithm updates the slack time for each task, which is defined as the amount of time a task has remaining before its deadline, sorts the tasks based on the slack time and priority, selects the task with the least slack time as the next task to be scheduled, adds the task along with its start and end times to the new list, updates the current time with the end time of the scheduled task. The process continues until all tasks have been scheduled. If there are tasks that must be executed in a particular order, then the priority number assigned to the task will take care of the scheduling. For simplicity purposes, this priority case is not considered.

[H] Least Slack Time [1]

task_scheduling*taskslist* current time = 0 schedule tasks = [ ] initialising a list for scheduled tasks *taskslist not empty slack time* = *deadline*−(*current time*+$WCET$) slack time for all the tasks in the list Sort the tasks list in ascending based on slack time *next task* = *tasklist*[0]task with least slack time is the next task *schedule tasks.append*(*next task*) *current time*+ = $AET$[*nexttask*] *taskslist.drop*(*next task*)

## 4 Methodology

This section will explore three detailed case studies highlighting different approaches to optimizing energy consumption. Subsection 4.1 examines DVS as a technique for energy optimization. Subsection 4.2 delves into DPM and its application for reducing energy consumption. Finally, subsection 4.3 integrates the principles of DPM and DVS to provide a comprehensive strategy for maximizing energy efficiency.

In these energy optimization approaches, the temperature is considered an outlier and therefore excluded from the present scope of analysis. Our primary objective here is to validate the effectiveness of DVS and DPM techniques under controlled conditions. We also recognize that comparisons against state-of-the-art techniques and standardized benchmarks would further strengthen the evaluation. While such comprehensive benchmarking is not included in the current work, it represents an important direction that we plan to pursue in future research.

### 4.1 Case Study 1: Dynamic Voltage Scaling (DVS) for Energy Optimization

The power of a CMOS circuit can be divided into static power, which is independent of the frequency and dynamic power, which is affected by frequency and voltage change.

#### 4.1.1 DVS energy model

The energy model for Dynamic Voltage Scaling aiming to reduce energy consumption is defined here.

- **Dynamic power**
  It is calculated by the following formula [15]:

$$(P_{dyn} = aC_L V^2 f)$$

where a is the switching probability, $C_L$ is the load capacitance, V is the voltage, and f is the frequency. In the processor, scaling and $C_L$ are not affected by frequency. When the frequency is scaled by a factor of $s$, as an effect, voltage is scaled by $s$; the current is scaled by a factor of $s^2$, and the dynamic power is scaled by $s^3$ which is given by, $P_{dyn}(s) = (1)^{-3} * P_{dyn}(1)$, where $P_{dyn}$ (1) is the dynamic power when there is no scaling factor applied to the system ($s$=1).

- **Static power**
  Static power, $P_{static}$, is defined as the sum of leakage power and power consumption of components that are not scalable. While leakage power changes linearly with the voltage, power consumed by non-scalable components is not affected by the voltage. In this analysis, static power is considered to be constant, and no external device is considered scalable for the simplicity of the model.

- **Optimal Scaling Factor**
  The amount of energy the CPU uses changes depending on the frequency and voltage. Reducing the voltage doesn't always lead to lower energy consumption because it also increases the time it takes to complete a task. This results in the CPU using more energy while active and when the device is in standby. The rest of this section determines the maximum scaling factor for a DVS system that results in minimum energy consumption without considering any deadlines. This is referred to as the optimal scaling factor.

  For a task, consider an arbitrary value for WCET and AET, where AET $\leq$ WCET. when the frequency is scaled by a factor of $s$, the execution time is also extended by the same factor.
  The static CPU energy consumption is given by [15]

$$(E_{static}(s) = P_{static}(s) * (s * AET) = s * P_{static}(1) * AET = s * E_{static}(1))$$

  The dynamic CPU energy consumption is given by [15]

$$(E_{dyn}(s) = P_{dyn}(s) * (s * AET) = s^{-2} * P_{dyn}(1) * AET = s^{-2} * E_{dyn}(1))$$

  Total energy consumption, which is the sum of static and dynamic power, is given by

$$E_{cpu}(s) = E_{dyn}(s) + E_{static}(s) = (s - 2 * P_{dyn}(1) + s * P_{static}) * AET$$

  Consider $Q_{dyn} = s^{-2} * P_{dyn}(1), Q_{static} = s * P_{static}, and\ Q_{cpu} = Q_{dyn} + Q_{static}.$

  Then $E_{cpu} = Q_{cpu} * AET$. The optimal scaling factor is achieved by minimizing energy consumption, which involves minimizing $Q_{cpu}$. Which results in
  $s_{opt} = (2 * P_{dyn}(1)/P_{static})^{1/3}$ [15].

  For the above plot, dynamic power is considered to be 125 watts, and static power is considered to be 75 watts. The dynamic power consumption, $P_{dyn}(1)$, is related to the amount of switching activity, a, which can vary from task to task. Therefore, the optimal scaling factor is specific to each task. We must consider static processor utilization if all the tasks are known. This refers to the amount of processor time the task occupies when

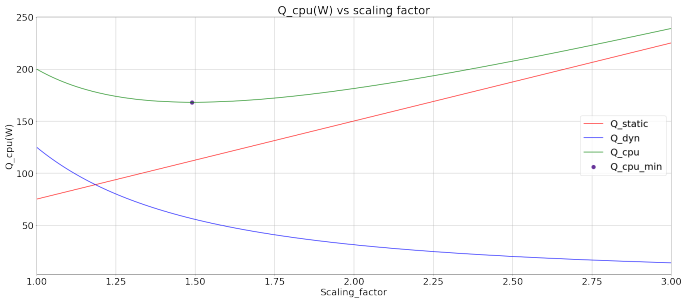


Figure 1: $Q_{CPU}(W)$ vs Scaling factor

executed periodically. It is calculated as the ratio of the total execution time of the task throughout the task [15].

Static processor utilization for $n$ periodic tasks with $P$ as the period is given by $\mu = \Sigma_{k=1}^{n} WCET^k / p^{[k]}$

The actual scaling factor is considered as the minimum of static power utilization and optimal scaling factor.

$s_{\text{act}} = \min(\mu, s_{\text{oPt}})$

Table 1: Task specification.

| Tasks | Deadline | WCET | Priority |
|---|---|---|---|
| Task 1 | 100 | 40 | 1 |
| Task 2 | 50 | 45 | 2 |
| Task 3 | 100 | 30 | 3 |
| Task 4 | 70 | 20 | 4 |

Table 2: Task execution order based on Least Slack Time scheduling algorithm

| Tasks | Deadline | WCET | Priority |
|---|---|---|---|
| Task 2 | 50 | 45 | 2 |
| Task 4 | 70 | 20 | 4 |
| Task 1 | 100 | 40 | 1 |
| Task 3 | 110 | 30 | 3 |

#### 4.1.2 Results

The frequency values are expressed as a proportion of the highest frequency and the energy values are expressed as a proportion of the energy used in the non-DVS scheduling profile. From Figure 2, applying DVS increases the time it takes to complete tasks as the processor runs at a lower frequency. However, this leads to a reduction in overall energy consumption by 84.03%.

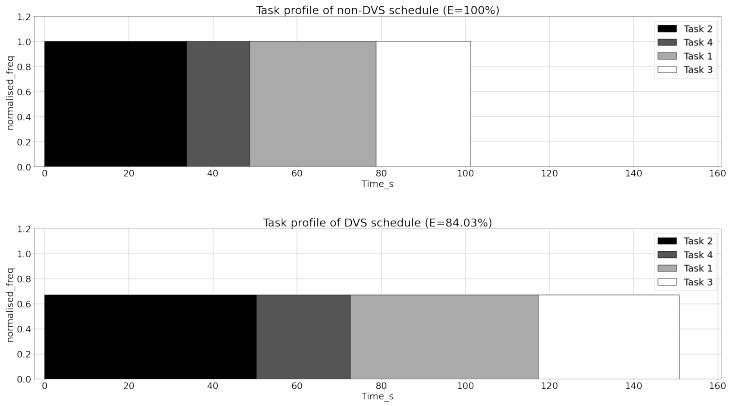


Figure 2: DVS Task profiles for the task set in Table 2

## 4.2 Case Study 2: Dynamic Power Management for energy consumption optimization

If there are a lot of external devices or sensors connected to the system, then DPM is implemented to reduce energy consumption by switching the states of inactive devices.

### 4.2.1 Device energy model

The devices are assumed to have three modes of operation: active, standby (idle), and off. Let $D_j$ be the number of devices required by the task $j$. The device consumes some power depending on the state of the device; $P_j^{act}$ for active power consumption, $P_j^{idle}$ for standby (idle) power consumption and $P_j^{off}$=0 for off state. This model does not consider Transition energy consumption to reduce the complexity, and the device is assumed to remain in the same state until the task is executed.

- Active energy
  Assume that the device ($D_j$ is in active state and consumes an active energy $E_j^{act}$, which is constant but in reality, it would not be constant. Active energy is given by
  $E_j^{act} = P_j^{act} * AET_j$ [15]

- Idle energy
  Assume that the device $D_j$ is in idle state and consumes an active energy $E_j^{idle}$ which is constant. Idle energy is given by

  $E_j^{idle} = P_j^{idle} * AET_j$ [15]

  If the processor's frequency is scaled by a factor $s$, then the execution time is also scaled by the same factor, which increases the device's energy consumption.

### 4.2.2 Results

The energy values are expressed as a proportion of the energy used in the non-DPS scheduling profile. The total energy consumed by the devices is reduced to 66.76%. In this example, the

Table 3: Task specification:deadline, WCET and priority can be considered from Table 1

| Tasks | Device 1 | Device 2 | Device 3 |
|---|---|---|---|
| Task 1 | Active | Active | Active |
| Task 2 | Idle | Active | Idle |
| Task 3 | Off | Idle | Idle |
| Task 4 | Active | Active | Off |

Table 4: Task execution order based on Least Slack Time scheduling algorithm

| Tasks | Device 1 | Device 2 | Device 3 |
|---|---|---|---|
| Task 2 | Idle | Active | Idle |
| Task 4 | Active | Active | Off |
| Task 1 | Active | Active | Active |
| Task 3 | Off | Idle | Idle |

processor is considered to be running at max frequency as DVS is not applied, as an effect the execution time is not scaled.

## 4.3 Case Study 3: Integration of DPM and DVS to optimize energy consumption

DPM is implemented first for the combination of DVS and DPM, then DVS is implemented along with the device power. DPM implementation is similar to case study 2 (see section 4.2).

### 4.3.1 Modification of DVS model

Table 5: Task specification:deadline, WCET, and priority is considered from Table 1, and state of devices is considered from Table 3

| Tasks | Deadline | WCET | Priority | Device 1 | Device 2 | Device 3 |
|---|---|---|---|---|---|---|
| Task 1 | 100 | 40 | 1 | Active | Active | Active |
| Task 2 | 50 | 45 | 2 | Idle | Active | Idle |
| Task 3 | 100 | 30 | 3 | Off | Idle | Idle |
| Task 4 | 70 | 20 | 4 | Active | Active | Off |

Consider a system with CPU and a set of devices/sensors. The total energy consumption now includes the energy consumed by the device ($E_{dev}(s)$) along with energy consumed by CPU ($E_{CPU}(s)$. As different tasks trigger different sets of devices, the optimal scaling factor for each task is different. Also, the device is assumed to stay active throughout the execution of a task, so it can be added to static power. The total energy of the system for a task $k$ is given by
$E(s) = (P_{\text{dyn}}(s) + P_{\text{static}}) \cdot s \cdot AET + P_{\text{dev}}^{[k]} \cdot s \cdot AET$
$E(s) = (s^{-2} \cdot P_{\text{dyn}}(1) + s \cdot (P_{\text{static}} + P_{\text{dev}}^{[k]})) \cdot AET$

Similar to DVS in case study1,

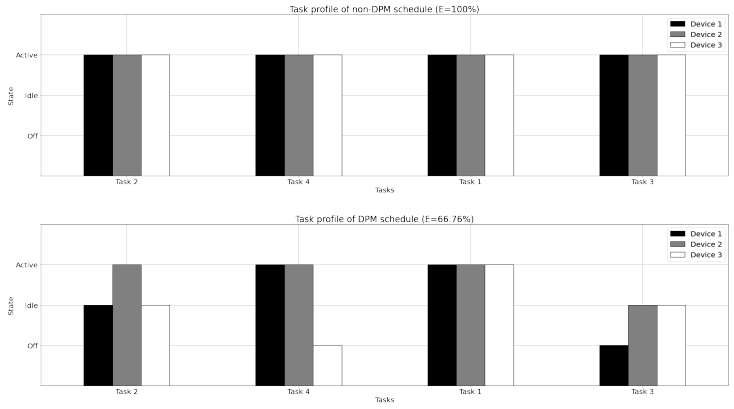


Figure 3: DPM Task profiles for the task set in Table 4

The optimal scaling factor is achieved by minimizing energy consumption, which involves minimizing $Q_{cpu}$. Which results in

$$s_{opt} = \left[ \frac{2 \cdot P_{dyn}}{P_{static} + \left( P_{dev}^{[k]} \right)} \right]^{\frac{1}{3}}$$

Similar to DVS, static processor utilization, $\mu$ is also considered
$\mu = \Sigma_{(k=1)}^{n} (WCET^{[k]})/p[k]$

The actual scaling factor is considered as the minimum of static power utilization and optimal scaling factor.

$$s_{\text{act}} = \min(\mu, s_{\text{o}Pt})$$

If $P_{static} + P_{dev}^{[k]} \geq 2 * P_{dyn}(1)$, the optimal scaling factor is $S_{opt} \leq 1$ which implies that DVS should not be applied.

Table 6: Task execution order based on Least Slack Time scheduling algorithm

| Tasks | Deadline | WCET | Priority | Device 1 | Device 2 | Device 3 |
|---|---|---|---|---|---|---|
| Task 1 | 100 | 40 | 1 | Active | Active | Active |
| Task 2 | 50 | 45 | 2 | Idle | Active | Idle |
| Task 3 | 100 | 30 | 3 | Off | Idle | Idle |
| Task 4 | 70 | 20 | 4 | Active | Active | Off |

### 4.3.2 Results

Consider the order of tasks from Table 6 and DPM is applied for each task as it is task dependent.

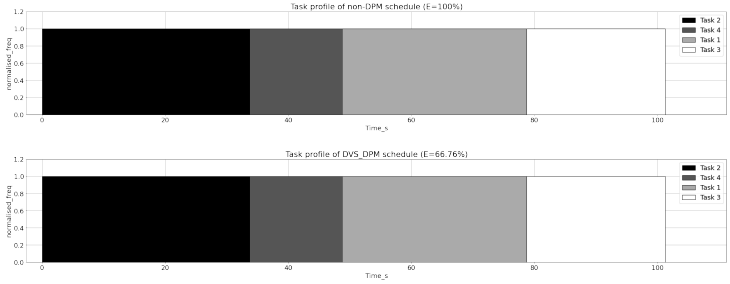


Figure 4: DPM Task profiles for the task set in Table 6

As shown in the above graph, the energy savings percentage is equal to that of only DVS, 83.22%. However, as this system includes external devices, this method will have more energy savings in absolute numbers than that of DVS (mention section number). The reason for less energy savings is that the external devices are in their respective states throughout the task execution time.

# 5 Challenges

- As most of the systems are not programmed to operate in full voltage region. Usage of DVS is very limited in the lower regions of the voltage spectrum and there are very few applications that can take advantage of scaling at low voltage levels.
- In digital circuits, Propagation delay is inversely proportional to supply voltage. Most mathematical models do not consider leakage current.
- DVS can cause stability issues such as system crashes and hardware failures if not implemented correctly.
- DVS can affect the temperature of the system, and the thermal management system needs to be designed appropriately to handle the temperature changes.
- Due to the policy limitations in DPM, sometimes the estimation of idling periods might not be accurate which results in under or over-estimation of the idle periods.
- A potential loss in energy saving might occur if there is an underestimation of idling periods. Similarly, an overestimation can result in both performance delay and energy loss.
- The appropriate combination of DVS and DPM techniques presents various issues because aggressive DVS schemes result in short device idle times, limiting the effectiveness of DPM solutions; similarly, solutions with DPM as the focus may result in excessive CPU power usage. As a result, at the system level, there is a trade-off spectrum between DVS and DPM

# 6 Conclusion

This research aims to propose an optimal model for a work schedule that minimizes energy usage, develop a suitable scheduling algorithm for DPM and DVS, and assess the effectiveness

of power management approaches. This objective strategy is easily adaptable to different needs. The findings of our study demonstrate that DVS and DPM can significantly reduce the energy consumption of computing devices. Applying DVS results in an 84.03% reduction in overall energy consumption, although it increases the time required to complete tasks due to the lower processor frequency.

On the other hand, using DPM reduces the total energy consumed by devices by 66.76% without impacting the execution time, as the processor is considered to be running at its maximum frequency. Furthermore, the combination of DVS and DPM yields similar energy savings to those of DVS alone (83.22%), with the potential for even greater energy savings due to the involvement of external devices. Although the use of a combination of DVS and DPM can lead to longer execution times, the significant reduction in energy consumption makes it a worthwhile trade-off for energy-efficient computing.

In the context of e-business information systems, the findings of this research are particularly relevant, as energy-efficient computation directly contributes to sustainable and cost-effective data processing in online business platforms. Cloud-based e-business applications rely heavily on continuous data transactions, resource allocation, and backend computation—all of which can benefit from power-optimized scheduling models. The integration of DPM and DVS can enhance the operational efficiency of e-business infrastructures by reducing operational energy costs while maintaining the required performance levels. Thus, the proposed approach not only aligns with technical optimization goals but also supports the broader sustainability and green computing initiatives that modern e-business systems increasingly adopt.

**Future Work.** First, we plan to implement a more advanced scheduling algorithm to break down the task into multiple subtasks, thereby increasing efficiency. Second, we aim to use a more advanced Device model, considering the transition times and their respective powers. The device transition can also take break-even time as a parameter, which can help avoid unnecessary state transitions. Additionally, we will implement a different scheduling algorithm in which devices are active for only a fraction of the task execution time, rather than remaining active throughout the task. A different type of combination can be implemented for more energy saving, including Hybrid DPM, Fine-Grained Power Management, and Application-Specific Power Management. Finally, the work presented here is limited to simulation, so future efforts should focus on real-world validation and correlation.

# References

[1] Ali, I., Jo, Y.I., Lee, S., Lee, W.Y., Kim, K.H.: Reducing dynamic power consumption in mixed-critical real-time systems. Applied Sciences **10**(20), 7256 (2020)

[2] Benini, L., Bogliolo, A., De Micheli, G.: A survey of design techniques for system-level dynamic power management. IEEE transactions on very large scale integration (VLSI) systems **8**(3), 299–316 (2000)

[3] Bhuiyan, A., Guo, Z., Saifullah, A., Guan, N., Xiong, H.: Energy-efficient real-time scheduling of dag tasks. ACM Transactions on Embedded Computing Systems (TECS) **17**(5), 1–25 (2018)

[4] Bhuiyan, A., Pivezhandi, M., Guo, Z., Li, J., Modekurthy, V.P., Saifullah, A.: Precise scheduling of dag tasks with dynamic power management. In: 35th Euromicro Conference on Real-Time Systems (ECRTS 2023). Schloss Dagstuhl-Leibniz-Zentrum für Informatik (2023)

[5] Bhuiyan, A., Sruti, S., Guo, Z., Yang, K.: Precise scheduling of mixed-criticality tasks by varying processor speed. In: Proceedings of the 27th International Conference on Real-Time Networks and Systems. pp. 123–132 (2019)

[6] Chen, G., Huang, K., Knoll, A.: Energy optimization for real-time multiprocessor system-on-chip with optimal dvfs and dpm combination. ACM Transactions on Embedded Computing Systems (TECS) **13**(3s), 1–21 (2014)

[7] Cheng, H., Goddard, S.: Online energy-aware i/o device scheduling for hard real-time systems. In: Proceedings of the Design Automation & Test in Europe Conference. vol. 1, pp. 6–pp. IEEE (2006)

[8] iea.org: https://www.iea.org/reports/data-centres-and-data-transmission-networks. iea.org

[9] Mehalaine, R., Boutekkouk, F.: Energy consumption reduction in real time multiprocessor embedded systems with uncertain data. In: Artificial Intelligence and Bioinspired Computational Methods: Proceedings of the 9th Computer Science On-line Conference 2020, Vol. 2 9. pp. 46–55. Springer (2020)

[10] Sheikh, S.Z., Pasha, M.A.: An improved model for system-level energy minimization on real-time systems. In: 2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS). pp. 276–282. IEEE (2019)

[11] Shipleyenergy: https://www.shipleyenergy.com/resources/efficiency/the-importance-of-unplugging-unused-appliances/. shipleyenergy.com

[12] Simunic, T., Benini, L., Acquaviva, A., Glynn, P., De Micheli, G.: Dynamic voltage scaling and power management for portable systems. In: Proceedings of the 38th annual Design Automation Conference. pp. 524–529 (2001)

[13] Tang, Z., Qi, L., Cheng, Z., Li, K., Khan, S.U., Li, K.: An energy-efficient task scheduling algorithm in dvfs-enabled cloud environment. Journal of Grid Computing **14**, 55–74 (2016)

[14] Thirunavukkarasu, M., Sawle, Y., Lala, H.: A comprehensive review on optimization of hybrid renewable energy systems using various optimization techniques. Renewable and Sustainable Energy Reviews **176**, 113192 (2023)

[15] Zhuo, J., Chakrabarti, C.: Energy-efficient dynamic task scheduling algorithms for dvs systems. ACM Transactions on Embedded Computing Systems (TECS) **7**(2), 1–25 (2008)