Fast Worst-Case Peak Temperature Evaluation for Real-Time Applications on Multi-Core Systems

Lars Schor, Iuliana Bacivarov, Hoeseok Yang, Lothar Thiele
Computer Engineering and Networks Laboratory
ETH Zurich, 8092 Zurich, Switzerland
firstname.lastname@tik.ee.ethz.ch

***Abstract*—The reliability of multi-core systems is nowadays threatened by high chip temperatures leading to long-term reliability concerns and short-term functional errors. In real-time systems, high chip temperatures are even adherent to potential deadline violations. Therefore, correct functionality can only be guaranteed if the worst-case peak temperature is incorporated in real-time analysis. However, calculating the peak temperature of hundreds of design alternatives during design space exploration is time-consuming. In this paper, we address this challenge and present a fast analytic method to calculate a non-trivial upper bound on the maximum temperature of a multi-core real-time system with non-deterministic workload. The considered thermal model is able to address various thermal effects like heat exchange between neighboring cores and temperature-dependent leakage power. Finally, the proposed method is applied to a multi-core ARM platform to validate its efficiency and accuracy.**

***Index Terms*—real-time systems; multi-core systems; compositional analysis; worst-case peak temperature; thermal analysis**

I. Introduction

Nowadays, the reliability of embedded multi-core systems is threatened by increased temperatures induced by the high utilization of all on-chip components. This may not only lead to long-term reliability concerns and short-term functional errors [1], but exceeding the threshold temperature could even decrease the performance. Therefore, providing guarantees on maximum temperature is nowadays as important as functional correctness and timeliness.

Reactive thermal management techniques such as [2], [3] are typically used to address thermal issues. For example, dynamic voltage and frequency scaling and stop-go scheduling are evaluated in [2], and a temperature aware frequency assignment is proposed in [3]. Causing substantial performance degradation or leading to expensive run-time overhead, reactive thermal management is often undesirable in today's embedded systems, in particular when tackling real-time constraints. The alternative is to already incorporate temperature analysis at design-time.

Thermal aware task allocation and scheduling for MPSoC platforms are explored in [1], [4], where temperature analysis is performed by either simulation [5] or steady-state analysis [6]. However, due to the complexity of today's systems, it is difficult to identify corner cases that actually lead to the maximum temperature of the system. Consequently, simulation-based thermal analysis may lead to undesired underestimations of the maximum temperature.

When considering real-time systems, the use of reactive thermal management techniques may even lead to violation of real-time constraints. In particular, real-time deadlines can only be guaranteed if the worst-case peak temperature of the system is incorporated into real-time analysis, at design-time. To this end, Rai et al. [7] proposed a method to calculate the worst-case peak temperature of a single-node system with non-deterministic workload. By incorporating the heat transfer among neighboring components into thermal analysis, the method is extended in [8] to multi-core systems. As the proposed method uses linear search to calculate a tight bound on the worst-case peak temperature, its evaluation time is too long for the design space exploration of multi-core systems with tens of processing components.

This paper addresses this challenge and describes a method to calculate an upper bound on the maximum temperature of a multi-core system with a time complexity that only depends on the number of processing components. The contributions of this paper can be summarized as follows:

- The considered system is formally described along with an overview of worst-case peak temperature analysis.
- A mathematical expression for a non-trivial upper bound on the worst-case peak temperature of a multi-core system is derived that has time complexity $O(n^2)$, where n is proportional to the number of processing components.
- The proposed method is used to explore the temperature distribution of a video-decoding application running on a 25-core ARM platform.

II. System Models

This section introduces the formal models to analyze a real-time application on a multi-core system.

Notation: Bold characters are used for vectors and matrices and non-bold characters for scalars. For example, $\mathbf{T}$ denotes a vector whose k-th element is T_k.

A. Computational Model

The considered computational model is expressed using abstractions in real-time calculus [9]. Events with a total workload of $R_\ell(s,t)$ time units arrive at component ℓ in time interval $[s,t)$. The arrival curve α_ℓ upper bounds all possible cumulative workloads:

$$R_\ell(s,t) \leq \alpha_\ell(t-s) \quad \forall s < t \tag{1}$$

with $\alpha_\ell(0) = 0$. Each event has a constant workload of Δ_ℓ^A time units.

We suppose that the processing components are work-conserving. In other words, they will be in 'active' mode as long as there are events in their ready queue. The accumulated computing time $Q_\ell(s,t)$ describes the amount of time units that component ℓ is spending to process an incoming workload of $R_\ell(s,t)$ time units. It is upper bounded by $\gamma_\ell(\Delta)$ for all intervals of length $\Delta < t$:

$$Q_\ell(t-\Delta,t) \leq \gamma_\ell(\Delta) = \inf_{0\leq\lambda\leq\Delta} \{(\Delta-\lambda)+\alpha_\ell(\lambda)\}. \quad (2)$$

For any fixed s with $s < t$, the accumulated computing time $Q_\ell(s,t)$ is monotonically increasing and has either slope 1 or 0. When the slope is 1, the component is in 'active' mode, i.e., it is processing events. When the slope is 0, the component is 'idle', i.e., sleep mode. We express the processing mode by the mode function:

$$S_\ell(t) = \frac{\mathrm{d}Q_\ell(s,t)}{\mathrm{d}t} \in \{0,1\}. \quad (3)$$

Similar to [8], we assume that an event stream can be specified by a period p_ℓ, a jitter j_ℓ, and a minimum inter-arrival distance d_ℓ [10]. Therefore, the computing time is characterized by the length of the first active interval b_ℓ, also called *burst*, the length of every other active interval Δ_ℓ^A, and the length of every idle interval Δ_ℓ^I. While b_ℓ and Δ_ℓ^A are constant for the considered computational model, we also assume for computational simplicity, that the upper bound on the accumulated computing time is selected such that all non-increasing intervals have the same length.

B. Thermal Model

The considered thermal model of a multi-core system describes the temperature evolution by means of an equivalent RC circuit [4], [5]. The vertical layout of the chip is modeled by four layers, namely the heat sink, heat spreader, thermal interface, and silicon die. Each layer is divided into a set of blocks according to the architecture-level units, i.e., processing components. As additional nodes are introduced in the heat spreader and heat sink layers, a multi-core system with χ processing components is modeled by $n = 4 \cdot \chi + 12$ nodes.

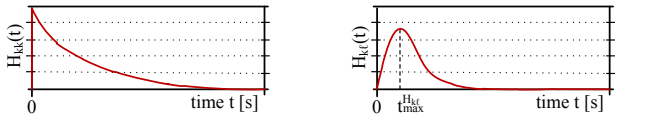
In particular, the n-dimensional temperature vector $\mathbf{T}(t)$ at time t is described by a set of first-order differential equations:

$$\mathbf{C} \cdot \frac{\mathrm{d}\mathbf{T}(t)}{\mathrm{d}t} = \left(\mathbf{P}(t) + \mathbf{K} \cdot \mathbf{T}^{\mathrm{amb}}\right) - (\mathbf{G}+\mathbf{K}) \cdot \mathbf{T}(t) \quad (4)$$

with $\mathbf{C}$ the thermal capacitance matrix, $\mathbf{G}$ the thermal conductance matrix, $\mathbf{K}$ the thermal ground conductance matrix, $\mathbf{P}$ the power dissipation vector, and $\mathbf{T}^{\mathrm{amb}} = T^{\mathrm{amb}} \cdot [1,\ldots,1]'$ the ambient temperature vector. The initial temperature vector is denoted as $\mathbf{T}^0$ and the system is assumed to start at time $t^0 = 0$.

A linear dependency of power dissipation on temperature [4], [11] is assumed due to leakage power:

$$\mathbf{P}(t) = \boldsymbol{\phi} \cdot \mathbf{T}(t) + \boldsymbol{\psi}(t) \quad (5)$$



(a) Self-impulse response $H_{kk}(t)$. (b) General impulse response $H_{k\ell}(t)$.

Fig. 1. Impulse responses.

where $\boldsymbol{\phi}$ is a diagonal matrix with constant coefficients, and $\boldsymbol{\psi}$ a vector. Finally, the state-space representation of the thermal model is expressed by:

$$\frac{\mathrm{d}\mathbf{T}(t)}{\mathrm{d}t} = \mathbf{A} \cdot \mathbf{T}(t) + \mathbf{B} \cdot \mathbf{u}(t) \quad (6)$$

with the input vector $\mathbf{u}(t) = \boldsymbol{\psi}(t) + \mathbf{K} \cdot \mathbf{T}^{\mathrm{amb}}$, $\mathbf{A} = -\mathbf{C}^{-1} \cdot (\mathbf{G}+\mathbf{K}-\boldsymbol{\phi})$, and $\mathbf{B} = \mathbf{C}^{-1}$. As the thermal system is linear and time-invariant, the temperature of node k is:

$$T_k(t) = T_k^{\mathrm{init}}(t) + \sum_{\ell=1}^{n} T_{k,\ell}(t). \quad (7)$$

$\mathbf{T}^{\mathrm{init}}(t) = e^{\mathbf{A}\cdot t} \cdot \mathbf{T}^0$ and $T_{k,\ell}(t)$ is the convolution of input u_ℓ and $H_{k\ell}$, i.e., the impulse response between nodes ℓ and k:

$$T_{k,\ell}(t) = \int_0^t H_{k\ell}(\xi) \cdot u_\ell(t-\xi)\,\mathrm{d}\xi. \quad (8)$$

The input u_ℓ depends on the processing mode of the architectural unit corresponding to node ℓ:

$$u_\ell(t) = S_\ell(t) \cdot u_\ell^a + (1 - S_\ell(t)) \cdot u_\ell^i \quad (9)$$

with $u_\ell^a = \psi_\ell^a + K_{\ell\ell} \cdot T^{\mathrm{amb}}$ if $S_\ell(t) = 1$ and $u_\ell^i = \psi_\ell^i + K_{\ell\ell} \cdot T^{\mathrm{amb}}$ if $S_\ell(t) = 0$.

Similar to [8], we assume that $H_{k\ell}(t)$ is a non-negative unimodal function that has its maximum at time $t_{\max}^{H_{kj}}$, see Fig. 1 for an illustration.

III. Thermal Analysis

In this section, we start by presenting some key results from peak temperature analysis, and then introduce a novel method to calculate a non-trivial upper bound on the maximum temperature without simulating the transient temperature evolution.

A. Peak Temperature Analysis

The worst-case peak temperature T_S^* of a multi-core platform with n nodes is the maximum temperature of all nodes:

$$T_S^* = \max\left(T_1^*,\ldots,T_n^*\right) \quad (10)$$

where T_k^* is the worst-case peak temperature of node k. The temperature $T_k^*(\tau)$ at a certain observation time τ is found by simulating the system with the set of critical accumulated computing times $\mathbf{Q}^{\{k\}} = \left[Q_1^{\{k\}}(0,\Delta),\ldots,Q_n^{\{k\}}(0,\Delta)\right]'$, where $\{k\}$ indicates that $\mathbf{Q}^{\{k\}}$ leads to the worst-case peak temperature of node k. The remaining question is how to calculate the set of critical accumulate computing times $\mathbf{Q}^{\{k\}}$.

First, we note that the critical accumulated computing time $Q_\ell^{\{k\}}(0,\Delta)$ of node ℓ can be calculated independently of all other accumulated computing times [8]. In particular, the

Algorithm. 1. Calculation of the critical accumulated computing time function $Q_\ell^{\{k\}}(0,\Delta)$ for all $0 \le \Delta \le \tau$ with v_ℓ defined in (11).

Input: $b_\ell, \Delta_\ell^I, \Delta_\ell^A, p_\ell, \widetilde{t}_{\max}^{H_{kj}}, \tau, H_{k\ell}$
Output: $Q_\ell^{\{k\}}(0,\Delta)$

for all $t^{(r)}$ in $[\widetilde{t}_{\max}^{H_{kj}}, \widetilde{t}_{\max}^{H_{kj}} + b_\ell - \Delta_\ell^A]$ **do** ▷*find position of the burst*
  for all $t^s \in [0, \Delta_\ell^I]$ **do** ▷*find gap btw burst and suc. active interval*
    $t^{(l)} = t^{(r)} - b_\ell + \Delta_\ell^A$
    $S_\ell(t) = \begin{cases} 1 & t \in [t^{(r)} - b_\ell + \Delta_\ell^A, t^{(r)}) \\ 0 & \text{otherwise} \end{cases}$
    for $i = 1$ to $\left\lceil \frac{\tau - t^{(r)}}{p_\ell} \right\rceil$ **do** ▷*make trace for* $t > t^{(r)}$
      $S_\ell(t) = S_\ell(t) + v_\ell\left(t, t^s + t^{(r)} + (i-1)\cdot p_\ell\right)$
    end for
    for $i = 1$ to $\left\lceil \frac{t^{(l)}}{p_\ell} \right\rceil$ **do** ▷*make trace for* $t < t^{(l)}$
      $S_\ell(t) = S_\ell(t) + v_\ell\left(t, t^s + t^{(l)} - i\cdot p_\ell\right)$
    end for
    $\Upsilon = \int_0^t S_\ell(\xi)\cdot H_{k\ell}(t-\xi)\,\mathrm{d}\xi$
    if $\Upsilon > \Upsilon^*$ **then** ▷Υ *comparison*
      $\Upsilon^* = \Upsilon, \quad S_\ell^{\{k\}} = S_\ell$
    end if
  end for
end for
$Q_\ell^{\{k\}}(0,\Delta) = \int_0^\Delta S_\ell^{\{k\}}(\xi)\,\mathrm{d}\xi$ for all $0 \le \Delta \le \tau$

problem is equivalent to finding the accumulated computing time $Q_\ell^{\{k\}}(0,\Delta)$ that maximizes $T_{k,\ell}$ defined as in (8). For simplicity, we define $\widetilde{t}_{\max}^{H_{kj}} = \tau - t_{\max}^{H_{kj}}$, where τ is the predefined observation time of the peak temperature. Furthermore, we introduce the auxiliary function v_ℓ of node ℓ, which is, starting at time t^s, one for Δ_ℓ^A time units:

$$v_\ell(t,t^s) = \begin{cases} 1 & 0 \le t^s \le t \le \min(t^s + \Delta_\ell^A, \tau) \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

The next theorem follows from the results of [8] and provides a method to calculate the critical accumulated computing time $\mathbf{Q}^{\{k\}}$ leading to worst-case peak temperature T_k^* of node k.

Theorem 1. *Suppose that the accumulated computing time function* $\mathbf{Q}^{\{k\}}(0,\Delta) = \left[Q_1^{\{k\}}(0,\Delta), \ldots, Q_n^{\{k\}}(0,\Delta)\right]'$ *for all* $0 \le \Delta \le \tau$ *with* $Q_\ell^{\{k\}}(0,\Delta)$ *constructed by Algorithm 1 leads to* $T_k^*(\tau)$ *at time* τ. *When the scheduler is work-conserving,* $T_k^*(\tau)$ *is an upper bound on the highest possible value of temperature* $T_k(\tau)$ *at time* τ. *Furthermore, when* $(\mathbf{T}^\infty)^i$ *is the steady-state temperature vector if all nodes are in 'idle' mode,* $T_k^*(\tau) \ge T_k(t)$ *for all* $0 \le t \le \tau$ *and any set of feasible workload traces with the same initial temperature vector* $\mathbf{T}^0 \le (\mathbf{T}^\infty)^i$.

Algorithm 1 calculates the critical accumulated computing time $Q_\ell^{\{k\}}$ by altering both the position of the burst and the gap between burst and first successive active interval, see Fig. 2 for an illustration of the algorithm. Then, $Q_\ell^{\{k\}}$ is the computing time that maximizes the sum of all areas below $H_{k\ell}$ where the node is in 'active' processing mode. Afterwards, the peak temperature $T_k^*(\tau)$ of node k is obtained by simulating the system with $\boldsymbol{Q}^{\{k\}}$.

Calculating an upper bound on the maximum temperature T_S^* has time complexity $O(n^2 * m)$ with n the number of nodes. The factor m reflects the time to execute Algorithm 1 and is

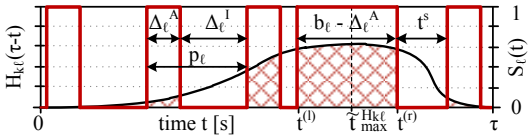


Fig. 2. Sketch of Algorithm 1 by illustrating the convolution between $H_{kl}(\tau - t)$ and $S_\ell(t)$.

inversely proportional to the selected time step. Increasing the time step can improve the execution time, but might lead to a reduced accuracy.

B. Fast Temperature Evaluation

As indicated, Algorithm 1 might by time-consuming and hence not suited for design space exploration. Therefore, we first derive an analytical expression for the accumulated computing time that leads to a non-trivial upper bound on the peak temperature. Afterwards, we use the obtained computing time to propose a novel mathematical expression for an upper bound on the maximum temperature.

The first lemma simplifies Algorithm 1 so that it is constant time. The resulting upper bound on the maximum temperature at observation time τ, $\widehat{T}_k^*(\tau)$, is not smaller than the worst-case peak temperature of the system. $\widehat{\mathbf{Q}}^{\{k\}}$ denotes the critical accumulated computing time that leads to $\widehat{T}_k^*(\tau)$.

Theorem 2. *Suppose that the accumulated computing time function* $\widehat{\mathbf{Q}}^{\{k\}}(0,\Delta) = \left[\widehat{Q}_1^{\{k\}}(0,\Delta), \ldots, \widehat{Q}_n^{\{k\}}(0,\Delta)\right]'$ *for all* $0 \le \Delta \le \tau$ *with:*

$$\widehat{Q}_\ell^{\{k\}}(0,\Delta) = \begin{cases} \gamma(\widetilde{t}_{\max}^{H_{kj}}) - \gamma(\widetilde{t}_{\max}^{H_{kj}} - \Delta) & 0 \le \Delta < \widetilde{t}_{\max}^{H_{kj}} \\ \gamma(\widetilde{t}_{\max}^{H_{kj}}) + \gamma(\Delta - \widetilde{t}_{\max}^{H_{kj}}) & \widetilde{t}_{\max}^{H_{kj}} \le \Delta < \tau \end{cases} \tag{12}$$

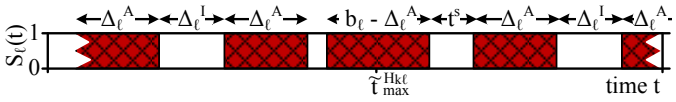
leads to $\widehat{T}_k^*(\tau)$ *at time* τ. *When the scheduler is work-conserving,* $\widehat{T}_k^*(\tau)$ *is an upper bound on the highest value of temperature* $T_k(\tau)$ *at time* τ. *Furthermore, when* $(\mathbf{T}^\infty)^i$ *is the steady-state temperature vector if all nodes are in 'idle' mode,* $\widehat{T}_k^*(\tau) \ge T_k(t)$ *for all* $0 \le t \le \tau$ *and any set of feasible workload traces with the same initial temperature vector* $\mathbf{T}^0 \le (\mathbf{T}^\infty)^i$.

Proof: We will prove this theorem by translating $\mathbf{Q}_\ell^{\{k\}}$ calculated by Algorithm 1 into $\widehat{\mathbf{Q}}_\ell^{\{k\}}$ calculated by (12) and show that in every step, the temperature will not decrease. To this end, we observe that the temperature does not decrease if the amount of 'active' time units is either increased or shifted closer to $\widetilde{t}_{\max}^{H_{kj}}$ [8].

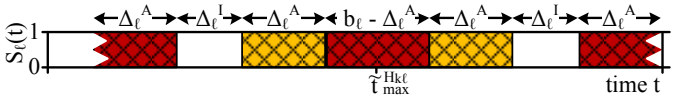
In the first step, the precedent and successive active intervals of the burst are moved to the burst so that the node is continuously active for $b_\ell + \Delta_\ell^A$ time units, compare Figs. 3(a) and 3(b) for an illustration. The second step makes the search for the position of the burst obsolete. To this end, the length of the burst is extended so that it covers all possible positions of the burst, i.e., $\widehat{S}_\ell(t) = 1$ for all $t \in [\widetilde{t}_{\max}^{H_{kj}} - b_\ell + \Delta_\ell^A, \widetilde{t}_{\max}^{H_{kj}} + b_\ell - \Delta_\ell^A]$, see Fig. 3(c) for an illustration of this step.

Algorithm 2 is the result of these two translations. One can readily prove that in both steps, the amount of 'active' time units is either increased or shifted closer to $\widetilde{t}_{\max}^{H_{kj}}$. Finally, we note that (12) is equivalent to Algorithm 2, and therefore $\widehat{T}_k^*(\tau) \ge T_k^*(\tau)$. ■

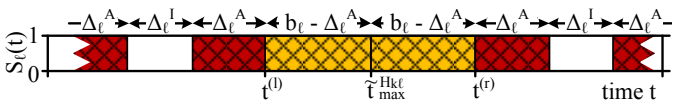
As Algorithm 2 is constant time, computing an upper bound on the maximum temperature of a multi-core system has



(a) Starting position.



(b) Translation step 1: the precedent and successive active intervals of the burst are moved to the burst.



(c) Translation step 2: the length of the burst is extended so that it covers all possible positions of the burst.

Fig. 3. Translation of Q_ℓ^* into $\widehat{Q}_\ell^*$.

time complexity $O(n^2)$, where n is the number of nodes. As illustrated in Fig. 3(c), the system continuously switches between active and idle mode except during the burst. Next, we show that calculating the peak temperature can further be simplified by running the processing component with constant slope $\delta_\ell = \frac{\Delta_\ell^A}{\Delta_\ell^A+\Delta_\ell^I}$ for all time units except during the burst. This lemma provides the foundation for the main theorem of this section.

Lemma 3. *Suppose that the mode function:*

$$\breve{S}_\ell(t) = \begin{cases} 1 & \widetilde{t}^{H_{kj}}_{\max} - b_\ell \le t \le \widetilde{t}^{H_{kj}}_{\max} + b_\ell \\ \delta_\ell & \textit{otherwise} \end{cases} \tag{13}$$

with utilization $\delta_\ell = \frac{\Delta_\ell^A}{\Delta_\ell^A+\Delta_\ell^I}$ leads to $\breve{T}^*_{k,\ell}(\tau)$ at time τ. When the scheduler is work-conserving, $\breve{T}^*_{k,\ell}(\tau)$ is an upper bound on the highest value of $T_{k,\ell}(\tau)$ at time τ.

Proof: Rewriting (8) with (9) leads to:

$$T_{k,\ell}(t) = u_\ell^i \cdot \int_0^t H_{k\ell}(t-\xi)\,\mathrm{d}\xi + (u_\ell^a - u_\ell^i)\cdot \int_0^t S_\ell(\xi)\cdot H_{k\ell}(t-\xi)\,\mathrm{d}\xi.$$

As we know from Theorem 2 that $\widehat{S}(t) = \frac{\mathrm{d}\widehat{Q}_\ell^{\{k\}}(0,\Delta)}{\mathrm{d}t}$ leads to $\widehat{T}_{k,\ell}(\tau)$ with $\widehat{T}_{k,\ell}(\tau) \ge T_{k,\ell}(\tau)$, we have to show that $\breve{T}_{k,\ell}(\tau) \ge \widehat{T}_{k,\ell}(\tau)$:

$$\begin{aligned}
&\left(\breve{T}_{k,\ell}(\tau) - \widehat{T}_{k,\ell}(\tau)\right) / \left(u_\ell^a - u_\ell^i\right) \\
&= \int_0^\tau \breve{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi - \int_0^\tau \widehat{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \\
&= \int_0^{\theta^{(l)}} \breve{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi - \int_0^{\theta^{(l)}} \widehat{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \\
&\quad + \int_{\theta^{(r)}}^\tau \breve{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi - \int_{\theta^{(r)}}^\tau \widehat{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi.
\end{aligned}$$

where we used that $\breve{S}_\ell(t) = \widehat{S}_\ell(t) = 1$ for $t \in [\theta^{(l)}, \theta^{(r)}]$ with $\theta^{(l)} = \widetilde{t}^{H_{kj}}_{\max} - b_\ell$ and $\theta^{(r)} = \widetilde{t}^{H_{kj}}_{\max} + b_\ell$. By rewriting the integral from 0 to $\theta^{(l)}$ as a sum, we get:

$$\begin{aligned}
&\int_0^{\theta^{(l)}} \left(\breve{S}_\ell(\xi) - \widehat{S}_\ell(\xi)\right)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \\
&= \textstyle\sum_{i=0}^{\varrho} \left(\int_{\theta^{(l)}-(i+1)\cdot p_\ell}^{\theta^{(l)}-i\cdot p_\ell} \left(\breve{S}_\ell(\xi) - \widehat{S}_\ell(\xi)\right)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi\right)
\end{aligned}$$

where ϱ is selected so that $p_\ell\cdot(\varrho-1) \le \theta^{(l)} \le p_\ell\cdot\varrho$. In particular, $\widehat{S}(t) = 0$ in $t \in [\theta^{(l)} - i\cdot p_\ell - \Delta_\ell^I, \theta^{(l)} - i\cdot p_\ell]$, $\widehat{S}(t) = 1$ in

Algorithm 2. Calculation of the critical accumulated computing time function $\widehat{Q}_\ell^{\{k\}}(0,\Delta)$ for all $0 \le \Delta \le \tau$.

Input: $b_\ell, \Delta_\ell^I, \Delta_\ell^A, p_\ell, \widetilde{t}^{H_{kj}}_{\max}, \tau$
Output: $\widehat{Q}_\ell^{\{k\}}(0,\Delta)$

$t^{(r)} = \widetilde{t}^{H_{kj}}_{\max} + b_\ell - \Delta_\ell^A$, $t^{(l)} = \widetilde{t}^{H_{kj}}_{\max} - b_\ell + \Delta_\ell^A$
$\widehat{S}_\ell^{\{k\}}(t) = \begin{cases} 1 & t \in [t^{(l)}, t^{(r)}] \\ 0 & \text{otherwise} \end{cases}$ ▷*make extended burst*
for $i = 1$ to $\left\lceil \frac{\tau - t^{(r)}}{p_\ell} \right\rceil$ **do** ▷*make trace for $t > t^{(r)}$*
  $\widehat{S}_\ell^{\{k\}}(t) = \widehat{S}_\ell^{\{k\}}(t) + v_\ell\left(t, t^{(r)} + (i-1)\cdot p_\ell\right)$
end for
for $i = 1$ to $\left\lceil \frac{t^{(l)}}{p_\ell} \right\rceil$ **do** ▷*make trace for $t < t^{(l)}$*
  $\widehat{S}_\ell^{\{k\}}(t) = \widehat{S}_\ell^{\{k\}}(t) + v_\ell\left(t, \Delta_\ell^I + t^{(l)} - i\cdot p_\ell\right)$
end for
$\widehat{Q}_\ell^{\{k\}}(0,\Delta) = \int_0^\Delta \widehat{S}_\ell^{\{k\}}(\xi)\,\mathrm{d}\xi$ for all $0 \le \Delta \le \tau$

$t \in [\theta^{(l)} - (i+1)\cdot p_\ell, \theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A]$, and $\theta^{(l)} - i\cdot p_\ell - \Delta_\ell^I = \theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A$, see Fig. 4 for illustration. Then, we find:

$$\begin{aligned}
&\int_{\theta^{(l)}-(i+1)\cdot p_\ell}^{\theta^{(l)}-i\cdot p_\ell} \left(\breve{S}_\ell(\xi) - \widehat{S}_\ell(\xi)\right)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \\
&= \delta_\ell \cdot \int_{\theta^{(l)}-(i+1)\cdot p_\ell}^{\theta^{(l)}-i\cdot p_\ell + \Delta_\ell^A} H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \\
&\quad - (1-\delta_\ell)\cdot \int_{\theta^{(l)}-(i+1)\cdot p_\ell}^{\theta^{(l)}-i\cdot p_\ell - \Delta_\ell^I} H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi
\end{aligned}$$

where we subtracted the two integrals in the interval $[\theta^{(l)} - (i+1)\cdot p_\ell, \theta^{(l)} - i\cdot p_\ell - \Delta_\ell^I]$. Next, we lower bound the value between $\theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A$ and $\theta^{(l)} - i\cdot p_\ell$ by means of $H_{k\ell}\left(\tau - \left(\theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A\right)\right)$, and upper bound the value between $\theta^{(l)} - (i+1)\cdot p_\ell$ and $\theta^{(l)} - i\cdot p_\ell - \Delta_\ell^I$ by means of $H_{k\ell}\left(\tau - \left(\theta^{(l)} - i\cdot p_\ell - \Delta_\ell^I\right)\right) = H_{k\ell}\left(\tau - \left(\theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A\right)\right)$, as well:

$$\begin{aligned}
&\int_{\theta^{(l)}-(i+1)\cdot p_\ell}^{\theta^{(l)}-i\cdot p_\ell} \left(\breve{S}_\ell(\xi) - \widehat{S}_\ell(\xi)\right)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \\
&\ge \delta_\ell \cdot \Delta_\ell^I \cdot H_{k\ell}\left(\tau - \left(\theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A\right)\right) \\
&\quad - (1-\delta_\ell)\cdot \Delta_\ell^A \cdot H_{k\ell}\left(\tau - \left(\theta^{(l)} - (i+1)\cdot p_\ell + \Delta_\ell^A\right)\right) = 0
\end{aligned}$$

where we used the fact that $\delta_\ell = \frac{\Delta_\ell^A}{\Delta_\ell^A+\Delta_\ell^I}$ and $\delta_\ell\cdot\Delta_\ell^I - (1-\delta_\ell)\cdot\Delta_\ell^A = 0$. Similarly, we can show that:

$$\int_{\theta^{(r)}}^\tau \breve{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi - \int_{\theta^{(r)}}^\tau \widehat{S}_\ell(\xi)\cdot H_{k\ell}(\tau-\xi)\,\mathrm{d}\xi \ge 0$$

and therefore, $\breve{T}_{k,\ell}(\tau) - \widehat{T}_{k,\ell}(\tau) / \left((u_\ell^a - u_\ell^i)\right) \ge 0$. ∎

Based on Lemma 3, we will present the main result of this section. The following theorem provides a mathematical expression to calculate a non-trivial upper bound on the maximum temperature $\breve{T}^*_k(\tau)$ of node k. Finally, an upper bound on the maximum temperature of the system can be obtained according to (10) by calculating the maximum of all individual upper bounds.

Theorem 4. *Suppose that $T_k(t)$ is the temperature of node k at time instant t for a set of workload functions $\mathbf{R}(s,t)$ that are bounded by the set of arrival curves $\boldsymbol{\alpha}$. When the scheduler is work-conserving, the following statements hold:*

- *The temperature:*

$$\breve{T}^*_k(\tau) = T_k^{\text{init}}(\tau) + \sum_{\ell=1}^{n} \breve{T}^*_{k,\ell}(\tau) \tag{14}$$

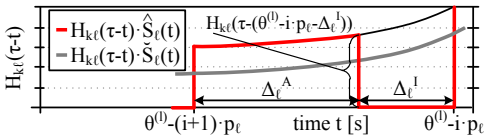


Fig. 4. Illustration of the proof of Lemma 3. The impulse response function $H_{k\ell}(t)$ is plotted for the interval $[\theta^{(l)} - (i+1)\cdot p_\ell, \theta^{(l)} - i\cdot p_\ell]$.

with:

$$\breve{T}^*_{k,\ell}(\tau) = \left(u^i_\ell + \delta_\ell \cdot \left(u^a_\ell - u^i_\ell\right)\right) \cdot \int_0^\tau H_{k\ell}(t-\xi)\,\mathrm{d}\xi + \left(u^a_\ell - u^i_\ell\right)\cdot(1-\delta_\ell)\cdot \int_{t^{\widetilde{H}_{kj}}_{\max}-b_\ell}^{t^{\widetilde{H}_{kj}}_{\max}+b_\ell} H_{k\ell}(t-\xi)\,\mathrm{d}\xi \tag{15}$$

and $\delta_\ell = \frac{\Delta^A_\ell}{\Delta^A_\ell + \Delta^I_\ell}$ *is an upper bound on the highest temperature of node* k *at time* τ*, i.e.,* $\breve{T}^*_k(\tau) \geq T_k(\tau)$.

- *In addition, if* $(\mathbf{T}^\infty)^i$ *is the steady-state temperature vector if all nodes are in 'idle' mode,* $\widehat{T}^*_k(\tau) \geq T_k(t)$ *for all* $0 \leq t \leq \tau$ *and any set of feasible workload traces with the same initial temperature vector* $\mathbf{T}^0 \leq (\mathbf{T}^\infty)^i$.

Proof: First, we rewrite (8) with (13) to derive (15). As Lemma 3 states that $\breve{T}^*_{k,\ell}(\tau) \geq \breve{T}_{k,\ell}(\tau)$ for all ℓ, and $T_k(t) = T^{\text{init}}_k(t) + \sum_{\ell=1}^n T_{k,\ell}(t)$, we get $\breve{T}^*_k(\tau) \geq T_k(\tau)$.

The second item is a simple consequence of Theorem 2. As $\breve{T}^*_k(\tau) \geq \widehat{T}^*_k(\tau)$, $\breve{T}^*_k(\tau) \geq T_k(t)$ for all $t \leq \tau$. ∎

Totally three different methods to calculate an upper bound on the maximum temperature have been presented in this section. The first method calculates the critical accumulated computing time by Algorithm 1 leading to the worst-case peak temperature T^*_k of node k. The second method calculates the accumulated computing time according to (12) leading to $\widehat{T}^*_k$, and the last method calculates an upper bound on the maximum temperature $\breve{T}^*_k$ of node k by the mathematical expression defined in (14). The relation between these three different bounds on the maximum temperature is as follows:

$$\breve{T}^*_k(\tau) \geq \widehat{T}^*_k(\tau) \geq T^*_k(\tau). \tag{16}$$

IV. Case Studies

In order to evaluate the proposed method, we extended the modular performance analysis (MPA) framework [12] with the ability to calculate the maximum temperature by the discussed algorithms.

A. System Description

We consider a multi-task motion JPEG (MJPEG) decoder application running on a reconfigurable homogeneous multi-core ARM platform with a variable number of processing components. The MJPEG decoder decompresses several frames in parallel by concurrently running tasks. In addition, a task splits up the input sequence into individual frames so that the frames can be distributed to the decompressing tasks. In addition, another task merges the decoded frames back into a stream. Each task is characterized by its best-case and worst-case execution demand that have been determined by simulating the MJPEG decoder on the MPARM virtual platform [13]. Fixed priority preemptive scheduling is used on all cores while a TDMA policy is employed on the shared bus that connects all processing components. Intermediate streams that cannot be represented by a period, a jitter, and a minimum interarrival distance are upper-bounded by the method presented in [14] and the observation time τ is set to five seconds.

TABLE I
Power dissipation parameters of node ℓ.

Parameter	Symbol	Value
Slope of power [W/K]	$\phi_{\ell\ell}$	0.023
Constant power in 'active' mode [W]	ψ^a_ℓ	8.684
Constant power in 'idle' mode [W]	ψ^i_ℓ	−5.512

The temperature-dependency of leakage power is addressed by linearizing the model described in [15]. Table I summaries the parameters of the considered power model defined in (5). As we consider a homogeneous platform, every component has the same power values. HotSpot [5] is used to calculate the thermal parameters of the platform, i.e., $\mathbf{C}$, $\mathbf{G}$, and $\mathbf{K}$ matrices, see [8] for the detailed thermal configuration. In all experiments, the traces start from the steady-state temperature in 'idle' mode, i.e., $\mathbf{T}^0 = (\mathbf{T}^\infty)^i$.

B. Efficiency and Accuracy

In the first case study, the MJPEG decoder consists of five tasks that run on three processing components, thus the thermal model has order 24. The application is driven by an input stream with a periodic invocation interval of 450 ms and a jitter of 600 ms.

To evaluate our method, both the time to calculate an upper bound on the maximum temperature and the quality of this bound are analyzed. To this end, we first calculate three different upper bounds on the maximum temperature:

1) The critical accumulated computing time is computed with Algorithm 1 leading to the worst-case peak temperature T^*_S.
2) The upper bound $\widehat{T}^*_S$ is calculated by simulating the system with the critical accumulated computing time defined in (12).
3) $\breve{T}^*_S$ is calculated according to (14).

Afterwards, we compare T^*_S, $\widehat{T}^*_S$, and $\breve{T}^*_S$ as well as the durations to calculate the bounds. The time increment of Algorithm 1 is set to 1 ms and all experiments have been performed on an Intel Core i7 M 620 processor with 4 GB of RAM. The peak temperatures and the durations to calculate these temperatures are listed in Table II for three different mapping configurations. $(\mathbf{T}^\infty)^a$ and $(\mathbf{T}^\infty)^i$ are the steady-state temperature vectors if all nodes are in 'active' and 'idle' mode, respectively.

Calculating $\breve{T}^*_S$ is on average 605 times faster than calculating T^*_S, but note that the execution time of Algorithm 1 depends on the selected time increment. Furthermore, the execution time depends on the actual mapping as shown in Table II(a). We quantify the accuracy of $\breve{T}^*_S$ by means of the worst-case peak temperature T^*_S. To this end, we introduce

TABLE II
EFFICIENCY AND ACCURACY COMPARISON.

(a) Duration of the compared temperature analysis methods.

	Mapping 1	Mapping 2	Mapping 3
duration to calculate T_S^*	30.609 s	29.444 s	34.844 s
duration to calculate $\widehat{T}_S^*$	0.443 s	0.454 s	1.554 s
duration to calculate $\breve{T}_S^*$	0.044 s	0.043 s	0.080 s

(b) Maximum temperature comparison for different analysis methods.

	Mapping 1	Mapping 2	Mapping 3
T_S^*	352.921 K	350.634 K	366.180 K
$\widehat{T}_S^*$	352.935 K	350.649 K	366.909 K
$\breve{T}_S^*$	352.937 K	350.651 K	366.910 K
$\max(\mathbf{T}^\infty)^a$	424.782 K	424.782 K	424.782 K
$\max(\mathbf{T}^\infty)^i$	310.714 K	310.714 K	310.714 K

the following notation of a relative error, that measures the normalized distance between $\breve{T}_S^*$ and T_S^*:

$$\text{error} = \frac{\breve{T}_S^* - T_S^*}{\max(\mathbf{T}^\infty)^a - \max(\mathbf{T}^\infty)^i}. \tag{17}$$

Applying this formula, the average error of our results is found to be only $0.22\,\%$. This confirms our approach to upper bound the peak temperature by (14) instead of using Algorithm 1 to calculate the critical accumulated computing time and then simulating the system with the critical computing time. Overall, calculating $\breve{T}_S^*$ instead of T_S^* is desirable in the design flow of embedded real-time systems as the three order of magnitude reduction in evaluation time enables a faster and more exhaustive design space exploration.

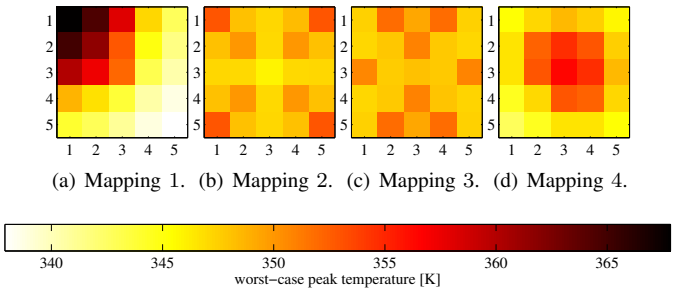
C. Temperature Distribution on a 25-Core Processor

Next, we consider a multi-core system with 25 processing components executing an MJPEG decoder with 10 tasks. The processing components are arranged in a grid with five rows and the corresponding thermal model has order 112. We will show that the temperature distribution, and thereby the worst-case peak temperature of the system, is affected by the assignment of tasks to processing components.

Figure 5 shows the worst-case peak temperature distribution of the system for four different mappings. In Fig. 5(a), the tasks are mapped onto components situated in the left top corner of the chip. Next, in Fig. 5(b), the tasks are distributed among components in all four corners. In Fig. 5(c), the tasks are distributed all over the chip, and finally, in Fig. 5(d), the tasks are only mapped onto components in the middle of the chip. The highest peak temperature occurs in Fig. 5(a) and the lowest one in Fig. 5(c). In particular, the difference between their worst-case peak temperatures is of about 16 K. This shows that the worst-case peak temperature can be reduced by spreading the workload over the chip. In this case, intermediate cores with no workload act like a passive cooling system and keep hot spots separated.

V. CONCLUSION

In this paper, we presented a fast thermal analysis method to calculate an upper bound on the maximum temperature of a real-time application with non-deterministic workload running on a multi-core system. The considered thermal model is able to address various thermal effects like temperature-dependent leakage power and heat exchange between neighboring cores to accurately model the thermal behavior of multi-core systems. In order to handle a broad range of uncertainties, task arrivals are modeled as periodic event streams with jitter and delay. Finally, the proposed method is applied to a 25-core ARM platform to validate the effectiveness of the proposed approach.



(a) Mapping 1. (b) Mapping 2. (c) Mapping 3. (d) Mapping 4.

Fig. 5. Worst-case peak temperature distribution for a 25-core processor when executing an MJPEG decoder application. The processing components are arranged in a grid with five rows.

ACKNOWLEDGMENTS

This work was supported by EU FP7 projects EURETILE and PRO3D, under grant numbers 247846 and 249776.

REFERENCES

[1] A. K. Coskun *et al.*, "Temperature Aware Task Scheduling in MPSoCs," in *Proc. DATE*, 2007.

[2] J. Donald *et al.*, "Techniques for Multicore Thermal Management: Classification and New Exploration," in *Proc. ISCA*, 2006.

[3] S. Murali *et al.*, "Temperature-Aware Processor Frequency Assignment for MPSoCs Using Convex Optimization," in *Proc. CODES+ISSS*, 2007.

[4] T. Chantem *et al.*, "Temperature-Aware Scheduling and Assignment for Hard Real-Time Applications on MPSoCs," in *Proc. DATE*, 2008.

[5] W. Huang *et al.*, "HotSpot: A Compact Thermal Modeling Methodology for Early-Stage VLSI Design," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 5, pp. 501–513, 2006.

[6] C.-Y. Yang *et al.*, "Energy-Efficient Real-Time Task Scheduling with Temperature-Dependent Leakage," in *Proc. DATE*, 2010.

[7] D. Rai *et al.*, "Worst-Case Temperature Analysis for Real-Time Systems," in *Proc. DATE*, 2011.

[8] L. Schor *et al.*, "Worst-Case Temperature Guarantees for Real-Time Applications on Multi-Core Systems," in *Proc. RTAS*, 2012.

[9] L. Thiele *et al.*, "Real-Time Calculus for Scheduling Hard Real-Time Systems," in *Proc. ISCAS*, 2000.

[10] R. Henia *et al.*, "System Level Performance Analysis - The SymTA/S Approach," *IEEE Proc. Computers and Digital Techniques*, vol. 152, no. 2, pp. 148–166, 2005.

[11] Y. Liu *et al.*, "Accurate Temperature-Dependent Integrated Circuit Leakage Power Estimation is Easy," in *Proc. DATE*, 2007.

[12] S. Chakraborty *et al.*, "Interface-Based Rate Analysis of Embedded Systems," in *Proc. RTSS*, 2006.

[13] L. Benini *et al.*, "MPARM: Exploring the Multi-Processor SoC Design Space with SystemC," *J. VLSI Signal. Proces.*, vol. 41, no. 2, pp. 169–182, 2005.

[14] S. Künzli *et al.*, "Combined Approach to System Level Performance Analysis of Embedded Systems," in *Proc. CODES+ISSS*, 2007.

[15] K. Skadron *et al.*, "Temperature-Aware Microarchitecture: Modeling and Implementation," *ACM Trans. Architec. Code Optim.*, vol. 1, no. 1, pp. 94–125, 2004.