A Scheduling Algorithm for Hard Real-Time System with the Constraints of Harvesting Energy in DVFS Platform

Abhilasha Tiwari, Mr.Sanjay Gupta, Mrs.Mona Kumari

***Abstract*— Real-time systems are computer system that monitor, respond to, or control an external environment. This environment is connected to the computer system through sensors, actuators, and other input-output interfaces. It may consist of physical or biological objects of any form and structure. The computer system must meet various timing and other constraints that are imposed on it by the real-time behavior of the external world to which it is interfaced. Hence come the name *real time*. Another name for many of these systems is *reactive systems*, because their primary purpose is to respond to or react to signals from their environment. A real-time computer system may be a component of a larger system in which it is embedded; reasonably, such a computer component is called an *embedded system*.**
Real-time embedded system that must carry their own power source and cannot depends on the power outlet on the wall, apart from feasibly schedule the set of tasks, power management is also the major issue because without power the system is useless.

In this propose paper a harvesting aware hard real-time scheduling algorithm with dynamic variable speed assignment and set of frequency selection scheme to set of periodic tasks aims to reduce the energy consumption while feasibly schedule the set of periodic tasks within their deadline. This can be achieved by DVFS(Dynamic Voltage and frequency Selection), executing the task with the speed such that a task can consume as much energy which is quite sufficient to complete it successfully within its deadline.

***Index Terms*— Real-Time Scheduling, DVFS(Dynamic Voltage and Frequency Selection), Energy Harvesting, Periodic Tasks, Embedded System, Power Management., Real time hard scheduling.**

I. INTRODUCTION

A real-time system is one that must process information and produce a response within a specified time, else risk severe consequences, including failure. That is, in a system with a real-time constraint it is no good to have the correct action or the correct answer ***after*** a certain deadline: it is either by the deadline or it is useless! In a lose approach all practical systems can be said to be real-time systems because they must produce an output or respond to the user's commands within a reasonable amount of time (insurance company responding to letters, word processor displaying what was typed on the screen, mobile phones responding with delays that allow 'comfortable' conversation). These systems where 'uncomfortably' long response times are a nuisance (windows2000 springs to my mind) but the system still functions even if deadlines are sometimes not met are called **soft real-time systems**. Systems where failure to meet response time constraints leads to catastrophic system failure (aircraft crashing, car skidding, patient dying before corrective action is performed) are called **hard real-time systems**.

Most of time these light weighted devices remains beyond the scope of recharges the battery. This may be due to unavailability of recharging point or time required for recharging due to mobility. Thus, for smooth functioning of this light weight mobile device one has to facilitate it either with large capacity battery or powerful power management technique to enlarge the battery back-up time. However in some applications, replacing/recharging a battery is costly or impractical, wireless sensor net- work is one of such application, where the sensor nodes are deployed in a wide wild area for environment surveillance. Hence, ideally such a system should be designed to operate perpetually with the battery being the only energy source [1, 3].

Advanced technical developments have increased the efficiency of devices in capturing trace amounts of energy from the environment and transforming them into electrical energy. In addition, advancements in microprocessor technology have increased power efficiency, effectively reducing power consumption requirements. In combination, these developments have sparked interest in the engineering community to develop more and more applications that utilize energy harvesting for power. Energy harvesting from a natural source where a remote application is deployed, and where such natural energy source is essentially inexhaustible, is an increasingly attractive alternative to inconvenient wall plugs and costly batteries. This essentially free energy source, when designed and installed properly, is available maintenance-free and is now available throughout the lifetime of the application. Such systems can be more reliable than wall plugs or batteries. In addition, energy harvesting can be used as an alternative energy source to supplement a primary power source and to

Manuscript received May 29, 2014.
Abhilasha Tiwari, M.Tech 4 th Sem Student, Computer Science & Engineering, VITS Engineering College, Jabalpur
Mr.Sanjay Gupta, Assistant Professor Dept of Computer Science & Engineering, VITS Engineering College, Jabalpur
Mrs.Mona Kumari, Assistant Professor Dept of Computer Science & Engineering, G.L.Bajaj Engineering College,, Noida

enhance the reliability of the overall system and prevent power interruptions.[4-5]

Any system in which the time at which output is produced is significant. This is usually because the input corresponds to some event in the physical world, and the output has to relate to that same event. The lag from input time to output time must be sufficiently small for acceptable timeliness.[8-10]

In a Today's environment most of the hard real-time embedded system executing on the platform that are mobile and carry their own power source in the form of battery and do not depend on power outlet on wall. Most of the time this mobile device remains beyond the scope of recharging their battery due to mobility or unavailability of recharging point, for example mobile video phone applications require light weight device movable across the globe [11, 13].

A. Sources of Energy harvesting hard real time system

- Mechanical Energy – from sources such as vibration, mechanical stress and strain
- Thermal Energy – waste energy from furnaces, heaters, and friction sources
- Light Energy – captured from sunlight or room light via photo sensors, photo diodes, or solar panels
- Electromagnetic Energy – from inductors, coils and transformers
- Natural Energy – from the environment such as wind, water flow, ocean currents, and solar
- Human Body – a combination of mechanical and thermal energy naturally generated from bio-organisms or through actions such as walking and sitting
- Other Energy – from chemical and biological sources

It is important to note, that all these energy sources are virtually unlimited and essentially free, if they can be captured at or near the system location.

B. Features of Energy harvesting hard real- time systems

1. System can be operate on lowest standby current to maximize storage of energy.
2. It consume lowest possible energy when device is active.
3. Ability to turn on and off instantaneously.
4. Analog capability for sensor interfacing and measurement.
5. System can operate with low voltage range.
6. It has a Lowest leakage current to maximize harvested energy.

C. Extremities of Energy harvesting hard real-time systems

1. Renewable energy sources available in environment are unstable.
2. Intensity of energy from renewable energy sources varies with time, for example in case of solar energy at day time the intensity of light is very high but during night it will be zero.
3. Limited size of energy storage or battery.

Apart from all these constraints, to maximize the possible numbers of tasks to be scheduled, an efficient power management technique is required. Present study focuses on scheduling periodic tasks with deadline, on a uniprocessor platform and variable speed system which is powered by renewable energy storage with limited capacity such as battery or a capacitor.

The content of present study is summarized in different sections. Section-2 introduces the related research works. The energy harvesting system model and some assumptions are described in Section-3. Section-4 explains the proposed methodology with example. Simulation results and discussions are present in Section-5. Finally Section-6 summarizes present study.

D. Components of an Energy harvesting hard real time System

An energy harvesting system generally requires an energy source such as vibration, heat, light or air flow and three other key electronic components, including:

- An energy conversion device such as a piezoelectric element that can translate the energy into electrical form
- An energy harvesting module that captures, stores and manages power for the device
- An end application such as a ZigBee-enable wireless sensor network or control and monitoring devices

E. 1.5 Energy Harvesting Applications

Many real life applications using energy harvesting system power are now practical. Wireless sensor network systems such as ZigBee systems often benefit from energy harvesting power sources. For example, when a wireless node is deployed at a remote site where a wall plug or a battery is either unreliable or unavailable, energy harvesting can augment or supply power. In another example, a remote control node running on energy harvesting can be implemented as a self-powered electronic system. And in yet other situations, multiple energy sources can be used to enhance the overall efficiency and reliability of any system.

II. RELATED WORK

In [1] Real-time scheduling refers to the problem in which there is a deadline associated with the execution of a task. In this paper, Author's addresses the scheduling problem for a uniprocessor platform that is powered by a renewable energy storage unit and uses a recharging system such as photovoltaic cells. First, Author's describes a model where two constraints need to be studied: energy and deadlines. Since executing tasks require a certain amount of energy,

classical task scheduling like earliest deadline is no longer convenient. Author's present an on-line scheduling scheme, called earliest deadline with energy guarantee (EDeg) that jointly accounts for characteristics of the energy source, capacity of the energy storage as well as energy consumption of the tasks, and time. In order to demonstrate the benefits of our algorithm, Author's evaluate it by means of simulation.Author's show that EDeg outperforms energy non-clairvoyant algorithms in terms of both deadline miss rate and size of the energy storage unit.

In [2] real time scheduling; preemption is one of the causes of run time overhead and large memory requirements. This paper focuses on reducing the number of preemptions in Earliest Deadline First (EDF) scheduling using a technique called Dynamic Preemption Threshold Scheduling (DPTS) in uniprocessor platform. This method is an improvement over existing threshold algorithms, but the complexity is slightly higher. The simulation results show that context switches are reduced by about 91% on an average. This technique is also applied for scheduling of sporadic requests along with periodic tasks. Preemptions that occur when tasks share resources and are required to synchronize are also reduced in this work. Their work also focuses on task set generation with limited hyper period (L.C.M. of periods of the tasks) as compared to previous scheduling algorithm it is not that much efficient because it is not saving energy.

In [3] real-time embedded system, that must carry their own power source and cannot depends on the power outlet on the wall, apart from feasibly schedule the set of tasks, power management is also the major issue because without power the system is useless. In this paper, Author's propose a harvesting aware real-time scheduling algorithm with variable speed assignment scheme to set of periodic tasks aims to reduce the energy consumption while feasibly schedule the set of periodic tasks within their deadline. This can be done by DVS(Dynamic Voltage and frequency Selection), executing the task with the speed such that a task can consume as much energy which is quite sufficient to complete it successfully within its deadline. this scheduling algorithm is good in terms of energy saving and job assignment for a periodic task ,But in this scheduling algorithm number of variables are more and it is very lengthy also that causes the more complexity as compare to previous algorithm,

In [5] this paper Author's propose a low-complexity and effective task mapping, scheduling and power management method for multi-core real-time embedded systems with energy harvesting. The proposed method is based on the concept of task CPU utilization, which is defined as the worst-case task execution time divided by its period. This work mathematically proves that by allocating the new task to the core with the lowest utilization, we can achieve the lowest overall energy dissipation. This method, combined with a new dynamic voltage and frequency selection (DVFS) algorithm with energy harvesting awareness and task slack management (TSM) forms the proposed UTilization Based (UTB) algorithm. With periodical tasks in a multi-core platform, this partitioned scheduling method is optimal for energy dissipation if the proposed utilization-based scheduling and DVFS algorithm is applied on each core. Experimental results show that new algorithm achieves better performance in terms of deadline miss rate in a single-core environment, comparing to the best of existing algorithm. When applied on a multi-core platform, the UTB algorithm achieves better efficiency in utilizing the harvested energy and overflowed energy.

In [10] The optimality of the Earliest Deadline First scheduler for uniprocessor systems is one of the main reasons behind the popularity of this algorithm among real-time systems. The ability of fully utilizing the computational power of a processing unit however requires the possibility of preempting a task before its completion. When preemptions are disabled, the schedulability overhead could be significant; leading to deadline misses even at system utilizations close to zero. On the other hand, each preemption causes an increase in the runtime overhead due to the operations executed during a context switch and the negative cache effects resulting from interleaving tasks' executions. These factors have been often neglected in previous theoretical works, ignoring the cost of preemption in real applications. A hybrid limited-preemption real-time scheduling algorithm is derived here, that aims to have low runtime overhead while scheduling all systems that can be scheduled by fully preemptive algorithms. This hybrid algorithm permits preemption where necessary for maintaining feasibility, but attempts to avoid unnecessary preemptions during runtime. The positive effects of this approach are not limited to a reduced runtime overhead, but will be extended as well to a simplified handling of shared resources.

III. HARD REAL TIME SYSTEM MODEL AND HYPOTHESIS

System consists of a uniprocessor system with a set of independent preemptive periodic tasks p_1, p_2, p_3 .....P_n. And each task p_i has an attribute:

a_i = arrival time of task p_i

$e_i(s_i)$ = worst case execution time of task p_i at speed s_i

$$E_{Source}(t_1,t_2) = (t_2 - t_1)*r \quad \text{..... (1)}$$

In energy harvesting hard real time system it is considered that the rate of harvesting is approximately equals to 1J/sec and is constant though out the day, and at night time the harvesting rate equals to zero.

A. *Energy Storage*

Here, we assume that a limited energy storage that may be charged up to its capacity C. If no tasks are executed and the stored energy has reaches its capacity leading to energy overflow.

$$0 \le E_C(t) \le C \quad \text{.....} \quad (2)$$

For executing the task, power $P_D(t)$ and the respective energy $E_D(t_1,t_2)$ is drained from the storage to execute tasks. We have the following relation:

$$E_C(t_2) \leq E_C(t_1) + E_{source}(t_1,t_2) - E_D(t_1,t_2) \; \forall \; t_2 > t_1$$

Therefore

,

$$E_D(t_1,t_2) \leq E_C(t_1) + E_{source}(t_1,t_2) \; \forall \; t_2 > t_1 \ldots\ldots\ldots\ldots\ldots (3)$$

B. *Energy Drain*

Energy is the function of speed level s_i.The energy drain in time interval

$$(t_1,t_2)E_D(t_1t_2)=\int P_D(s_i)$$

$E_i(s_i)$ = Energy demand of task p_i to complete its execution with speed s_i

d_i = deadline of task p_i

Here a dynamic priority scheduling algorithm i.e, earliest deadline first scheduling algorithm is considered for assigning the priority and scheduling the set of independent periodic tasks. The DVFS processor that is capable of operating at three different voltage levels V_1, V_2 and V_3 with the corresponding speed levels s_1, s_2 and s_3 is considered in this system. The speed s_1 is the lowest operating speed level measured at voltage V_1 whereas the maximum speed s_3 at the voltage level V_3. Here the processor runs at any of the speed level between s_1 to s_3. Power or Energy consumption at the speed s_i is given as:

$$P_i = C(s)^3 \ldots\ldots\ldots\ldots (4)$$

Response time of taskp_i at speed s_i is the sum of its own execution time requirement and the execution time of its higher priority tasks preempting it.

The System modeled with energy source, energy storage, energy drain, DVFS processor and real time periodic tasks as follows.

C. *Energy Source*

Harvesting source of energy is dependent on environmental factors. Such as solar, wind etc. They are highly varying with time. Where $P_D(s_i)$ is the power drain at speed level s_i .

IV. PROPOSED METHODOLOGY

Here our main goal of power management device is to minimize the energy consumption (or) to maximize the lifetime achieved while meeting the required performance constraints in a battery powered device.

In a proposed energy aware hard real time scheduling, idea is to save energy by slowing down the processor just enough to meet the deadline of a task and avoid energy overflow. In this work, we proposed a harvesting aware hard real-time scheduling algorithm which reduce the energy as well as timing over- head by utilizing speed in such a way that response time of task is less than or just equal to the existing approach even though on the cost of lesser energy consumption. Here the execution speed of a task is selected based on the stored energy as well as available energy through harvesting and deadline of a task.

A. *The proposed method*

Tasks are executed at minimum energy consuming state even though the systems have enough energy to complete the task at maximum speed.

Whenever any periodic task p_i arrives there are two possible cases

Case-1: When task requirement at maximum speed is less than available energy $E_C(t)$, then we will calculate slack time for task and if slack $(p_i) > 0$, reserve a time slot from latest start time of a task to deadline, and energy equals to the energy required by p_i to complete its execution at maximum speed.

- Compute the lowest feasible speed for a task subjected to:

EC(ai) + EH(ei(si)) – E consumption(pi(si))) ≤ C ....(5)

and

ei(si) ≤ di ..... (6)

- After that execute the task at assigned speed up to completion of task (or) slack time whichever is earlier

- If task has not finished before the latest start time of a task, then execute the remaining portion of task at full speed in reserve slot with reserve energy.

Case-2: If energy requirement of task at maximum speed is greater than total available energy EC(t), then again we will calculate lowest feasible speed for a task execute the task at assigned speed until its completion and arrival of some higher priority task whichever is earlier.

B. ***Algorithm 1*** *Harvesting Aware Hard Real-Time Schedule (HA-HRTS):*

Input: A set of n real time periodic taskspi =p1 ,p2 , ...τn and a DVS based processor that support different frequency (or) speed levels.

i. Initialize a ready queue Q.
ii. Initialize a reserve queue R.
ii. Enter tasks ready to execute in a ready queue Q.
iii. Sort the tasks in Q on EDF basis.
iv. While (Q=NIL) do
v. Process the taskp_i
vi. While ($E_C(t) > 0$ and $E_i(s_{max}) \leq E_C(t)$) do
vii. Calculate $s_{earliest}$ and s_{late} of p_i
viii. If $(s_{earliest} - s_{late}) > 0$, then
ix. Enter taskp_i in reserve queue R
x. Reserve time slot (s_{late}, d_i) forp_i
xi. Reserve energy forp_i , $E_{reserve}(p_i) = E_i(s_{max})$
Therefore, $E_{avail}(t) = E_C(t) - E_{reserve}(\tau_i)$
xii. Compute a lowest feasible speed(s_i) for p_i using Algorithm 2.
xiii. Execute() taskp_i with speed s_i until its completion

xiv. if any other taskp_j arrive at time $a_j < f_i(\tau_i)$ and $d_j < d_i$
xv. Update ready queue Q
xvi. p_j preempt p_i
xvii. Update $E_{avail}(t)$
$E_{avail}(t) = E_{avail}(t) - E_{consume}(\tau_i(a_j))$
xviii. If $(E_j(s_{max}) \leq E_{avail}(t))$
xix. Go to step 8., else go to step 27.
xx. end if
xxi. Else p_i continue its execution at (s_i)
xxii. If p_i completes before s_{late}, then
xxiii. Remove p_i from Q and R
xxiv. Frees the reserve energy and time slot for p_i
xxv. $E_{avail}(t) = E_{avail}(t) + E_{reserve}(\tau_i)$
xxvi. Else remaining portion of p_i executed at s_{max} and
xxvii. $E_{avail}(t) = E_{avail}(t) + (E_{reserve}(\tau_i) - E_{consume}(\tau_i)$
xxviii. $E_{avail}(t) = E_C(t)$

end if
end if
end if
end while
xxix. While $(E_C(t) > 0$ and $E_i(s_{max}) > E_C(t))$ do
xxx. Compute a lower feasible speed (s_i) forp_i using Algorithm 2.
xxxi. Execute() taskp_i with speed s_i until its completion
end while
xxxii. While $(E_C(t) \leq 0)$ do
xxxiii. Calculate common slack for all tasks ready to execute in Q at time t when $E_C(t)=0$
xxxiv. While $(E_C(t) \leq E_{max}$ and slack(t)$> 0)$
xxxv. Wait and recharge the battery
xxxvi. t = t + 1
if a_k= arrival time of taskp_k, then insertp_k in Q and update slack(t)
end if
end while
end while
end while

C. *Algorithm 2*

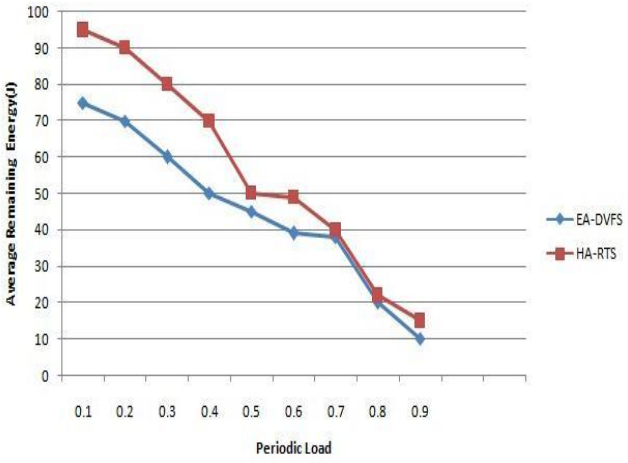
Speed Allocation algorithm:

i. Prepare a list of all possible different speed levels (or) frequency levels used in this case.
ii. Sorting of speed levels in increasing order.
iii. While $(s_i = s_n)$ do
iv. if $(E_C(a_i) + E_H(e_i(s_i)) - E_{consumption}(\tau_i(s_i))) \leq C$ and $e_i(s_i) \leq d_i$, then
v. $s_i = s_i$
vi.
lse $s_i = s_i + 1$
vii.
o to
step 4.
viii.
nd if
ix. end while

V. EXPERIMENTAL RESULTS

Here, we compare the performance of proposed approach Harvesting Aware hard Real-Time Scheduling (HA-HRTS) algorithm with existing approach Energy Aware Dynamic Voltage and Frequency Selection(EA-DVFS)[16]. The key parameters for performance measurement are remaining energy and tasks acceptance ratio. In the following section we measure the effect of variation in periodic tasks load on the average energy consumption and acceptance ratio of task set. The effect of load on the remaining energy consumption can be seen from the figure

Figure1, Average remaining energy

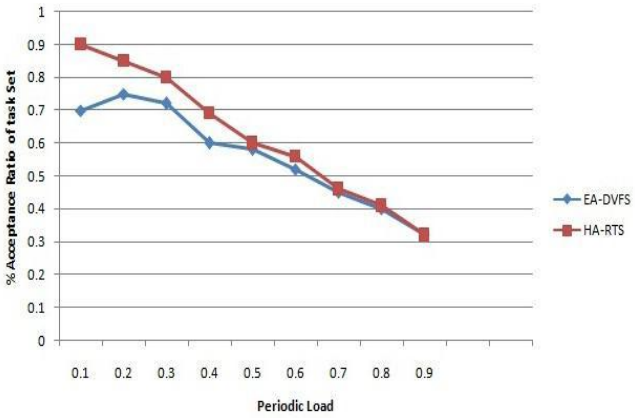


The storage capacity is to 2000J. We observe from the figure 1 when the periodic tasks load increases the remaining energy will decreases. When we varies periodic load from 10%to 90% we observe from the figure as load increase remaining energy of the system decreases. At lower periodic load (10% to 40%) our proposed approach have significant saving in energy almost store 20% more energy as compared to existing approach. This is due to, at lower periodic load our approach run the whole computation of task most of the time at slower speed and same speed level however, existing approach execute some portion at lower speed and remaining computation time at maximum speed level even there is no any higher priority tasks. Energy consumption of task at maximum speed level increases exponentially as compare to energy consumption at lower speed level. However, at higher periodic load (70% to 90%) our proposed approach has slightly saving in energy. Most of the time our proposed approach as well as existing approach, execute the task at maximum speed

level at higher periodic load. This is due to that at higher periodic load in both approaches processor rarely has chance to slowdown the speed of task.

Effect of load on acceptance ratio: The effect of load on the acceptance ratio of set of periodic tasks can be seen from the figure 2, compare the performance of existing approach and proposed approach. In this we set the storage capacity is to 2000J. We observe from the figure 2 when the tasks load increases the acceptance ratio will decreases. At lower periodic load (10% to 40%) our proposed approach have accept 10% more periodic tasks as compared to existing approach. This is due to that at lower tasks load our approach run the whole computation of task most of the time at slower speed and execute whole computation at same speed level however, existing approach execute some portion at lower speed and remaining computation time at maximum speed level even though there is no any other tasks. Thus, the future arriving task may miss their deadline due to the shortage of energy. However, at higher tasks load (70% to 90%) our proposed approach and existing approach both accept almost same number of periodic tasks. This is due to at higher tasks load both approaches execute task at maximum speed level most of the time.

Figure2. Percentage of acceptance ratio of task set



Reduction in rejection ratio: The effect of storage capacity on the rejection ratio of periodic tasks can be seen from the above figures. Our objective is concerning on the relative capacity savings achieved with our algorithms, we are especially interested in the smallest capacities C necessary to avoid any deadline violations due to the shortage of energy. We can observe from the figure when periodic load is 20%, 40%, 60% and 80% the proposed approach significantly reduces almost 50%, 30%, 20% and 10% on average respectively. We can also observe that the storage capacities saving are 48%, 40%, 10% and 5% at periodic load 20%, 40%, 60% and 80% respectively.

VI. CONCLUSION

Here, a general scheduling algorithm that maximizes the utility of harvested energy for hard real time embedded system with voltage scalable processor is presented. The proposed approach decides operating speed that reduce the energy overhead as well as timing overhead due to the speed switching.

The simulation studies shows that the proposed scheduling algorithm improves the overall aver- age remaining stored energy. The average remaining stored energy of the system is approximately 5% more than the existing approach when a load varied from 70% to 90% while 20% more energy will be stored at lower tasks load varied from 10% to 50%. When the tasks load is low say 10% to 50% our proposed approach accepts 8% more task than existing approach. However, at higher tasks load both approach Perform almost same. Thus, extensive simulation shows that our proposed approach is capable of performing better in terms of average stored remaining energy of the system as well as acceptance ratio of periodic tasks.

VII. FUTURE WORK

Present study solves, the problem of energy minimization for periodic load in a uniprocessor of hard real time system. So, we can extend this work for aperiodic and sporadic load along with periodic load in a uniprocessor system of hard real time.

REFERENCES

[1] Agrawal, S., Yadav, R. S. and Ranvijay, 2009. A Pre- emption Control Approach for Energy Aware Fault Tolerant Real Time System, International Journal of Recent Trends in Engineering, 381-386.

[2] Allavena, A. and Mosse, D., 2001. Scheduling of frame-based embedded systems with rechargeable batteries, Workshop Power Manage Real-time Embedded System.

[3] Bertogna, M. and Baruah, S., 2010. Limited Preemption EDF Scheduling of Sporadic Task Systems, IEEE Transactions on Industrial Informatics, 579 - 591.

[4] Dehghan and Maryam, 2010. Adaptive checkpoint placement in energy harvesting real-time systems , 18th Iranian Conference on Electrical Engineering (ICEE), 932 - 937.

[5] Hussein, E. L. G., Chetto, M. and Chehade, R.H., 2011. EH- EDF: An On-line Scheduler for Real-Time Energy Har- vesting Systems, 18th IEEE International Conference on Electronics Circuit and System (ICECS), 776-779.

[6] Hussein, E. L. G., Chetto, M. and Chehade, R.H., 2011. A real- time scheduling framework for embedded systems with environmental energy harvesting, Elsevier on Computers andElectrical Engineering 37.

[7] Liu, S., Lu, J., Wu, Q. and Qiu, Q., 2011. Harvesting-Aware Power Management for Real-Time Systems With Renewable Energy, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 1 - 14.

[8] Liu, S., Lu, J., Wu, Q. and Qiu, Q., 2010. Load-matching adaptive task scheduling for energy efficiency in energy har- vesting real-time embedded systems, ACM/IEEE International Symposium on Low-Power Electronics and Design (ISLPED),325 –330.

[9] Liu, S., Qiu, Q. and Wu, Q., 2008. Energy Aware Dynamic Voltage and Frequency Selection for Real-Time Systems with Energy Harvesting, conference on Design, Automation and Test in Europe, 236 - 241.

[10] Lu, J. and Qiu, Q. 2011. Scheduling and mapping of periodic tasks on multi-core embedded systems with energy harvesting, Journal, Computers and Electrical Engineering archive, 498 -510.

[11] Moser, C., Brunelli, D., Thiele, L. and Benini, L., 2007. Real- time scheduling for energy harvesting sensor nodes, Real-Time Syst ; 37(3), 233-260.

[12] Moser, C., Thiele, L., Brunelli, D. and Benini, L., 2007. Adaptive Power Management in Energy Harvesting Systems , Design, Automation and Test in Europe Conference and Exhibition, 1 - 6.

[13] Niu, L. and Quan, G. 2006. System-Wide Dynamic Power Man- agement for Portable Multimedia Devices, Eighth IEEE International Symposium on Multimedia, 97 -104 .

[14] Niu, L. and Quan, G., 2006. Energy minimization for real-time systems with (m,k)-guarantee, IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 717-729.

[15] Paul, A. A. and Pillai, A.S. B., 2011. Reducing the Number of Con- text Switches in Real Time Systems, International Conference on Process Automation, Control and Computing (PACC), 1 - 6.
[16] Qadi, A., Goddard, S. and Farritor, S., 2003. A dynamic voltage scaling algorithm for sporadic tasks , 24th IEEE Conference on Real-Time Systems Symposium (RTSS), 52 -62.
[17] Zhu, L., Tongquan, Wei, T., Yonghe, Guo., Xiaodao, Chen. and Shiyan, Hu., 2010. Energy efficient fault-tolerance task allocation scheme for real-time energy harvesting systems, International Conference on Intelligent Control and Information Processing (ICICIP), 589 - 594.
[18] G. Bernat and A. Burns, "Combining (n;m)-hard deadlines and dual priority scheduling," in Proc. RTSS, Dec. 1997.
[19] W. Kim, J. Kim, and S. L. Min, "A dynamic voltage scaling algorithm for dynamic-priority hard real-time systems using slack analysis," in Proc. DATE, pp. 788, 2002
[20] H. Aydin, R. Melhem, D. Moss´e and P. Mejia-Alvarez, "Dynamic and Aggressive Power-Aware Scheduling Techniques for Real-Time Systems", in RTSS, 2001.