Thermal Feasibility Analysis of Periodic Real-Time Tasks with Jitter

Devesh Bhimsaria *

Department of Electrical and Computer Engineering
University of Wisconsin Madison
Madison, Wisconsin 53706

July 30, 2014

Supervised by Dr. Parmeswaran Ramanathan

*Email: bhimsaria@wisc.edu

Acknowledgments

I would like to thank Dr. Parmeswaran (Parmesh) Ramanathan for his constant support and guidance, especially for his precious time he gave to me, even on the weekends, hours and hours. I am also very grateful to Professor for his amazing inputs to my research, for keeping me focused, and providing me freedom to pursue my interest. Much more than research, I want to thank him for making me feel comfortable in his lab, with him I never felt away from home.

I also thank all the present and past members of Ramanathan lab for helping me in my project, specially Rehan Ahmed.

My studies and research, all became possible because of lot of sacrifices made by my mother Urmila Bhimsaria. I thank her for being my first teacher, my inspiration and ideal. Her overwhelming love gave me strength to pursue my research interest. I also thank my wife Sakshi Sharma Bhimsaria for sacrificing her career for my research. Her care for me motivated me to work harder.

I would also like to thank the National Institutes of Health, Keck foundation and University of Wisconsin Madison for funding. I thank University of Wisconsin Madison, Department of Biochemistry and Department of Electrical and Computer Engineering for accepting me as an exchange student earlier and then as a graduate student.

Signed .....Date .....

Contents

Abstract

It is well-known that spatial and temporal variations in power density within a processor core creates thermal hot spots where the temperatures can exceed 100 °C. Such temperatures can degrade device reliability and adversely affect processor performance. This problem is expected to become even more acute in the coming years because average power densities are expected to double in the next decade. As a result, estimating the worst-case peak temperature for a given processor workload is an important problem.

This report presents an algorithm to estimate the worst-case peak temperature for a given real-time workload in a multi-core system. The challenge is to find the worst-case arrival pattern for the given real-time workload that results in the maximum peak temperature when the tasks are executed using a work-conserving scheduler. The proposed algorithm improves a solution from literature by relaxing some of its restrictive assumptions. In the cases where the assumptions are satisfied, the estimates from the proposed algorithm and the scheme from literature are the same. In all other cases, the estimates from the proposed algorithm are much less conservative and hence, more accurately estimate the achievable worst-case temperature.

Keywords- real-time systems; multi-core systems; multiple task set; worst-case peak temperature; thermal analysis

Chapter 1

Introduction

Until recently, thermal management was not a major issue in processor design and use. However, it has become a major issue in recent years because on-chip power densities have increased considerably with technology scaling. In fact, on-chip power densities are expected to double in the next decade. As density increases, spatio-temporal variations in power consumption create unacceptable temperature hot spots within the processor cores. Since higher temperatures cause more device failures, increased leakage currents, and lower performance, managing the run time thermal profile of a processor has become a major challenge.

A large number of recent papers have proposed thermal management schemes for both non-real-time and real-time applications [1][2][3][4][5]. Many of these schemes rely on Dynamic Voltage and Frequency Scaling (DVFS) to keep the processor temperatures below pre-determined thresholds. In reactive schemes such as those in [6][7] the tasks in the application are scheduled at a high speed until the processor temperature exceeds a specified threshold. Once the processor temperature exceeds the threshold, the speed is lowered and thereby cooling the processor. Since real-time applications often need design time guarantees of schedulability, Wang et al. [6][7] derive schedulability bounds to determine whether a given periodic task set can be feasibly executed without violating the processor's temperature constraints. Proactive schemes [8][9][10], on the other hand, strive to look ahead into the future when scheduling the tasks. Based on the projected temperatures into the future, processor speeds are dynamically adjusted well before the processor gets too hot. Once again schedulability tests are derived to determine whether a task set can be executed within the thermal constraints of the processor. The key assumption in most of these papers that derive schedulability tests is that all the tasks are perfectly periodic. In fact, some of these papers [6][7] make an even stronger assumption that all tasks have the same period.

Some recent papers have considered aperiodic tasks [11]. They do not provide design time guarantees. Recent papers that consider periodic tasks with possible jitter in the arrival process are [12][13][14]. In [12], Rai et al. propose a method to calculate the worst-case peak temperature of single-core processor system at design time for periodic tasks with jitter. The basic idea is to identify the worst-case arrival pattern of real-time tasks which will result in highest processor temperature. This method is extended in [13] to multi-core processor systems. Since exact analysis is very difficult, the scheme in [13] is modified to compute an upper bound on the worst-case peak temperature at much lower computational complexity [14]. However, a key limitation of the analyses in [12][13][14] is that they assume that maximum jitter must satisfy certain integrality constraint. If this assumption is not satisfied, then one must model the task arrival process using a more conservative bound for maximum jitter which satisfies the assumption. However, when one chooses such a conservative bound, the corresponding estimate of the worst-case peak temperature is also conservative.

One of the main contributions of this work is that it relaxes this integrality requirement. In particular, it is shown that this extended analysis results in provably tighter estimates of the worst-case processor temperatures than the ones in [13][14]. Numerical evaluations show that the difference in the estimates may be as large as 6-7 °C, which is substantial when dealing with temperature close to processor tolerance limits. We also extend and evaluate our scheme for a real time application with multiple periodic tasks in each core. This is in contrast to [13][14] where the evaluations are based on a single task on each core. Our approach for extending to the multiple task scenario is based on the idea in [15]. However, the idea in [15] is tailored to estimate the worst-case peak temperature.

The rest of this report is organized as follows- The system model is presented in chapter II. The proposed thermal analysis for a single task case is described in chapter III. The proofs of theorems in chapter III are included in the Appendix. An extension to the multiple task case is proposed in chapter IV. Results of an empirical evaluation of the proposed analysis are discussed in chapter V. The report concludes with chapter VI.

Chapter 2

System Model

This chapter describes the computational and thermal model for single and multiple tasks for real-time application on multi-core system. The system model is taken from [13].

Notation: Bold characters represent vectors and matrices and non-bold represent scalar. For example $\mathbf{T}$ denotes a vector whose k-th element is denoted as T_k.

2.1 Computational Model

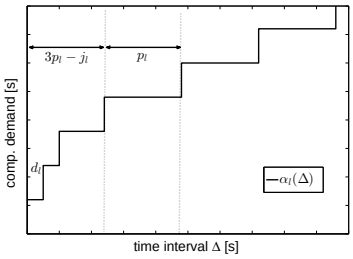
The computational model is based on real-time calculus. Let us assume that task instances with a total workload of $R_l(s,t)$ time arrives in the time interval $[s,t)$ at processor core l. The arrival curve α_l upper bounds the cumulative workload, i.e.,

$$R_l(s,t) \leq \alpha_l(t-s)\ \forall s < t \tag{2.1}$$

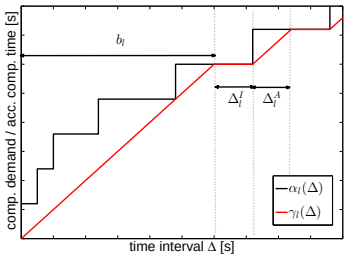
with $\alpha_l(0) = 0$. Period Jitter Delay (PJD) model for each task can be described with following parameters- a constant workload of Δ_l^A time units, period p_l, jitter j_l and a minimum arrival time d_l. Figure 2.1a shows a typical arrival curve. The task scheduler is assumed to be work-conserving i.e., a processor core is 'active' when there are instances in the queue. Time spent by core l to process an incoming workload of $R_l(s,t)$ time units, is denoted by the accumulated computing time $Q_l(s,t)$. $Q_l(t-\Delta,t)$ is upper bounded by $\gamma_l(\Delta)$ for all intervals of length $\Delta < t$:

$$Q_l(t-\Delta,t) \leq \gamma_l(\Delta) = \inf_{0\leq\lambda\leq\Delta} (\Delta-\lambda) + \alpha_l(\lambda) \tag{2.2}$$

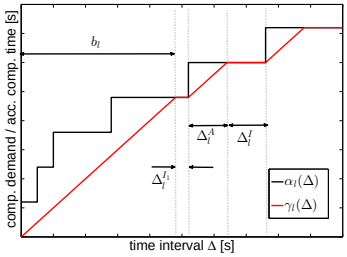
$Q_l(s,t)$ is monotonically increasing and has either slope 1 or 0 for fixed s with $s < t$. When the slope is 1 or 0 means the core is in 'active' mode (executing



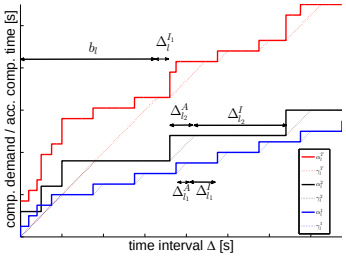
(a) Arrival curve α_l



(b) Accumulated computing time for constrained task



(c) Accumulated computing time for unconstrained task



(d) Accumulated computing time for two unconstrained tasks

Figure 2.1: Computational Model

task instances) or 'idle' mode correspondingly. The processing mode can be expressed by the mode function:

$$S_l(t) = \frac{\mathrm{d}\, Q_l(s,t)}{\mathrm{d}\, t} \in \{0,1\} \tag{2.3}$$

Length of the first increasing interval is represented by burst b_l, the length of every other active interval is Δ_l^A and every idle interval is Δ_l^I. If the jitter j_l is an integral multiple of Δ_l^I (considering minimum arrival time d_l is not greater than workload Δ_l^A), then all the non-increasing intervals are of the same length. A task satisfying the assumption that its jitter is an integral multiple of idle interval Δ_l^I is referred to as constrained task in this report. Otherwise, the task is said to be an unconstrained task. Figure 2.1b illustrates a typical arrival curve and its corresponding upper bound on the accumulated computing time for constrained task. Figure 2.1c illustrates the general case of unconstrained task, where jitter is not an integral multiple of Δ_l^I, in such case length of the first idle interval is $\Delta_l^{I_1}$ and every other idle interval is Δ_l^I. If the encountered task is unconstrained (as in Figure 2.1c), a tight upper bound for it can be approximated to slightly conservative upper bound by increasing the jitter to make it integral multiple of Δ_l^I, thus transform it to constrained task (as in Figure 2.1b). This approach is taken in [13]. However it leads to a conservative estimate of the worst-case temperature. Figure 2.1d displays two unconstrained tasks set of different period, min-delay, jitter and workload, there is a curve which shows summation of the two arrival curves. Their corresponding arrival curve ($\alpha_l^T(\Delta)$) is also shown, b_l denotes the burst for the arrival curve.

2.2 Thermal Model

The assumed thermal model of the multi-core processor is also taken from [13]. The model describes the temperature evolution by means of an equivalent *RC* circuit taken from [13]. The n-dimensional temperature vector $\mathbf{T}(t)$ at time t is given by a set of first-order differential equations:

$$\mathbf{C}.\frac{\mathrm{d}\, \mathbf{T}(t)}{\mathrm{d}\, t} = (\mathbf{P}(t) + \mathbf{K}.\mathbf{T}^{\mathrm{amb}}) - (\mathbf{G} + \mathbf{K}).\mathbf{T}(t) \tag{2.4}$$

with n is the number of nodes of the *RC* circuit, $\mathbf{C}$, $\mathbf{G}$, $\mathbf{K}$, and $\mathbf{P}$ represents the thermal capacitance matrix, thermal conductance matrix, thermal ground conductance matrix and power dissipation vector respectively and $\mathbf{T}^{\mathrm{amb}} = T^{\mathrm{amb}}.[1, ..., 1]'$ represents the ambient temperature vector. $\mathbf{T}^0$ denotes the initial temperature and the system is assumed to start at time

$t = 0$. $\mathbf{G}$ is a non-positive matrix with zero diagonal elements and $\mathbf{K}$ is a non-negative diagonal matrix.

A linear dependency of power dissipation on temperature [4] is assumed due to leakage power. Processing components have two processing modes, namely 'active' and 'idle'. If component is in 'active' processing mode at time t, the mode function $S_l(t)$ defined by (2.3) is 1 and otherwise, 0. The leakage power is assumed to be independent of its processing mode, and thus can characterize the power dissipation as:

$$\mathbf{P}(t) = \boldsymbol{\phi}.\mathbf{T}(t) + \boldsymbol{\psi}(t) \tag{2.5}$$

where

$$P_l(t) = \begin{cases} P_l^a = \phi_{ll}.T_l(t) + \psi_l^a & if S_l(t) = 1 \\ P_l^i = \phi_{ll}.T_l(t) + \psi_l^i & if S_l(t) = 0 \end{cases} \tag{2.6}$$

where $\boldsymbol{\phi}$ is a diagonal matrix with constant coefficients, and $\boldsymbol{\psi}$ a vector. Thus from above equations the state-space representation of the thermal model can be expressed as:

$$\frac{\mathrm{d}\,\mathbf{T}(t)}{\mathrm{d}\,t} = \mathbf{A}.\mathbf{T}(t) + \mathbf{B}.\mathbf{u}(t) \tag{2.7}$$

where the input vector $\mathbf{u}(t) = \boldsymbol{\psi}(t) + \mathbf{K}.\mathbf{T}^{\mathrm{amb}}$, $\mathbf{A} = -\mathbf{C}^{-1}.(\mathbf{G} + \mathbf{K} - \boldsymbol{\phi})$, and $\mathbf{B} = \mathbf{C}^{-1}$. Input u_l of node l can be associated with the workload of the corresponding component:

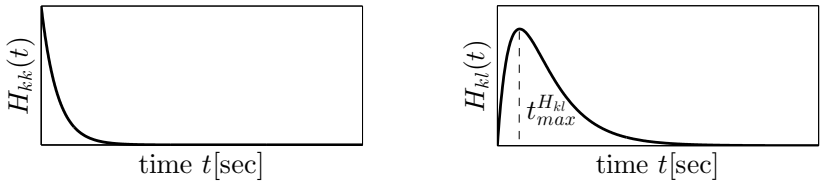
$$u_l(t) = u_l^a.S_l(t) + u_l^i.(1 - S_l(t)) \tag{2.8}$$

where $u_l^a = \psi_l^a + K_{ll}.T^{\mathrm{amb}}$ and $u_l^i = \psi_l^i + K_{ll}.T^{\mathrm{amb}}$ i.e., input of node l at time t is u_l^a (or u_l^i) if the corresponding component is in 'active' (or 'idle') processing mode at time t, that is, $S_l(t) = 1$ (or $S_l(t) = 0$). This is true assuming that there is a constant power leakage in case of active mode as well as idle mode. Since the thermal system is linear and time-invariant, the temperature of node k is :

$$T_k(t) = T_k^{\mathrm{init}}(t) + \sum_{l=1}^{n} T_{k,l}(t) \tag{2.9}$$

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

$$T_{k,l}(t) = \int_0^t H_{kl}(\xi).u_l(t - \xi)\,\mathrm{d}\,\xi. \tag{2.10}$$



(a) Self response $H_{kk}(t)$ (b) General response $H_{kl}(t)$

Figure 2.2: Impulse Response

The class of impulse responses can be defined by considering the duality of thermal network and grounded capacitor RC circuit, which is relevant for analysing the thermal characteristics of multi-core systems. As shown in Figure 2.2 [13], it can be assumed that $H_{kl}(t)$ is a non-negative unimodal function that has its maximum at time $t_{\max}^{H_{kl}} = 0$ if $k = l$, or at time $t_{\max}^{H_{kl}} > 0$ if $k \neq l$. This describes that the temperature rises with power on the node that produces power without a delay and on a neighbour of the node that produces power only after a delay. In the following analysis, we assume that every impulse response belongs to the described class.

2.3 Problem Statement

The rest of this report addresses the question of determining the worst-case peak temperature of a multi-core real-time system. The problem statement can be formulated as follows:

For a given work-conserving computing model (2.2), power model (2.5), the distributed thermal model (2.4), and a bound on workload $\boldsymbol{\alpha}$, the goal is to find the worst-case peak temperature T_S^* of the system for all computing demands $\mathbf{R}$ that follows $\boldsymbol{\alpha}$.

Chapter 3

Thermal Analysis for Single Unconstrained Task

Existing algorithms for worst-case peak temperature estimation considers constrained tasks, [13] presents algorithm for exactly calculating worst possible peak temperature and [14] presents faster but approximate version of the same algorithm. Both these algorithm can only accommodate single task and that too constraint by assuming length of every idle interval as constant Δ_l^I. We introduce an algorithm which accurately estimates the worst-case peak temperature for tasks not constrained in this fashion. This chapter provides a non-conservative approach for unconstrained task, which can be then simplified to get conservative faster approach.

To accurately find the worst-case peak temperature for such tasks, we must first find a possible schedule for the given situation. We have to find the number of instances of the task that must occur as burst to maximize the core temperature(s). For an unconstrained task, keeping everything else constant, increasing jitter from zero results in a sharp transition in the number of instances of the task that can occur in a burst, each time jitter crosses the mark of integral multiple of Δ_l^I.

So to counter such sudden jump we consider two possibilities, worst of the two will result in worst-case temperature, further we'll show in results

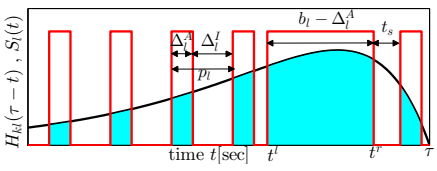


Figure 3.1: Illustration of Algorithm 1 (taken from [13])

Input: $b_l, \Delta_l^I, \Delta_l^{I_1}, \Delta_l^A, p_l = \Delta_l^I + \Delta_l^A, \widetilde{t}_{\max}^{H_{kj}}, \tau, H_{kl}$

Output: $\mathring{Q}_l^{\{k\}}(0, \Delta)$

1: $S_l^{\{k1\}} = 0$
2: **for all** $t^{(r)} \in [\widetilde{t}_{\max}^{H_{kj}}, \widetilde{t}_{\max}^{H_{kj}} + b_l - \Delta_l^A]$ **do**
3: **for all** $t^s \in [0, \Delta_l^{I_1}]$ **do**
4: $S_l^{\{k1\}} = maximize\_area(t^s, t^{(r)}, \Delta_l^I - \Delta_l^{I_1}, S_l^{\{k1\}})$
5: **end for**
6: **end for**
7: $S_l^{\{k2\}} = 0$
8: **for all** $t^{(r)} \in [\widetilde{t}_{\max}^{H_{kj}}, \widetilde{t}_{\max}^{H_{kj}} + b_l]$ **do**
9: **for all** $t^s \in (\Delta_l^{I_1}, \Delta_l^I]$ **do**
10: $S_l^{\{k2\}} = maximize\_area(t^s, t^{(r)}, -\Delta_l^{I_1}, S_l^{\{k2\}})$
11: **end for**
12: **end for**
13: $S_l^{\{k\}} = max(S_l^{\{k1\}}, S_l^{\{k2\}})$
14: $\mathring{Q}_l^{\{k\}}(0, \Delta) = \int_0^{\Delta} S_l^{\{k\}}(\xi)\,\mathrm{d}\,\xi$ for all $0 \le \Delta \le \tau$

Algorithm 1: Calculation of the critical accumulated computing time function $\mathring{\mathbf{Q}}^{\{k\}}(0, \Delta)$ for an unconstrained task for all $0 \le \Delta \le \tau$, Appendix A contains subroutine maximize_area and Appendix B gives the proof for the algorithm

that worst of the two is continuous with respect to jitter change. The two different possible bursts size are considered which encompasses $\widetilde{t}_{\max}^{H_{kl}}$: burst 1 $b_{l1} = b_l - \Delta_l^A$ (as considered by [13]) and burst 2 $b_{l2} = b_l$. We use two such different burst for temperature estimation. The corresponding temperatures caused by the bursts are considered as $\mathring{T}_k^{*b_{l1}}(\tau)$ and $\mathring{T}_k^{*b_{l2}}(\tau)$. Maximum of the two is taken as worst case temperature. We can then prove the following theorem. This theorem is an extension of the Theorem 5 in [13] to deal with unconstrained tasks. Following theorem provides a method to calculate the critical accumulated computing time $\mathring{Q}_l^{\{k\}}(0, \Delta)$ for a constrained task leading to worst-case peak temperature $\mathring{T}_k^*(\tau)$ of node k.

Theorem 1. Suppose that the accumulated computing time $\mathring{\mathbf{Q}}^{\{k\}}(0, \Delta)$ for all $0 \leq \Delta \leq \tau$ is constructed by Algorithm 1 leads to temperature $\mathring{T}_k^*(\tau)$ at time τ. When the scheduler is work-conserving, $\mathring{T}_k^*(\tau)$ is an achievable upper bound on the highest possible value of temperature $T_k(\tau)$ at time τ. Furthermore , $\mathring{T}_k^*(\tau) \geq \mathring{T}_k^*(t) \geq T_k(t)$ for all $0 \leq t \leq \tau$ with same initial temperature vector $\mathbf{T}^0 \leq (\mathbf{T}^\infty)^i$, where $(\mathbf{T}^\infty)^i$ is the steady-state temperature vector if all nodes are in 'idle' mode.

Proof: See Appendix B for formal details. Similar to the algorithm in [13] the proposed Algorithm 1 also calculates the critical accumulated computing time $\mathring{Q}_l^{\{k\}}(0, \Delta)$ by maximizing the area under the curve by altering both the position of the burst and the gap between burst and first successive active interval. This is illustrated in Figure 3.1. The sum of the gaps between the burst and active interval either side in case of constrained Algorithm is fixed to Δ_l^I, whereas for unconstrained Algorithm it is $\Delta_l^{I_1}$ and $\Delta_l^{I_1} + \Delta_l^I$ for bursts b_{l1} and b_{l2} respectively. In case of b_{l2} the minimum distance between the burst and active interval on either side is $\Delta_l^{I_1}$.

The relation between four different bounds calculated on the maximum temperature for unconstrained task is as follows:

$$\breve{T}_k^*(\tau) \geq \hat{T}_k^*(\tau) \geq T_k^*(\tau) \geq \mathring{T}_k^*(\tau) \tag{3.1}$$

where $\breve{T}_k^*(\tau)$, $\hat{T}_k^*(\tau)$ and $T_k^*(\tau)$ are the worst-case peak temperatures calculated by Theorem 4,2 and 1 of [14] respectively, when an unconstrained task is made a constrained one by changing the jitter as mentioned in [13]. In other words, the peak temperature using our algorithm is the least conservative.

Chapter 4

Extension to Multiple Unconstrained Task

This chapter presents an algorithm for multiple unconstrained tasks on each core of the processor. The method used in previous chapter for finding an achievable bound to peak temperature is extended here to find an upper bound to the peak temperature.

It can be seen from Figure 2.1d that there is a single burst b_l that comes from the sum of the arrival curves of the individual tasks. The worst case burst which can encompass $\widetilde{t}_{\max}^{H_{kj}}$ is thus b_l, but the next active period can be minimum $\Delta_l^{I_1}$ distance away from the burst b_l and the next active period can be of maximum computation time $\Delta_{l_T}^A = \sum_{i=1}^{n_T} \Delta_{l_i}^A$ i.e., sum of computation time of all the tasks, where n_T is the number of tasks and $\Delta_{l_i}^A$ is the computation time for task i. Taking an upper bound to it, we consider that maximum computation time can occur adjacent to burst b_l (considering $\Delta_l^{I_1} = 0$). So the final burst which can encompass $\widetilde{t}_{\max}^{H_{kj}}$ is $b_l' = b_l + \Delta_{l_T}^A$. Now processing component of each task should be separated from this burst (both sides of burst) with at least their corresponding idle time.

Further, the maximum utilization for a processor for the given set of tasks is $\delta_T = \frac{\Delta_{l_T}^A}{\Delta_{l_T}^A + \Delta_{l_T}^I}$, where $\Delta_{l_T}^I = \sum_{i=1}^{n_T} \Delta_{l_i}^I$ i.e., sum of idle time of all the tasks (period minus corresponding computational load). Now we get an upper bound for peak temperature for multiple tasks where we can vary the position of burst $b_l + \Delta_{l_T}^A$ around $\widetilde{t}_{\max}^{H_{kj}}$ and run the processing component with constant slope at all other instances as δ_T. Since the impulse response is monotonically non-increasing on either side of $\widetilde{t}_{\max}^{H_{kj}}$, a constant slope of δ_T on both sides will give a small upper-bound (proof for that is intuitive and given in [14]). Suppose maximum temperature bound thus obtained by this

theorem is $\breve{T}_k^{*m}(\tau)$, which is upper bound to temperature $T_k^m(\tau)$ on processor k at time τ due to multiple tasks on multi processor system. We are skipping formal proof (can be followed from [14]) and statement for this theorem. Instead putting the next theorem (similar to [14]) and the main result of this chapter, which is a further more conservative but faster approach as-

Theorem 2. Suppose at time instant t, $\hat{T}_{k,l}^{*m}(\tau)$ is the incremental temperature (upper bound) of node k due to task executed at node l and $\hat{T}_k^{*m}(\tau)$ is the absolute temperature (upper bound) of node k for a set of workload functions $\mathbf{R}(s;t)$ that are bounded by the set of summed arrival curves $\boldsymbol{\alpha}$, (arrival curve obtained after summing the multiple tasks on each processor). When the scheduler is work-conserving, then the temperature:

$$\hat{T}_k^{*m}(\tau) = T_n^{\text{init}}(\tau) + \sum_{l=1}^{n} \hat{T}_{k,l}^{*m}(\tau) \tag{4.1}$$

with:

$$\begin{aligned} \hat{T}_{k,l}^{*m}(\tau) = (u_l^i + \delta_T.(u_l^a - u_l^i)). \int_0^{\tau} H_{kl}(t-\xi)\,\mathrm{d}\,\xi \\ +(u_l^a - u_l^i).(1-\delta_T). \int_{\widetilde{t}_{\max}^{H_{kj}} - b_l - \Delta_{l_T}^A}^{\widetilde{t}_{\max}^{H_{kj}} + b_l + \Delta_{l_T}^A} H_{kl}(t-\xi)\,\mathrm{d}\,\xi \end{aligned} \tag{4.2}$$

and $\delta_T = \frac{\Delta_{l_T}^A}{\Delta_{l_T}^A + \Delta_{l_T}^I}$ is an upper bound on the highest temperature of node k at time τ i.e., $\hat{T}_k^{*m}(\tau) \geq T_k^m(\tau)$. Furthermore , $\hat{T}_{k,l}^{*m}(\tau) \geq \hat{T}_{k,l}^{*m}(t) \geq T_k(t)$ for all $0 \leq t \leq \tau$ with same initial temperature vector $\mathbf{T}^0 \leq (\mathbf{T}^\infty)^i$, where $(\mathbf{T}^\infty)^i$ is the steady-state temperature vector if all nodes are in 'idle' mode.

Proof: Instead of varying the burst $b_l + \Delta_{l_T}^A$ around the peak of impulse response $\widetilde{t}_{\max}^{H_{kj}}$, we are simply summing up all the possibilities of burst to find the new upper bound. Thus for the time period $\widetilde{t}_{\max}^{H_{kj}} + b_l + \Delta_{l_T}^A$ to $\widetilde{t}_{\max}^{H_{kj}} - b_l - \Delta_{l_T}^A$ in places where the processing component should have been 1 at some place and δ at other places to calculate $\breve{T}_k^{*m}(\tau)$, now we are assuming it to be 1 always to find $\hat{T}_k^{*m}(\tau)$ for faster computation, keeping the rest as same. Thus, $\hat{T}_k^{*m}(\tau) \geq \breve{T}_k^{*m}(\tau)$. Since $\breve{T}_k^{*m}(\tau) \geq T_k(\tau)$ is already known, which proves $\hat{T}_k^{*m}(\tau) \geq T_k(\tau)$. Proof for second part of Theorem 2 is similar to that of Theorem 1.

Chapter 5

Results

First, the results for the unconstrained task are compared to that from the algorithms in [13][14]. Then, the results for multiple unconstrained tasks are shown. The algorithms are implemented in MATLAB and the temperature estimates come from the results of algorithm on applying HotSpot [16] .

We took a multi-core system with 9 processors in a 3X3 design (as shown in figure 5.1) and maximized temperature on processor 1 due to different tasks on processor 1 and other processors. The temperature on each core was calculated using HotSpot [16], different thermal parameters used in HotSpot are shown in Table 5.1.

5.1 Unconstrained single task

On processor 1 and 2 (see figure 5.1) a task with period=50 ms, min delay=1 ms and workload=2 ms is modeled. The jitter for the task is varied from 0 to 200 ms and τ is set to 1000 ms, i.e., time where maximum temperature is

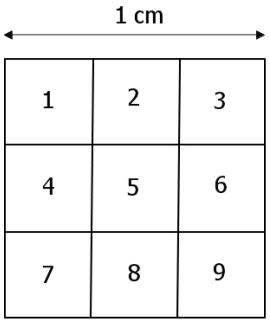
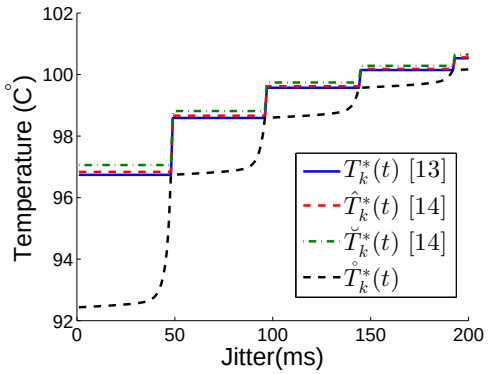
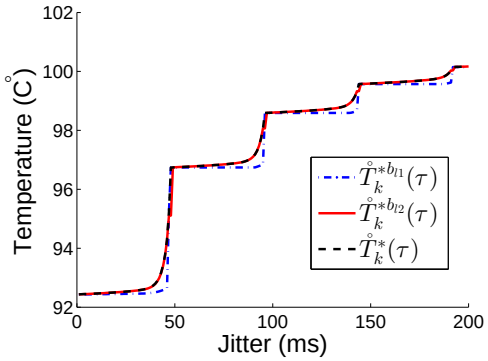


Figure 5.1: Design of 3X3 multi processor used for analysis

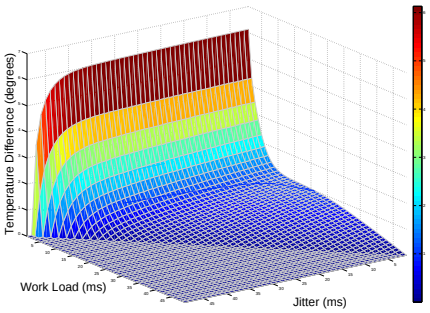


(a) Temperature variation with respect to jitter for unconstrained task by different methods

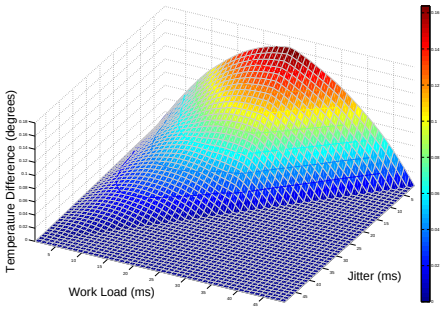


(b) Temperature variation with respect to jitter for unconstrained task by algorithm defined in this report

Figure 5.2: Temperature variation with respect to jitter for unconstrained single task



(a) Difference in peak temperatures on processor 1 calculated by proposed and previous algorithm($T_k^*(\tau) - \mathring{T}_k^*(\tau)$) with respect to jitter and work load for unconstrained tasks on processors 1 and 2



(b) Difference in peak temperatures on processor 1 calculated by proposed and previous algorithm($T_k^*(\tau) - \mathring{T}_k^*(\tau)$) with respect to jitter and work load for unconstrained task on processor 3

Figure 5.3: Temperature variation with respect to jitter and workload

Table 5.1: HotSpot Thermal Parameters

Parameter	Value
chip thickness in meters	0.00015
silicon thermal conductivity in W/(m-K)	100.0
silicon specific heat in J/(m^3-K)	1.75e6
temperature threshold for DTM (kelvin)	354.95
convection capacitance in J/K	140.4
convection resistance in K/W	0.1
heatsink side in meters	0.06
heatsink thickness in meters	0.0069
heatsink thermal conductivity in W/(m-K)	400.0
heatsink specific heat in J/(m^3-K)	3.55e6
ambient temperature in kelvin	333.15
initial temperature in kelvin	333.15

calculated. Note, here we took min delay not more than workload. A function for impulse response H_{kl} is obtained from HotSpot [16] as shown in Figure 2.2. Average power delivered when in active state is taken to be $100W/cm^2$ and $10W/cm^2$ in idle state. Unconstrained tasks for Theorem 1, 2 and 4 of [14] can be transformed to constrained task by increasing the jitter to make it integral multiple of idle period Δ_l^I (period-work load) leading to a over-approximation of the worst-case temperature. Figure 5.2a shows temperature on core 1 due to tasks on core 1 & 2 (since superposition principle works here, effect of each can be considered independent) calculated from Theorem 1, Theorem 2 & Theorem 4 from [14] and compared to algorithm described here. Figure 5.2a proves the equation (3.1) presented for the unconstrained task. It can be seen there is an overestimation of around 5 °C by algorithms defined in [13][14] for this case. The maximum temperature difference occurs when jitter is just above zero and $T_k^*(\tau)$, $\mathring{T}_k^*(\tau)$ coincide only when jitter is integral multiple of idle time Δ_l^I.

Figure 5.2b shows curves for two bounds that were considered in Theorem 2 i.e., $\mathring{T}_k^{*b_{l1}}(\tau)$ and $\mathring{T}_k^{*b_{l2}}(\tau)$ for b_{l1} and b_{l2} cases of bursts respectively. Upper bound of both the curves forms the curve for $\mathring{T}_k^*(\tau)$, it can be seen that the change in $\mathring{T}_k^*(\tau)$ is continuous as opposed to $T_k^*(\tau)$, this happens because two different cases of bursts are considered instead. Note that $\mathring{T}_k^{*b_{l2}}(\tau)$ is not

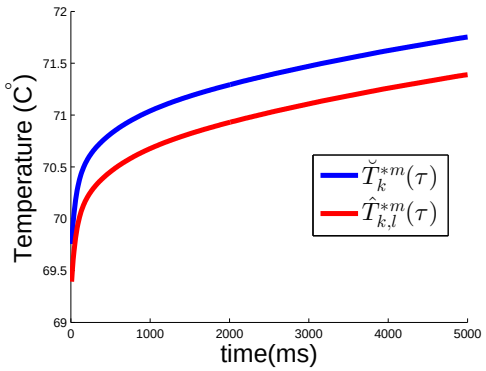


Figure 5.4: Temperature variation with respect to τ for unconstrained multiple tasks

defined when jitter is integral multiple of Δ_l^I, at those instance $\mathring{T}_k^*(\tau)$ is equal to $\mathring{T}_k^{*b_{l1}}(\tau)$. H_{kl} for chosen processor setting is such that $\mathring{T}_k^*(\tau)$ follows the curve $\mathring{T}_k^{*b_{l2}}(\tau)$ except when jitter is integral multiple of Δ_l^I, but this is not necessary the case always as we show in Appendix B.

Figure 5.3a shows the difference in the peak temperatures calculated by proposed algorithm ($\mathring{T}_k^*(\tau)$) and previous algorithm ($T_k^*(\tau)$) with respect to work load, which is varied from 1 ms to period and jitter, which is varied from 1 ms to (period-workload), while keeping period as 50 ms and τ as 1000 ms. It can be seen from the figure that maximum temperature difference is more than 6 degrees for workload of 1 ms and jitter 1 ms. The difference is not consistent even if we fix any two out of period, jitter and workload. Figure 5.3b shows difference in peak temperature calculated by the two methods for processor 1 due to the same task set on processor 3 (see figure 5.1) . It can be seen that the maximum difference in the peak temperature is not achieved for this case at low work load, but somewhere in the middle. Thus there can be several local maxima or varying slopes (when multiple processors are assigned tasks) for the temperature difference curve, which highly depends on the impulse response. Thus previous methods gives an upper bound to peak temperature which can differ from actually achievable temperature by more than 6 degrees for considered case (depending on PJD model). The proposed methods instead gives an achievable bound.

5.2 Unconstrained multiple tasks

Figure 5.4 shows the increase in peak temperature of a core for multiple tasks on each core of multi-core system (all 9 cores). Here two tasks with pe-

riod=50 ms, jitter=50 ms, min delay=1 ms, workload=1 ms and period=20 ms, jitter=50 ms, min delay=1 ms, workload=1 ms are considered. Temperature is then maximized for different time points τ, shown in figure 5.4. The upper bound to the temperature is calculated using Theorem 2 ($\breve{T}_k^{*m}(\tau)$) and also by varying position of the burst ($\hat{T}_k^{*m}(\tau)$). Since there is no need to vary burst position to calculate the temperature bound in Theorem 2, $\breve{T}_k^{*m}(\tau)$ gives a higher upper bound to the worst case peak temperature, but is much faster.

Chapter 6

Conclusion

In this report, I presented an algorithm to calculate the maximum temperature of a real-time application for unconstrained task 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. An empirical evaluation shows that a considerable improvement (over 6 °C) in peak temperature estimation is achieved using the proposed algorithms as compared to algorithms in literature.

Bibliography

[1] J. Donald and M. Martonosi, "Techniques for multicore thermal management: Classification and new exploration," in *Proceedings of IEEE International Symposium on Computer Architecture (ISCA)*, pp. 78–88, June 2006.

[2] S. Murali, A. Mutapcic, D. Atienza, R. Gupta, S. Boyd, and G. De Micheli, "Temperature-aware processor frequency assignment for MPSoCs using convex optimization," in *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis*, pp. 111–116, 2007.

[3] A. Coskun, T. Rosing, and K. Whisnant, "Temperature aware task scheduling in MPSoCs," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, pp. 1659–1664, 2007.

[4] T. Chantem, R. Dick, and X. Hu, "Temperature-aware scheduling and assignment for hard real-time applications on MPSoCs," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, pp. 288–293, 2008.

[5] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. Stan, "HotSpot: A compact thermal modeling methodology for early-stage VLSI design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 5, pp. 501–513, 2006.

[6] S. Wang and R. Bettati, "Delay analysis in temperature-constrained hard real-time systems with general task arrivals," in *Proceedings of the Real-Time Systems Symposium (RTSS)*, pp. 323–334, 2006.

[7] S. Wang and R. Bettati, "Reactive speed control in temperature-constrained real-time systems," *Real-Time Systems*, vol. 39, no. 1-3, pp. 73–95, 2008.

[8] J.-J. Chen, S. Wang, and L. Thiele, "Proactive speed scheduling for frame-based real-time tasks under thermal constraints," in *Proceedings of the Real Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 141–150, Apr. 2009.

[9] G. Quan, Y. Zhang, W. Wiles, and P. Pei, "Guaranteed scheduling for repetitive hard real-time tasks under the maximal temperature constraint," in *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis*, pp. 267–272, 2008.

[10] G. Quan and Y. Zhang, "Leakage aware feasibility analysis for temperature-constrained hard real-time periodic tasks," in *Proceedings of the Euromicro Conference on Real-Time Systems (ECRTS)*, pp. 207–216, 2009.

[11] R. Ahmed, P. Ramanathan, K. K. Saluja, and C. Yao, "Scheduling aperiodic tasks in next generation embedded real-time systems," in *Proceedings of International Conference on VLSI Design*, pp. 25–30, Jan. 2013.

[12] D. Rai, H. Yang, I. Bacivarov, J.-J. Chen, and L. Thiele, "Worst-case temperature analysis for real-time systems," in *Proceedings of the Conference on Design, Automation and Test in Europe (DATE)*, pp. 631–636, Mar. 2011.

[13] L. Schor, I. Bacivarov, H. Yang, and L. Thiele, "Worst-case temperature guarantees for real-time applications on multi-core systems," in *Proceedings of the Real Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 87–96, Apr. 2012.

[14] L. Schor, I. Bacivarov, H. Yang, and L. Thiele, "Fast worst-case peak temperature evaluation for real-time applications on multi-core systems," in *Proceedings of Latin American Test Workshop (LATW)*, pp. 1–6, Apr. 2012.

[15] L. Thiele, L. Schor, I. Bacivarov, and H. Yang, "Predictability for timing and temperature in multiprocessor system-on-chip platforms," *ACM Transactions in Embedded Computing Systems (TECS)*, Mar. 2012.

[16] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis - the SymTA/S approach," in *IEE Proceedings Computers and Digital Techniques*, vol. 152, pp. 148–166, July 2005.

Chapter 7

Appendix

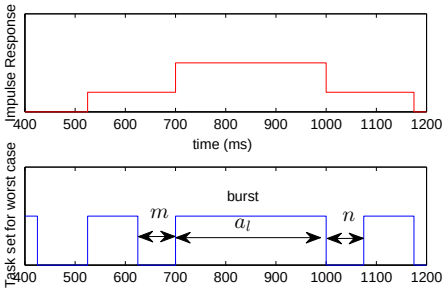
7.1 Subroutine maximize_area

Input: $t^s, t^{(r)}, d, S_l^{\{k\}}$
Output: $S_l^{\{k\}}$
1: $t^{(l)} = t^{(r)} - b_l + \Delta_l^A$
2: $S_l(t) = \begin{cases} 1 & t \in [t^{(r)} - b_l + \Delta_l^A, t^{(r)}) \\ 0 & otherwise \end{cases}$
3: **for** $i = 1$ *to* $\left\lceil \frac{\tau - t^{(r)}}{p_l} \right\rceil$ **do**
4: $\quad S_l(t) = S_l(t) + v_l(t, t^s + t^{(r)} + (i-1).p_l)$
5: **end for**
6: **for** $i = 1$ *to* $\left\lceil \frac{t^{(l)}}{p_l} \right\rceil$ **do**
7: $\quad S_l(t) = S_l(t) + v_l(t, t^s + t^{(l)} + d - i.p_l)$
8: **end for**
9: $\Upsilon = \int_0^t S_l(\xi).H_{kl}(t-\xi)\,\mathrm{d}\,\xi$
10: **if** $\Upsilon > \Upsilon^*$ **then**
11: $\quad \Upsilon^* = \Upsilon^*, S_l^{\{k\}} = S_l$
12: **end if**

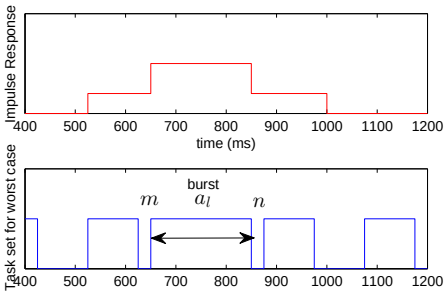
Algorithm 2: Calculating and maximizing area under the curve for impulse response and task set [13]

Subroutine is shown in Algorithm 2, where function v_l is defined as-

$$v_l(t, t^s) = \begin{cases} 1 & 0 \le t^s \le t \le \min(t_s + \Delta_l^A, \tau) \\ 0 & otherwise \end{cases} \tag{7.1}$$



(a) Case where impulse response is such that chosen $i = k = 3$



(b) Case where impulse response is such that chosen $i = k - 1 = 2$

Figure 7.1: Different impulse response can result in two different cases of burst to be considered

7.2 Proof for Algorithm 1

First we prove that the two cases used in algorithm 1 are at-least needed for an unconstrained task. Then we prove by contradiction that only these two cases are needed to find the worst case scenario which results in maximum temperature.

It can be seen from figure 2.1c that the burst is always an integral multiple of work load i.e., burst $b_l = k * \Delta_l^A$, where $k \geq 0$. Burst containing the peak of impulse response (check [13] to see why burst should contain the peak of impulse response) can be of length $a_l = i * \Delta_l^A$, where $0 \leq i \leq k$.

Let us consider jitter is not an integral multiple of Δ_l^I (that is task is not constrained) i.e., excluding the boundary condition for simplicity (which was proved in [13]). Suppose the gap between the burst and nearest active interval is m and n on left and right side respectively (figure 7.1a). For worst case temperature we have to maximize Area Under Curve (AUC) of impulse

response and the active interval [13]. Thus $m+n = \max(0, \Delta_l^{I_1} + (i-k+1) * \Delta_l^I)$ & $\max(0, \Delta_l^{I_1}+(i-k)*\Delta_l^I) \leq m, n \leq \max(0, \min(\Delta_l^I, \Delta_l^{I_1}+(i-k+1)*\Delta_l^I))$ for worst case and to satisfy the arrival curve (can be deduced easily).

Now consider a case period=200 ms, min delay=50 ms, work load=100 ms and jitter=250 ms, in such case burst $b_l = 300$ ms ($k = 3$) and $\Delta_l^{I_1} = 50$ ms, $\Delta_l^I = 100$ ms & $\Delta_l^A = 100$ ms. For this case consider 1st part of figure 7.1a as the impulse response. Then among all the possibilities $i = k = 3$ i.e., $a_l = 300$ ms will give maximum AUC, $m+n = \Delta_l^{I_1} + \Delta_l^I$ & $\Delta_l^{I_1} \leq m, n \leq \Delta_l^I$, $m = n = 75$ ms gives maximum area and accordingly the task set are plotted in 2nd part of the figure. Similarly in figure 7.1b, 1st part gives the impulse response for which $i = k - 1 = 2$ i.e., $a_l = 200$ ms will give maximum AUC, $m + n = \Delta_l^{I_1}$ & $0 \leq m, n \leq \Delta_l^{I_1}$, further $m = n = 25$ ms gives maximum area.

Thus, we have proved here that according to different impulse response we need at-least the two cases $i = k$ & $i = k - 1$. Now we'll prove that we need only these two cases. In other words if we decrease i further i.e., $i \leq k - 2$ then we'll get AUC less than what we got for $i = k$ or $i = k - 1$.

To maximize area for $i \leq k - 2$ we'll get $m = n = 0$, thus the total burst length (including the neighboring active interval region) will become $(i + 2) * \Delta_l^I$. So for $i = k - 2$ burst length will become equal to burst for $i = k$, but this time the gap between the burst and the next active interval will be Δ_l^I which is more than that of $i = k$ (where the gap was m & n), thus the area will be lesser for $i = k-2$. Similarly we can prove for $i = k-3$ and this thus holds for any $i \leq k - 2$.

Hence proved that there are two and only two cases (presented above) which are needed to find worst case peak temperature for an unconstrained task.

Proof for $\mathring{T}_k^*(\tau) \geq \mathring{T}_k^*(t) \geq T_k(t)$ for all $0 \leq t \leq \tau$ (with given conditions satisfied) [13]- Since

- $H_{kl}(t) \geq 0 \; \forall t, k, l$, and
- Temperature caused by any active process is $\geq$ temperature of a system that operates in 'idle' mode for the same time period (follows from equations (2.8),(2.9),(2.10)).
- Temperature at time zero $\mathbf{T}(0) = \mathbf{T}^0 \leq (\mathbf{T}^\infty)^i$.
- $\mathring{T}_k^*(\tau)$ gives maximum temperature at any instant τ.

If follows from above that: $\mathring{T}_k^*(\tau) \geq \mathring{T}_k^*(t)$ for all $0 \leq t \leq \tau$. Since $\mathring{T}_k^*(t) \geq T_k(t)$ from first part of the theorem, thus it proves $\mathring{T}_k^*(\tau) \geq \mathring{T}_k^*(t) \geq T_k(t)$ for all $0 \leq t \leq \tau$.