Schedulability and Energy Consumption Analysis of Energy-Aware Fixed-Priority Task Scheduling Schemes

Kiran Arora[1,2*], Savina Bansal[3,4] and Rakesh Kumar Bansal[3,4]
[1]*Research Scholar, IKGPTU, Jalandhar, India,*
[2]*Department of CSE, BHSBIET, Lehragaga, Sangrur, India*
[3]*Department of ECE, GZSCCET, Bathinda, India*
[4]*Maharaja Ranjit Singh Punjab Technical University, Bathinda, India*
(erkiranarora@gmail.com)

***Abstract*—** The state-of-art real-time systems offer huge computation power to deal with complex engineering applications. However, this capability comes at the cost of increased energy consumption and in-turn higher heat dissipation. Dynamic voltage scaling and dynamic power management are two most commonly used energy management techniques. Modern processors are enabled with capability to run on different voltage levels. Accordingly, for real time systems, energy management is done by reducing supply voltage/frequency of processor while respecting deadline constraints. For static-priority tasksets, selecting minimum frequency to maximize energy saving is a crucial research issue as interference of higher-priority tasks must be taken into account. In the current work, a comparative analysis of state-of-the-art energy-aware fixed-priority task-scheduling techniques is carried out in terms of their task schedulability and energy savings.

***Keywords*—** *energy, task-scheduling, real-time systems, fixed-priority.*

I. INTRODUCTION

Due to rapid development in processor technology, applications of real time systems have widespreaded such as intelligent transportation, navigation, medical care, and automated surveillance. About 70% of processors developed in industry are meant for real-time embedded applications [1]. The worldwide market for embedded systems was valued at $68.9 billion in 2017 and is predicted to grow to $105.7 billion up to 2025[2]. Energy management has always been a hot topic in these systems [3]. The reason behind this concern is increased heat dissipation due to miniaturization of electronic chips in present-day computing systems. Approximately 0.5% of world's entire power usage is projected to rise by four times by 2020 [4]. Dynamic Voltage Scaling (DVS) and Dynamic Power Management (DPM) are two prominent techniques for minimizing energy consumption as proposed in literature by various researchers from time to time.

Over the last two-decades, dynamic voltage scaling has achieved appreciable attention of research community due to its quadratic property of saving energy with reduction in supply-voltage. With DVS technique, reduction in energy consumption is achieved by switching processor to operate on low voltage levels. However, decreased supply-voltage is unfavourable from performance point of view as it results in increased execution-time. As the correct operation of real time systems depends not only on logical output but also on timeliness of results, so, blindly reducing voltage may result in missing task deadlines. Another issue which restricts the level of voltage reduction is critical frequency, below which the utility of DVS starts diminishing due to leakage current. The DPM technique, on the other hand, puts processor/system components to sleep state in idle intervals whenever possible [5], to achieve energy efficiency. In fact, off chip devices have an *active* state and at least one low-power *sleep* state. However, considerable transition energy/time overheads may be involved in state transitions of devices. Therefore, only a smart use of DVS and DPM together can give optimised energy saving[6][7][8].

Task scheduling assists in improving performance and energy efficiency in real time systems [9]. Traditional real time scheduling algorithm such as Rate Monotonic Scheduling (RMS) and Earliest Deadline First (EDF) are based on ideal characteristics of tasks and can be differentiated on the basis of priority assignment [10]. RMS and EDF are fixed-priority and dynamic-priority task scheduling algorithms, respectively. In the current work, fixed-priority task scheduling algorithm has been considered owing to its wider applicability and simplicity.

In this paper, energy aware task-scheduling approach for fixed-priority real-time periodic taskset on a uniprocessor system has been studied and explored. Energy-efficient processor speed selection techniques available in literature have been analyzed and compared. Rest of the paper is organized as follows: Section II contains the related work. Section III presents models and assumptions. Section IV details energy efficient speed selection techniques. Section IV shows performance analysis and Section V concludes the work done.

II. RELATED WORK

Substantial amount of research has been conducted for energy aware scheduling for real-time applications on DVS

processors in the recent past. These approaches differ in many facets such as- scheduling policy (RM /EDF), type of tasks (periodic/mix) etc. DVS techniques exploit available slack time to set processor speed, which may be same for all tasks or it may differ from task to task. The slack is basically unused processor time of a job before its deadline.

Energy saving techniques using DVFS can be classified as inter-task and intra-task algorithms[11][12]. In intertask algorithms, a job is executed at the same speed level until it completes or is preempted by another (high-priority) job [13][8][6]. Whereas in intratask algorithms, speed selected may differ from task to task[14][15]. The intertask algorithms form the majority of the existing DVFS solutions as it incurs low runtime overhead.

To decrease energy consumption, there are many real time scheduling algorithms with DVS technology assuming worst-case execution times and dynamic power. The motivation of trading processing speed for energy savings was first proposed by Weiser et al. [16], where processor frequency and subsequent supply voltage is adjusted by means of utilization based estimates. Yao et al. [17] illustrated a polynomial-time static off-line scheduling algorithm for autonomous tasks running with variable frequencies.

Optimal execution speed that consumes minimum energy can be found in polynomial time for dynamic priority algorithms using technique proposed in [17], but for fixed priority scheme, problem becomes NP-hard due to its property of assigning high priority to smaller period tasks [18]. In order to reduce energy consumption and meet deadline constraint, speed of processor can be set equal to workload normalized with respect to Liu-Layland utilization Bound [10] for Rate-Monotonic Algorithm [19][20]. To better exploit slack available using DVS, energy efficient speed can be calculated based on response-time of each task[21][22] [18][23]. Saewong et al. [21] claims that Sys-Clock algorithm gives optimal energy-efficient speed for fixed-priority preemptive scheduling.

Above discussed works provide generalized schemes to find energy-efficient speed for static-priority real-time tasks executing on uniprocessor. For specific application areas, different techniques have been proposed in literature to set execution speed for minimizing energy consumption. Melhem et al. exploited slack available to minimize energy consumption based on application-level DVS [13]. Niu et al. have proposed a frequency selection approach that considers a tradeoff between the use of DVS and DPM [8]. Based on dynamic voltage scaling (DVS) technique, Mosse proposed and analyzed a number of schemes to dynamically adjust processor speed with slack reclamation [24], where statistical information about task's execution time was used to slow down processor speed uniformly. The best proposal is an adaptive one that takes an aggressive approach while safeguarding from the deadline constraint[25][26]. Aydin et al. have proposed a dynamic reclaiming scheduling algorithm for periodic tasks, where on the early completion of higher priority task instance, slack time is claimed to scale down the processor speed[27].

Wei et al. proposed low-cost schemes that unite voltage scaling and feasibility analysis for hard real-time systems based on the exact characterization of Rate Monotonic Algorithm (ECRMA)[20]. Moghaddas et al. [28] proposed energy-efficient scheduling method for fixed-priority tasksets that employs both DVS and DPM and reduce speed to total utilization available on processor. Haque et al. [6] assigned frequency value to tasks based on Sys-Clock algorithm [21].

In this paper, existing intertask DVS techniques for fixed priority real time tasks available in literature are analyzed and compared for energy efficiency and schedulability.

III. MODELS AND ASSUMPTIONS

A. Task Model

Real time fixed-priority periodic task system has been taken into consideration. A taskset τ contains n tasks such that $\tau = \{\tau_1 \ldots \tau_n\}$. A task τ_i is commonly represented by three parameters: Worst Case Execution Time C_i under the maximum frequency, Relative Deadline D_i and Period P_i. Utilization U_i of a task τ_i is calculated as $\frac{C_i}{P_i}$. The total utilization of taskset i.e. sum of utilizations of all tasks is represented by U_{tot}. The Hyperperiod $Hyp(\tau)$ of a task-set is defined as Least Common Multiple (LCM) of task periods.

B. Power Model

The power consumption of embedded systems can be categorized as dynamic power and static power. The dynamic power (ρ_{dyn}) consumption arises due to charging and discharging of the load capacitance [12][29] [6]. Thus,

$$\rho_{active} = \rho_{dyn} + \rho_{static} \quad (2)$$

Specifically, the power consumption ρ is a function of supply voltage (υ) and clock frequency ($\mathcal{F}$) [12]:

$$\rho_{active} = \varrho C_{ef} \upsilon^2 \mathcal{F} + \rho_{ind} + \rho_{static} \quad (3)$$

where C_{ef} is the total capacitance, ϱ is the gate activity factor. The supply voltage (υ) has a linear relationship with frequency $\mathcal{F}$. Thus, Eq. (2) can be written as,

$$\rho_{active} = \varrho C_{ef} \mathcal{F}^3 + \rho_{ind} + \rho_{static} \quad (4)$$

The maximum CPU frequency $\mathcal{F}_{max}$ has been set to 1. All other values are normalized with respect to it. ρ_{static} is a static power which is dominated by leakage current whereas ρ_{ind} corresponds to speed-independent power. There exists a critical frequency value i.e. $\mathcal{F}_{crit}$, also called energy-efficient frequency, below which the DVS does not remain effective. Critical frequency value depends on ρ_{static} and ρ_{ind}.

IV. ANALYSIS OF ENERGY EFFICIENT SPEED SELECTION TECHNIQUES

In this paper, DVS scheduling techniques for fixed-priority (FP) real-time system have been taken into consideration. Fixed-priority algorithms are of great practical importance, and most real-time scheduling algorithms, especially hard real-time systems, use fixed-priority assignments due to their low overhead and predictability [18].

A. Schedulability Tests

With rate-monotonic scheduling for periodic taskset, task with smaller period is assigned a higher priority. Tasks are

generally sorted according to increasing period. Taskset is feasible if U_{tot} is no more than Liu-Layland Utilization Bound (LL_UB) [10] as shown in (1):

$$U_{tot} \leq n\left(2^{\frac{1}{n}} - 1\right) \quad (1)$$

LL_UB is first feasibility test for RM scheduling on a uniprocessor system. A periodic taskset is schedulable if it successfully schedules all tasks without missing any deadline. Joseph et al. [30] introduced an exact schedulability condition for fixed priority scheduling. In this test, the response time of each task r_i is obtained, and if $r_i \leq d_i$, then task τ_i meets its deadline. Response time r_i can be calculated with time-demand analysis (TDA) technique[6]. TDA computes worst-case response time at the critical instant of task τ_i. Critical instant of task refers to the time, when task τ_i is released along with all its high priority tasks HP(τ_i). The worst-case response time of a task τ_i can be calculated iteratively with (2) [6](2):

$$r_i^{m+1} = c_i + \sum_{\tau_j \in HP(\tau_i)} \left\lceil \left(\frac{r_i^m}{t_j}\right) \right\rceil \times c_j \quad (2)$$

r_i^0 initialize from c_i and iteration continues until $r_i^{m+1} = r_i^m$ where *m* is number of iterations. If r_i^m exceeds d_i, task τ_i is said to be unfeasible in worst case situation. Otherwise, response time of task τ_i is a value of last iteration where $r_i^{m+1} = r_i^m$.

As task takes more time to complete when its speed is reduced, so careful schedulability analysis is necessary before applying energy-management scheme. Time-demand analysis helps in utilizing slack time available for reducing speed by precisely finding the amount of workload-demand for taskset.

B. Speed/Frequency Calculation Techniques

While calculating reduced speed for energy management, it is necessary to consider the schedulability test discussed above for fulfilling deadline constraint. Frequency assignment is generally based on the total utilization of taskset and slack available. Key objective while calculating frequency of executing task is to achieve maximum energy saving while fulfilling timing constraint in real-time systems. When a suitable frequency is selected, all tasks run on common frequency. Various frequency selection schemes are discussed below:

1) Uniform Frequency Utilization-Bound (UF_UB):

First feasibility test for RM scheduling base on Liu-Layland Utilization Bound is used in this technique. In order to find reduced processor speed in polynomial time which can meet deadline constraint for rate monotonic scheduling, frequency of processor is set equal to taskset utilization normalized with respect to LL_UB[19][20][31] , that is,

$$f = \frac{U_{tot}}{n\left(2^{\frac{1}{n}} - 1\right)} \quad (3)$$

Although it is not optimal energy-efficient frequency, but gives satisfactory results. The benefit of this scheme is that solution can be found in polynomial time.

2) Uniform Frequency-Sysclock (UF_Sys):

Techniques fall under this category perform the complete response-time analysis and finds optimal[21]/sub-optimal[23] solution. The optimal sys-clock algorithm [21] has been implemented and analyzed. It considers task's own execution time as well as preemption by higher-priority tasks while calculating workload needed to complete task's execution. It finds minimum frequency that enables each task to complete before deadline with maximum energy saving. But the problem becomes NP-Hard as these algorithms consider larger set of schedulability points. Algorithms fall under this category are being used by many researchers to find energy-efficient speed [6][32][31].

3) Uniform Frequency Utilization (UF_U):

In this scheme, the frequency f of processor is lower down to total utilization of taskset U_{tot}. This method gives excellent energy savings at lightly loaded applications but schedulability decreases with heavy-load fixed-priority applications. It shows that general model to reduce speed based on workload [17], as discussed above in section II, cannot be directly applied to fixed-priority periodic real-time tasksets rather it works well for dynamic-priority tasksets.

V. PERFORMANCE ANALYSIS

In this section, performance of various speed scaling techniques discussed in above section are evaluated by simulation for schedulability analysis and energy consumption. Taskset is schedulable if it runs successfully during hyper-period without missing deadline. Schedulability analysis gives percentage of tasksets that are schedulable with given technique. Energy consumption is an average of energy consumed by schedulable tasksets during its execution for hyper-period. It has been normalized with respect to No-Power Management Scheme (NPM). NPM assumes that no energy management technique has been applied and tasks run at maximum frequency. Priorities have been assigned to tasks according to RM policy.

A. Task-Generation

For each task-set utilization value, 1000 periodic task-sets are generated. UUnifast [33][6] algorithm is used to generate utilizations, where average task utilization is set as $u_{avg} = 0.1$ and $u\_avg = 0.05$, respectively. Periods are randomly generated between 10ms to 100ms. Deadline is set equal to period and worst-case execution time is calculated as a product of utilization and period. For each task set, enough number of tasks are generated so that taskset utilization reaches a given target value.

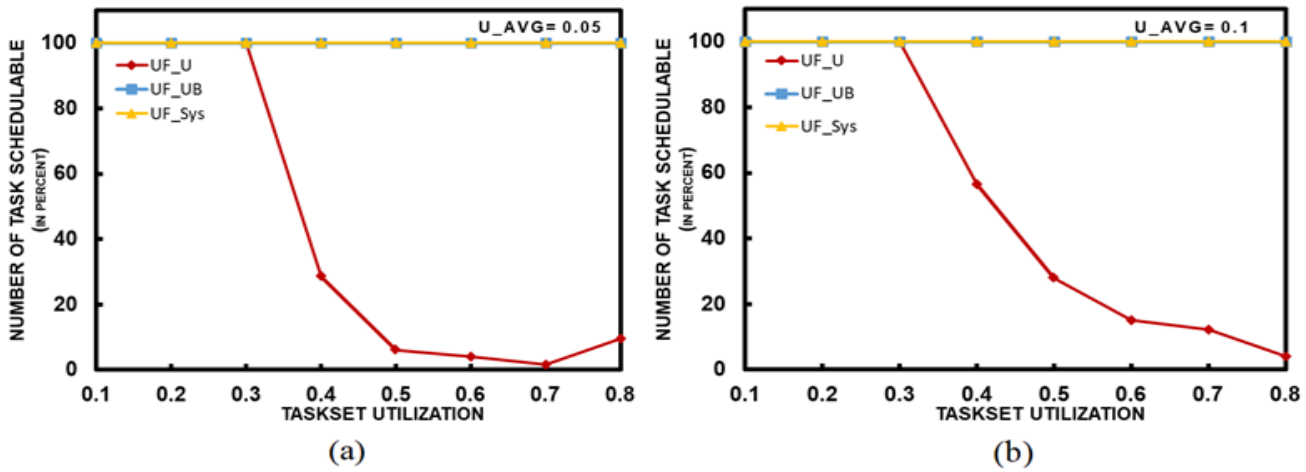

Figure 1. Schedulable tasksets

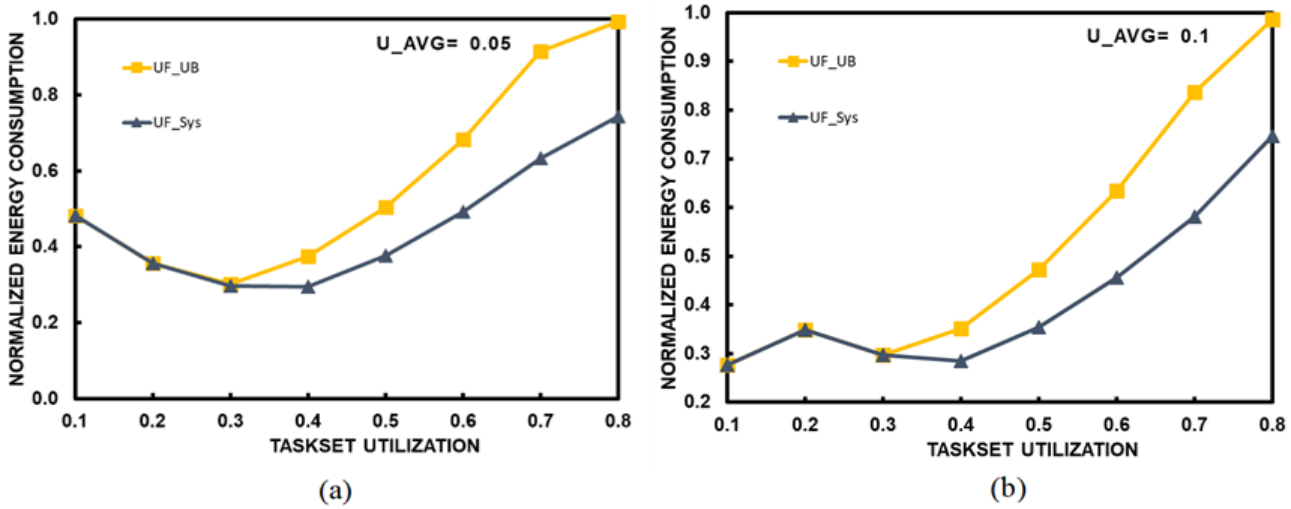

Figure 2 . Normalised Energy Consumption

B. Simulation

To evaluate performance of proposed scheme experimentally, simulator has been constructed in JAVA. In the given system, simulation ran up to hyper-period for each taskset to correctly analyze the technique.

Theoretically, an ideal processor should support continuous voltage levels. But, using continuous variable voltages is not feasible since the switching overhead to support several operational levels would be very high. Thus, latest processors support only fixed number of discrete-level clock-speeds [34][35]. Intel Xeon 5500 supports 15 operating states[36]. In this work, it is assumed that processor has eight major operational frequency levels (which are {0.4, 0.5, 0.6, 0, 7, 0.8, 0.9, 1.0}) which are further divided into minor level with a step of 0.05 for example 0.4,0.45,0.5…. etc. Further critical frequency is set to 0.4 and breakeven time is 1.5ms [6] [28].

1) Schedulability analysis:

UF_U, UF_UB and UF_Sys are tested for utilization from 0.1 to 0.8. Taskset is considered unschedulable if it does not fulfill deadline constraint for any task. Each point in the graph gives a percentage of tasksets schedulable among 1000 tasksets. Fig. 1(a) and Fig. 1(b) shows that UF_UB and UF_Sys are 100% schedulable for average task utilization 0.05 and 0.1, respectively. But after $U_{tot} = 0.3$, number of schedulable tasksets for UF_U are very low in both cases. This is because for rate-monotonic scheduling, interference of high-priority tasks has not been considered by UF_U. So, by lowering frequency equal to workload of application will result in missing deadline by low priority task.

2) Energy Consumption Analysis:

Normalized energy consumption with respect to NPM by UF_UB and UF_Sys has been shown in Fig. 2. Each point in the plot shown in Fig. 2(a) and Fig. 2(b) gives an average energy consumption of 1000 tasksets for average task utilization 0.05 and 0.1, respectively. As UF_Sys considers response-time and preemption of low priority tasks by high priority tasks, frequency selected by this technique utilizes slack to its best. Due to better slack usage energy consumed by UF_Sys is lower than UF_UB. Percentage of improvement in energy saving by UF_Sys over UF_UB are shown in Table 1. Each row in the table shows an average energy saving of

1000 tasksets. UF_Sys saves on an average 17% and 16 % more energy than UF_UB for $u\_avg = 0.1$ and $u\_avg = 0.05$, respectively for 7000 tasksets.

TABLE I. IMPROVEMENT IN ENERGY SAVING BY UF_SYS OVER UF_UB.

Utilzation of task-set	$u\_avg = 0.05$ (%)	$u_{avg} = 0.1$ (%)
0.1	0	0
0.2	0	0
0.3	2	0
0.4	21	19
0.5	26	25
0.6	28	28
0.7	31	30
0.8	25	24
Average	**17**	**16**

VI. CONCLUSIONS

Energy consumption can be lowered by reducing frequency only up to critical frequency. While reducing frequency deadline constraint has also been considered because lower frequency lengthens execution time of task. In this paper, frequency assignment techniques to minimize energy consumption in real-time fixed-priority applications are analyzed and compared. To find reduced frequency in polynomial time, UF_UB technique can be helpful. But it does not give optimal results. To have optimized energy saving UF_Sys technique can be used. Some algorithms falling under UF_Sys also give sub-optimal results in lesser time by having small set of schedulability points. Thus, among the techniques presented here, UF_Sys is used in energy-critical systems.

REFERENCES

[1] R. K. Bansal, *Fault-Tolerant Real-Time Scheduling for Multiprocessor Systems*. United Kingdom: LAP Lambert Academic Publishing, 2011.

[2] "5 Key Trends for Embedded Systems in 2019," 2018. [Online]. Available: https://news.thomasnet.com/featured/5-key-trends-for-embedded-systems-in-2019/. [Accessed: 02-Mar-2019].

[3] K. Arora, S. Bansal, and R. Kumar Bansal, "Energy Aware Fault Tolerant Fixed Priority Task Scheduling in Multiprocessor System," in *Proceedings of the 8th International Conference Confluence 2018 on Cloud Computing, Data Science and Engineering, Confluence 2018*, 2018, pp. 658–663.

[4] N. Kaur, S. Bansal, and R. K. Bansal, "Energy conscious scheduling with controlled threshold for precedence-constrained tasks on heterogeneous clusters," *Concurr. Eng.*, vol. 25, no. 3, pp. 276–286, 2017.

[5] V. Swaminathan and K. Chakrabarty, "Energy-Conscious , Deterministic I / O Device Scheduling in Hard Real-Time Systems," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 22, no. 7, pp. 847–858, 2003.

[6] M. A. Haque, H. Aydin, and D. Zhu, "Energy-aware standby-sparing for fixed-priority real-time task sets," *Sustain. Comput. Informatics Syst.*, vol. 6, pp. 81–93, 2015.

[7] V. Devadas and H. Aydin, "On the interplay of voltage/frequency scaling and device power management for frame-based real-time embedded applications," *IEEE Trans. Comput.*, vol. 61, no. 1, pp. 31–44, 2012.

[8] L. Niu and W. Li, "Energy-efficient fixed-priority scheduling for real-time systems based on threshold work-demand analysis," *2011 Proc. Ninth IEEE/ACM/IFIP Int. Conf. Hardware/Software Codesign Syst. Synth.*, pp. 159–168, 2011.

[9] R. K. Bansal, K. Singh, and S. Bansal, "Performance Analysis of a Real-Time 1-Fault-Tolerant Scheduling Algorithm on Heterogeneous Computing Platform," *Int'l J. Comput. Sci. & Syst. Anal. IJCSSA, Ser. Publ.*, vol. 3, no. 2, pp. 65–70, 2009.

[10] C. L. Liu and J. W. Layland, "Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment," *J. ACM*, vol. 20, no. 1, pp. 46–61, Jan. 2002.

[11] V. Venkatachalam and M. Franz, "Power reduction techniques for microprocessor systems," *ACM Comput. Surv.*, vol. 37, no. 3, pp. 195–237, 2006.

[12] M. Bambagini, M. Marinoni, H. Aydin, and G. Buttazzo, "Energy-Aware Scheduling for Real-Time Systems," *ACM Trans. Embed. Comput. Syst.*, vol. 15, no. 1, pp. 1–34, 2016.

[13] R. Melhem, D. Mossé, and E. Elnozahy, "The Interplay of Power Management and Fault Recovery in Real-Time Systems," *IEEE Trans. Comput.*, vol. 53, no. 2, pp. 217–231, 2004.

[14] Y. W. Zhang and R. F. Guo, "Power-aware fixed priority scheduling for sporadic tasks in hard real-time systems," *J. Syst. Softw.*, vol. 90, no. 1, pp. 128–137, 2014.

[15] Y. W. Zhang and R. F. Guo, "Power-aware scheduling algorithms for sporadic tasks in real-time systems," *J. Syst. Softw.*, vol. 90, no. 1, pp. 128–137, 2014.

[16] M. Weiser, B. Welch, A. Demers, and S. Shenker, "Scheduling for Reduced CPU Energy," *Proc. 1st USENIX Conf. Oper. Syst. Des. Implement.*, pp. 13–23, 1994.

[17] F. Yao, A. Demers, and S. Shenker, "A scheduling model for reduced CPU energy," in *Proceedings of IEEE 36th Annual Foundations of Computer Science*, 1995, pp. 374–382.

[18] G. Quan and X. S. Hu, "Energy efficient DVS schedule for fixed-priority real-time systems," *ACM Trans. Embed. Comput. Syst.*, vol. 6, no. 4, p. 29–es, 2007.

[19] T. A. AlEnawy and H. Aydin, "Energy-Aware Task Allocation for Rate Monotonic Scheduling," 2005, pp. 213–223.

[20] T. Wei, P. Mishra, K. Wu, and H. Liang, "Fixed-priority allocation and scheduling for energy-efficient fault tolerance in hard real-time multiprocessor systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 11, pp. 1511–1526, 2008.

[21] S. Saewong and R. (Raj) Rajkumar, "Practical Voltage-Scaling for Fixed-Priority RT-Systems," in *Proceedings of the The 9th IEEE Real-Time and Embedded Technology and Applications Symposium*, 2003, p. 106--.

[22] P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," *ACM SIGOPS Oper. Syst. Rev.*, vol. 35, no. 5, pp. 89–102, 2001.

[23] E. Bini, G. Buttazzo, and G. Lipari, "Minimizing CPU energy in real-time systems with discrete speed

management," *ACM Trans. Embed. Comput. Syst.*, vol. 8, no. 4, pp. 1–23, 2009.

[24] D. Mossé, H. Aydin, B. R. Childers, and R. Melhem, "Compiler-assisted dynamic poweraware scheduling for real-time applications," *Proc. Work. Compil. OS Low Power*, 2000.

[25] N. AbouGhazaleh, D. Moss, B. Childers, and R. Melhem, "Toward the Placement of Power Management Points in Real-Time Applications," *Compil. Oper. Syst. low power, Springer US*, pp. 37–52, 2003.

[26] H. Aydin, R. Melhem, D. Moss, and P. Mej, "Dynamic and Aggressive Scheduling Techniques for Power-Aware Real-Time Systems," in *IEEE Real-Time Systems Symposium*, 2001, pp. 95–105.

[27] H. Aydin, R. Melhem, D. Mossé, and P. Mejía-Alvarez, "Dynamic and aggressive scheduling techniques for Power-Aware Real-Time systems," *Proc. - Real-Time Syst. Symp.*, pp. 95–105, 2001.

[28] V. Moghaddas, M. Fazeli, and A. Patooghy, "Reliability-oriented scheduling for static-priority real-time tasks in standby-sparing systems," *Microprocess. Microsyst.*, vol. 45, pp. 208–215, 2016.

[29] S. Mittal, "A Survey of Techniques For Improving Energy Efficiency in Embedded Computing Systems," *Proc. Int. J. Comput. Aided Eng. Technol.*, pp. 440–459, 2014.

[30] M. Joseph and P. Pandya, "Finding Response Times in a Real-Time System.," *Comput. J.*, vol. 29, no. 5, pp. 390–395, 1986.

[31] D. Zhu, X. Qi, and S. Antonio, "Priority-Monotonic Energy Management for Real-Time Systems with Reliability Requirements," *25th Int. Conf. Comput. Des. 2007 (ICCD 2007)*, pp. 629–635, 2007.

[32] H. Cheng and S. Goddard, "SYS-EDF: a system-wide energy-efficient scheduling algorithm for hard real-time systems," *Int. J. Embed. Syst.*, vol. 4, no. 2, p. 141, 2009.

[33] D. J. Hand, H. Mannila, and P. Smyth, "Measuring the Performance of a Classifier," *Princ. Data Min.*, p. 546, 2001.

[34] N. Min-Allah, H. Hussain, S. U. Khan, and A. Y. Zomaya, "Power efficient rate monotonic scheduling for multi-core systems," *J. Parallel Distrib. Comput.*, vol. 72, pp. 48–57, 2012.

[35] Y. Guo, D. Zhu, H. Aydin, J. J. Han, and L. T. Yang, "Exploiting primary/backup mechanism for energy efficiency in dependable real-time systems," *J. Syst. Archit.*, vol. 78, pp. 68–80, 2017.

[36] P. Gepner, M. F. Kowalik, D. L. Fraser, and R. Tylman, "New multi-core Intel Xeon processors help design energy efficient solution for high performance computing," in *Proceedings of the International Multiconference on Computer Science and Information Technology, IMCSIT '09*, 2009, vol. 4, pp. 567–571.

Kiran Arora received her B.Tech degree in 2005 from Punjab Technical University, Jalandhar in Computer Science and Engineering. She did M.Tech in Computer Science and Engineering from Punjab Technical University, Jalandhar in 2011. She is pursuing Ph.D in Computer Sc. & Engg. from IK-Gujral Punjab Technical University, Jalandhar. Presently she is working as Assistant Professor in Computer Science & Engineering in BHSBIET, Lehragaga, Sangrur, India. Her current areas of interest include energy aware multiprocessor scheduling, and real time scheduling techniques.

Savina Bansal earned BSc in 1984 from Punjab University, Chandigarh, and Bachelor of engineering in Electronics & Electrical Communication from Punjab Engineering College, Chandigarh, India in 1988. She did Masters' in Computer Sc from Thapar Institute of Engg & Technology University, Patiala and PhD from IIT Roorkee (India) in parallel & distributed computing in 2003. Currently she is Professor with Giani Zail Singh College of Engg & Tech, a Campus College of Maharaja Ranjit Singh Punjab Technical University Bathinda, India. Her research interests include energy-efficient and fault tolerant multiprocessor scheduling, irregular interconnection networks, and wireless sensor networks. She is Fellow of IE(I), & IETE and senior member CSI. You can contact her at: savina [dot] bansal [at] gmail [dot] com.

Rakesh Kumar Bansal did Bachelor and Masters of engineering both from Thapar Institute of Engg & Technology TIET University, Patiala. Later, he did PhD (Engg) from Punjabi University Patiala in Punjab State of India. He started his professional career in technical education with, TIET University in 1986 and is presently Professor with Giani Zail Singh College of Engg & Tech, a campus college of Maharaja Ranjit Singh Punjab Technical University Bathinda, India. His areas of interest include wireless-sensor-networks, fault-tolerant real-time scheduling in parallel systems and energy-efficient computing. He is available at: drrakeshkbansal [at] gmail [dot] com.