VOL. 2, NO. 4, AUGUST 2007 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

A SIMULATION SOFTWARE DEVELOPMENT FOR PERFORMANCE ANALYSIS OF DVS ALGORITHM FOR LOW POWER EMBEDDED SYSTEM

A. Chilambuchelvan[1], S. Saravanan[2] and J. Raja Paul Perinbam[3]
[1]Department of ECE, IRTT, Erode, Tamilnadu, India
[2]Department of EEE, JCE, Chennai, Tamilnadu, India
[3]Department of ECE, CEG, Anna University, Tamilnadu, India
E-mail: chilambuchelvan_a@yahoo.com

ABSTRACT

Energy efficiency is an important property of mobile and pervasive computing devices. Dynamic voltage scaling (DVS) is an energy saving technique, achieve this property by reducing energy dissipation of the core by lowering the supply voltage and operating frequency. In this paper a simulation environment for testing of different DVS algorithms under EDF and RM have been discussed- as there are no standard simulators readily available in market. The simulator environment provides a framework for objective performance evaluations of different DVS algorithms. Several key DVS algorithms recently proposed for hard real-time periodic task sets have been compared; analyzed for their energy efficiency and discussed the performance differences quantitatively. It is shown through simulations that the real time DVS algorithms closely approach the theoretical lower bound on energy consumption and can easily reduce energy consumption (15-20%) in an embedded real-time system.

Keywords: RT-DVS, EDF, RM, idle factor, utilization, energy consumption.

1. INTRODUCTION

Dynamic voltage scaling (DVS) is an emerging technology to reduce the power consumption of handheld and mobile devices. The energy usage of computer systems is becoming more important, especially for battery operated portable devices. The displays, disks and CPUs are the most power consuming components of a computer system. The power management of most of the systems handles the display and disk power with ease. They get to save power by switching on and off the power of the display and disks at idle time. The power of the system is based on CPU's speed of execution. And the only way to reduce the power consumed by the CPU is to reduce the speed. With the hunger for more speed and performance, the reduction of CPU speed for saving power becomes unacceptable. There has to be a system that would reduce the CPU power at times when there is no CPU usage and will clock at full speed when there is heavy processor load. Answer to this requirement is the Dynamic voltage scaling. Dynamic Voltage Scaling is the technique of dynamically reducing the processor speed and thereby reducing the voltage and power consumption. Reducing energy consumption is important in portable computers due to their limited battery capacity. An energy saving technology that has recently begun appearing in modern portable computers are dynamic voltage scaling, the ability to change processor's voltage without rebooting. DVS tries to tradeoff between performance and battery life by taking into account the important characteristics of most current computer systems: (1) the peak computing rate needed is much higher than the average throughput that must be sustained; and (2) the processors are based on CMOS logic. Since the energy dissipated per cycle with CMOS circuitry scales quadratically to the supply voltage ($E \propto V^2$), DVS can potentially provide a very large net energy savings through frequency and voltage scaling [1, 2].

In this paper, it is described DVS simulator, an integrated simulation environment for DVS algorithms, which can be used in comparing the energy efficiency of various DVS algorithms. Real-Time scheduler integration and basic RT-DVS algorithms are explained briefly in section 2. The simulation environments are discussed in section 3. Simulation studies and result analysis are shown in section 4. Conclusions and future directions are given in last section.

2. REAL-TIME DVS ALGORITHM

In order to realize the reduced energy-consumption benefits of DVS in a real-time embedded system, a new DVS algorithm that is tightly coupled with the actual real-time scheduler of the operating system is needed. In 1973, Liu [3] presented the rate monotonic (RM) algorithm as an optimal fixed priority-scheduling algorithm, and the earliest-deadline-first (EDF) algorithm as optimal dynamic priority scheduling algorithm. RM is a static priority scheduler, and assigns task priority according to period. EDF is a dynamic priority scheduler that sorts tasks by deadlines.

In a classical model of real time system there is a set of 'n' independent tasks that need to be executed periodically. Each task (Ti) has an associated period (Pi), deadline (Di) and a worst-case computation time (Ci). Let U be the total utilization of this task set. A sufficient condition [4, 5] for feasible scheduling of the task set with real time scheduler is

VOL. 2, NO. 4, AUGUST 2007 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

$$U = \sum_{i=1}^{n} Ci/\min(Di,Pi) \le n(2^{\frac{1}{n}} - 1) \qquad \text{for RM}$$

$$U = \sum_{i=1}^{n} Ci/\min(Di,Pi) \le 1 \qquad \text{for EDF}$$

Research into DVS algorithms can be classified into two categories: (1) Algorithms that attempt to estimate the future utilization of the processor based on the past information and (2) Algorithms that use task deadlines to guide performance setting decisions. The disadvantages of the former are its unresponsiveness to dynamically changing workloads, but are simpler in their implementation. For hard real-time systems, there are two types of voltage scheduling approaches [6, 7] depending on the voltage scaling granularity: intra-task DVS (Intra DVS) and inter-task DVS (Inter-DVS). The intra-task DVS algorithms adjust the voltage within an individual task boundary, while the inter-task DVS algorithms determine the voltage on a task-by-task basis at each scheduling point. The main difference between the two approaches is whether the slack times [8] are used for the current task or for the tasks that follow. Inter DVS algorithms distribute the slack times from the current task to the following tasks, while Intra DVS algorithms use the slack times from the current task for the current task itself.
Three heuristics DVS algorithms were brought up by Pillai and Shin [9] working with RM or EDF task scheduling. The first one, static scheduling, selects only one lowest possible operating frequency to let all tasks meet all the deadlines. The second one, cycle-conserving scheduling, determines the lowest frequency for each schedule task satisfying the acceptance test. In the acceptance test, the bound of the total utilization is decreased to the optimum speed of the system. Then, the system updates the actual utilization, according to the full speed, that the previous task used in order to calculate the next task speed. The last heuristic, look ahead scheme tries to defer as much work as possible, and sets the operating frequency to meet the minimum work that must be done now to ensure that all future deadlines are met. The simulation results are shown that the look-ahead scheduling is the best among three heuristics in almost all cases.

3. DVS SIMULATOR

A simulator is developed for the operation of hardware capable of voltage and frequency scaling with real-time EDF/RM scheduling. It takes periodic real time tasks as an input which is scheduled under EDF/RM scheduling policy. It supports all the inter DVS algorithms and it can be used to compare the energy efficiency of different inter DVS algorithms using the same task set specification under the same machine configuration (Figure-1). Using this evaluation, one can decide the best DVS algorithm for the given application on the given hardware platform. It can be used as well when evaluating a given DVS algorithm under various evaluation conditions.

The following subsection describes our simulator and the assumptions made in its design. Later, it is shown some simulation results and provides insight into the most significant system parameters affecting RT-DVS energy savings.

3.1. Simulator

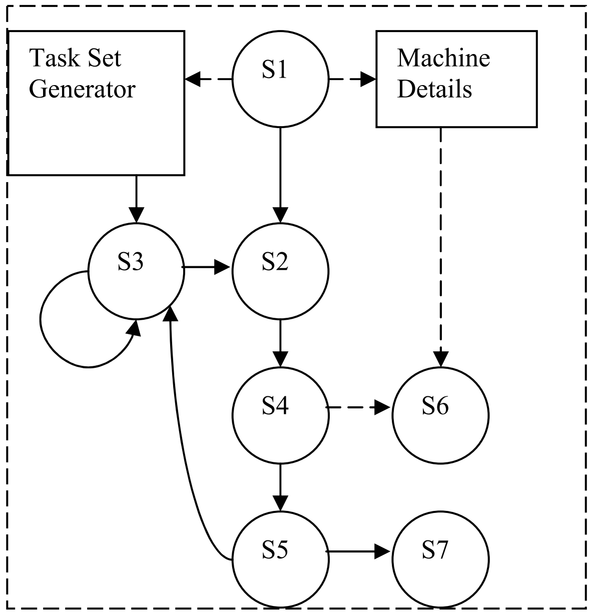
It requires three inputs; a task set specification, duration of simulation and a machine specification. The task set specification describes various task set characteristics that affect the energy efficiency of a DVS algorithm while the machine specification describes the machine characteristics that affect the energy efficiency of a DVS algorithm. The Inter DVS Module is responsible for the whole operation of simulator.

3.2. Inter DVS module

The Inter DVS Module is responsible for scheduling tasks and plays a role of a real-time scheduler in a hard real-time system. The priority-based scheduling can be implemented by maintaining two queues [10], one called run queue and the other called delay queue. The run queue holds tasks that are waiting to run and the tasks in the queue are ordered by priority (state 3). The task that is running on the processor is called the active task (state 2). The delay queue holds tasks that have already run in their periods and are waiting for their next periods to start again. They are ordered by the time at which their release is due. When the scheduler is invoked, it searches the delay queue to see if any tasks should be moved to the run queue. If some of the tasks in the delay queue are moved to the run queue, the scheduler compares the active task to the task at the head of the run queue.

The Inter DVS Module consists of three sub modules one for estimating available slack times, one for execution of task and the another one for estimating energy consumed by a task in a particular invocation. The slack estimation is done by the slack estimation module, which computes the total available time for the scheduled task while the task execution module (state 4), which determines the operating speed for the scheduled task and simulates the execution of the task, does the slack distribution. The energy estimation module (state 6) takes the timing and speed information from the task execution module and computes the energy consumption of the current task execution using the current machine specification. Energy consumption is calculated using $E \propto$ Exe time * $f_i * V_{DD}^2$ where V_{DD} is the supply voltage, f_i is the operating frequency and Exe time is execution time of task. The state machine diagram for real time DVS scheduling of the simulator is shown in Figure-1.

VOL. 2, NO. 4, AUGUST 2007 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences



S1: Initializing scheduler; S2: Task selection; S3: Task set update; S4: Task execution; S5: End time check; S6: Energy estimation state; S7: Halt state.

Figure-1. State machine diagram for RT – Scheduling.

4. SIMULATION AND RESULT ANALYSIS

Using C++ and VB we have developed a simulator for the operation of hardware capable of voltage and frequency scaling with real-time EDF/RM scheduling. The following subsection describes the assumptions made in its design. Later, we show some simulation results and provide insight into the most significant system parameters affecting RT-DVS energy savings.

4.1. Assumptions

I. The processor considered has discrete and finite frequencies (f1<f2…. <fm).
II. The switching between frequency levels may occur anywhere within the task and the switching overheads are negligible in comparison with the task deadlines.
III. The scheduler follows EDF/RM Scheme.
IV. For simplicity, only task execution and idle (halt) cycles are considered.

4.2. Simulation results and discussions

It is assumed that a task set of 3 tasks [Table-1] is to be run on a DVS simulated machine[11,12] that provides three relative operating frequencies (0.5, 0.75 and 1.0) and corresponding voltages (3, 4 and 5) respectively. RT-DVS algorithms were simulated to determine the most important system parameters that affect energy consumption. Since many factors affect the energy efficiency of DVS algorithms the comparative study cannot answer all the DVS performance questions. In this paper, the performance of RT-DVS algorithms is compared for various parameters like idle factors, and worst case execution times using the simulator. A theoretical lower bound was included for energy dissipation. This lower bound reflects execution throughputs only, and does not consider any timing issues. No real algorithm can do better than this theoretical lower bound. From simulation graphs, it is interesting to note that look ahead algorithm is closer to this theoretical lower bound.

Table-1. Example specifications.

Ti	Ci (WCET)	Pi	Invocation 1	Invocation 2
1	3 ms	8 ms	2 ms	1 ms
2	3 ms	10 ms	1 ms	1 ms
3	1 ms	14 ms	1 ms	1 ms

Varying idle level

To evaluate the impact of halt feature of the processor on energy efficiency of the RT-DVS algorithm, different idle specifications were tested. Idle factor is defined as the ratio of energy consumed in a cycle while the processor is halted to the energy consumed in a normal execution cycle. To see how an imperfect halt feature affects power consumption, several simulations of RT-DVS algorithm were performed varying the idle level factors between 0.01 and 1.0 as shown in Figure-2. When idle level is 0 the impact of dynamic voltage scaling algorithm is relatively marginal compared to static algorithm. But when idle level increases to 1 (same energy consumption as in normal operation) the percentage saving with voltage scaling improves because dynamic voltage scaling algorithms switch to the lowest frequency and voltage while the static one does not. From simulated results it is observed that the performance of the look ahead EDF algorithm is relatively better than other algorithms because idle time is utilized efficiently.

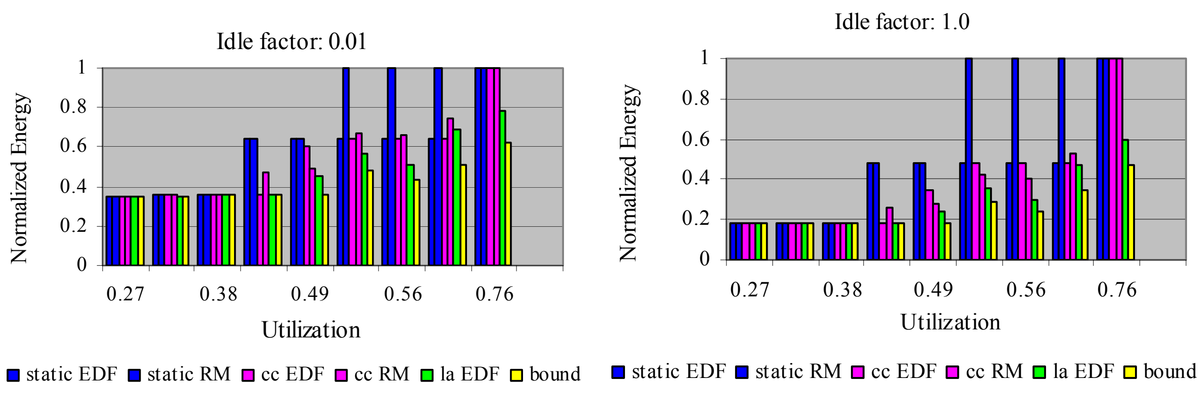


Figure-2. Normalized energy consumption of simulated machine with utilization factor 0.8 and idle level factors 0.01 and 1.0

Varying computation time

To evaluate the impact of worst case processor utilization (WCPU) of task set on the energy efficiency of the RT-DVS algorithm, different WCPU specifications were tested. The real-time task sets are specified by their period and worst-case computation time and actual computation. These tasks are generated by randomly choosing parameters assuming uniform distribution. When the WCPU of a given task set is less than 1.0, the tasks have inherent static slack times. Figure 3 shows simulation results for tasks that require a constant 90% and 70% of their worst-case execution cycles (WCEC) for each invocation. The results indicate that when the task set utilization is low, the look ahead EDF algorithms consume the same amount of energy, because task set with low utilizations usually have enough slack and idle slots, so that task set can be operated at the lowest speed level and it is closer to the theoretical lower bound.

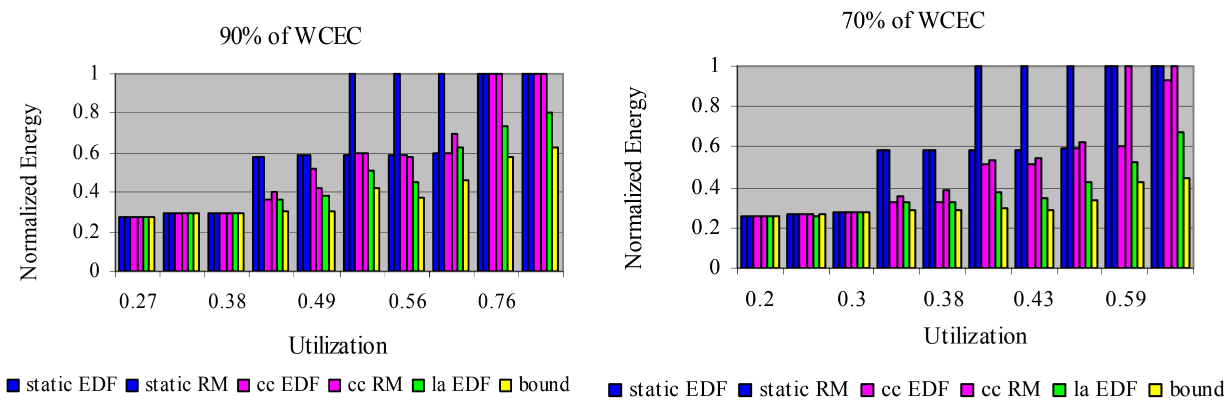


Figure-3. Normalized energy consumption of simulated machine with idle factor 0.2 and worst-case execution cycles (WCEC) of 0.90 and 0.70

CONCLUSIONS

In this paper it is discussed about a simulation environment to test the performance of various DVS algorithms under EDF and RM. The simulator has been designed such a way that any new DVS algorithm can be evaluated easily by adding new task execution module to it. It also supports to add new machines specifications to in it. It is shown that the most significant parameters affecting energy conservation through RT-DVS mechanisms and the extent to which CPU power dissipation can be reduced. Furthermore, look-ahead and cycle-conserving RT-DVS mechanisms can achieve close to the theoretical lower bound on energy. From the simulated results it is indicated that 15% to 20% energy savings can be achieved, even including irreducible system energy overheads and using task sets with high values for both worst-case and average-case utilizations.

REFERENCES

[1] Burd, T. D., and Broderen, R. W. 1995. Energy efficient CMOS microprocessor design. Proceedings of the 28th Annual Hawaii International Conference on System Sciences. Volume 1: Architecture (Los Alamitos, CA, USA, January), T. N. Mudge and B. D. Shriver, Eds., IEEE Computer Society Press. pp. 288-297.

[2] T. Pering, T. Burd, and R. Brodersen. 1998. Dynamic voltage scaling and the design of a low-power microprocessor system. Power Driven Micro architecture Workshop, attached to ISCA98.

[3] C.L. Liu and J.W. Layland. 1973. Scheduling Algorithms for Multiprogramming in a hard real time environment. Journal of the Association for Computing Machinery. Vol. 20(1): 44-61.

[4] Liu, J.W.-S. 2000. Real-time systems. Prentice Hall.

[5] Hong, M. Potkonjak and M.B. Srivastava. 1998. On-line Scheduling of Hard Real-time Tasks on Variable Voltage Processor. Proceedings of the IEEE/ACM International Conference on Computer-Aided Design. pp. 653-656.

[6] H. Aydin, R. Melhem, D. Mosse and P. M. Alvarez. 2001. Dynamic and Aggressive Scheduling Techniques for Power- Aware Real-Time Systems. Proceedings of IEEE Real- Time Systems Symposium. December.

[7] W. Kim, J. Kim and S. L. Min. 2002. A Dynamic Voltage Scaling Algorithm for Dynamic-Priority Hard Real-Time Systems Using Slack Time Analysis. Proceedings of Design, Automation and Test in Europe (DATE’02). pp. 788-794. March.

[8] J. Lehoczky and S. Thuel. 2000. Algorithm for scheduling hard aperiodic tasks in fixed priority systems using slack time stealing. Proc. of the IEEE Real-Time Systems Sympopsium.

[9] P. Pillai and K.G. Shin. 2001. Real-Time Dynamic Voltage Scaling for Low-Power Embedded Operating Systems. ACM symposium on operating system principles.

[10] Y. Shin, K. Choi and T. Sakurai. 2000. Power Optimization of Real-Time Embedded Systems on Variable Speed Processors. Proceedings of the International Conference on Computer-Aided Design. pp. 365-368. November.

[11] Advanced Micro Devices Corp. 2000. Mobile AMD-K6+ Processor Data Sheet. Publication # 23446.

[12] Intel Corporation. 2000. Mobile Intel Pentium-III Processor in BGA2 and MicroPGA2 Packages.