International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-1, November 2019

Fault-Tolerant Energy-Aware Task Scheduling on Multiprocessor System for Fixed-Priority Real-Time Tasks

Kiran Arora, Savina Bansal, Rakesh Kumar Bansal

Abstract*: *Energy-aware real-time scheduling is gaining attention in recent years owing to environmental concerns and applications in numerous fields. System reliability also gets affected adversely with increasing energy dissipations posing serious challenges before the researchers. Keeping these in view, in recent times researchers have diverted to combining issues of fault-tolerance and energy efficiency. In literature, DVFS and DPM, most commonly used techniques for power management in task scheduling, are often combined with Primary/Backup technique to achieve fault tolerance against transient and permanent faults. Optimal algorithms, Earliest deadline first (EDF) and Rate-Monotonic (RM), meant for scheduling dynamic and fixed priority tasks respectively, have mainly been analyzed using a dual-processor approach for fault-tolerance and energy efficiency. In this paper, to handle higher workload of fixed-priority real-time tasks, energy-aware fault-tolerant scheduling algorithms are proposed for multiprocessor systems with balanced and unbalanced number of main and auxiliary processors. Simulations over extensive task-sets indicate that balanced approach is more energy-efficient than the unbalanced one.

Keywords*: - *energy-aware, fault-tolerance, fixed-priority, rate-monotonic, task scheduling

I. INTRODUCTION

Owing to rapid development in processor technology, real-time systems have become pervasive part of our day-to-day lives. Embedded systems can be found in abundance in 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][3]. Due to the growth in scale and complexity of real-time embedded systems, higher performance at a minimum energy consumption rate has become a need of time. Subsequently, many embedded systems are now implementing multiprocessor architectures into their design ranging from simple consumer electronics to space systems.

Revised Manuscript Received on November 30, 2019.
***** Correspondence Author
Kiran Arora*, Research Scholar, I.K. Gujral Punjab Technical University, Jalandhar, Punjab, India. erkiranarora@gmail.com.
Savina Bansal, Department of Electronics and Communication Engineering, Giani Zail Singh Campus College of Engineering and Technology, Bathinda, Punjab, India. savina.bansal@gmail.com.
Rakesh Kumar Bansal, Department of Electronics and Communication Engineering, Giani Zail Singh Campus College of Engineering and Technology, Bathinda, Punjab, India. drrakeshkbansal@gmail.com.

Energy-management has become a hot topic in research areas and industrial environments because of increased heat dissipation due to complex microarchitectural designs of modern multiprocessor computing platforms.

Dynamic voltage scaling (DVS) and dynamic power management (DPM) are two eminent and effective schemes at the operating system level that are typically employed for lessening energy consumption[4], [5].

Over the last two-decades, dynamic voltage scaling has attained substantial attention of the research community due to its quadratic property of saving energy with a decrease in supply voltage. Using DVS technique, the processor operates on low voltage levels to reduce energy consumption. However, decreased supply-voltage is unfavorable from performance perspective as it results in increased execution time of application. 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 to sleep state in idle intervals whenever possible [6], to achieve energy efficiency. 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 provide optimized energy saving [7].

Apart from guaranteeing the timely execution of tasks, computing systems for the execution of real-time applications should provide high standards of reliability, confirming proper functioning of the system even in the presence of faults. Faults are classified as transient and permanent faults [8]. Transient faults arise more frequently than permanent faults and are a major reason for soft errors also known as single event upset (SEUs) [9]. The rate of radiation-induced transient faults is 100 times more than permanent faults [10]. Also, rate of transient faults increases with a decrease in operating frequency showing negative effect of DVFS used for reducing energy consumption [11].

To deal with the adverse effect of reducing working voltage on system reliability, numerous works have been proposed in the series of reliability-aware power management [12]. Basic idea is to exploit slack for additional execution of replica to achieve fault-tolerance and providing desirable reliability. Execution of replica of the same task for the sake of decoupling the results of faulty executions from the output of the system attains excellent levels of reliability. Similarly,

for tolerating a single permanent fault while ensuring system reliability with respect to transient faults, the Standby-Sparing (SS) schemes have been explored for both aperiodic and periodic tasks [13] running on a dual-processor system based on the traditional Primary/Backup (PB) fault-tolerance approach.

Basically, SS schemes schedule primary and backup tasks separately on the main and auxiliary processors, respectively, to tolerate one permanent fault. However, this scheme enforces substantial energy overhead. Thus, attaining reliability and energy-saving together in the system is however challenging as they are conflicting with each other.

Judicious task allocation and scheduling, however, assists in achieving the above said objectives to a great extent [4], [5]. Earliest Deadline First (EDF) and Rate Monotonic (RM) are two optimal scheduling algorithms for real-time systems meant for fixed-priority and dynamic-priority tasks respectively[14]. The utilization bound of an algorithm is the maximum utilization that it can successfully handle without missing a deadline, which is 1 for EDF and $n * (2^{\frac{1}{n}} - 1)$ for RM for n tasks. Utilization bound for RM is generally referred to as Liu-Layland Bound (LLB). In the current work, fixed-priority task scheduling algorithm has been considered owing to its wider applicability in applications like embedded controller to space and avionic applications.

To the best of our knowledge, there is no existing work that schedules fixed-priority periodic real-time tasks on multiprocessor systems to save energy with both DPM and DVFS schemes along with tolerating one permanent fault and preserving system reliability against transient faults as well. In the current work, primary/backup approach on the main-auxiliary dual-processor system [15] for scheduling fixed priority periodic tasks has been extended to multiprocessor system.

In particular, contributions of the current work are summarized as follows:

1. First, primary/backup task scheduling is extended to a multiprocessor system with equal or balanced number of processors for both main and auxiliary processors, which execute primary and backup tasks, respectively.
2. Second, an unbalanced primary-backup approach with different number of main and auxiliary processors is implemented and analyzed for energy-efficiency.
3. Finally, the two proposed balanced and unbalanced fault-tolerant energy-aware task scheduling schemes, B-FEAT and U-FEAT, are evaluated through exhaustive simulations.

The rest of the paper is organized into following sections: Section 2 gives the related work. Section 3 elaborates models and assumptions. Section 4 explains fault-tolerant energy-aware task-scheduling scheme. Section 5 shows performance analysis. Finally, section 6 concludes the paper.

II. RELATED WORK

Energy-aware fault-tolerant task scheduling schemes can be differentiated based on redundancy used. Recovery task placement and checkpointing techniques refer to time redundancy whereas task replication techniques like primary/backup approach belong to the category of hardware redundancy. Hardware redundancy-based techniques basically use slack time on the processor to exploit DVFS and additional hardware for fault tolerance. Slack time (difference between execution time of task and its deadline) available is basically used for saving energy.

The use of DVS for energy management reduces reliability of a system due to increased amount of transient errors at lower operating frequency [9]. Reliability aware power management framework (RA-PM)[16][12] preserves reliability of a system in the presence of DVS. It schedules recovery copy of a task, which has been considered for voltage reduction, using available slack before applying DVFS. RA-PM was first proposed by Zhu et al. for few tasks selected from task-set/application for applying RA-PM[17]. Zhao et al. [18] claim that instead of placing recovery task copies statically for all tasks of periodic task set, modest recovery allowance with dynamically allocated recovery copies helps in achieving high reliability. Zhao et al. [19] has considered reliability preservation for energy-constrained systems for frame-based task system (for precedence constraint tasks) as well as for periodic task system to maximize reliability under given energy budget.

According to Zhao et al. [20][21], RA-PM schemes discussed above are conservative, because allocating multiple recovery blocks decreases the prospects for energy saving by reducing available slack for DVS. Thus, a new 'shared recovery' approach for RA-PM has been introduced by them, where in spite of separate recovery copies for scaled tasks, one global shared recovery block is reserved, which can be used by any task at any time in the situation of fault. Han et al. [22] also worked on energy-efficient fault tolerance with shared recovery block.

Unsal et al. have proposed an energy-aware fault tolerance approach using primary-backup scheme on distributed real-time systems [23]. Haque et al. used replication to attain reliability while saving energy with DVS [24] and presented the interplay of energy, reliability, frequency, and replication. Similar work has been done for dependent tasks by Salehi et al. [25] for three-level of redundancy that includes single execution (SE), dual modular redundancy (DMR) and triple modular redundancy (TMR).

Energy has been saved with DVS and DPM policies in the standby-sparing system by Aminzadeh et al. [26]. Ejlali et al. have proposed a low energy standby sparing (LESS) system, DVS and DPM are used on primary and spare processors, respectively for saving energy [27][28]. Guo et al. have proposed an energy-efficient fault tolerance scheme where fault has been tolerated with hardware redundancy while energy is saved with DVS [29] for earliest deadline first scheduling algorithm. The standby-sparing technique has also been exploited by Haque et al. for fixed priority real time periodic tasks with dual queue mechanism used for delaying start time of backups on secondary processor [13].

Ejlali et al. worked on a combined approach of energy management and fault tolerance on standby-sparing system for aperiodic non-preemptive tasks where main processor only executes primary tasks using DVFS whereas backup tasks are scheduled on spare processor that use DPM to save energy [27][28]. Guo et al. [29] and Haque et al.

[13] extended the work for real-time periodic tasks using dynamic and fixed-priority scheduling algorithms, respectively. A combination of backup deallocation and overlap reduction techniques with a primary/backup approach has been implemented for EDF scheduling algorithm by Guo et al. [29] on multiprocessor system.

Going by the related works, it is gathered that most of the energy-aware fault-tolerant techniques for fixed-priority real-time tasks concentrated on uni-processor or dual processor. As far as we understand, none of the energy-aware primary-backup approach tackled the problem on multiprocessor system for fixed-priority real time periodic tasks. So, keeping this in mind, our work focuses on energy-aware fault-tolerant mapping and scheduling of fixed-priority tasks on multiprocessor system.

III. MODELS AND ASSUMPTIONS

A. Task Model

The system under consideration consists of r homogeneous processors and handles fixed-priority periodic real-time tasks. A task-set Γ contains tasks such that $\Gamma = \{\tau_1 \ldots \tau_n\}$. Task τ_i has three parameters: $<c_i , t_i , d_i>$ where c_i represents worst-case execution time (WCET) at maximum operating frequency f_{max}, t_i is its period and d_i is relative deadline. Here, implicit deadlines are considered having $d_i = t_i$. An example of periodic task is shown in Fig. 1. Utilization u_i of task τ_i is given by the ratio of its worst-case execution time to the period, as $\frac{c_i}{t_i}$. Each task τ_i must complete before d_i so as to satisfy the real-time application constraints. The job $j_{i,j}$ is the j^{th} instance of task τ_i. The total utilization of the task-set U_{tot} is the sum of utilization of all tasks. The hyper-period $\hbar(\Gamma)$ of a task-set is defined as Least Common Multiple (LCM) of task periods. $HP(\tau_i)$ is a set of tasks having priority higher than τ_i. Utilization bound $LLB(n)$ for rate monotonic scheduling $n * (2^{\frac{1}{n}} - 1)$.

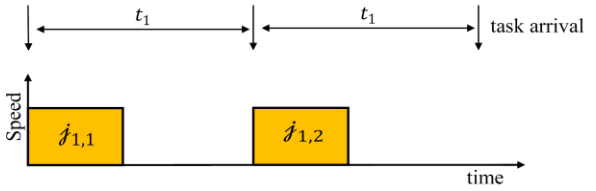


Fig. 1. A periodic task τ_1 with two jobs

Each primary task τ_i has a backup copy τ_i^b associated with it. Accordingly, Γ^B represents a set of all backup tasks τ_i^b, having identical parameters as τ_i, corresponding to task-set Γ. Similarly, job $j_{i,j}^b$ represents j^{th} instance of backup task τ_i^b.

B. Power Model

The processors are assumed to be capable of dynamic voltage scaling (DVS) and dynamic power management (DPM). Accordingly, a processor can operate in three power modes- active, idle and sleep.

Active Mode: In active mode, power consumed by a processor is modeled as sum of dynamic and static power, and is expressed as:

$$P_{active} = P_{dyn} + P_{static} \quad (1)$$

Since dynamic power (P_{dyn}) consists of frequency-dependent and frequency independent power and former one arises due to charging and discharging of load capacitance [30][13]. So, eq (1) can be written as [30] :

$$P_{active} = \varrho C_{ef} v^2 f + P_{ind} + P_{static} \quad (2)$$

where C_{ef} is total capacitance, ϱ is a gate activity factor. The supply voltage (v) has a linear relationship with frequency f [13]. Thus, Eq. (2) can be written as

$$P_{active} = \varrho C_{ef} f^3 + P_{ind} + P_{static} \quad (3)$$

The maximum processor frequency f_{max} has been set to 1. All other frequency values are normalized with respect to it. There exists a critical frequency value f_{crit}, below which DVS does not remain effective. Critical frequency depends on the total capacitance and independent power of the system [21][31].

Idle Mode: When a processor is not executing anything, then it switches to low dynamic power state called idle state. Power consumed in idle state will consist of static power and low dynamic power (where f = 0) expressed as

$$P_{idle} = P_{d0} + P_{static} \quad (4)$$

Sleep Mode: When a processor is in sleep state, it consumes minimum power needed to keep the clock running and maintaining the basic circuit of system. Only static energy is consumed during sleep state. But transition to and from sleep state takes a significant time and energy overhead. Each device has some minimum transition time from one state to another called device breakeven time [32]. When idle interval is greater than device breakeven time, only then processor is put to sleep to take advantage of DPM.

C. Fault Model

The real-time computing system under consideration is assumed to tolerate transient faults. Generally, transient faults are modeled with Poisson distribution with an average fault rate γ [19]. Melhem et al. [31] showed that reduced supply voltage results in exponentially increased fault rate. Thus, average arrival rate of soft error caused by transient faults in DVS enabled processor at scaled processing frequency f is given as:

$$\gamma(f) = \gamma_0 . g(f) \quad (5)$$

where γ_0 is average fault rate corresponds to the maximum processing frequency f_{max}. Zhu et al. suggested an exponential fault rate model for soft errors caused due to transient faults as follows:

$$\gamma(f) = \gamma_0 . g(f) = \gamma_0 . 10^{\frac{s(1-f)}{1-f_{crit}}} \quad (6)$$

where $s(> 0)$ indicates the sensitivity of the rate of soft error to dynamic voltage and frequency scaling DVFS and f_{crit} is minimum energy-efficient processing frequency.

Problem Statement: The objective of the work is to map and schedule a given set of real-time fixed-priority periodic tasks on a multiprocessor system so that all tasks are able to complete within their respective deadlines in the presence of transient as well as permanent faults with minimum energy consumption.

IV. FAULT-TOLERANT ENERGY-AWARE TASK SCHEDULING ON MULTIPROCESSOR SYSTEM

The problem of fault-tolerant energy-aware task scheduling on multiprocessor system can be divided into three phases:

- Task mapping
- Frequency assignment
- Scheduling of tasks on processors

A. Task mapping

This phase deals with the allocation of tasks on different processors so as to have balanced workload on different processors. A processor executing primary copies is called main processor and the one executing backup copies is referred as auxiliary processor in this work. Energy consumption gets directly affected by total workload on a processor as slack exploitation is basically used for saving energy. Delaying of backup copies for as late as possible execution and maximum exploitation of DVFS scheme depends on available slack. With higher workload, scope of reducing operating voltage diminishes due to lesser availability of slack. To have economical schedule in terms of energy consumption, all processors must have a well-adjusted workload. Therefore, allocation of all tasks is done with WFD scheme so as to have comparable load on all processors. Due to the Liu-Layland bound, maximum utilization that can be handled successfully by a system using RM scheduling is given as $\frac{r}{LLB}$. Also, schedulability condition must be satisfied with time demand analysis [13] as task mapping is acceptable if all processors satisfy deadline constraints. Further, main and auxiliary processors can be organized in different configurations such as balanced and unbalanced. In Balanced mapping scheme, half of the total number of processors will act as main processors and other half as auxiliary processors. Every main processor has its associated auxiliary processor having same subset of tasks.

Algorithm 1: BALANCED MAPPING scheme

Input: task sets Γ and Γ^B ; r is number of processors; n is size of task-set;

Output: processor mapping with balanced allocation;

Set $A = r/2, B = r - A$;

Sort tasks in Γ and Γ^B in decreasing order of utilization

for $(i \rightarrow 1\ to\ n)$

Map with the WFD allocation scheme $\tau_i \in \Gamma$ to $R(1:A)$ such that $U(R^{main}(k)) \leq LLB$.

Map associated backup copy $\tau^b \in \Gamma^B$ to $R^{aux}(B+k)$.

end for

Algorithm 2: UNBALANCED MAPPING scheme

Input: task sets Γ and Γ^B ; r is number of processors; n is size of task-set;

Output: processor mapping with unbalanced allocation;

Set $A = \frac{U_{tot}}{LLB}$; $E_{min} = \infty$; X^{minE}=A

Sort tasks in Γ and Γ^B in decreasing order of utilization.

for $(k \rightarrow A\ to\ r - A\ )$

$B = r - A$;

for $(i \rightarrow 1\ to\ n)$

Map with the WFD allocation scheme $\tau_i \in \Gamma$ to $R(1:A)$ such that $U(R^{main}(k)) \leq LLB$.

Map with the WFD allocation scheme $\tau^b \in \Gamma^B$ to $R(1:B)$ such that $U(R^{aux}(k)) \leq LLB$.

endfor

Get energy consumed E by scheduling with the current configuration.

If $(E < E_{min})$

$E_{min} = E$

X^{minE}=X

end if

end for

Another way to map tasks on multiprocessor system is an unbalanced mapping scheme where no pairing of processors into couple exists. The subset of tasks on given main processor is not necessarily same on any other auxiliary processor. Number of main processors varies between the range $\frac{U_{tot}}{LLB}$ to $(r - \frac{U_{tot}}{LLB})$. Different configurations will have varied impacts on energy savings of multiprocessor system as analyzed in this work. Algorithm 1 and Algorithm 2 present the balanced and unbalanced mapping techniques.

B. Frequency assignment

This phase deals with the assignment of different operating frequencies to primary tasks assuming the underlying main processor to be DVFS enabled. Dynamic voltage/frequency scaling scheme is employed on main processor which helps in reducing energy consumption by lowering execution speed of processor. This phase is of great significance as operating frequency or speed of tasks directly affects energy consumption of the system as voltage reduction has quadratic effect on power saving. Main processor can reduce its speed between the range of critical frequency to maximum available frequency value. The frequency of main processor is chosen based on Sys-clock algorithm [33] as it provides minimum frequency that satisfies time demand analysis. However, due to technical limitations, processors can run only on few discrete frequencies depending on the selected hardware. Since lowering of operating frequency adversely affects the system reliability against transient faults, so, auxiliary processor running backup copies of tasks executes with maximum frequency. Energy management on auxiliary processor is achieved using DPM in this work.

C. Scheduling of tasks

This phase deals with assigning start times to the allocated tasks. Primary tasks on main processors are scheduled using rate monotonic scheduling at an assigned frequency (phase 2) whereas on auxiliary processors tasks are scheduled in a delayed manner using dual priority scheduling [15]. Dual priority scheduling maintains two queues- lower and upper. Tasks are assigned to lower queue at the time of generation and upgraded to upper queue at activation time. Activation time is the latest possible time up to which a task can be delayed without missing deadline and can be found using time demand analysis. In the upper queue, tasks are maintained with RM priority.

To save energy on auxiliary processor backup copy of task is canceled as soon as its associated primary copy completes successfully.

Algorithms explaining mapping, frequency assignment and scheduling procedure with balanced (B-FEAT) and unbalanced (U-FEAT) approach are shown in Algorithm 3 and Algorithm 4.

Transient as well as permanent faults can be handled with the proposed algorithms. As each task has an associated backup copy, so all primary copies can tolerate transient faults together. For permanent fault, the B-FEAT algorithm has exactly same copies of tasks on main and auxiliary processor, so system can handle one permanent fault of processor per couple while U-FEAT can tolerate fault of either main cores or auxiliary cores at a time.

Algorithm 3: U-FEAT

Input: task sets Γ and Γ^B ; r is number of processors; n is size of task-set;

Output: schedule with balanced allocation;

----------**Phase 1**----------

Call Algorithm 1 for balanced allocation.

----------**Phase 2 and Phase 3**----------

for (each primary processor $R^{main}(k)$: $k = 1 \rightarrow Y$) do

Get f = Sys-Clock[R^{main} (Γ)]

Set f = min{$f_i | f_i \geq f_{critical}$, i = 1 , . . . , L } for all tasks on R^{main};

Generate RM schedule.

end for

for (each auxiliary processor $R^{aux}(k)$: $k = 1 \rightarrow Z$) do

Set f = 1 or all tasks on R^{aux};

Generate the schedule using dual-priority scheduling for tasks on R^{aux}.

end for

Algorithm 4: U-FEAT

Input: task sets Γ and Γ^B ; r is number of processors; n is size of task-set;

Output: schedule with unbalanced allocation;

----------**Phase 1**----------

Call Algorithm 2 for unbalanced allocation.

----------**Phase 2 and Phase 3**----------

for (each primary processor $R^{main}(k)$: $k = 1 \rightarrow Y$) do

Get f = Sys-Clock[R^{main} (Γ)]

Set f = min{$f_i | f_i \geq f_{critical}$, i = 1 , . . . , L } for all tasks on R^{main};

Generate RM schedule.

end for

for (each auxiliary processor $R^{aux}(k)$: $k = 1 \rightarrow Z$) do

Set f = 1 or all tasks on R^{aux};

Generate the schedule using dual-priority scheduling [34] for tasks on R^{aux}.

End for

V. PERFORMANCE EVALUATION

A. Simulation Parameters

In this section, performance of B-FEAT and U-FEAT allocation schemes is evaluated on 16 multiprocessor systems for energy efficiency. For each simulation, 100 periodic task sets are generated. The UUnifast algorithm [15] has been used for generating task-set utilization. In each task-set, number of tasks are generated to attain the value of system utilization as 3, 4, and 5. For each task-set average task utilization has been set to 0.1 and 0.05 to indicate large and small task sizes. Average number of tasks in the task-set will be $\frac{U_{tot}}{u_{avg}}$ for system utilization U_{tot}. Deadline is set equal to period and worst-case execution time is calculated as a product of utilization and period. The system under consideration has seven normalized discrete speeds with respect to maximum processor frequency (0.4,0.5,0.6,0.7,0.8,0.9 and 1).

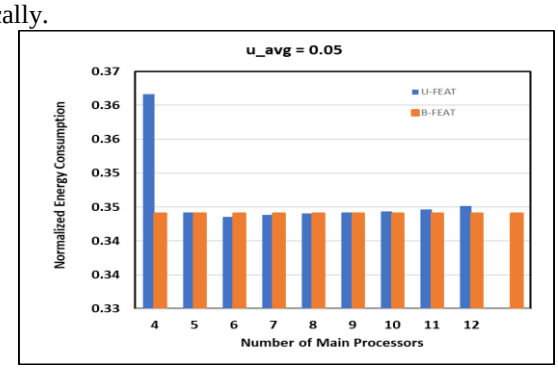
Table-I. Simulation parameters

Parameter	Value
γ_0	.000001
Breakeven time	1.5 ms
Energy overhead	0.1mJ
s	2
P_{static}	5% of the maximum dependent power
P_{ind}	15% of the maximum dependent power

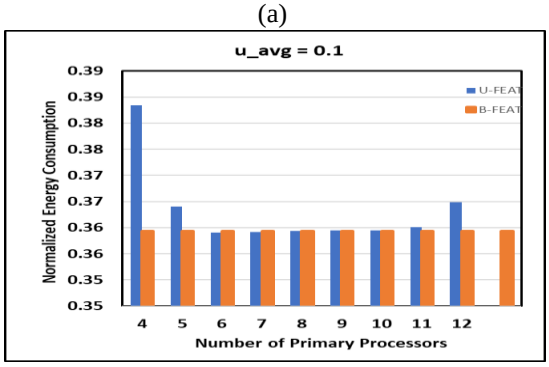
Results are normalized with respect to No-Power Management scheme (NPM) where tasks are allocated with WFD allocation scheme and both primary and backup copy is executed at maximum frequency.

B. Impact of varying number of main processors

For 16-processor multiprocessor system, B-FEAT and U-FEAT schemes are evaluated for system utilizations 2.0, 3.0 and 4.0. It is assumed that all tasks run up to their WCET where backup task has maximum execution speed and primary tasks execute at lower frequencies assigned to them statically.



(a)

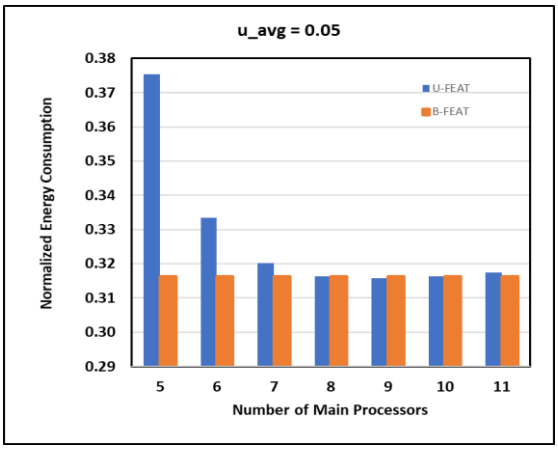


(b)

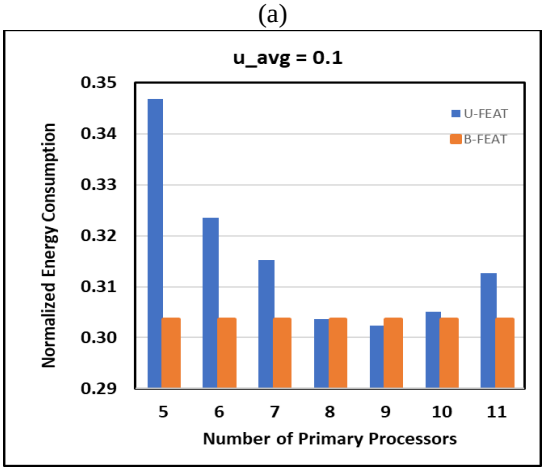
Fig. 2. Energy consumption with system utilization = 2.0

For B-FEAT and U-FEAT scheme (for lower number of primary processors) energy consumption is high for system utilization 2.0, 3.0, and 4.0 as shown in Fig. 2, Fig. 3 and Fig. 4, respectively. However, for fixed-priority real-time task scheduling, configuration with nearly equal number of main and auxiliary processors consumes least amount of energy. It has also been noticed that energy consumption also rises for very low number of auxiliary processors. Thus, this fact supports the truthfulness of scheme B-FEAT, which shows that 50% main and auxiliary processor configuration is a best energy-saving configuration.

If there are lesser number of auxiliary processors, workload on each auxiliary processor will be comparatively high that gives lesser chance to delay backup copies. Execution of extra backup copies at maximum frequency gives rise to more energy consumption.



(a)



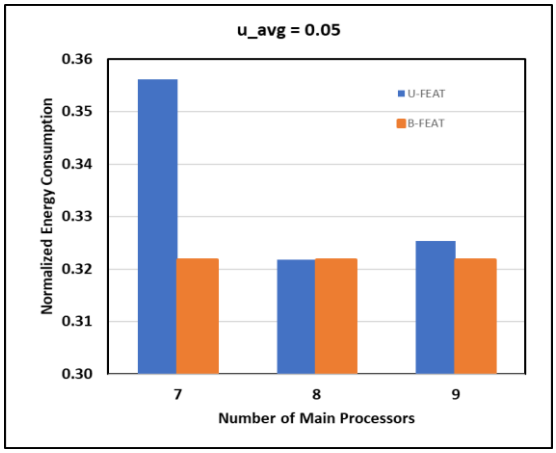
(b)

Fig. 3. Energy consumption with system utilization = 3.0

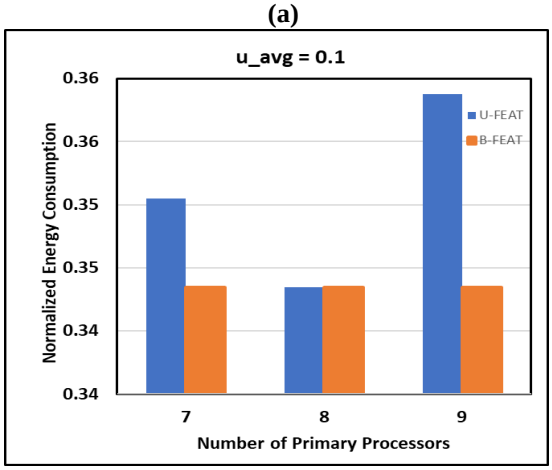
Energy-saving over NPM scheme by B-FEAT and U-FEAT are shown in Table 2 with small and large tasks have average task utilization 0.05 and 0.1, respectively.

Table-II. Energy Improvement over NPM (in percent)

	U_avg=0.05		U_avg = 0.1	
System Utilization	**B-FEAT**	**U-FEAT**	**B-FEAT**	**U-FEAT**
2.0	66%	65%	64%	64%
3.0	68%	67%	70%	68%
4.0	68%	67%	66%	64%



(a)



(b)

Fig. 4. Energy consumption with system utilization = 4.0

VI. CONCLUSION

In this paper, two energy-aware fault-tolerant task scheduling algorithms on the multiprocessor system for fixed-priority real-time tasks have been proposed. B-FEAT takes up the concept of balanced configuration of processors while mapping tasks, that is number of main processors and auxiliary processors are same to organize them in processor couples. U-FEAT, however, can have varied number of main and auxiliary processors and it does not require to make pairs of processors. DVFS and DPM are used for energy management and primary/backup strategy is used to handle fault tolerance. Both the proposed DVFS based task scheduling schemes result in minimum 64% energy savings in comparison to no-power management scheme though the balanced B-FEAT algorithm showing marginally better performance than unbalanced U-FEAT algorithm for all workloads for fixed-priority real-time tasks.

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. K. Bansal, "Schedulability and Energy Consumption Analysis of Energy-Aware Fixed-Priority Task Scheduling Schemes," *Int. J. Res. Electron. Comput. Eng.*, vol. 7, no. 1, pp. 1254–1259, 2019.
4. N. Kaur, S. Bansal, and R. K. Bansal, "Duplication-controlled static energy-efficient scheduling on multiprocessor computing system," *Concurr. Comput.* , vol. 29, no. 12, 2017.
5. N. Kaur, S. Bansal, and R. K. Bansal, "Energy efficient duplication-based scheduling for precedence constrained tasks on heterogeneous computing cluster," *Multiagent Grid Syst.*, vol. 12, no. 3, pp. 239–252, 2016.
6. 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.
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. D. K. Pradhan, Ed., *Fault-tolerant Computer System Design*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1996.
9. D. Zhu, R. Melhem, and D. Moss, "The Effects of Energy Management on Reliability in Real-Time Embedded Systems," *IEEE/ACM Int. Conf. Comput. Des. 2004*, pp. 35–40, 2004.
10. 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.
11. D. Zhu, R. Melhem, and D. Mossé, "The effects of energy management on reliability in real-time embedded systems," in *Proceedings of the 2004 IEEE/ACM International conference on Computer-aided design*, 2004, pp. 35–40.
12. D. Zhu and H. Aydin, "Reliability-aware energy management for periodic real-time tasks," *IEEE Trans. Comput.*, vol. 58, no. 10, pp. 1382–1397, 2009.
13. 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.
14. 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. 1973.
15. M. A. Haque, H. Aydin, and D. Zhu, "Energy-aware task replication to manage reliability for periodic real-time applications on multicore platforms," in *2013 International Green Computing Conference Proceedings*, 2013, pp. 1–11.
16. D. Zhu, "Reliability-Aware Dynamic Energy Management in Dependable Embedded Real-Time Systems," *ACM Trans. Embed. Comput. Syst.*, vol. 10, no. 2, p. 27, 2010.
17. D. Zhu and H. Aydin, "Energy Management for Real-Time Embedded Systems with Reliability Requirements," in *Proceedings of the 2006 IEEE/ACM international conference on Computer-aided design*, 2006, pp. 528–534.
18. B. Zhao, H. Aydin, and D. Zhu, "Energy management under general task-level reliability constraints," in *2012 IEEE 18th Real Time and Embedded Technology and Applications Symposium*, 2012, pp. 285–294.
19. B. Zhao, H. Aydin, and D. Zhu, "On maximizing reliability of real-time embedded applications under hard energy constraint," *IEEE Trans. Ind. Informatics*, vol. 6, no. 3, pp. 316–328, 2010.
20. B. Zhao, H. Aydin, and D. Zhu, "Enhanced reliability-aware power management through shared recovery technique," in *Proceedings of the 2009 International Conference on Computer-Aided Design*, 2009, pp. 63–70.
21. B. Zhao, H. Aydin, and D. Zhu, "Shared Recovery for Energy Efficiency and Reliability Enhancements in Real-time Applications with Precedence Constraints," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 18, no. 2, pp. 23:1–23:21, 2013.
22. Q. Han, L. Niu, G. Quan, S. Ren, and S. Ren, "Energy efficient fault-tolerant earliest deadline first scheduling for hard real-time systems," *Real-Time Syst.*, vol. 50, no. 5–6, pp. 592–619, 2014.
23. O. S. Unsal, I. Koren, and C. M. Krishna, "Towards Energy-Aware Software-Based Fault Tolerance in Real-Time Systems," in *Proceedings of the 2002 international symposium on Low power electronics and design*, 2002, pp. 124–129.
24. M. A. Haque, H. Aydin, and D. Zhu, "On Reliability Management of Energy-Aware Real-Time Systems Through Task Replication," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 3, pp. 813–825, 2017.
25. M. Salehi *et al.*, "DRVS: Power-efficient reliability management through Dynamic Redundancy and Voltage Scaling under variations," in *2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 2015, pp. 225–230.
26. S. Aminzadeh and A. Ejlali, "A Comparative Study of System-Level Energy Management Methods for Fault-Tolerant Hard Real-Time Systems," *Computers, IEEE Transactions on*, vol. 60, no. 9. pp. 1288–1299, 2011.
27. A. Ejlali, B. M. Al-Hashimi, and P. Eles, "A Standby-sparing Technique with Low Energy-overhead for Fault-tolerant Hard Real-time Systems," in *Proceedings of the 7th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis*, 2009, no. ACM, pp. 193–202.
28. A. Ejlali, B. M. Al-hashimi, and P. Eles, "Low-Energy Standby-Sparing for Hard Real-Time Systems," *IEEE Trans. Comput. Des. Integr. Circuits Syst.*, vol. 31, no. 3, pp. 329–342, 2012.
29. 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.
30. 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.
31. Dakai Zhu, R. Melhem, and D. Mosse, "The effects of energy management on reliability in real-time embedded systems," in *IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004.*, 2004, pp. 35–40.
32. J. J. Chen and T. W. Kuo, "Procrastination determination for periodic real-time tasks in leakage-aware dynamic voltage scaling systems," in *2007 IEEE/ACM International Conference on Computer-Aided Design*, 2007, pp. 289–294.
33. S. Saewong and R. Rajkumar, "Practical voltage-scaling for fixed-priority rt-systems," in *The 9th IEEE Real-Time and Embedded Technology and Applications Symposium, 2003. Proceedings.*, 2003, pp. 106–114.
34. R. Davis and A. Wellings, "Dual priority scheduling," in *Proceedings 16th IEEE Real-Time Systems Symposium*, 2002, pp. 100–109.

AUTHORS PROFILE

Kiran Arora received her B.Tech degree in 2005 from I.K. Gujral Punjab Technical University, Jalandhar in Computer Science and Engineering and M.Tech in Computer Science and Engineering from I.K. Gujral Punjab Technical University, Jalandhar in 2011. She is pursuing Ph.D in Computer Science & Engg. from I.K. 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, fault-tolerant real-time scheduling. She is available at erkiranarora@gmail.com

Savina Bansal from India earned her PhD (Engg.) from IIT Roorkee, Masters in Computers from TIET, Patiala, Bachelors in Engg. from PEC, and Bachelors in Science from Punjab University, Chandigarh respectively. She is a Professor at Giani Zail Singh Campus College of Engineering &Technology, Maharaja Ranjit Singh Punjab Technical University, Bathinda, since 2005. Earlier Dean (R&D), she is now Dean (Academics) at Maharaja Ranjit Singh Punjab Technical University, Bathinda, India. Her current areas of interest include energy-efficient and fault-tolerant scheduling on parallel & distributed computing systems, wireless sensor networks, image processing and wireless communication systems. She had guided 4 PhD scholars and currently 4 candidates, including the author, are working with her. She has more than 100 publications at the International level. She is available at savina.bansal@gmail.com

Rakesh Kumar Bansal from India earned his PhD (Engg.) from Punjabi University, Patiala and Masters in Computers and Bachelors in Engg. both from TIET, Patiala. He is a Professor at Giani Zail Singh Campus College of Engineering & Technology, Maharaja Ranjit Singh Punjab Technical University, Bathinda, and Dean (Students Welfare) at Maharaja Ranjit Singh Punjab Technical University, Bathinda, India. His areas of interest include real time fault-tolerant multiprocessor scheduling, distributed computing and wireless sensor networks. He has 60 publications at the International level. He is available at drrakeshkbansal@gmail.com