Energy Consumption Optimization in Real-Time Embedded Systems

Xuefeng Piao*, Heeheon Kim*, Yookun Cho*, Moonju Park†, Sangchul Han††, Minkyu Park‡ and Seongje Cho§

*School of Computer Science and Engineering, Seoul National University, Seoul, Korea
Email: {hbpark, hhkim, cho}@os.snu.ac.kr
†Dept. of Computer Science and Engineering, University of Incheon, Incheon, Korea
Email: mpark@incheon.ac.kr
‡Dept. of Computer Science, Konkuk University, Chungju, Korea
Email: {schan, minkyup}@kku.ac.kr
§Div. of Information and Computer Science, Dankook University, Yongin, Korea
Email: sjcho@dankook.ac.kr

Abstract

Minimizing energy consumption with guaranteeing real-time constraints in low-power embedded systems is gaining more importance as real-time applications become more widely used in embedded systems. Dynamic voltage scaling is a technique to reduce energy consumption by lowering supply voltage. However, lowering supply voltage may interfere with scheduling algorithms, so that tasks may not be successfully scheduled. In this paper, we formulate the problem of minimizing energy consumption for Pre-scheduling as an optimization problem, and show that the problem is a nonlinear convex optimization with linear constraints which can be solved by sequential quadratic programming. By solving the problem, we can obtain the optimal supply voltage and successful scheduling of all tasks is guaranteed.

1. Introduction

Embedded systems are generally resource constrained, special purpose computer systems with real-time computing constraints. The real-time constraint is often modeled as deadline before which all the tasks must complete execution. On the other hand, real-time embedded devices such as portable or mobile devices have a common limitation in power supply. As real-time embedded systems are more widely used in various fields, minimizing energy consumption as well as guaranteeing real-time constraints is the most important issue concerning low-power embedded systems.

Consider a Wireless Sensor Network (WSN) systems, recent research is focused on creating energy-efficient large-scale networks using energy saving technologies. Reducing energy consumption of battery operated sensor node in WSN is one of effective solution when each sensor node has a target operational lifetime.

Furthermore, the embedded system can in general be described to be either polling or reactive[1]. Each sensor node in WSN collects data and passes these data via network link to a repository at a fixed time interval. Also, the sensor node may perform mode changes and relay control data among sensor nodes. Task scheduling approach is an import part in energy-aware software design. This design approach allows the task scheduler to work in a reactive manner with polling. It requires that the task scheduling approach should provides strong predictability. In this paper, polling can be characterized as time-driven tasks and reactions can be characterized as event-driven tasks.

In task scheduling, static scheduling algorithm is well accepted for its predictability and simplicity, but it is difficult to schedule event-driven tasks. On the contrary, dynamic scheduling algorithm, such as the Earliest Deadline First scheduling algorithm [2], schedules event-driven tasks effectively, but it does not provide strong predictability and its on-line complexity is $O(\log n)$ which is higher than that of static scheduling algorithm $O(1)$. For a balanced solution, Wang et al. proposed an optimal Pre-Scheduling algorithm that produces a preschedule off-line for time-driven tasks with reserving sufficient time to schedule event-driven tasks[3], [4]. Since the Pre-Scheduling algorithm provides strong predictability in terms of guaranteeing the ordering of tasks, it is suitable for scheduling real-time tasks in embedded systems.

The problem of minimizing energy consumption while guaranteeing real-time constraints has been widely addressed in real-time literature. Voltage scaling is a technique that exploits the hardware characteristic of processors to reduce energy consumption by lowering supply voltage. Since the energy consumption has a quadratic dependency on the supply voltage, a small reduction in supply voltage can produce a significant reduction in energy consumption.

In general, there are two kinds of voltage scaling approaches for periodically running tasks. An off-line voltage scaling approach determines supply voltage before execution exploiting that the processor utilization must be less than or equal to one for guaranteeing successful schedule of all tasks, so that the processor can be fully utilized when
period is identical to relative deadline. On the other hand, an
on-line voltage scaling approach determines supply voltage
during run time to achieve lower energy consumption by
observing that the actual execution time is usually less
than the worst case execution time in common real-time
applications [5]. Recent research addressed other situations
such as task synchronization [6].

In previous investigations to minimize energy consump-
tion, some works focus on periodic tasks with assumption
that relative deadlines of the tasks are equal to their periods
[7], [8], [9], [10]. When the workload of system is mixed
task set, these approaches for periodic tasks cannot be ap-
plied. Other works addressed on-line approaches for a mixed
task set by observing the variation of system utilization at
run time [11], [12], [13]. The weakness of these approaches
is that the on-line complexity is higher than that of off-line
approaches. In this sense, for mixed task set has relative
deadlines of tasks less than their periods, none of existing
off-line approaches can be applied. Note that when the
relative deadlines of tasks are less than their respective
periods, since a feasible schedule may be produced even
if the density of a system is greater than one.

In this paper, we address the problem of minimizing
energy consumption for real-time embedded system in which
task scheduler works in reactive manner with polling.
Pre-Scheduling is adopted to optimally schedule for real-
time embedded system, we determine the optimal supply
voltage for each execution instance to minimize energy
consumption. We call each instance of tasks in a hyper
period as a job. A periodic job

\[ \tau = (C, D, D) \]

is defined by a 3-tuple of

\[ (C, D, D) \]

where \( C \) represents relative deadline,
\( D \) represents absolute deadline, and
\( L \) represents the worst case execution time in common
real-time systems. The weakness of these approaches
is that the on-line complexity is higher than that of off-line
approaches. In this sense, for mixed task set has relative
deadlines of tasks less than their periods, none of existing
off-line approaches can be applied. Note that when the
relative deadlines of tasks are less than their respective
periods, since a feasible schedule may be produced even
if the density of a system is greater than one.

In this paper, we address the problem of minimizing
energy consumption for real-time embedded system in which
task scheduler works in reactive manner with polling.
Pre-Scheduling is adopted to optimally schedule for real-
time embedded system, we determine the optimal supply
voltage for each execution instance to minimize energy
consumption. To this end, we formulate the problem of
energy consumption minimization for Pre-Scheduling as an
optimization problem and show the problem is a nonlinear
convex optimization problem. As a solution to the problem,
the sequential quadratic programming solver can be used
[14].

The primary contribution of this paper is that a valid
prescheduler with optimal supply voltage can be obtained
by solving the energy consumption optimization problem.
The second contribution of this paper is that the energy
consumption optimization can be applied to any scheduling
policy in which relative deadline of task may smaller than
its periods, and the task may have start time other than
the beginning of its period. The third contribution is that
the energy consumption optimization model can be used as
schedulability test for tasks in reactive real-time systems.

The rest of this paper is organized as follows. Section
2 outlines the problem of minimizing energy consumption
in voltage scaling. Section 3 briefly reviews Pre-Scheduling
algorithm. Section 4 formulates the problem of minimizing
energy consumption for Pre-Scheduling as an optimization
problem. An example is presented in Section 5. Section 6
concludes our work.

2. Dynamic Voltage Scaling

2.1. System Model

The workload of real-time systems is made up of indepen-
dent time-driven and event-driven tasks, and they are charac-
terized as periodic tasks and sporadic tasks, respectively. A
periodic task is defined by a 4-tuple of

\[ (O, L, C, P) \]

where \( O \) represents initial offset, \( L \) represents relative deadline, \( C \)
represents the worst case execution time at the maximum
supply voltage, and \( P \) represents period, respectively. A
sporadic task is defined by a 3-tuple of

\[ (P, L, C) \]

where \( P \) represents minimum inter-arrival time between two
consecutive instances, \( L \) represents relative deadline, and \( C \)
represents the worst case execution time at the maximum
supply voltage, respectively. The actual ready time of any
sporadic job is unknown until it arrives. For both periodic
and sporadic tasks, relative deadlines of a task is less than
or equal to its period or inter-arrival time \( (L \leq P) \).

The time line is considered as an infinite number of
consecutive hyper periods. The hyper period \( H \) is the least
common multiple of periods of all periodic and sporadic
tasks. Since the schedule of periodic tasks repeats itself after
each \( H \) time units, we only need to consider the tasks that
are scheduled during a hyper period in minimizing energy
consumption. We call each instance of tasks in a hyper
period as a job. A periodic job \( J_k \) is defined by a 3-tuple of

\[ (R, D, C) \]

where \( R \) represents ready time, \( D \) represents absolute
deadline, and \( C \) represents the worst case execution time.
A sporadic job \( J_k \) is defined by a 2-tuple of

\[ (L, C) \]

where \( L \) represents relative deadline, and \( C \) represents
the worst case execution time.

2.2. Energy Consumption

The energy consumption of a processor is dominated
by dynamic power dissipation \( P_D \), which is given by

\[ P_D = C_{ef} \cdot V_{dd}^2 \cdot f \]

where \( C_{ef} \) represents effective switched
capacitance, \( V_{dd} \) represents supply voltage, and \( f \)
represents clock frequency [15]. The circuit delay \( t_d \) is inversely
related to the supply voltage \( V_{dd} \) as given by the formula

\[ t_d = \frac{k \cdot V_{th}}{V_{dd}} \]

where \( k \) is a constant and \( V_{th} \) is the threshold
voltage. Since the clock frequency is inversely proportional
to the circuit delay, it can obtain striking power savings if
the supply voltage and the clock frequency are reduced. In
this sense, voltage scaling technique is based on obtaining
energy savings by reducing the supply voltage and the clock
frequency, at the expense of increased latency.

The ratio of the actual supply voltage to the maximum
supply voltage of the processor is called scaling factor
which is denoted as \( \eta(0 < \eta \leq 1) \). Let \( V_{max} \) denotes
the maximum supply voltage. Then the actual supply voltage
can be denoted as

\[ V_{dd} = \eta \cdot V_{max} \]

We first consider energy consumption by a job \( J_k \) which is given by

\[ P_D \cdot ACET_i \]
where $ACET_i$ represents the actual execution time of the job when the supply voltage is $\eta_i \cdot V_{max}$. Given a maximum processor speed $S_{max}$, the worst case execution cycles required by the job can be denoted as $C_i \cdot S_{max}$. The processor speed can be dynamically varied by adjusting the supply voltage and the clock frequency. In case that there is an imposing bound on the rate of voltage change, the voltage change overhead can be incorporated in the worst case workload of each task. The energy consumption of a job can be written as

$$C_{ef} \cdot (\eta_i \cdot V_{max})^2 \cdot C_i \cdot S_{max} \quad (1)$$

By summing up energy consumptions of jobs that are scheduled in a hyper period, the total energy consumption of periodic jobs can be formulated as in the objective function given by (2) with the timing constraint given by (3), where $n$ represents the number of periodic jobs in the hyper period. The objective function represents the total energy consumption of all periodic jobs scheduled in a hyper period. The constraint represents the total actual execution time of all periodic jobs in a hyper period must not exceed the length of the hyper period. It must be satisfied, or no feasible schedule exists [9].

$$\text{minimize}$$

$$\sum_{i=1}^{n} C_{ef} \cdot (\eta_i \cdot V_{max})^2 \cdot C_i \cdot S_{max} \quad (2)$$

subject to

$$\sum_{i=1}^{n} \frac{C_i}{\eta_i} \leq P \quad (3)$$

In the above energy consumption formulation, the relative deadlines of tasks must be equal to its periods. If the relative deadline of a task is less than its period, the successful scheduling of all tasks is not guaranteed even though (3) is satisfied. For example, suppose that there is only one periodic task defined as $(0,4,3,5)$. Then only one periodic job $(0,4,3)$ exists in the hyper period 5. Let us assume that the maximum supply voltage is $V_{max} = 1$. Then the objective function has the minimum value when the scaling factor is 0.6. However, with scaling factor of 0.6, the job completes its execution at time $\frac{1}{0.6} \cdot 3 = 5$, so it misses its deadline at time 4.

Note that, in our work, the relative deadlines of jobs can be less than their periods as shown in 4.2. We assume that all tasks are scheduled on single variable voltage processor which can change its clock speed dynamically at any clock cycle by adjusting supply voltage. The power loss of voltage switching and time overhead to change supply voltage or clock speed are assumed to be negligible.

### 3. Pre-Scheduling

Pre-Scheduling algorithm[3] consists of an off-line component and an on-line component. The off-line component produces a static preschedule in the form of list of execution instances for periodic jobs, which are called fragments[3]. Each fragment defines an execution time and a time interval; a job should be scheduled for at least the specified execution time in the time interval. The on-line component schedules fragments and sporadic jobs by EDF algorithm with an extra constraint: the execution order of the fragments must be kept as defined in the static preschedule.

The off-line component determines the execution times of fragments with the following three sets of constraints:

1) Non-negative constraints: The execution time of each fragment must be equal to or larger than 0.
2) Sufficient constraints: For every periodic job in a hyper period, the aggregate execution time of its fragments must be equal to its execution time.
3) Slack-reserving constraints on critical intervals: The aggregate execution time of fragments and all sporadic jobs that must be scheduled within a critical interval must be equal to or less than the length of critical time interval. A time interval $(X, Y)$ is critical if and only if the following conditions are all true.

(a) $0 < Y - X \leq P$.
(b) Time $X$ is between $(0, P)$ and there exists a job $J_x$ in a hyper period, such that $X = R_x$.
(c) There exists a job $J_y$ in a hyper period, such that $Y = D_y$.

### 4. Energy Optimization for Pre-Scheduling

#### 4.1. Energy Consumption Model

In this subsection, we address the problem of energy consumption in Pre-Scheduling. Suppose that a periodic job $J_i$ is split into $m$ fragments. The $j^{th}$ fragment of $J_i$ is denoted as $F_{i,j}$ and its execution time is denoted as $C_{i,j}$. Then the energy consumption for the periodic job $J_i$, $E(J_i)$, and the energy consumption for a sporadic job $J_k^S$, $E(J_k^S)$, are formulated as below, where $\eta_{i,j}$ represents the scaling factor of $F_{i,j}$ and $\eta_k^S$ represents the scaling factor of $J_k^S$.

$$E(J_i) = \sum_{j=1}^{m} C_{ef} \cdot (\eta_{i,j} \cdot V_{max})^2 \cdot C_{i,j} \cdot S_{max} \quad (4)$$

$$E(J_k^S) = C_{ef} \cdot (\eta_k^S \cdot V_{max})^2 \cdot C_k^S \cdot S_{max} \quad (5)$$

Then the total energy consumption during a hyper period is given by (6) which is the objective function in the problem of

1. Pre-scheduling is presented in the context of scheduler composition in [4], and our approach can be easily applied to that environments.
minimizing energy consumption for Pre-Scheduling, where \( w \) is the maximum number of sporadic jobs in a hyper period.

\[
\sum_{i=1}^{n} \sum_{j=1}^{m} (\eta_{i,j})^{2} \cdot C_{i,j} + \sum_{k=1}^{w} (\eta_{k}^{S})^{2} \cdot C_{k}^{S} \cdot C_{ef} \cdot V_{max}^{2} \cdot S_{max} \tag{6}
\]

Let \( I_{(X,Y)} \) denote a set of execution instances that include all fragments and sporadic jobs scheduled during a critical time interval \((X,Y)\). Then, when the execution instances run at the actual supply voltages \((\eta_{i,j} \cdot V_{max} \text{ and } \eta_{k}^{S} \cdot V_{max})\), the non-negative, sufficient, and slack-reserving constraints of Pre-Scheduling are given by (7)-(9), respectively.

\[
\forall F_{i,j} \quad C_{i,j} \geq 0 \tag{7}
\]
\[
\forall J_{i} \quad \sum_{j=1}^{m} C_{i,j} = C_{i} \tag{8}
\]
\[
\forall I_{(X,Y)} \quad \sum_{\forall F_{i,j} \in I_{(X,Y)}} C_{i,j} + \sum_{\forall J_{k} \in I_{(X,Y)}} C_{k}^{S} \leq Y - X \tag{9}
\]

Note that if the execution instances run at the maximum supply voltages \((\eta_{i,j} = 1 \text{ and } \eta_{k}^{S} = 1)\), then the slack-reserving constraints can be given by (10).

\[
\forall I_{(X,Y)} \quad \sum_{\forall F_{i,j} \in I_{(X,Y)}} C_{i,j} + \sum_{\forall J_{k} \in I_{(X,Y)}} C_{k}^{S} \leq Y - X \tag{10}
\]

Thus, the three sets of constraints (7), (8), (10) are intuitive constraints for a valid preschedule. The three sets of constraints are called execution-time constraints for a valid preschedule.

The constraints given in (7)-(9) are not sufficient to obtain the optimal supply voltages because jobs may violate real-time constraint. For example, there are two periodic tasks \((0, 5, 4, 10)\) and \((5, 5, 3, 10)\), and a sporadic task \((10, 1, 1)\). Then the hyper period of all periodic and sporadic tasks is 10, and a job set in the first hyper period is: \(J_{1}(0, 5, 4), J_{2}(5, 10, 3), J_{3}(1, 1)\). Under the three sets of constraints in (7)-(9), the scaling factors minimizing the objective function are given by \(\eta_{1} = 0.8, \eta_{2} = 0.8, \text{ and } \eta_{3}^{S} = 0.8\). Applying the scaling factors, the actual execution time of \(J_{3}^{S}\) becomes 1.25 time units, but in fact it should run at the maximum supply voltage since the relative deadline is equal to its worst case execution time.

The constraint (9) is nonlinear since \(\eta_{i,j}, \eta_{k}, \text{ and } C_{i,j}\) are all variables. Minimizing the nonlinear objective function (6) subject to linear constraints (7), (8), and nonlinear constraints (9) is a nonlinearly constrained optimization problem. Unfortunately, there is no general guaranteed procedure for solving the nonlinearly constrained optimization problem even though the sequential quadratic programming method is widely considered as a general technique for solving nonlinear problems. Generally speaking, problems with nonlinear constraints are typically more difficult to solve than problems with linear constraints, so that longer time is needed to solve the problems with nonlinear constraints than problems with linear constraints [16].

In the following subsection, we modify the problem and add new constraints so that jobs can meet real-time constraints. Then we show that the optimal solution to that problem can be obtained by showing that the problem is a convex optimization problem.

### 4.2. Energy Consumption Optimization

Let \(ST_{i,j}\) denote allocable slack time of a fragment \(F_{i,j}\) and \(ST_{k}^{S}\) denote allocable slack time of a sporadic job \(J_{k}^{S}\). Then the actual execution times of fragments can be expressed as \(C_{i,j} + ST_{i,j}\) and the actual execution times of a sporadic job can be expressed as \(C_{k}^{S} + ST_{k}^{S}\). The scaling factors of fragments and sporadic jobs can be expressed as

\[
\eta_{i,j} = \frac{C_{i,j}}{C_{i,j} + ST_{i,j}} \quad \text{and} \quad \eta_{k}^{S} = \frac{C_{k}^{S}}{C_{k}^{S} + ST_{k}^{S}} \tag{11}
\]

We reformulate the objective function as (12) and the constraints as (13)-(17) by replacing scaling factors in (6)-(9) by those in (11).

For the calculation of allocable slack time, we introduce allocable slack time constraint, which defines the upper bound of allocable slack time for each job. The upper bounds of allocable slack times for fragments are given by (16) and the upper bound of allocable slack times for sporadic jobs are given by (17).

\[
\text{minimize } \left[ \sum_{i=1}^{n} \sum_{j=1}^{m} \left( C_{i,j}^{3} \right) + \sum_{k=1}^{w} \left( C_{k}^{S} + ST_{k}^{S} \right)^{3} \right] \cdot C_{ef} \cdot V_{max}^{2} \cdot S_{max} \tag{12}
\]

subject to

\[
\forall F_{i,j} \text{ and } \forall J_{k}^{S} \quad C_{i,j} \geq 0, \quad ST_{i,j} \geq 0, \quad ST_{k}^{S} \geq 0 \tag{13}
\]
\[
\forall J_{i} \quad \sum_{j=1}^{m} C_{i,j} = C_{i} \tag{14}
\]
\[
\forall I_{(X,Y)} \quad \sum_{\forall F_{i,j} \in I_{(X,Y)}} (C_{i,j} + ST_{i,j}) + \sum_{\forall J_{k} \in I_{(X,Y)}} (C_{k}^{S} + ST_{k}^{S}) \leq Y - X \tag{15}
\]
\[
\forall J_{k} \quad \sum_{j=1}^{m} ST_{i,j} + \sum_{J_{k}^{S} \in I_{(X,Y)}} (C_{k}^{S} + ST_{k}^{S}) \leq D_{i} - R_{i} - C_{i} \tag{16}
\]
\[
\forall J_{k}^{S} \quad ST_{k}^{S} \leq L_{k}^{S} - C_{k}^{S} \tag{17}
\]
Now we show that minimizing the objective function (12) under the constraints (13)-(17) is in fact an instance of nonlinear convex optimization problem. The proof is based on the mathematical fundamentals on convex functions.

Given a nonlinear function \( f(x) \) with constraints \( C, C = \{ x \in \mathbb{R}^n \mid A \cdot x \leq b, Aeq \cdot x = beq, lb \leq x \leq ub \} \), where \( x, b, beq \), \( lb \) and \( ub \) are vectors, and \( A \) and \( Aeq \) are matrices, a vector \( x^* \) is called optimal, if it has the smallest objective value among all vectors in the feasible set \( C \).

The following propositions describe the characteristics of convex function [17].

**Propositions:**

P1. A linear function is convex.

P2. The weighted sum of convex functions with positive weights, is convex.

P3. Let \( C \subset \mathbb{R}^n \) be a convex set and let \( f : \mathbb{R}^n \rightarrow \mathbb{R} \) be twice continuously differentiable over \( C \). If \( \nabla^2 f(x) \) is positive semi-definite for all \( x \in C \), then \( f \) is convex over \( C \).

**Lemma 4.1.** The problem of minimizing energy consumption for Pre-Scheduling is nonlinear convex optimization problem.

**Proof:** In Pre-Scheduling energy consumption model, since the constraints (13), (14), (15), (16) and (17) are all linear, the constraints are all convex by Proposition P1. So the feasible set \( C \) is convex. The objective function (12) can be denoted as \( f(x) = p(x) + s(x) \), where \( p(x) = p_1(x) + p_2(x) + \cdots + p_n(x) \) and \( s(x) = s_1(x) + s_2(x) + \cdots + s_m(x) \). We only need to prove that each instance \( p_i(x) \) of \( p(x) \) and each instance \( s_i(x) \) of \( s(x) \) are convex by Proposition P2. On the basis of Proposition P3, we can obtain the grad \( \nabla^2 p_i(x) \) and \( \nabla^2 s_i(x) \), and easily show that they are all positive semi-definite. Therefore the objective function (12) is convex function.

A fundamental property of convex optimization problems is that a locally optimal point is also globally optimal [18]. By Lemma 4.1, energy consumption optimization for Pre-scheduling is convex optimization problem. We can obtain optimal execution times and allocable slack times for fragments and sporadic jobs, consequently optimal voltage scaling factors.

This paper assumes that the speed of the processor can be varied continuously. More practically, for processors that provide a finite number of discrete speeds, the optimal supply voltage changes can be computed in polynomial time by the methods proposed in [19], [20].

**5. Example**

The periodic task set \( T_p \) and sporadic task set \( T_s \) is defined below.

\[
T_p = \{ \tau_1(90,30,5,225), \tau_2(0,75,10,75), \tau_3(0,225,120,225) \} \\
T_s = \{ (225,225,225) \}
\]

The hyper period of the all task sets is 225. The set of periodic jobs in a hyper period is as follows.

\[
J_1(0,75,10), J_2(75,150,10), J_3(0,225,120), \\
J_4(150,225,10), J_5(90,120,5).
\]

The fragments of periodic jobs and the execution order of these fragments are as follows. However, the execution time of each executive is not known yet.

\[
F_{1,1}(0,75, C_{1,1}), F_{3,1}(0,120, C_{3,1}), F_{2,1}(75,120,C_{2,1}), F_{5,1}(90,120, C_{5,1}), F_{2,2}(90,150, C_{2,2}), F_{3,2}(90,225, C_{3,2}), F_{4,1}(150,225, C_{4,1}).
\]

Since \( C_{ef} \cdot V_m^2 \cdot S_m \) is constant, the objective function can be rewritten as

\[
\begin{align*}
&\frac{C_{1,1}^3}{(C_{1,1} + ST_{1,1})^2} + \frac{C_{3,1}^3}{(C_{3,1} + ST_{3,1})^2} + \frac{C_{2,1}^3}{(C_{2,1} + ST_{2,1})^2} \\
&+ \frac{C_{6,1}^3}{(C_{5,1} + ST_{5,1})^2} + \frac{C_{2,2}^3}{(C_{2,2} + ST_{2,2})^2} + \frac{C_{3,2}^3}{(C_{3,2} + ST_{3,2})^2} \\
&+ \frac{C_{4,1}^3}{(C_{4,1} + ST_{4,1})^2} + \frac{C_{7}^3}{(C_{7} + ST_{7})^2}
\end{align*}
\]

where

\[
C_{1,1} = 10, \quad C_{4,1} = 10, \quad C_{5,1} = 5, \quad C_{7} = 25.
\]

The real-time constraints of the jobs are given as follows.

\[
C_{3,1} + C_{2,1} + ST_{3,1} + ST_{2,1} \leq 80, \\
C_{2,2} + C_{3,2} + ST_{3,2} + ST_{2,2} + ST_{1} \leq 95, \\
C_{2,1} + C_{2,2} = 10, \\
C_{3,1} + C_{3,2} = 120, \\
ST_{2,1} \leq 15, ST_{2,2} \leq 30, ST_{1,1} \leq 40, ST_{5,1} \leq 40.
\]

The execution times of executives and the allocable slack times for voltage scaling can be obtained by using the SQP. The result is shown in the following.

We suppose that the sporadic job \( J^S \) arrived at time 100. Figure 1 shows the on-line execution of all executives and sporadic jobs in a hyper period. The actual supply voltages of execution instances are represented on vertical line, and horizontal solid line represents the time line. The dotted line represents the maximum supply voltage. For each execution instance is represented as a horizontal line segment. The length of box between two ends represents the actual execution time. The execution order of execution instances is \( F_{1,1}, F_{3,1}, F_{2,1}, F_{5,1}, F_{2,2}, J^S, F_{3,2}, F_{4,1}, \). The proposed
algorithm reduces energy consumption 24% approximately (from 155 to 117.8).

<table>
<thead>
<tr>
<th>$F_{i,j}$</th>
<th>$C_{i,j}$</th>
<th>$ST_{i,j}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F^*_1$</td>
<td>$C^*_1$</td>
<td>$3$</td>
</tr>
<tr>
<td>$F^*_2$</td>
<td>$C^*_2$</td>
<td>$1$</td>
</tr>
<tr>
<td>$F^*_3$</td>
<td>$C^*_3$</td>
<td>$2$</td>
</tr>
<tr>
<td>$F^*_4$</td>
<td>$C^*_4$</td>
<td>$3$</td>
</tr>
<tr>
<td>$F^*_5$</td>
<td>$C^*_5$</td>
<td>$0$</td>
</tr>
</tbody>
</table>

Table 1. Execution times and allocable slack times

Figure 1. On-line execution of execution instances in a hyper period

6. Conclusion

In this paper, we have addressed the energy consumption optimization problem for real-time embedded systems. We formulated the problem of energy consumption minimization in Pre-scheduling as an optimization problem, and showed that it can be formulated as a nonlinear convex problem with linearly constraints. For a given periodic task set and a given sporadic task set in a real-time embedded systems, the optimal supply voltage of each execution instance can be obtained by solving the proposed optimization problem in energy consumption whenever a feasible schedule exists. It was also shown that the produced preschedule is valid in the sense that it guarantees all periodic tasks and sporadic tasks are scheduled successfully.

References


