Adaptive Power Management for Real-Time Event Streams

Kai Huang¹   Luca Santinelli²   Jian-Jia Chen¹
Lothar Thiele¹   Giorgio C. Buttazzo²

¹Computer Engineering and Networks Laboratory
ETH Zurich, Switzerland

²Scuola Superiore Sant’Anna of Pisa, Italy

January 19, 2010
Power Dissipation

- Dynamic power consumption
- Static power consumption (leakage)
Power Dissipation
- Dynamic power consumption
- Static power consumption (leakage)

Energy Saving
- Dynamic Voltage Scaling
- Dynamic Power Management
Figure 1: ITRS Technology Roadmap: Power Trends.
The leakage power is comparable to or even more than the dynamic power dissipation

⇒ Reducing the leakage power is crucial
Challenges

- When to turn off
  - mode switch overhead
  - Break Even Time
Challenges

- When to turn off
  - mode switch overhead
  - Break Even Time
When to turn off

- mode switch overhead

⇒ Break Even Time
Challenges

- When to turn off
  - mode switch overhead
  - Break Even Time

Break Even Time

- turn on
- turn off
Challenges

- **When to turn off**
  - mode switch overhead
  - ⇒ Break Even Time

- **When to turn on**
  - as late as possible
  - cope with future burstiness
Challenges

- **When to turn off**
  - mode switch overhead
  - \[\Rightarrow\] Break Even Time

- **When to turn on**
  - as late as possible
  - cope with future burstiness
Challenges

► When to turn off
  ▶ mode switch overhead
  ⇒ Break Even Time

► When to turn on
  ▶ as late as possible
  ▶ cope with future burstiness
Challenges

- **When to turn off**
  - mode switch overhead
  - \( \Rightarrow \) Break Even Time

- **When to turn on**
  - as late as possible
  - cope with future burstiness
Challenges

- **When to turn off**
  - mode switch overhead
  - ⇒ Break Even Time

- **When to turn on**
  - as late as possible
  - cope with future burstiness
Challenges

- **When to turn off**
  - mode switch overhead
  - \( \Rightarrow \) Break Even Time

- **When to turn on**
  - as late as possible
  - cope with future burstiness
Challenges

► When to turn off
  ▶ mode switch overhead
  ⇒ Break Even Time

► When to turn on
  ▶ as late as possible
  ▶ cope with future burstiness
Challenges

- **When to turn off**
  - mode switch overhead
  - ⇒ Break Even Time

- **When to turn on**
  - as late as possible
  - cope with future burstiness
  - ⇒ more complex for multiple streams
Contributions

- Extend our online algorithms in RTSS’09 which adaptively control the power mode of a system (device) by
  - predicting the next moment for mode switch by considering both historical and future event arrivals
  - procrastinating the buffered and future events as late as possible without violating the timing and backlog constraints
- Propose method to cope with multiple streams with
  - preemptive fixed-priority scheduling
  - preemptive earliest-deadline-first scheduling
- Apply to streams with different characteristics to demonstrate the effectiveness
Outline

1. Introduction
2. Underlying Mathematical Model
3. Our Algorithms
4. Experimental Results
Traditionally

- Periodic Real-Time Tasks
- Sporadic Real-Time Tasks
- ............
Model of Event Arrivals

- Traditionally
  - Periodic Real-Time Tasks
  - Sporadic Real-Time Tasks
  - ...........

- Nowadays
  - Arrival curves
  - Maximum/minimum arriving demand in *any interval* of length, e.g. 3 s
Model of Event Arrivals

- Traditionally
  - Periodic Real-Time Tasks
  - Sporadic Real-Time Tasks
  - ...........

- Nowadays
  - Arrival curves
  - Maximum/minimum arriving demand in *any interval* of length, e.g. 3 s
Model of Event Arrivals

- Traditionally
  - Periodic Real-Time Tasks
  - Sporadic Real-Time Tasks
  - ...........

- Nowadays
  - Arrival curves
  - Maximum/minimum arriving demand in any interval of length, e.g. 3 s
Model of Event Arrivals

Traditionally
- Periodic Real-Time Tasks
- Sporadic Real-Time Tasks
- ...........

Nowadays
- Arrival curves
  ⇒ Maximum/minimum arriving demand in any interval of length, e.g. 3 s
Model of Event Arrivals

- Traditionally
  - Periodic Real-Time Tasks
  - Sporadic Real-Time Tasks
  - ...........

- Nowadays
  - Arrival curves
  - Maximum/minimum arriving demand in *any interval* of length, e.g. 3 s
Model of Event Arrivals

Traditionally
- Periodic Real-Time Tasks
- Sporadic Real-Time Tasks
- ...........

Nowadays
- Arrival curves
  \[ \Rightarrow \] Maximum/minimum arriving demand in \emph{any interval} of length, e.g. 3 s

Time domain
\[ \Delta = 3 \]

6/22
Model of Event Arrivals

Traditionally
- Periodic Real-Time Tasks
- Sporadic Real-Time Tasks
- ...........

Nowadays
- Arrival curves
- Maximum/minimum arriving demand in *any interval* of length, e.g. 3 s
Traditionally
- Periodic Real-Time Tasks
- Sporadic Real-Time Tasks
- ...........

Nowadays
- Arrival curves
- Maximum/minimum arriving demand in *any interval* of length, e.g. 3 s
Model of Event Arrivals

Traditionally
- Periodic Real-Time Tasks
- Sporadic Real-Time Tasks
- ..........

Nowadays
- Arrival curves
- Maximum/minimum arriving demand in any interval of length, e.g. 3 s
Model of Event Arrivals

- Traditionally
  - Periodic Real-Time Tasks
  - Sporadic Real-Time Tasks
  - .........

- Nowadays
  - Arrival curves
    - Maximum/minimum arriving demand in *any interval* of length, e.g. 3 s
  - Arrival curves generalize traditional event models and are suitable to represent complex characteristics of event streams
Service curves

⇒ Maximum/minimum available service in *any interval* of length $\Delta$ for the whole time span
Service curves

⇒ Maximum/minimum available service in *any interval* of length $\Delta$ for the whole time span
Model of Resource Service

- Service curves
  - Maximum/minimum available service in *any interval* of length $\Delta$ for the whole time span
Service curves

⇒ Maximum/minimum available service in *any interval* of length $\Delta$ for the whole time span

Service curves generalize different resource models
Delay and Backlog Analysis

Given:

- $\alpha$ is the stream arrival curve
- $\beta$ is the service guarantee

$\Rightarrow$ maximum delay $D$

$\Rightarrow$ maximum backlog $B$
Delay and Backlog Analysis

Given:
- $\alpha$ is the stream arrival curve
- $\beta$ is the service guarantee

$\Rightarrow$ maximum delay $D$
$\Rightarrow$ maximum backlog $B$
Delay and Backlog Analysis

Given:
- $\alpha$ is the stream arrival curve
- $\beta$ is the service guarantee

$\Rightarrow$ maximum delay $D$
$\Rightarrow$ maximum backlog $B$

Scheduling Analysis

Suppose:
- $\alpha$ is the stream arrival curve
- $D$ is the deadline of the stream
- $\beta^A = \alpha(\Delta - D)$ is service demand of $\alpha$
- $\beta^G$ is the service guarantee

$\Rightarrow$ the event stream is schedulable iff

$$\beta(\Delta) \geq \beta^A = \alpha(\Delta - D), \quad \forall \Delta \geq 0$$
Delay and Backlog Analysis

Given:
- $\alpha$ is the stream arrival curve
- $\beta$ is the service guarantee

$\Rightarrow$ maximum delay $D$
$\Rightarrow$ maximum backlog $B$

Scheduling Analysis

Suppose:
- $\alpha$ is the stream arrival curve
- $D$ is the deadline of the stream
- $\beta^A = \alpha(\Delta - D)$ is service demand of $\alpha$
- $\beta^G$ is the service guarantee

$\Rightarrow$ the event stream is schedulable iff
- $\beta(\Delta) \geq \beta^A = \alpha(\Delta - D)$, $\forall \Delta \geq 0$
Outline

1. Introduction
2. Underlying Mathematical Model
3. Our Algorithms
4. Experimental Results
System model

- A device is managed by a controller
- Distributed backlogs to buffer events for each stream
- Events of different streams scheduled with EDF/FP policies

Power model

- Active mode with power consumption $P_a$
- Standby mode with leakage power consumption $P_s$
- Sleep mode with power consumption $P_\delta$
- Mode switch overhead: Break even time $T_{BET}$
The Control Flow of Our Approach

Control flow

- sleep
- activation/decision
- active/standby
- deactivation/decision
The Control Flow of Our Approach

Control flow

Activation & Deactivation scheduling decisions

- History Aware Deactivation (HAD) algorithm
- Worst Case Greedy (WCG) activation algorithm
- Event Driven Greedy (EDG) activation algorithm
Basic Routines

- Bounded delay function: \( \text{bdf}(\Delta, \tau) \)
Basic Routines

- Bounded delay function: \( \text{bdf}(\Delta, \tau) \)
Bounded delay function: $\text{bdf}(\Delta, \tau)$
Bounded delay function: $\text{bdf}(\Delta, \tau)$

$\beta^A(\Delta) = \alpha^u_1(\Delta - D_1)$
Basic Routines

- **Bounded delay function:** \( \text{bdf}(\Delta, \tau) \)

\[
\beta^A(\Delta) = \alpha_1^u(\Delta - D_1)
\]
Bounded delay function: \( bdf(\Delta, \tau) \)

\[
\beta^A(\Delta) = \alpha^u_1(\Delta - D_1)
\]

\[
\Delta
\]

\[
Q \cdot w_1
\]

\[
D_1
\]

\[
\tau^*
\]
Bounded delay function: $\text{bdf}(\Delta, \tau)$

$$\delta^* = D_1 bdf(\Delta, \tau^*)$$

$$\alpha_1^u(\Delta)$$

$$\beta^A(\Delta) = \alpha_1^u(\Delta - D_1)$$
Basic Routines

- Bounded delay function: \( \text{bdf}(\Delta, \tau) \)

\[
\Delta_1^{\alpha}(\Delta) = \alpha_1^{\alpha}(\Delta - D_1)
\]

\[
\beta^A(\Delta) = \alpha_1^{\alpha}(\Delta - D_1)
\]
Basic Routines

- Bounded delay function: \( \text{bdf}(\Delta, \tau) \)

\[ bdf(\Delta, \tau^*) = \alpha^u_1(\Delta - D_1) \]

- History aware arrival curve: \( \alpha^u(\Delta, t) \)

\[ \alpha^u_1(\Delta) \]

\[ Q \cdot w_1 \]

\[ D_1 \]

\[ \tau^* \]

\[ \delta^* \]

\[ \beta^A(\Delta) = \alpha^u_1(\Delta - D_1) \]
Basic Routines

★ Bounded delay function: \( \text{bdf}(\Delta, \tau) \)

★ History aware arrival curve: \( \alpha^u(\Delta, t) \)
Basic Routines

- Bounded delay function: \( bdf(\Delta, \tau) \)

\[
\Delta \alpha_1^u(\Delta) = \alpha_1^u(\Delta - D_1)
\]

- History aware arrival curve: \( \alpha^u(\Delta, t) \)

\[
\Delta h
\]
Basic Routines

- Bounded delay function: \( \text{bdf}(\Delta, \tau) \)

\[
\alpha_1^u(\Delta) = \text{bdf}(\Delta, \tau^*)
\]

\[
\beta^A(\Delta) = \alpha_1^u(\Delta - D_1)
\]

- History aware arrival curve: \( \alpha^u(\Delta, t) \)

\[
H_i(\Delta, t')
\]

\( \Delta^h \)
Basic Routines

- Bounded delay function: \( bdf(\Delta, \tau) \)

\[
\alpha_1^u(\Delta) = \beta_A^u(\Delta) = \alpha_1^u(\Delta - D_1)
\]

- History aware arrival curve: \( \alpha^u(\Delta, t) \)

\[
H_i(\Delta, t') \quad \Delta^h \quad \alpha^u(\Delta)
\]
Basic Routines

- Bounded delay function: \( bdf(\Delta, \tau) \)

- History aware arrival curve: \( \alpha^u(\Delta, t) \)
Basic Routines

- **Bounded delay function**: $\text{bdf}(\Delta, \tau)$

  \[
  \text{bdf}(\Delta, \tau) = \max\{0, (\Delta - \tau)\}, \quad \forall \Delta \geq 0
  \]

  \[
  \tau^* = \max\{\tau : \text{bdf}(\Delta, \tau) \geq \beta^A(\Delta), \quad \forall \Delta \geq 0\}
  \]

  \[
  \delta^* = \max\{0, \min\{\delta : \alpha^u_i(\Delta) - \text{bdf}(\Delta, \tau^* - \delta) \leq Q_i \cdot w_i, \quad \forall \Delta\}\}
  \]

- **History aware arrival curve**: $\alpha^u_i(\Delta, t)$

  \[
  H_i(\Delta, t') = \begin{cases} 
    R_i(t') - R_i(t - \Delta), & \text{if } \Delta \leq \Delta^h; \\
    R_i(t') - R_i(t' - \Delta^h), & \text{otherwise}.
  \end{cases}
  \]

  \[
  \alpha^u_i(\Delta, t') \leq \inf_{\lambda \geq 0} \{\alpha^u_i(\Delta + \lambda) - H_i(\lambda, t')\}
  \]
History Aware Deactivation (HAD) algorithm
  - idea: turn off when the sleep interval can be larger than the break even time

Worst Case Greedy (WCG) activation algorithm
  - idea: reevaluate when at the previous predication time

Event Driven Greedy (EDG) activation algorithm
  - idea: reevaluate upon every event arrival

Details refer to RTSS’09
History Aware Deactivation (HAD) algorithm
   - idea: turn off when the sleep interval can be larger than the break even time

Worst Case Greedy (WCG) activation algorithm
   - idea: reevaluate when at the previous predication time

Event Driven Greedy (EDG) activation algorithm
   - idea: reevaluate upon every event arrival

Details refer to RTSS’09

Key: How to compute a valid but tight service demand $\beta^A$
Computing Flow for Multiple-Stream Scenario

- History-aware future arrival: $\alpha^u_i$
- Backlogged-aware and backlog-constrained demands: $\beta^b_i, \beta^\dagger_i$
- Individual stream service demand $\beta^*_i$
- Total service demand $\beta^A_{total}$
- Bounded delay $\tau_{total}$ from $\beta^A_{total}$
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{\text{total}} \]

\[ \alpha_1 \rightarrow S_1 \]

\[ \alpha_2 \rightarrow S_2 \]

\[ \vdots \]

\[ \alpha_N \rightarrow S_N \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{\text{total}} \]

\[ \beta^*_1 \]

\[ \alpha_1 \rightarrow S_1 \]

\[ \alpha_2 \rightarrow S_2 \]

\[ \ldots \]

\[ \alpha_N \rightarrow S_N \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{\text{total}} \]

\[ \beta_1^* \]

\[ \alpha_1 \]

\[ S_1 \]

\[ \beta_2^* \]

\[ \alpha_2 \]

\[ S_2 \]

\[ \ldots \]

\[ \alpha_N \]

\[ S_N \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{\text{total}} \]

\[ \alpha_1 \rightarrow S_1 \]

\[ \beta_1^* \]

\[ \alpha_2 \rightarrow S_2 \]

\[ \beta_2^* \]

\[ \beta_3^* \]

\[ \vdots \]

\[ \alpha_N \rightarrow S_N \]

\[ \beta_N^* \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{\text{total}} \]

\[ \beta^*_1 \]

\[ \alpha_1 \rightarrow S_1 \]

\[ \beta^*_2 \]

\[ \alpha_2 \rightarrow S_2 \]

\[ \beta^*_3 \]

\[ \beta^*_N \]

\[ \alpha_N \rightarrow S_N \]

\[ \ldots \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{total} \]

\[ \begin{align*}
\alpha_1 & \quad \beta_1^* \\
\alpha_2 & \quad \beta_2^* \\
\alpha_N & \quad \beta_N^* \\
\end{align*} \]

\[ \beta_N^*(\Delta, t') = \max\{\beta_N^b(\Delta, t'), \beta_N^i(\Delta, t')\} \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{total} \]

\[ \alpha_1 \]

\[ S_1 \]

\[ \beta_1^* \]

\[ \alpha_2 \]

\[ S_2 \]

\[ \beta_2^* \]

\[ \vdots \]

\[ \alpha_N \]

\[ S_N \]

\[ \beta_N^* \]

\[ \beta^*_k(\Delta) = \max \{ \beta^\#_{k-1}(\Delta), \beta^b_{k-1}(\Delta, t'), \beta^t_{k-1}(\Delta, t') \} \]

\[ \beta^\#_{k-1}(\Delta) \geq \inf \{ \beta : \beta^*_k(\Delta, t') = \sup_{0 \leq \lambda \leq \Delta} \{ \beta(\lambda) - \alpha^u_{k-1}(\lambda, t') \} \} \]

\[ \beta^*_N(\Delta, t') = \max \{ \beta^b_N(\Delta, t'), \beta^t_N(\Delta, t') \} \]
Preemptive Fixed-Priority Scheduling

\[ \beta^A_{\text{total}}(\Delta) = \beta_1^*(\Delta) \]

\[ \beta^*_k(\Delta) = \max \{ \beta^k_{\#}(\Delta), \beta^k_{-1}(\Delta, t'), \beta^k_{\dag}(\Delta, t') \} \]

\[ \beta^k_{-1}(\Delta) \geq \inf \{ \beta : \beta^k(\Delta, t') = \sup_{0 \leq \lambda \leq \Delta} \{ \beta(\lambda) - \alpha^u_{k-1}(\lambda, t') \} \} \]

\[ \beta^*_N(\Delta, t') = \max \{ \beta^N_{\#}(\Delta, t'), \beta^N_{\dag}(\Delta, t') \} \]
Preemptive Fixed-Priority Scheduling

$$\beta_{total}^A$$

$$\beta_{total}^A(\Delta) = \beta^*_1(\Delta)$$

$$\beta_{k-1}^*(\Delta) = \max \{ \beta_{k-1}^\#, \beta_{k-1}^b(\Delta, t'), \beta_{k-1}^t(\Delta, t') \}$$

$$\beta_{k-1}^\#(\Delta) \geq \inf \{ \beta : \beta_k^*(\Delta, t') = \sup_{0 \leq \lambda \leq \Delta} \{ \beta(\lambda) - \alpha_{k-1}^u(\lambda, t') \} \}$$

$$\beta_N^*(\Delta, t') = \max\{ \beta_N^b(\Delta, t'), \beta_N^t(\Delta, t') \}$$

$$\tau_{total} = \max\{ \tau : \text{bdf}(\Delta, \tau) \geq \beta^*_1(\Delta), \forall \Delta \geq 0 \}$$
Preemptive Earliest-Deadline-First Scheduling

\[ \sum_{i \neq j} \alpha_i \]

\[ \beta^A_{\text{total}} \]

\[ \alpha_j^* \rightarrow S_j \]

\[ S \setminus \{S_j\} \]
Preemptive Earliest-Deadline-First Scheduling

\[ \beta^A_{\text{total}} \]

\[ \sum_{i \neq j} \alpha_i \]

\[ S \setminus \{s_j\} \]

\[ \beta_j^* \]

\[ \alpha_j^* \]

\[ s_j \]
Preemptive Earliest-Deadline-First Scheduling

\[ \beta_{total}^A \]

\[ \sum_{i \neq j} \alpha_i \]

\[ S \setminus \{s_j\} \]

\[ \beta_{j, \text{total}}^* \]

\[ \beta_j^* \]

\[ \alpha_j^* \]

\[ s_j \]
Preemptive Earliest-Deadline-First Scheduling

\[ \beta_{total}^A \]

\[ \beta_{j, total}^* \]

\[ \sum_{i \neq j} \alpha_i \]

\[ S \setminus \{s_j\} \]

\[ \beta_j^* \]

\[ \alpha_j^* \]

\[ s_j \]
Preemptive Earliest-Deadline-First Scheduling

\[
\beta^A_{\text{total}}
\]

\[
\sum_{i \neq j} \alpha_i \\
S \setminus \{S_j\}
\]

\[
\beta_j^{*} \\
\alpha_j^{*} \\
S_j
\]

\[
\beta_j^*(\Delta, t') = \max\{\beta_j^b(\Delta, t'), \beta_j^t(\Delta, t')\}
\]
Preemptive Earliest-Deadline-First Scheduling

\[ \sum_{i \neq j} \alpha_i \]

\[ S \setminus \{S_j\} \]

\[ \beta_j^{\ast} \]

\[ \beta_j^{\ast}(\Delta) = \max \{ \beta_j^{\ast}(\Delta), \sum_{i \neq j} \beta_i^{b}(\Delta, t') \} \]

\[ \beta_j^{\ast}(\Delta) \geq \inf \{ \beta : \beta_j^{\ast}(\Delta, t') = \sup_{0 \leq \lambda \leq \Delta} \{ \beta(\lambda) - \sum_{i \neq j} \alpha_i^{u}(\lambda, t') \} \} \]

\[ \beta_j^{\ast}(\Delta, t') = \max \{ \beta_j^{b}(\Delta, t'), \beta_j^{\dagger}(\Delta, t') \} \]
Preemptive Earliest-Deadline-First Scheduling

\[ \beta^A_{\text{total}}(\Delta) = \max_{i \in N} \{\beta^*_i(\Delta)\} \]

\[ \beta^*_{j,\text{total}}(\Delta) = \max \{\beta^*_j(\Delta), \sum_{i \neq j}^{N} \beta^b_i(\Delta, t')\} \]

\[ \beta^*_j(\Delta, t') = \max\{\beta^b_j(\Delta, t'), \beta^\dagger_j(\Delta, t')\} \]

\[ \beta^*_j(\Delta) \geq \inf \{\beta : \beta^*_j(\Delta, t') = \sup_{0 \leq \lambda \leq \Delta} \{\beta(\lambda) - \sum_{i \neq j}^{N} \alpha^u_i(\lambda, t')\}\} \]
Preemptive Earliest-Deadline-First Scheduling

\[ \beta^A_{\text{total}}(\Delta) = \max_{i \in N} \{\beta^*_i, \text{total}(\Delta)\} \]

\[ \beta^*_{j, \text{total}}(\Delta) = \max \{\beta^*_j(\Delta), \sum_{i \neq j} \beta^b_i(\Delta, t')\} \]

\[ \beta^\sharp_j(\Delta) \geq \inf \{\beta : \beta^*_j(\Delta, t') = \sup_{0 \leq \lambda \leq \Delta} \{\beta(\lambda) - \sum_{i \neq j} \alpha^u_i(\lambda, t')\}\} \]

\[ \beta^*_j(\Delta, t') = \max\{\beta^b_j(\Delta, t'), \beta^\dagger_j(\Delta, t')\} \]

\[ \tau_{\text{total}} = \max \{\tau : \text{bdf}(\Delta, \tau) \geq \max_{i \in N} \{\beta^*_i, \text{total}(\Delta)\}, \forall \Delta \geq 0\} \]
Outline

1. Introduction
2. Underlying Mathematical Model
3. Our Algorithms
4. Experimental Results
Experiment Setup

- **Event Stream Setting**

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
<th>$S_5$</th>
<th>$S_6$</th>
<th>$S_7$</th>
<th>$S_8$</th>
<th>$S_9$</th>
<th>$S_{10}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>p (msec)</td>
<td>198</td>
<td>102</td>
<td>283</td>
<td>354</td>
<td>239</td>
<td>194</td>
<td>148</td>
<td>114</td>
<td>313</td>
<td>119</td>
</tr>
<tr>
<td>j (msec)</td>
<td>387</td>
<td>70</td>
<td>269</td>
<td>387</td>
<td>222</td>
<td>260</td>
<td>91</td>
<td>13</td>
<td>302</td>
<td>187</td>
</tr>
<tr>
<td>d (msec)</td>
<td>48</td>
<td>45</td>
<td>58</td>
<td>17</td>
<td>65</td>
<td>32</td>
<td>78</td>
<td>-</td>
<td>86</td>
<td>89</td>
</tr>
<tr>
<td>c (msec)</td>
<td>12</td>
<td>7</td>
<td>7</td>
<td>11</td>
<td>8</td>
<td>5</td>
<td>13</td>
<td>14</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

- **Power Profiles for the Device**

<table>
<thead>
<tr>
<th>Device Name</th>
<th>$P_a$ (Watt)</th>
<th>$P_s$ (Watt)</th>
<th>$P_\sigma$ (Watt)</th>
<th>$t_{sw}$ (sec)</th>
<th>$E_{sw}$ (mJ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM Microdrive</td>
<td>1.3</td>
<td>0.5</td>
<td>0.1</td>
<td>0.012</td>
<td>9.6</td>
</tr>
</tbody>
</table>

- Schemes to compare: HAD-EDG & HAD-WCG
- Bursting and sparse traces: $R^u$ and $R^l$
- Implemented using RTC ToolBox
- Simulated on 1.7 GHz processor
Average Idle Power Consumption (Watt)

- Idle power is reduced for both traces $R^u$ and $R^l$. 

$R^u$ and $R^l$ refer to different sets of traces or conditions, indicating a comparison between two sets of data.
EDG activation is varied according to the traces
WCG activation is affected by the deadline
Both schemes require a small computation time

The increment for longer relative deadline is small
Extend online algorithms which adaptively control the on/off of a device for multiple event streams with

- preemptive fixed-priority scheduling
- preemptive earliest-deadline-first scheduling

Guarantee hard real-time requirements with respect to both timing and backlog constraints

Experiments prove the effectiveness of the algorithms
The principle is to deactivate the device only when energy saving is possible.
The principle is to deactivate the device only when energy saving is possible.
The principle is to deactivate the device only when energy saving is possible.
The principle is to deactivate the device only when energy saving is possible.
Deactivation Algorithm (HAD)

The principle is to deactivate the device only when energy saving is possible.

\[ t_\epsilon \leftarrow \min_{t > t^T} t \text{ such that } \bar{\alpha}_1^u(t - t^T, t^T) > 0 \]

\[ \tau^T = \max \{ \tau : \text{bdf}(\Delta, \tau) \geq \alpha_1^u(\Delta - D_1, t_\epsilon) \} \]

\[ \delta^T = \max \{ 0, \min \{ \delta : \alpha_1^u(\Delta, t_\epsilon) - \text{bdf}(\Delta, \tau^T - \delta) \leq Q \cdot w_1, \forall \Delta \} \} \]
Activiation Algorithms

- **Worst Case Greedy (WCG) Algorithm**
  - Time triggered reevaluation
  - Suitable for bursty event arrival

- **Event Driven Greedy (EDG) Algorithm**
  - Event triggered reevaluation
  - Suitable for sparse event arrival
Worst Case Greedy Algorithm for Activation

on

turn off
Worst Case Greedy Algorithm for Activation
Worst Case Greedy Algorithm for Activation
Worst Case Greedy Algorithm for Activation
Worst Case Greedy Algorithm for Activation

\[ \tau^\perp - \delta^\perp \]

\[ t^\perp \]

\[ e_0 \]

\[ t^\perp \]

\[ \tau^\perp \]
Worst Case Greedy Algorithm for Activation

\[ \tau \perp - \delta \tau \perp - \delta \perp \]

\[ t_{\perp} \]

\[ \theta_0 \]

\[ \text{turn off} \]

\[ \text{turn on} \]
Worst Case Greedy Algorithm for Activation
Worst Case Greedy Algorithm for Activation

\[ \tau - \delta \tau - \delta \]

\[ t^\perp \]

\[ e_0 \]

\[ \text{turn off} \]

\[ \text{on} \]
Worst Case Greedy Algorithm for Activation
Worst Case Greedy Algorithm for Activation
Worst Case Greedy Algorithm for Activation

don $\tau$ $\perp$ $- \delta$ $\perp$ $t$ $\perp$ $- \delta$ $\perp$ $t$ $\perp$ $- \delta$ $\perp$

$\tau$ $\perp$ $- \delta$ $\perp$ $- \delta$ $\perp$ $- \delta$ $\perp$

$\epsilon_0$

turn off $\quad t \perp \quad t \perp \quad t \perp \quad$ turn on
Worst Case Greedy Algorithm for Activation

\[ \beta^A(\Delta) = \alpha_1^u(\Delta - D_1, t^\perp) + w_1 \cdot B_1(\Delta, t^\perp) \]

\[ \tau^\perp = \max \{ \tau : \text{bdf}(\Delta, \tau) \geq \beta^A(\Delta) \} \]

\[ \delta^\perp = \max \left\{ 0, \min \{ \delta : \alpha_1^u(\Delta, t^\perp) - \text{bdf}(\Delta, \tau^\perp - \delta) \leq (Q - |E(t^\perp)|) \cdot w_1, \forall \Delta \} \right\} \]
Event Driven Greedy Algorithm for Activation
Event Driven Greedy Algorithm for Activation
Event Driven Greedy Algorithm for Activation

\[ e_0 \]

\[ t \]

\[ t' \]
Event Driven Greedy Algorithm for Activation
Event Driven Greedy Algorithm for Activation
Event Driven Greedy Algorithm for Activation

turn off

e_0 e_1

t \perp t' \perp

turn on

\[ t' \]

[Image 280x5 to 356x25]
Event Driven Greedy Algorithm for Activation
Event Driven Greedy Algorithm for Activation

\[ t_0 \quad t_1 \quad t_2 \]

\[ \text{turn off} \quad \text{on} \]

\[ e_0 \quad e_1 \quad e_2 \]

\[ \text{turn on} \]

\[ \text{turn off} \]

\[ t_0 \quad t_1 \quad t_2 \]
Event Driven Greedy Algorithm for Activation

don

turn off

e_0
e_1e_2

t

t'

turn on

off
Event Driven Greedy Algorithm for Activation

\[ \tau^\perp - \delta^\perp \]
Event Driven Greedy Algorithm for Activation

diagram showing event-driven activation

turn off $t_0$, $e_1$, $e_2$

on $t'$

$\tau - \delta$
Event Driven Greedy Algorithm for Activation

\[ \beta^A(\Delta) = \alpha^u_1(\Delta - D_1, t') + w_1 \cdot B'_1(\Delta, t') \]

\[ \tau^\perp = \max\{\tau : bdf(\Delta, \tau) \geq \beta^A(\Delta)\} \]

\[ \delta^\perp = \max\left\{0, \min\{\delta : \alpha^u_1(\Delta, t') - bdf(\Delta, \tau^\perp - \delta) \right\} \]

\[ \leq (Q - |E(t^\perp)| - \bar{\alpha}'_1(\epsilon)) \cdot w_1, \forall \Delta \} \]