Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2014, Article ID 617346, 8 pages
http://dx.doi.org/10.1155/2014/617346

Research Article

Optimization of Processor Clock Frequency for Sensor Network Nodes Based on Energy Use and Timing Constraints

Youngmin Kim, Heeju Joo, and Chan-Gun Lee

Department of Computer Science and Engineering, Chung-Ang University, Dongjak-gu, Seoul 156-756, Republic of Korea

Correspondence should be addressed to Chan-Gun Lee; cglee@cau.ac.kr

Received 16 December 2013; Accepted 16 April 2014; Published 16 June 2014

Academic Editor: Jongsung Kim

The effectiveness of sensor networks depends critically on efficient power management of the sensor nodes. Dynamic voltage frequency scaling (DVFS) and dynamic power management (DPM) have been proposed to enable energy-efficient scheduling for real-time and embedded systems. However, most power-aware scheduling algorithms are designed to deal with only those cases in which the task execution time is determined solely by the clock frequency of the processor. In this study, we propose an extended task execution model that is appropriate for the sensor nodes and an algorithm that determines the optimal clock frequency for a node's processor. We analyze the extended model and verify that our algorithm calculates the clock frequency that optimizes energy savings while satisfying the timing constraints.

1. Introduction

A typical sensor network consists of multiple sensor nodes and wireless networks that connect these nodes. Each sensor in a sensor network runs on a battery with a limited power supply [1]. Hence, it is considered critical for the sensor node to operate in an energy-saving manner. Numerous approaches have been reported recently for saving the sensors' energy [2, 3].

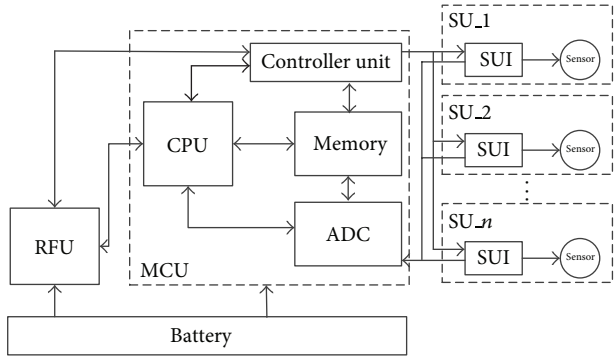
Each sensor node is composed of units for sensing, processing, radio frequency (RF) transmission, and battery power supply. A typical sensor executes real-time applications with timing constraints; the application periodically operates sensing units, processes the collected data, and transmits the processed data to the wireless network; thus, the challenge at the sensor node is to finish the above tasks with minimal energy expenditure while satisfying the timing constraints. There are real-time scheduling algorithms that are designed to address energy issues; among them, DVFS [4] and DPM [5, 6] are the most widely used schemes in the field.

DVFS is a technique in which the clock frequency and the operating voltage of a processor are adjusted during the execution. A processor operates at different clock frequencies depending on the core voltage. Reducing the core voltage and the clock frequency not only reduces energy use, but also reduces execution speed; thus, in any reduction scheme of this type there is a limit on the energy savings that can be realized without violating the timing constraints.

Real-Time DVFS technology has also been developed to exploit DVFS while also addressing the issue of timing constraints. Various other approaches have been reported in the literature to exploit DVFS, including the use of various models to describe the relevant tasks and systems [7–9].

The DPM technique functions differently, switching a device into a sleep mode when there is no task to execute. A device consumes minimal energy while in sleep mode but is also inactive, and waking the device typically entails a certain delay. This issue should be considered when developing a scheduling algorithm that not only uses sleep mode but also needs to guarantee real-time responsiveness. Benini et al. [10] have studied the question of when to switch a device into sleep mode in real-time scheduling.

Recently, Devadas and Aydin [11] and Zhao and Aydin [12] studied the interplay of DVFS and DPM for a real-time embedded system. They analyzed the combined use of DVFS and DPM for a real-time application that uses devices. However, these approaches employ an execution model in which the execution time of a real-time task depends only



MCU: Microcontroller unit
ADC: Analog digital convertor
SU: Sensor unit
RFU: Radiofrequency unit

Figure 1: Architecture of the sensor node.

on the clock frequency of the processor, which means that every device is impacted by the clock frequency. In practice, there are devices that do not depend on the clock frequency. We argue that an extended task execution model should be studied that takes both the processor and the devices into account.

In this study, we propose an extended energy model in which both of clock-dependent and clock-independent task executions are considered and present analysis results on this model. In addition, we provide an algorithm for calculating the clock frequency of the processor that optimizes the system's energy consumption; this algorithm was developed considering the interplay of DVFS and DPM for the extended model. This paper is extended from our preliminary work [13] for the nodes of sensor network.

The rest of this paper is organized as follows. Section 2 describes the system architecture of a sensor node. Section 3 proposes an extended model that considers both the clock-dependent and clock-independent task executions. In Section 4, we analyze the extended model and derive an algorithm to determine the optimal clock frequency; that is, the frequency that minimizes energy consumption while satisfying the timing constraints. Section 5 presents simulation results and analysis. We summarize the paper and suggest future work in Section 6.

2. Sensor Node Architecture

Figure 1 shows the system architecture of a typical sensor node with multiple sensor units.

As shown in Figure 1, a sensor node consists of a microcontroller unit (MCU), a RF Unit (RFU), a battery, and multiples sensor units (SUs). The analog signal detected by each SU is translated into digital data after being processed by the analog digital convertor (ADC) of the MCU. Hence, the resources such as CPU, memory, ADC, and RFU are shared by multiple SUs. The control unit controls the other components within the MCU.

It should be noted that each SU is designed to operate at a fixed frequency that is typically determined by the system designer; hence, the sensor unit is a system clock-independent device. In addition, the RFU operates at yet another frequency and depends neither on the clock frequency of the CPU nor on that of the sensor unit.

3. System Models

In this section, we present models for the task execution, the device's operation, and the system's energy consumption. The task execution model is designed to consider the different clock frequencies of the processor and the device. We normalize the clock frequencies of the processor and the device into the range of [0, 1]. In addition, our discussion will be based on the systems running on frame-based real-time scheduling.

3.1. Task Execution Model. Recent studies on the interplay of DVS and DPM for a real-time embedded system [11, 12] proposed the following model for task execution:

$$C(f) = \frac{C_{\max}}{f}. \tag{1}$$

By using this model, we can calculate a task's execution time by a processor with the clock frequency f. In the equation, $C_{\max}$ is the execution time for the case with the processor running at the clock frequency $f_{\max}$, which is normalized to 1.

However, the above model does not consider the case in which the execution time also depends on the clock frequency of the device. We present an extended model for considering the frequencies of both the processor and the device as shown in (2). In this equation, $C_{\max}$ represents

the execution time for a task when the processors and the devices run at their maximum clock frequencies:

$$C(f_1,\ldots,f_n) = \frac{C_{\max}}{f_1}\emptyset_1 + \cdots + \frac{C_{\max}}{f_n}\emptyset_n + C_{\max}\left(1-(\emptyset_1+\cdots+\emptyset_n)\right). \quad (2)$$

Each $\emptyset_i$ represents the ratio of the dependency of the task on f_i; these also ranges from 0 to 1. The expression $(1-(\emptyset_1+\cdots+\emptyset_n))$ is a ratio representing the device's dependency on its clock frequency.

Equation (3) shows how to determine $\emptyset_i$. $C^i_{\min}$ is calculated by setting only f_i to its minimum, which is referred to as $f^i_{\min}$, and the others to their maximums. Note that $C^i_{\min}$ is the task execution time when the ith device operates at its minimum clock frequency:

$$\emptyset_i = \frac{C_{\max}-C^i_{\min}}{C_{\max}}\frac{f^i_{\min}}{1-f^i_{\min}}. \quad (3)$$

Equations (4) shows that $C_{\max}$, the execution time when the ith device operates at its maximum clock frequency, is derived from (2) by setting f_i to $f^i_{\max}$. Similarly, $C^i_{\min}$ can be derived by setting f_i to $f^i_{\min}$ as shown in (5):

$$C_{\max} = \frac{C_{\max}}{f^1_{\max}}\emptyset_1 + \cdots + \frac{C_{\max}}{f^i_{\max}}\emptyset_i + \cdots + \frac{C_{\max}}{f^n_{\max}}\emptyset_n + C_{\max}\left(1-(\emptyset_1+\cdots+\emptyset_n)\right), \quad (4)$$

$$C^i_{\min} = \frac{C_{\max}}{f^1_{\max}}\emptyset_1 + \cdots + \frac{C_{\max}}{f^i_{\min}}\emptyset_i + \cdots + \frac{C_{\max}}{f^n_{\max}}\emptyset_n + C_{\max}\left(1-(\emptyset_1+\cdots+\emptyset_n)\right). \quad (5)$$

3.2. Device Model. In this paper, we assume that the devices attached to the system support both DVFS and DPM. There are two device modes: active mode and sleep mode. In active mode, the device is ready to process requests, while in sleep mode, the device goes into a low-power mode and cannot process requests. In addition, we assume that the device is in an active mode at the beginning of the task and stays in this the mode until the task execution ends; this is typically referred to as intertask device scheduling. The following list gives various notations for device parameters:

(i) P_a: power consumption in active mode;

(ii) P_s: power consumption in sleep mode;

(iii) E_{sd} and T_{sd}: power consumption and time delay, respectively, that are incurred in changing from active mode to sleep mode;

(iv) E_{wu} and T_{wu}: power consumption and time delay, respectively, that are incurred in changing from sleep mode to active mode.

The break-even time B represents the minimum duration of the sleep mode that will compensate for the added power consumption incurred in changing between the active and sleep mode. Devadas and Aydin [11] calculated the break-even time as shown in (6) and we adopt this calculation in our extended execution model:

$$B = \frac{E_{sd}+E_{wu}-(T_{sd}+T_{wu})P_s}{P_a-P_s}. \quad (6)$$

3.3. Energy Model. The system energy E is partitioned into the static energy E_s and the dynamic energy E_d; specifically, E_s is the static energy consumed in operating the system clock, while E_d is dynamically varying energy that relates directly to the clock frequency of the processor. In this paper, we focus on dynamic energy consumption. We propose an extended energy model that utilizes (1) from [11]. However, as some portion of the task does not depend on any processor frequency, it is difficult to exactly determine the execution time of a task. In this paper, the energy model considers the execution time model represented by (2). Equation (7) represents the energy model, which is an extension of that presented in [11]. $\delta(f)$ represents the slack time in the frame at the frequency f. $\mathbf{D}_a$ and $\mathbf{D}_s$ represent the sets of devices transitioned to active and sleep mode, respectively, during the slack time of the frame. The total active power of devices P_{ind} is independent of the frequencies of CPUs and devices:

$$E_d(f_1,\ldots,f_n) = \left(af^3+P_{\text{ind}}\right)C(f_1,\ldots,f_n) + \sum_{i|D_i\in\mathbf{D}_a} P^i_a\delta(f_1,\ldots,f_n) + \sum_{i|D_i\in\mathbf{D}_s}(E_{sd}+E_{wu}). \quad (7)$$

4. Optimal Frequency Decision in SUs

In order to simplify the discussion, we assume that a sensor node is equipped with a sensing unit that runs at a fixed frequency. The following equation represents an extended energy model for this scenario:

$$E_d(f) = \left(af^3+P_{\text{ind}}\right)C(f) + \sum_{i|D_i\in\mathbf{D}_a}\delta(f)P^i_a + \sum_{i|D_i\in\mathbf{D}_s}(E_{sd}+E_{wu}). \quad (8)$$

The analysis of our energy model is illustrated in the following sections. In addition, we discuss how to determine the optimal frequency that minimizes the sensor node's energy consumption. Note that the analysis result is very similar to [11] because our model is extended from theirs. In the following, we shall refer the left and the right areas based on $f = f_b$ to areas α and β, respectively, where f_b represents the break-even point of the device.

4.1. Minimum Energy Consumption Frequency Decision in Area α. In area α, the device is never switched into the sleep mode during its idle time because the duration of sleep mode would be shorter than the break-even time. Therefore, the

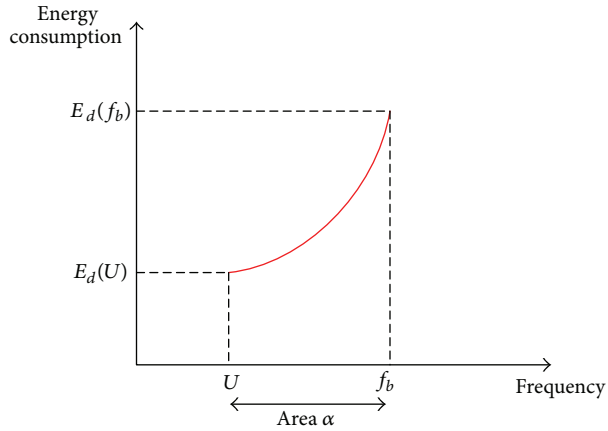


Figure 2: Energy consumption in area α.

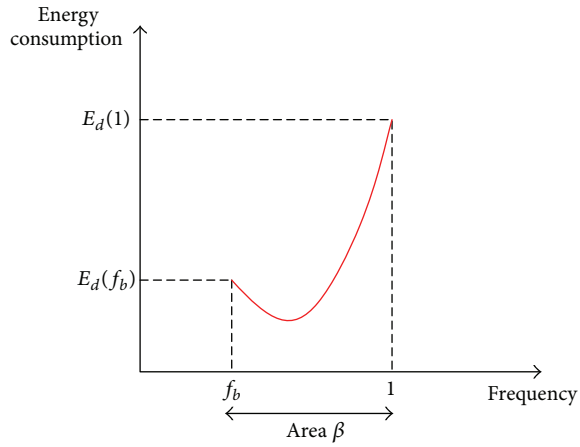


Figure 3: Energy consumption in area β.

```
(1) function FrequencyDecision()
(2)   f_opt1 ← U                          ▷the optimal frequency decision in area α
(3)   if x_1 > 0 then                     ▷the optimal frequency decision in area β
(4)     f_ee ← x_1
(5)   else
(6)     f_ee ← x_2
(7)   end if
(8)   if f_ee < f_b then
(9)     f_opt2 ← f_b
(10)  else if f_ee > 1 then
(11)    f_opt2 ← 1
(12)  else
(13)    f_opt2 ← f_ee
(14)  end if
(15)  E_diff ← E_d(f_opt1) − E_d(f_opt2)
(16)  if E_diff > 0 then                  ▷the optimal frequency decision on an entire system
(17)    f_opt ← f_opt2
(18)  else
(19)    f_opt ← f_opt1
(20)  end if
(21)  return f_opt
(22) end function
```

Algorithm 1: The optimal frequency decision algorithm.

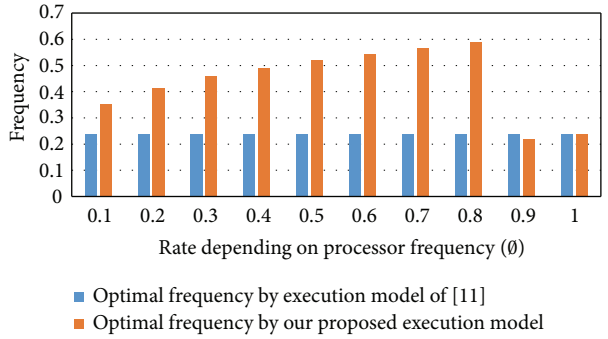


Figure 4: Optimal clock frequencies for various ∅, as determined by a previous approach and the proposed approach.

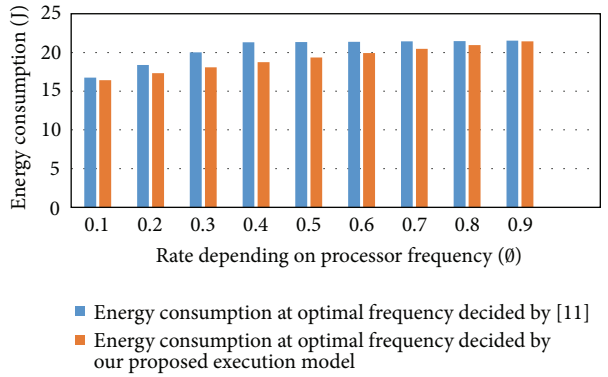


Figure 5: Energy consumption for various ∅.

energy consumption of the device is a constant value $P_a d$, where d is the deadline of the task. On the other hand, the energy consumption of a processor is proportional to the cube of processor frequency f^3; hence, lower frequencies correspond to lower energy consumptions.

Note that the execution of the task is finished exactly at its deadline when $f = U$, the lowest clock frequency that meets the timing constraint. Hence, U is the optimal clock frequency of the processor for this area. The utilization U is derived as shown below:

$$U = \frac{C_{\max}\emptyset}{d - C_{\max}(1-\emptyset)}. \tag{9}$$

Figure 2 shows the energy consumption trend in area α. The energy consumption increases monotonically as a function of frequency. Hence, $E_d(U)$ is minimal energy consumption in area α.

4.2. Minimum Energy Consumption Frequency Decision in Area β. In area β we need to consider the tradeoff between the energy consumption by the processor and the device with respect to the clock frequency. The energy model can be obtained as follows:

$$E_d(f) = \left(af^3 + P_a\right)\left(\frac{C_{\max}}{f}\emptyset + C_{\max}(1-\emptyset)\right) + E_{sd} + E_{wu}. \tag{10}$$

Equation (10) is strictly convex for $f > 0$; hence, there must be a minimum point in $f > 0$. The frequency that minimizes (10) can be determined by setting its derivative Equation (11) to zero. Figure 3 shows the relationship between frequency and the corresponding energy consumption within area β:

$$E_d'(f) = f^{-2}\left(3a(1-\emptyset)C_{\max}f^4 + 2a\emptyset C_{\max}f^3 - P_a\emptyset C_{\max}\right). \tag{11}$$

Solving the quartic formula yields four values. The derivative of our energy model consists of a second-order term that is positive infinite at $f = \infty$ and a negative second-order term that is negative infinite at $f \approx 0$. Thus, our energy model must have a minimum point for $f > 0$. One of the solutions for (11), which corresponds to x_1 or x_2 in (12), represents the minimum point.

Two Solutions of (11)*:*

$$
\begin{aligned}
x_1 \;=\; & -\frac{2a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}} \\
& -\frac{1}{2}\left(\left(\frac{4a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}\right)^2+\left(\frac{\sqrt[3]{2}R}{81(a(1-\emptyset))^3\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}\right.\right. \\
& \left.\left.+\frac{\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}{9\sqrt[3]{2}a(1-\emptyset)C_{\max}}\right)\right)^{1/2} \\
& +\frac{1}{2}\left(\left(\frac{\sqrt{32}a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}\right)^2-\left(\frac{\sqrt[3]{2}R}{81(a(1-\emptyset))^3\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}\right.\right. \\
& \left.\left.+\frac{\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}{9\sqrt[3]{2}a(1-\emptyset)C_{\max}}\right)\right) \\
& -\left(2a\emptyset C_{\max}\right)^3\times\left(108\left(a(1-\emptyset)C_{\max}\right)^3\left(\left(\frac{4a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}\right)^2+\left(\frac{\sqrt[3]{2}R}{81(a(1-\emptyset))^3\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}\right.\right.\right. \\
& \left.\left.\left.+\frac{\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}{9\sqrt[3]{2}a(1-\emptyset)C_{\max}}\right)\right)^{1/2}\right)^{-1}\right)^{1/2}, \\
x_2 \;=\; & -\frac{2a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}+\frac{1}{2}\left(\left(\frac{4a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}\right)^2+\left(\frac{\sqrt[3]{2}R}{81(a(1-\emptyset))^3\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}\right.\right. \\
& \left.\left.+\frac{\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}{9\sqrt[3]{2}a(1-\emptyset)C_{\max}}\right)\right)^{1/2} \\
& +\frac{1}{2}\left(\left(\frac{\sqrt{32}a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}\right)^2-\left(\frac{\sqrt[3]{2}R}{81(a(1-\emptyset))^3\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}\right.\right. \\
& \left.\left.+\frac{\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}{9\sqrt[3]{2}a(1-\emptyset)C_{\max}}\right)\right) \\
& +\left(2a\emptyset C_{\max}\right)^3\times\left(108\left(a(1-\emptyset)C_{\max}\right)^3\left(\left(\frac{4a\emptyset C_{\max}}{12a(1-\emptyset)C_{\max}}\right)^2\right.\right. \\
& \left.\left.+\left(\frac{\sqrt[3]{2}R}{81(a(1-\emptyset))^3\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}\right.\right.\right. \\
& \left.\left.\left.+\frac{\sqrt{\left(108a^2P_a\emptyset^3C_{\max}^3\right)+\sqrt{-4\left(36aP_a(1-\emptyset)C_{\max}^2\right)^3+\left(108a^2P_a\emptyset^3C_{\max}^3\right)^2}}}{9\sqrt[3]{2}a(1-\emptyset)C_{\max}}\right)\right)^{1/2}\right)^{-1}\right)^{1/2}
\end{aligned}
\tag{12}
$$

We can choose a clock frequency that minimizes the energy consumption in area β as that frequency that minimizes energy consumption for $f > 0$. We first find out whether the frequency that corresponds to the minimum point is in area β. The frequency of the minimum point and the optimal frequency of area β are denoted by f_{ee} and f_{opt2}, respectively, in the following:

(i) if $f_{\text{ee}} < f_b$, $f_{\text{opt2}} = f_b$;

(ii) if $f_b \leq f_{\text{ee}} \leq 1$, $f_{\text{opt2}} = f_{\text{ee}}$;

(iii) if $1 < f_{\text{ee}}$, $f_{\text{opt2}} = 1$.

4.3. Optimal Frequency Decision. Now, we can choose a frequency that optimizes the energy consumption by considering the results from both areas. The difference in the optimal energy consumption of each area is denoted by E_{diff}, and is calculated as shown in (13). If $E_{\text{diff}} > 0$, we set $f_{\text{opt}} = f_{\text{opt2}}$; otherwise, we set $f_{\text{opt}} = f_{\text{opt1}}$:

$$E_{\text{diff}} = E_d\left(f_{\text{opt1}}\right) - E_d\left(f_{\text{opt2}}\right). \tag{13}$$

Algorithm 1 shows an algorithm that derives the optimal clock frequency for a sensor node. After calculating the optimal frequency for each area using (10), we choose the optimal clock frequency between the two.

5. Simulation Results

In this section we evaluate our algorithm in various scenarios. Throughout this section, we assume a real-time task with $d = 42$ and $C_{\max} = 10$. The task uses a device with the following characteristics: $P_a = 0.5$, $E_{sd} = 5$, $E_{wu} = 5$, $T_{sd} = 10$, $T_{wu} = 10$, and $B = 20$. The switching capacitance of the processor is assumed to be $a = 1$. The parameters are set to the values used in [11] for the comparison purpose.

Figure 4 shows that the previous approach fails to handle the case in which the execution time of the task is not determined solely by the clock frequency. In the figure, the clock frequency dependency $\emptyset$ changes from 0.1 to 1, and only when the entire system depends on the clock frequency, that is, $\emptyset = 1$, does the previous approach [11] achieve an the optimal solution. In contrast, our proposed algorithm accurately determines the optimal frequency in all cases. In the worst case, using the result of the previous approach consumes 2.5 J more energy than the optimal solution.

This simulation results in Figure 5 show that our algorithm enables better power management than the previous approach [11]. This is because our algorithm is based on an extended model that more accurately reflects the architecture of the sensor node.

6. Conclusion and Future Work

A sensor node employs various devices, including a microcontroller unit for processing, a radio frequency unit for transmissions, a battery to supply power, and multiple sensor units for sensing. Some of these devices depend directly on the clock frequency and others do not.

In this study, we proposed an extended model that considers the case in which the execution time of the task is not determined only by the clock frequency of the processor.

Based on the proposed model, we presented an algorithm that calculates the optimal clock frequency of the processor to achieve system-wide energy savings. We validated our approach by both analysis and simulation results in various scenarios.

In future work, we shall extend the model to handle multiple devices; we also intend to perform experiments with actual sensor nodes.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by the Chung-Ang University Excellent Student Scholarship, the National Research Foundation (NRF-2011-0013924), and the MSIP (Ministry of Science, ICT, and Future Planning) under the ITRC (Information Technology Research Center) support program (NIPA-2014-H0301-14-1023) supervised by the NIPA.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–105, 2002.

[2] R. Poornachandran, H. Ahmad, and H. Çam, "Energy-efficient task scheduling for wireless sensor nodes with multiple sensing units," in *Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference (IPCCC '05)*, pp. 409–414, April 2005.

[3] B. Hohlt, L. Doherty, and E. Brewer, "Flexible power scheduling for sensor networks," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks (IPSN '04)*, pp. 205–214, April 2004.

[4] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for reduced cpu energy," in *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation*, 1994.

[5] H. Cheng and S. Goddard, "Online energy-aware I/O device scheduling for hard real-time systems," in *Proceedings of the Design, Automation and Test in Europe (DATE '06)*, March 2006.

[6] V. Devadas and H. Aydin, "Real-time Dynamic Power Management through device forbidden regions," in *Proceedings of the 14th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '08)*, pp. 34–44, April 2008.

[7] S. Saewong and R. Rajkumar, "Practical voltage scaling for fixed priority rt-systems," in *Proceedings of the IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '03)*, 2003.

[8] H. Aydin, R. Melhem, D. Mossé, and P. Mejia-Alvarez, "Power-aware scheduling for periodic real-time tasks," *IEEE Transactions on Computers*, vol. 53, no. 5, pp. 584–600, 2004.

[9] M. Marinoni and G. Buttazzo, "Elastic DVS management in processors with discrete voltage/frequency modes," *IEEE*

Transactions on Industrial Informatics, vol. 3, no. 1, pp. 51–62, 2007.

[10] L. Benini, A. Bogliolo, and G. de Micheli, "A survey of design techniques for system-level dynamic power management," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 3, pp. 299–316, 2000.

[11] V. Devadas and H. Aydin, "On the interplay of dynamic voltage scaling and dynamic power management in real-time embedded applications," in *Proceedings of the 7th ACM International Conference on Embedded Software (EMSOFT '08)*, pp. 99–108, October 2008.

[12] B. Zhao and H. Aydin, "Minimizing expected energy consumption through optimal integration of DVS and DPM," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD '09)*, pp. 449–456, November 2009.

[13] Y. Kim and C. G. Lee, "Determining frequency for energy saving in consideration of the frequency-dependent code ratio for real-time embedded systems," in *Proceedings of the Korea Conference on Software Engineering (KCSE '13)*, 2013.