On-Line Scheduling of Hard Real-Time Tasks on Variable Voltage Processor

Inki Hong†, Miodrag Potkonjak†, and Mani B. Srivastava‡
†Computer Science Department, University of California, Los Angeles, CA 90095-1596 USA
‡Electrical Engineering Department, University of California, Los Angeles, CA 90095-1596 USA

Abstract

We consider the problem of scheduling the mixed workload of both sporadic (on-line) and periodic (off-line) tasks on variable voltage processor to optimize power consumption while ensuring that all periodic tasks meet their deadlines and to accept as many sporadic tasks, which can be guaranteed to meet their deadlines, as possible. The proposed efficient algorithms result in the scheduling solutions, which are very close to the minimum bound achievable with the dynamically variable voltage approach. The effectiveness of the proposed algorithms is shown on extensive experiments with real-life design examples.

1 Introduction

The growing class of portable systems, such as personal computing and communication devices, demands data- and computation-intensive functionalities with low power consumption. In this paper, we target synthesis of application-specific system-on-chip with variable voltage processor core, mainly focusing on processor power optimization. The distribution of power dissipation by the components of application-specific system-on-chip depends on the actual applications running on the system. However, extensive studies indicate that the power consumption of the processor accounts for significant portion of the overall power consumption [1].

The most effective way to reduce power consumption of a processor core in CMOS technology is to lower the supply voltage level, which exploits the quadratic dependence of power on voltage [2]. Reducing the supply voltage however increases circuit delay and decreases clock speed. The resulting processor core consumes lower average power while meeting the deadlines. Unfortunately, this technique is ineffective when tight deadlines are present in systems. Another power optimization technique for processor cores is the predictive system shutdown [7, 14]. The predictive system shutdown technique, though usable even in the presence of tight deadlines, is inferior to the supply voltage reduction technique for the cases when both techniques can be applied. The limitations of the techniques arise due to the fact that systems are designed with a fixed supply voltage. The supply voltage reduction technique attempts to find a single optimal voltage level for the entire processor operation, while the predictive system shutdown technique makes a binary runtime decision whether to turn on or off the power supply.

Recent advances in power supply technology along with custom and commercial CMOS chips that are capable of operating reliably over a range of supply voltages make it possible to create processor cores with supply voltage that can be varied at run time according to application timing constraints [12, 15]. The variable voltage processor core can operate at different optimal points along its power vs. speed curve in order to achieve much higher energy efficiency than existing techniques for a wider class of applications.

A hard real-time system typically has a mixture of off-line and on-line workloads. The off-line requests support the normal functions of the system while the on-line requests are sporadic tasks to handle external events such as operator commands and recovery actions, which are usually unpredictable. This unpredictable nature of the external events makes on-line scheduling difficult. Supply voltage reduction technique is inapplicable because we do not have the complete knowledge of the sporadic tasks, when the decision for the voltage level is made. The predictive system shutdown technique can be applied, but the difficulty of estimating the future idle times without the complete knowledge of the on-line tasks [7, 14] renders the technique ineffective. In this paper, we consider the problem of preemptively scheduling this mixed workload on variable voltage processor to optimize power consumption while ensuring that all off-line requests meet their deadlines and to accept as many sporadic requests, which can be guaranteed to meet their deadlines, as possible.

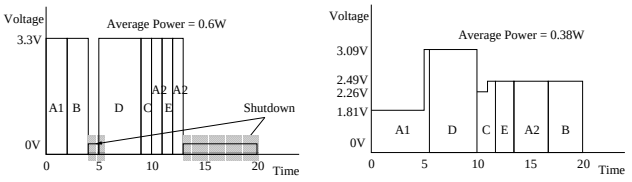
To illustrate the key point of the proposed dynamically variable voltage approach, we consider a set of tasks, shown in Table 1, as a motivational example. Five independent computations T_A, T_B, T_C, T_D and T_E need to be executed on an embedded processor core. The tasks T_A and T_B are periodic while the other three tasks are sporadic. The deadlines for the periodic tasks T_A and T_B are relative to their periods. Each task can be executed immediately after its arrival and must be finished by its deadline.

task	arrival	deadline	period	execution time at 3.3 V
A	0	10	10	2
B	0	20	20	2
C	5	15	-	1
D	5	10	-	4
E	11	18	-	1

Table 1: The characteristics of the 5 tasks used to illustrate the motivation for dynamically variable voltage approach.

Assume the maximum supply voltage to be $(V_{dd})_{ref} = 3.3$ volts. Power is normalized to its value at the reference point, i.e., $P(3.3\ volts) = 1$ Watt. Reducing supply voltage results in increased circuit delay and to a good accuracy, the circuit delay is given by $k \times \frac{V_{dd}}{(V_{dd}-V_t)^2}$, where V_t is the threshold voltage, and k is a constant [2]. We assume a typical value of $0.8\ volts$ for the threshold voltage. The system consumes 1 $Watt$ when no power reduction techniques are applied. We now consider the application of the predictive system shutdown technique and dynamically variable voltage approach. When applying the predictive system shutdown technique, we assume the complete knowledge of the idle periods, which will result in best improvement for the technique. The Earliest-Deadline-First (EDF) scheduling algorithm is used for both techniques since EDF is the optimal algorithm for the preemptive dynamic priority scheduling problem [11].

Figure 1 shows the scheduling solutions for the predictive system shutdown technique and dynamically variable voltage approach.



(a) Predictive system shutdown technique

(b) Dynamically variable voltage approach

Figure 1: Scheduling solutions for the predictive system shutdown technique and dynamically variable voltage approach

With the predictive system shutdown technique, the system will operate at $V_{dd} = 3.3\ volts$. All the tasks are executed in the intervals [0, 4] and [5, 13]. The processor can be shut down for the intervals [4,5] and [13, 20] and then be resumed for the next periodic tasks. The duty cycle of the processor is 60 %, so the average power consumption is $0.6\ Watts$.

With the variable voltage hardware, one can schedule the tasks such that the power consumption is minimized under the assumption of no knowledge of the arrival times of the sporadic tasks. The average power consumption is now $0.38\ Watts$, which is 37 % lower than that of the predictive system shutdown technique.

With the complete knowledge of the arrivals of sporadic tasks, the optimal strategy is to set the supply voltage to the single minimum voltage all the time such that all the tasks just meet their deadline. In this example, for the time interval [0,20], the optimal voltage is $2.48\ volts$, correponding to the 60 % of the highest processor speed, which results in average power consumption of $0.34 Watts$. This is the lowest bound we can achieve without assuming any knowledge of the arrivals of sporadic tasks.

The rest of the paper is organized in the following way. Section 2 presents the related work. Section 3 provides the background material on variable voltage systems, hard real-time scheduling and our experimental platform. In Sections 4 and 5, several scheduling algorithms are proposed for the various scheduling problems and experimental data are presented to evaluate the effectiveness of the proposed algorithms. The paper is concluded in Section 6.

2 Related Work

We review the research results relevant to low power systems based on dynamically variable voltage hardware. At the technology level, several researchers [12, 15] have developed efficient DC-DC converters that allow the output voltage to be rapidly changed under external control. At the hardware design level, research work has been performed on chips with dynamically variable supply voltage that can be adjusted based on process and temperature variations, and processing load [5]. A number of behavioral synthesis research groups have addressed the use of multiple (in their software implementation restricted to two or three) different voltages [3, 8, 10, 13].

Scheduling strategies for adjusting CPU speed to reduce power consumption have been proposed mostly in the context of non-real-time workstation-like environment [4]. Yao, Demers and Shenker [17] have provided an optimal preemptive off-line real-time scheduling algorithm for a set of independent tasks with arbitrary arrival times and deadlines on a variable speed processor. Several researchers have also addressed the issue of power in event-driven systems, and proposed various techniques for shutting down the system or parts of the system [7, 14].

Hong et al. [6] describes a design methodology for the real-time system-on-chip based on dynamically variable voltage processor core and provides an off-line scheduling heuristic for non-preemptive hard real-time tasks in addition to the selection of the processor core and the determination of the instruction and data cache size and configuration. In this paper, we develop several on-line preemptive scheduling algorithms for mixed workload of on-line and off-line tasks on variable voltage processor to optimize power consumption while ensuring that all off-line requests meet their deadlines and to accept as many on-line requests, which can be guaranteed to meet their deadlines, as possible.

3 Preliminaries

The variable voltage is generated by the DC-DC switching regulators in the power supply. [12, 15] reported efficient DC-DC switching regulators with fast transition times. The clock frequency also takes time to stabilize at the new value. The time overhead associated with voltage switching is on the order of 10 cycles in a microprocessor [12, 15]. This overhead is negligible since the computation itself can continue during the voltage and frequency transition.

There are two types of hard real-time tasks: periodic and sporadic tasks. Each periodic task $T_i(C_i, D_i, P_i)$ is characterized by its worst-case computation time C_i at the reference (highest) voltage, hard deadline D_i, and period P_i. The ready times of the tasks occur periodically with period P_i, the period of the task T_i. In most cases, the deadlines are assumed to be equal to the periods. Each sporadic task $S_i(A_i, C_i, D_i)$ is characterized by its arrival time A_i, worst-case computation time C_i at the reference voltage, and hard deadline D_i. The parameters of a sporadic task become known when the task arrives. On-line scheduling algorithms focus on dynamically performing feasibility checks. When a sporadic task arrives, an acceptance test is performed to check if the task can be scheduled to meet its deadline and all the tasks already scheduled are still guaranteed to meet their deadlines. We assume that sporadic tasks which cannot be feasibly scheduled at the arrival times of the tasks are not accepted by the scheduler.

We assume that the configuration of the system-on-chip is given. For experiments, we use the system configurations provided by [6]. For periodic tasks, we use the various mixes of the nine public domain benchmarks, part of the MediaBench [9]. For sporadic tasks, we use 10 small custom programs. We use the technique described in [6] for computing the worst-case execution times of the tasks on the system-on-chip.

4 Variable Voltage Scheduling for Sporadic Tasks

In this Section, we consider a special case, where only sporadic (on-line) tasks arrive to the system.

4.1 Acceptance Test

The ready time of each sporadic task is the instant of arrival of the task. We let $S = \{S_i(A_i, C_i, D_i) : 1 \leq i \leq m\}$ denote the stream of the tasks that have been accepted but uncompleted at the time t when an acceptance test is to be carried out. We present an acceptance test algorithm whose running time is $O(m)$. The tasks in S are indexed in ascending order of their deadlines. They are stored in a priority queue. Let $RC(S_i, t)$ denote the unfinished portion of the worst-case execution time at the highest speed of the task S_i at the time t. We keep track of the maximum utilization factor of the processor, $U(t)$, at time t which is defined as $U_j(t) = \frac{\sum_{i=1}^{j} RC(S_i,t)}{D_j - t}$ and $U(t) = max_{\forall j=1,\cdots,m} U_j(t)$.

If $U(t) > 1$, the set S can not be feasibly scheduled even at the highest speed using the EDF scheduling algorithm. When S_{new} arrives, we update $RC(S_1, t)$ for the currently running task S_1 and compute a new $U(t)$ including the new task so that if $U(t) > 1$, S_{new} is not accepted. It is easy to see that it takes $O(m)$ to compute $U(t)$ in the worst case. Note that we need to recompute $U_j(t)$ for the task S_j such that $D_j \geq D_{new}$

4.2 Variable Voltage Scheduling

The scheduler maintains a priority task queue in which tasks are ordered on the EDF basis. In the beginning, there is no task in

the task queue. A scheduling decision is made whenever any of the following two events occurs: (i) Event_1: a new sporadic task arrives and is accepted to the system by the acceptance test and (ii) Event_2: the current task completes its execution. When an Event_1 occurs, the scheduler updates the optimal voltage schedule of the processor for all the tasks including the new one in the task queue. The variable voltage scheduling algorithm is shown in Figure 2. When an Event_2 occurs, the completed task is removed from the task queue and we execute the task at the head of the task queue following the current voltage schedule.

Before we present an optimal variable voltage scheduling algorithm, we present some theoretical results on the shapes of optimal solutions. The algorithm is optimal when we have no knowledge of the arrivals of sporadic tasks. Lemma 1 shows that the optimal voltage value of a task with earlier deadline should be larger than or equal to that of the other task with later deadline. All proofs have been omitted due to space limitation.

STS Algorithm
Input: Current time t and the sorted set S of the m tasks in the system including the new task.
1. **Repeat** $\{$
2. Compute $U_j(t) = \frac{\sum_{i=1}^{j} RC(S_i,t)}{D_j - t}$ for all tasks in the set S;
3. Compute $U(t)$ and find k such that $U(t) = U_k(t)$, where k is the maximum value if there are ties;
4. Schedule only the tasks in $[t, D_k]$ at the voltage corresponding to processor speed $U(t)$ and remove the tasks from the set S;
5. Let $t = D_k$;
6. $\}$ **Until** (the set S is empty);

Figure 2: Pseudo code of the on-line scheduling algorithm called **STS** Algorithm for sporadic tasks on variable voltage processor

Lemma 1. Given the m sporadic tasks $S_i = (A_i, C_i, D_i)$ with $A_i = A$, $i = 1, \cdots, m$, sorted in the increasing order of their deadlines, the power optimal voltages V_i, $i = 1, \cdots, m$, for the m tasks are nonincreasing, i.e., $V_i \geq V_{i+1}, \forall i = 1, \ldots, m-1$, where A is constant.

Lemma 1 immediately implies that the following theorem holds.
Theorem 1. The optimal voltage schedule is in a "downstair" shape and at the edge of each "stair", a task completes at its deadline.

Theorem 1 leads to the optimal algorithm called **STS** (Sporadic Task Scheduling) Algorithm whose running time is $O(m)$. Note that the algorithm always guarantees all the accepted tasks by the acceptance test to be scheduled to meet their deadlines.

4.3 Experimental Results

To quantify the benefits of the proposed approach over the predictive system shutdown approach, we performed a number of simulations to compare the power consumption under the two strategies. We compared the **STS** algorithm with the predictive system shutdown technique and the **MIN** algorithm, which gives us the minimum bound for dynamically variable voltage approach. The predictive system shutdown technique assumes the complete knowledge of the idle periods while the **MIN** algorithm assumes the complete knowledge of the arrivals of sporadic tasks.

We varied the average processor utilization from the light workload to heavy workload. The idealized predictive system shutdown technique achieves exactly half of the original power consumption. The **MIN** algorithm results in a single voltage solution corresponding to the average minimum required processor speed to meet the deadlines of the tasks. We observe that the power consumption of the solutions of **STS** algorithm is close to that of the solutions of **MIN** algorithm.

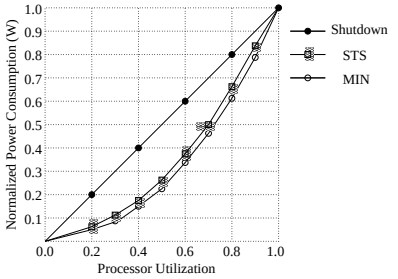


Figure 3: Dynamically variable voltage approach *vs.* Predictive system shutdown technique: Simulation results

5 Variable Voltage Scheduling for Periodic and Sporadic Tasks

In this Section, we consider a case, where periodic (off-line) tasks as well as sporadic tasks arrive to the system.

5.1 Acceptance Test

When a new sporadic task S_{new} arrives, the task is accepted if the scheduler can schedule the request to meet its deadline without causing any offline task or previously accepted sporadic request to miss its deadline using the highest speed of the processor. This problem has been studied by many researchers in real-time scheduling research community. Tia et al. [16] proposed an acceptance test whose time complexity is $O(n+m)$. The efficiency of the test was attained by carefully exploiting the properties of the EDF scheduling and periodic tasks, as well as by maintaining simple and easy to update data structures. This algorithm assumes that sporadic tasks span no more than one hyperperiod of the periodic tasks. We use their technique for acceptance test.

5.2 Variable Voltage Scheduling

We present two algorithms for this problem. The two algorithms are based on the similar ideas used for **STS** algorithm. The first algorithm called **OPASTS** (Optimal Periodic And Sporadic Task Scheduling) Algorithm is optimal when we have no knowledge of the arrivals of sporadic tasks. **OPASTS** Algorithm has time complexity of $O(N + m)$, where N is the total number of requests in each hyperperiod of the n periodic tasks in the system and m is the number of sporadic tasks that have been accepted but uncompleted at the time when a scheduling decision is made. Since N depends on the periods of the tasks, this time complexity is pseudo-polynomial. Since on-line scheduling decisions have to be made efficiently, we propose the second algorithm called **HPASTS** (Heuristic Periodic And Sporadic Task Scheduling) Algorithm whose time complexity is $O(m)$. **HPASTS** Algorithm is not optimal, but efficient and effective.

We define the utilization factor $U_{max}(T)$ to be the fraction of processor time to be spent in the execution of the task set T of n periodic tasks. It is easy to see that $U_{max}(T) = \sum_{i=1}^{n} \frac{C_i}{P_i}$ for the n periodic tasks in the set T. When there are no sporadic tasks accepted to the system, the processor speed should be set to $U_{max}(T)$.

5.2.1 Optimal Algorithm

We define the set $R(t)$ to be the set of l periodic tasks in each hyperperiod, which has not been completed at the time t. We use S_i and R_i to refer to an individual sporadic task and periodic task, respectively. We define the set W to be the union of the sets S and R. The tasks in the set $W = \{W_i : 1 \leq i \leq l+m\}$ are indexed in the ascending order of their deadlines. We assume that sporadic tasks span no more than one hyperperiod of the periodic tasks. The extension of the proposed algorithm should be straightforward, when

this assumption does not hold. When all the periodic tasks in W are completed, all N periodic tasks in another hyperperiod are inserted in W. The set W can have at most $N + m$ tasks at all times.

Lemma 2. Given the m sporadic tasks, and l periodic tasks in the hyperperiod which have not been completed, $W_i = (A_i, C_i, D_i)$, $i = 1, \cdots, l + m$, sorted in the increasing order of their deadlines, the power optimal voltages V_i, $i = 1, \cdots, m$, for the $l + m$ tasks are nonincreasing, i.e., $V_i \geq V_{i+1}, \forall i = 1, \ldots, l + m - 1$.

Lemma 2 immediately implies that the following theorem holds.

Theorem 2. The optimal voltage schedule is in a "downstair" shape and at the edge of each "stair", a task completes at its deadline.

Theorem 2 leads to the optimal algorithm called **OPASTS** (Optimal Periodic And Sporadic Task Scheduling) Algorithm whose running time is $O(N + m)$.

5.2.2 Heuristic Algorithm

HPASTS Algorithm is based on **STS** Algorithm for only sporadic tasks. The only difference exists when computing $U_j(t)$ such that for **HPASTS** Algorithm, $U_j(t) = U_{max}(T) + \frac{\sum_{i=1}^{j} RC(S_i,t)}{D_j - t}$ while for **STS** Algorithm, $U_j(t) = \frac{\sum_{i=1}^{j} RC(S_i,t)}{D_j - t}$. If the resulting $U(t) > 1$, we set $U(t)$ to be 1. It is easy to see that the voltage schedule obtained by **HPASTS** Algorithm yields a feasible schedule since the scheduler reserves enough processor speed, assuming the worst case, that all periodic and sporadic tasks meet their deadlines. The time complexity of **HPASTS** Algorithm depends linearly on the number of sporadic tasks in the system.

5.3 Experimental Results

We compared the **OPASTS** and **HPASTS** algorithms with the predictive system shutdown technique and the **MIN** algorithm, which gives us the minimum bound for dynamically variable voltage approach. The predictive system shutdown technique assumes the complete knowledge of the idle periods while the **MIN** algorithm assumes the complete knowledge of the arrivals of sporadic tasks.

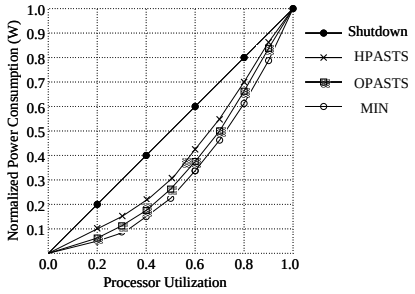


Figure 4: Dynamically variable voltage approach *vs.* Predictive system shutdown technique: Simulation results

We varied the average processor utilization from the light workload to heavy workload. The workload by the periodic tasks accounted for between 30-70 % of the total workload. The idealized predictive system shutdown technique achieves exactly half of the original power consumption. The **MIN** algorithm results in a single voltage solution corresponding to the average minimum required processor speed to meet the deadlines of the tasks. Similar as the workload with only the sproadic tasks, we observe that the power consumption of the solutions of **OPASTS** algorithm is close to that of the solutions of **MIN** algorithm. **HPASTS** algorithm produces solutions close to those of **OPASTS** algorithm, which consumed 15 % more power on average.

6 Conclusion

We proposed several on-line scheduling algorithms for scheduling the mixed workload of both sporadic (on-line) and periodic (off-line) hard real-time tasks on variable voltage processor to optimize power consumption while ensuring that all periodic tasks meet their deadlines and to accept as many sporadic tasks, which can be guaranteed to meet their deadlines, as possible. The proposed efficient algorithms result in the scheduling solutions for various scheduling scenarios and workloads, which are within 20 % of the minimum bound achievable with the dynamically variable voltage approach. The effectiveness of the proposed algorithms was shown on extensive experiments with numerous real-life design scenarios.

REFERENCES

[1] T. D. Burd and R.W. Brodersen. Processor design for portable systems. *Journal of VLSI Signal Processing*, 13(2-3):203–221, 1996.

[2] A. P. Chandrakasan, S. Sheng, and R.W. Broderson. Low-power CMOS digital design. *IEEE Journal of Solid-State Circuits*, 27(4):473–484, 1992.

[3] J.-M. Chang and M. Pedram. Energy minimization using multiple supply voltages. In *International Symposium on Low Power Electronics and Design*, pages 157–162, 1996.

[4] K. Govil, E. Chan, and H. Wasserman. Comparing algorithms for dynamic speed-setting of a low-power CPU. In *ACM International Conference on Mobile Computing and Networking*, pages 13–25, 1995.

[5] V. Gutnik and A. Chandrakasan. An efficient controller for variable supply-voltage low power processing. In *Symposium on VLSI Circuits*, pages 158–159, 1996.

[6] I. Hong, D. Kirovski, G. Qu, M. Potkonjak, and M. Srivastava. Power optimization of variable voltage core-based systems. In *Design Automation Conference*, 1998.

[7] C. Hwang and A.C.-H. Wu. A predictive system shutdown method for energy saving of event-driven computation. In *IEEE/ACM International Conference on Computer-Aided Design*, pages 28–32, 1997.

[8] M. C. Johnson and K. Roy. Datapath scheduling with multiple supply voltages and level converters. *ACM Transactions on Design Automation of Electronic Systems*, 2(3), 1997.

[9] C. Lee, M. Potkonjak, and W. H. Mangione-Smith. MediaBench: A tool for evaluating and synthesizing multimedia and communications systems. In *Micro 30*, 1997.

[10] Y.-R. Lin, C.-T. Hwang, and A.C.-H Wu. Scheduling techniques for variable voltage low power designs. *ACM Transactions on design Automation of Electronic Systems*, 2(2):81–97, 1997.

[11] C. L. Liu and J. Layland. Scheduling algorithms for multiprogramming in a hard real-time environment. *Journal of the ACM*, 20(1):46–61, 1973.

[12] W. Namgoong, M. Yu, and T. Meng. A high-efficiency variable-voltage CMOS dynamic DC-DC switching regulator. In *IEEE International Solid-State Circuits Conference*, pages 380–381, 1997.

[13] S. Raje and M. Sarrafzadeh. Variable voltage scheduling. In *International Symposium on Low Power Design*, pages 9–14, 1995.

[14] M. Srivastava, A. P. Chandrakasan, and R. W. Brodersen. Predictive system shutdown and other architectural techniques for energy efficient programmable computation. *IEEE Transactions on VLSI Systems*, 4(1):42–55, 1996.

[15] K. Suzuki, et al. A 300 MIPS/W RISC core processor with variable supply-voltage scheme in variable threshold-voltage CMOS. In *IEEE Custom Integrated Circuits Conference*, pages 587–590, 1997.

[16] T. Tia, J. W.-S. Liu, J. Sun, and R. Ha. A linear-time optimal acceptance test for scheduling of hard real-time tasks. Technical report, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana-Champaign, IL, 1994.

[17] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced CPU energy. In *IEEE Annual Foundations of Computer Science*, pages 374–382, 1995.