# Steady-State Dynamic Temperature Analysis and Reliability Optimization for Embedded Multiprocessor Systems

Ivan Ukhov
Linköping University
Sweden
ivan.ukhov@liu.se

Min Bao
Linköping University
Sweden
min.bao@liu.se

Petru Eles
Linköping University
Sweden
petru.eles@liu.se

Zebo Peng
Linköping University
Sweden
zebo.peng@liu.se

## ABSTRACT

In this paper we propose an analytical technique for the steady-state dynamic temperature analysis (SSDTA) of multiprocessor systems with periodic applications. The approach is accurate and, moreover, fast, such that it can be included inside an optimization loop for embedded system design. Using the proposed solution, a temperature-aware reliability optimization, based on the thermal cycling failure mechanism, is presented. The experimental results confirm the quality and speed of our SSDTA technique, compared to the state of the art. They also show that the lifetime of an embedded system can significantly be improved, without sacrificing its energy efficiency, by taking into consideration, during the design stage, the steady-state dynamic temperature profile of the system.

## Categories and Subject Descriptors

C.3 [**Special-Purpose and Application-Based Systems**]: Microprocessor/microcomputer applications, real-time and embedded systems; G.1.3 [**Numerical Linear Algebra**]: Sparse, structured, and very large systems; G.3 [**Probability and Statistics**]: Reliability and life testing; J.6 [**Computer-Aided Engineering**]: Computer-aided design.

## General Terms

Algorithms, Design, Performance, Reliability.

## Keywords

Multiprocessor System, Periodic Power Profile, Temperature Analysis, Leakage Power, Thermal Cycling Fatigue.

## 1. INTRODUCTION AND PRIOR WORK

Due to increasing power densities, temperature has evolved into a major concern for designers of modern embedded systems. Thus, temperature analysis has become an important component of current embedded system design frameworks.

Temperature-aware system-level design methods rely on the availability of temperature modeling and analysis tools. System-level temperature modeling approaches are mostly based on the duality between heat transfer and electrical phenomena [1]. The basic idea is to build an equivalent circuit of thermal resistances and capacitances capturing both the architecture blocks and elements of the thermal package. HotSpot [2], an architecture and system-level model and simulator, is the state of the art choice for system-level temperature analysis, as in [3, 4, 5, 6, 7, 8, 9].

However, temperature analysis time with HotSpot, or other similar approaches, is too long to be used inside a temperature-aware system-level optimization loop. The long thermal simulation time can severely limit the efficiency of the design space exploration. There has been some work on establishing fast system-level temperature analysis techniques. They also build on the duality between heat transfer and electrical phenomena and are based on restrictive assumptions in order to simplify the model. The approaches proposed in [10, 11], for example, are strictly restricted to monocore systems. The method described in [12] is restricted to homogeneous platforms and to applications in which the execution time of individual tasks is long, comparable with the thermal time constant of the package (in the order of 100 $s$).


*DAC'12,* June 3–7, 2012, San Francisco, California, USA.


Broadly speaking, there are two types of thermal analysis: (1) static temperature analysis, that produces a hypothetical temperature value (the steady-state temperature) at which the circuit is supposed to function if running for a long time under a certain constant (average) input power; (2) dynamic temperature analysis, that produces a transient temperature curve, describing the temperature behavior of the circuit as a function of time, when exposed to an arbitrary power profile.

The steady-state temperature, as produced by static analysis, is an approximation of the thermal behavior with limited applicability. It assumes that, eventually, the circuit will function at one constant temperature. This, however, is very often not the case in reality. In the context of a variable power profile applied periodically, the circuit will not reach a constant steady-state temperature but a steady state in which temperature is varying according to a certain periodic pattern. This pattern is captured by the steady-state dynamic temperature profile (SSDTP).

A typical design task, for which the SSDTP is of central importance, is temperature-aware reliability optimization. The impact of temperature on the lifetime of electronic circuits is well-known [3, 5, 8, 13]. The failure mechanisms commonly considered are electromigration, time-dependent dielectric breakdown, and thermal cycling, which are directly driven by the temperature [13]. What is important in this context is that not only average and maximum temperature, but also the amplitude and frequency of temperature oscillations, have a huge impact on the overall lifetime of the chip. Thus, efficient reliability optimization depends on the availability of the actual SSDTP.

Two approaches have been applied in the literature in order to obtain the SSDTP, as a prerequisite for reliability optimization. An approximate SSDTP can be produced by running a temperature simulator over one or more successive periods of the application until one can assume that a sufficient approximation of the thermal steady state has been reached [3]. Such an approach is both time consuming and potentially inaccurate. A very rough but fast approximation of the SSDTP is proposed in [7]. It constructs a stepwise temperature curve where each step corresponds to the static steady-state temperature that would be reached if a certain constant power was applied for a sufficiently long time. In Sec. 4 we will further elaborate on these two state of the art solutions. As our experiments show, they are too slow and/or too inaccurate in order to efficiently be used inside a temperature-aware system-level optimization loop for, e.g., reliability optimization.

In this paper we consider multiprocessor systems running applications exhibiting a power profile that can be considered periodic. Our contribution is twofold. First, we propose an approach that is both accurate and fast, for SSDTP calculation. Second, we show how our approach makes it possible to efficiently perform reliability optimization, based on the thermal cycling (TC) failure mechanism. More exactly, we propose a temperature-aware task mapping and scheduling technique that addresses the TC ageing effect. Experiments demonstrate the superiority of the proposed techniques, compared to the state of the art.

The rest of the paper is organized as follows. In Sec. 2 we introduce the architecture, power, and thermal models. The problem formulation is given in Sec. 3. The state of the art solutions are discussed in Sec. 4. In Sec. 5 we obtain an analytical formulation for the SSDTP calculation. In Sec. 6 and Sec. 7 we propose a fast and accurate technique to compute the SSDTP. Sec. 8 formulates the temperature-aware reliability optimization problem and proposes a solution based on our fast SSDTP calculation. Experimental results are given in Sec. 9 and Sec. 10 concludes the paper. Supplementary materials are given in the appendix, Sec. S1–S3.

## 2. ARCHITECTURE, POWER, AND THERMAL MODELS

We consider a heterogeneous multicore architecture with a set of processing elements $\Pi$ defined as the following:

$$\Pi = \{\pi_i = (V_i,\ f_i,\ N_{\text{gate}\,i}) :\ i = 0, \ldots, N_p - 1\}$$

where $V_i$, $f_i$, and $N_{\text{gate}\,i}$ are the supply voltage, frequency, and number of gates [4] of the $i$th core, respectively.

The total power dissipation of a processing element is defined as the sum of the dynamic and leakage power: $P = P_{\text{dyn}} + P_{\text{leak}}$. The dynamic part is modeled as $P_{\text{dyn}} = C_{\text{eff}} \cdot f \cdot V^2$ where $C_{\text{eff}}$ is the effective switched capacitance, $V$ and $f$ are the supply voltage and frequency, respectively. The leakage part of the power dissipation is defined as [4]:

$$P_{\text{leak}}(T) = N_{\text{gate}}\, V\, I_0 \left[ A\, T^2 e^{\frac{\alpha\, V + \beta}{T}} + B e^{(\gamma\, V + \delta)} \right] \qquad (1)$$

where $T$ and $V$ are the current temperature and supply voltage, respectively, $N_{\text{gate}}$ is the number of gates in the circuit, $I_0$ is the average leakage current at the reference temperature and supply voltage. $A$, $B$, $\alpha$, $\beta$, $\gamma$, and $\delta$ are the technology-dependent constants found in [4].

Our proposed technique is based on the RC thermal model that employs the analogy between electrical and thermal circuits [1]. Heat transfer is modeled with the following system of differential equations:

$$\mathbf{C}\, \frac{d\mathbf{T}(t)}{dt} + \mathbf{G}\, (\mathbf{T}(t) - \mathbf{T}_{\text{amb}}) = \mathbf{P}(t) \qquad (2)$$

where $\mathbf{T}$ is the temperature vector, $\mathbf{T}_{\text{amb}}$ is the ambient temperature vector, $\mathbf{C}$ is the thermal capacitance matrix, $\mathbf{G}$ is the thermal conductance matrix, and $\mathbf{P}$ is the power dissipation vector. The dimensions of the system are $N_n \times N_n$, where $N_n$ is the number of nodes in the equivalent RC thermal circuit, which is further discussed in Sec. S1.

## 3. PROBLEM FORMULATION

Consider a multicore system that consists of $N_p$ processing elements $\Pi = \{\pi_i : i = 0, \ldots, N_p - 1\}$ and executes a periodic application with a period $\tau$. We construct an equivalent RC thermal circuit of the system that contains $N_n$ thermal nodes. The dynamic power profile of the system is sampled into $N_s$ time intervals of duration $\Delta t$, called sampling interval, in such a way that the dynamic power dissipation and temperature of each node are assumed to be constant within an interval. The discrete dynamic power profile is defined as the following:

$$\mathbb{P}_{\text{dyn}} \stackrel{\text{def}}{=} \{P_{ij} :\ i = 0, \ldots, N_s - 1;\ j = 0, \ldots, N_n - 1\}$$

where $P_{ij}$ is the dynamic power dissipation during the $i$th time interval of the $j$th thermal node. After the steady state is reached, the corresponding temperature profile becomes periodic and is defined as:

$$\mathbb{T} \stackrel{\text{def}}{=} \{T_{ij} :\ i = 0, \ldots, N_s - 1;\ j = 0, \ldots, N_n - 1\}$$

where $T_{ij}$ is the temperature of the $j$th node in the $i$th time interval. The profile is called the steady-state dynamic temperature profile (SSDTP).

Given:

- A multicore system with a set of processing elements $\Pi$ executing a periodic application.
- The discrete dynamic power profile $\mathbb{P}_{\text{dyn}}$ of the system[1] with the sampling interval $\Delta t$.
- The floorplan of the chip corresponding to the level of details at which the thermal modeling is performed.
- The configuration of the thermal package, i.e., dimensions of the thermal interface material, heat spreader, and heat sink.
- The thermal parameters of the die and package, e.g., the thermal conductivity and thermal capacitance.

Find:

- The corresponding periodic temperature profile $\mathbb{T}$ of the system when the steady state is reached.

[1]Power dissipation of inactive nodes, i.e., the nodes that belong to the thermal package, is zero.

## 4. STATE OF THE ART SOLUTIONS

### 4.1 Iterative Simulation

A rough approximation of the SSDTP can be obtained by running a temperature simulation over successive periods of the application until it can be assumed that the system has reached the thermal steady state. The simulator performs the transient temperature analysis where the common approach is to solve Eq. (2) numerically, for instance, using the fourth-order Runge-Kutta method [14].

The number of iterations required to reach the SSDTP depends on the thermal characteristics of the system. In order to illustrate this aspect, we have considered an application with the period of 0.5 $s$ running on five hypothetical platforms with core areas between 1 and 25 $mm^2$. The configuration of the die and thermal package can be found in the appendix (Tab. S1). We have run the temperature simulation with HotSpot [2] for 50 successive periods. The temperature profile in each period has been compared with the actual SSDTP, obtained with our analytical approach (Sec. 6), and the normalized root mean square error (NRMSE) has been calculated. The result is shown in Fig. 1a. It can be observed that the number of successive periods over which the temperature simulation has to be performed, in order to achieve a satisfactory level of accuracy, is significant for the majority of configurations. For a 9 $mm^2$ die, for example, after 15 iterations, the NRMSE is still close to 20%. This leads to large computation times, making it difficult to apply the technique inside an intensive optimization loop.

### 4.2 Steady-State Approximation (SSA)

An approximation of the SSDTP has been proposed in [7]. Instead of solving the system of equations in Eq. (2), it is assumed that during each time interval $\Delta t_i$, in which the power is constant, the system stays in its steady state. The derivative $d\mathbf{T}/dt = 0$ and temperature can be calculated as $\mathbf{T}_i = \mathbf{G}^{-1}\mathbf{P}_i$. The result is a stepwise temperature curve where each step corresponds to the steady-state temperature $\mathbf{T}_i$ that would be reached if the constant power $\mathbf{P}_i$ was applied for a sufficiently long time.

An example of such an approximation (SSA) along with the corresponding SSDTP for an application with 10 tasks and period of 0.1 $s$ is given in Fig. 1b. The die area is 25 $mm^2$, the configuration of the chip is the same as in Tab. S1. The reduced accuracy of the SSA is due to the mismatch between the actual temperature within each interval $\Delta t_i$ and the hypothetical steady-state temperature. The inaccuracy depends on the thermal characteristics of the respective platform and on the application itself. To illustrate this, we have generated five applications with periods between 0.01 and 1 $s$ and computed approximated SSDTPs using the SSA for die areas between 1 and 25 $mm^2$. The NRMSE relative to the correct SSDTP is shown in Fig. 1c. It can be seen that, e.g., for a die area of 10 $mm^2$ and a period of 100 $ms$ the NRMSE with the SSA is close to 40%.

## 5. ANALYTICAL SOLUTION

As shown in Sec. 4, the state of the art solutions either produce inaccurate and, in many cases, completely useless results, or they are unacceptably slow. In this section we eliminate the first problem by obtaining an analytical solution for the SSDTP and tackle the second one in Sec. 6 where a fast solution technique is proposed.

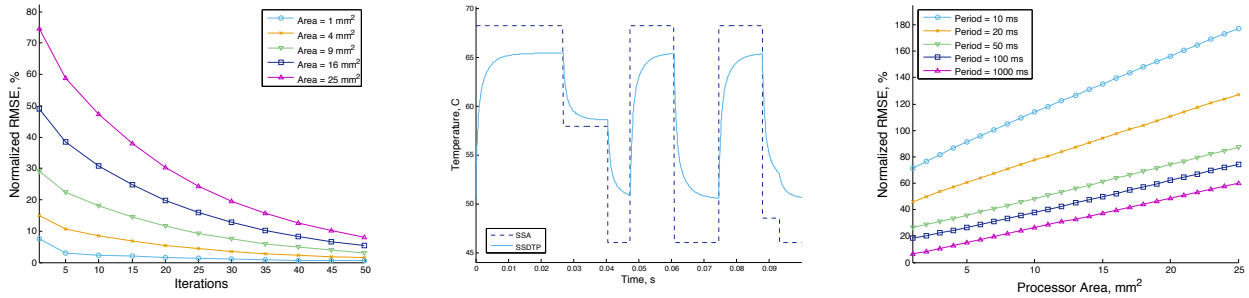
In the following explanation, without loss of generality, we assume $\mathbf{T}(t) \equiv \mathbf{T}(t) - \mathbf{T}_{\text{amb}}$. Let the power consumption vector $\mathbf{P}(t)$ be constant and equal to $\mathbf{P}$; then the system given by Eq. (2) is a system of ordinary differential equations (ODE) with the following solution:

$$\mathbf{T}(t) = e^{\mathbf{A}t}\, \mathbf{T}_0 + \mathbf{A}^{-1}(e^{\mathbf{A}t} - \mathbf{I})\, \mathbf{C}^{-1}\mathbf{P} \qquad (3)$$

where $\mathbf{A} = -\mathbf{C}^{-1}\, \mathbf{G}$, $\mathbf{T}_0$ is the initial temperature and $\mathbf{I}$ is the identity matrix. Therefore, given a discrete power profile, the corresponding temperature profile can be found using the following recurrence:

$$\mathbf{T}_{i+1} = \mathbf{K}_i\, \mathbf{T}_i + \mathbf{B}_i\, \mathbf{P}_i \qquad (4)$$

where $\mathbf{K}_i = e^{\mathbf{A}\Delta t_i}$ and $\mathbf{B}_i = \mathbf{A}^{-1}(e^{\mathbf{A}\Delta t_i} - \mathbf{I})\mathbf{C}^{-1}$. The approach can be used to perform the TTA as it is discussed in the appendix (Sec. S2.1).




**(a) NRMSE with HotSpot.** **(b) SSA and real SSDTP.** **(c) NRMSE with SSA.**

**Figure 1: State of the art solutions.**

For the SSDTA calculation the following system of linear equations can be derived from Eq. (4):

$$\begin{cases} \mathbf{K}_0\,\mathbf{T}_0 - \mathbf{T}_1 & = -\mathbf{B}_0\,\mathbf{P}_0 \\ \dots & \\ -\mathbf{T}_0 + \mathbf{K}_{N_s-1}\,\mathbf{T}_{N_s-1} & = -\mathbf{B}_{N_s-1}\,\mathbf{P}_{N_s-1} \end{cases}$$

where the last equation enforces the boundary condition, the equality of temperature values on both ends of the period:

$$\mathbf{T}_0 = \mathbf{T}_{N_s} \tag{5}$$

To get the whole picture, the system can be written as:

$$\underbrace{\begin{bmatrix} \mathbf{K}_0 & -\mathbf{I} & 0 & \cdots & 0 \\ 0 & \mathbf{K}_1 & -\mathbf{I} & & \vdots \\ \vdots & & \ddots & -\mathbf{I} & 0 \\ 0 & & & \mathbf{K}_{N_s-2} & -\mathbf{I} \\ -\mathbf{I} & 0 & \cdots & 0 & \mathbf{K}_{N_s-1} \end{bmatrix}}_{\mathbb{A}} \underbrace{\begin{bmatrix} \mathbf{T}_0 \\ \vdots \\ \mathbf{T}_{N_s-1} \end{bmatrix}}_{\mathbb{X}} = \underbrace{\begin{bmatrix} -\mathbf{B}_0\,\mathbf{P}_0 \\ \vdots \\ -\mathbf{B}_{N_s-1}\,\mathbf{P}_{N_s-1} \end{bmatrix}}_{\mathbb{B}} \tag{6}$$

where $\mathbb{A}$ is a $N_nN_s \times N_nN_s$ matrix, $\mathbb{X}$ and $\mathbb{B}$ are vectors with $N_nN_s$ elements. It can be seen that we have obtained a regular system of linear equations. Straight-forward techniques to solve it and their disadvantages are further discussed in the appendix (Sec. S2.2).

# 6. PROPOSED TECHNIQUE

In this section we propose a fast approach to solve the system in Eq. (6). The approach consists of an auxiliary transformation (Sec. 6.1) and the actual solution (Sec. 6.2).

The major problem with straight-forward techniques (see Sec. S2.2) is that (1) the sparseness of the matrix is not taken into account and/or (2) its specific structure is totally ignored, resulting in inefficiency and inaccuracy of the computations. Using direct dense and sparse solvers, for example, requires a computation time proportional to $N_n^3N_s^3$ [14]. Our proposed technique considers both features and delivers solutions in time proportional to $N_sN_n^3$ while operating only on a few $N_n \times N_n$ matrices. It is important that the dependency on $N_s$ (the number of steps in the power profile), which is by far dominating ($N_s \gg N_n$), is linear.

Observing the structure of the matrix in Eq. (6), nonzero elements are located only on the block diagonal, on one subdiagonal just above the block diagonal, and on one subdiagonal in the left bottom corner. The block diagonal is composed of $N_n \times N_n$ matrices while all elements of the subdiagonals are equal to $-1$. Linear systems with the same structure arise in boundary value problems for ODEs where a technique to solve them is to form a so-called condensed equation (CE), or condensed system [15].

## 6.1 Auxiliary Transformation

The analytical solution in Eq. (3) includes two computationally expensive operations, namely the matrix exponential and inverse involving $\mathbf{A} = -\mathbf{C}^{-1}\,\mathbf{G}$, which is an arbitrary square matrix. It is preferable to have a symmetric matrix to perform these computations, since for a real symmetric matrix $\mathbf{M}$ the following eigenvalue decomposition with independent eigenvectors holds [14]:

$$\mathbf{M} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathbf{T}} \tag{7}$$

where $\mathbf{U}$ is a square matrix of the eigenvectors, $\mathbf{U}^T$ is the transpose of $\mathbf{U}$, and $\mathbf{\Lambda}$ is a diagonal matrix of the eigenvalues $\lambda_i$ of $\mathbf{M}$. With such a decomposition, the calculation of the matrix exponential and inverse becomes trivial: $e^{\mathbf{M}} = \mathbf{U}\,e^{\mathbf{\Lambda}}\,\mathbf{U}^{\mathbf{T}}$ and $\mathbf{M}^{-1} = \mathbf{U}\,\mathbf{\Lambda}^{-1}\,\mathbf{U}^{\mathbf{T}}$, where the central matrices are diagonal with elements $e^{\lambda_i}$ and $\lambda_i^{-1}$, respectively.

The conductance matrix $\mathbf{G}$ is a symmetric matrix, since if a node A is connected to B, then B is also connected to A with the same conductance. However, as it is mentioned previously, the product of $\mathbf{G}$ with the inverse of the capacitance matrix $\mathbf{C}$ does not have this property. Since $\mathbf{C}$ is a diagonal matrix, we use the following transformation in order to keep the desired symmetry:

$$\tilde{\mathbf{T}}(t) = \mathbf{C}^{\frac{1}{2}}\mathbf{T}(t) \qquad \tilde{\mathbf{A}} = -\mathbf{C}^{-\frac{1}{2}}\mathbf{G}\,\mathbf{C}^{-\frac{1}{2}} \tag{8}$$

where $\tilde{\mathbf{A}}$ is symmetric[2]. Consequently, the system of ODEs (Eq. (2)) and its solutions (Eq. (3)) can be rewritten as the following:

$$\frac{d\tilde{\mathbf{T}}(t)}{dt} = \tilde{\mathbf{A}}\,\mathbf{Y}(t) + \mathbf{C}^{-\frac{1}{2}}\mathbf{P}$$
$$\tilde{\mathbf{T}}(t) = e^{\tilde{\mathbf{A}}t}\tilde{\mathbf{T}}_0 + \tilde{\mathbf{A}}^{-1}(e^{\tilde{\mathbf{A}}t} - \mathbf{I})\mathbf{C}^{-\frac{1}{2}}\mathbf{P}$$

where $\tilde{\mathbf{A}}$ is a symmetric matrix. Therefore, in the case of, e.g., the matrix exponential we have:

$$e^{\tilde{\mathbf{A}}t} = \mathbf{U}\,e^{\mathbf{\Lambda}t}\,\mathbf{U}^T = \mathbf{U}\,\mathrm{diag}\left(e^{t\lambda_0}, \dots, e^{t\lambda_{N_n-1}}\right)\,\mathbf{U}^T \tag{9}$$

where *diag* denotes a diagonal matrix and $\lambda_i$ are the eigenvalues of $\tilde{\mathbf{A}}$. A similar equation can be obtained for $\tilde{\mathbf{A}}^{-1}$.

The next step is to update the SSDTP system in Eq. (4):

$$\tilde{\mathbf{T}}_{i+1} = \tilde{\mathbf{K}}_i\,\tilde{\mathbf{T}}_i + \tilde{\mathbf{B}}_i\,\mathbf{P}_i \tag{10}$$
$$\tilde{\mathbf{K}}_i = e^{\tilde{\mathbf{A}}\,\Delta t_i} \qquad \tilde{\mathbf{B}}_i = \tilde{\mathbf{A}}^{-1}\left(e^{\tilde{\mathbf{A}}\Delta t_i} - \mathbf{I}\right)\mathbf{C}^{-\frac{1}{2}}$$

Using the eigenvalue decomposition, the last equation can be computed in the following way:

$$\tilde{\mathbf{B}}_i = \mathbf{U}\,\mathrm{diag}\left(\frac{e^{\Delta t_i\,\lambda_0} - 1}{\lambda_0}, \dots, \frac{e^{\Delta t_i\,\lambda_{N_n-1}} - 1}{\lambda_{N_n-1}}\right)\,\mathbf{U}^T\,\mathbf{C}^{-\frac{1}{2}}$$

## 6.2 Solution with Condensed Equation (CE)

In the recurrence given by Eq. (10) we denote $\mathbf{Q}_i = \tilde{\mathbf{B}}_i\,\mathbf{P}_i$:

$$\tilde{\mathbf{T}}_{i+1} = \tilde{\mathbf{K}}_i\,\tilde{\mathbf{T}}_i + \mathbf{Q}_i,\ i = 0, \dots, N_s - 1 \tag{11}$$
$$\tilde{\mathbf{T}}_0 = \tilde{\mathbf{T}}_{N_s}$$

Performing the iterative repetition of Eq. (11) leads to:

$$\tilde{\mathbf{T}}_i = \prod_{j=0}^{i-1}\tilde{\mathbf{K}}_j\,\tilde{\mathbf{T}}_0 + \mathbf{W}_{i-1},\ i = 1, \dots, N_s \tag{12}$$

where $\mathbf{W}_i$ are defined as follows:

$$\mathbf{W}_0 = \mathbf{Q}_0 \qquad \mathbf{W}_i = \tilde{\mathbf{K}}_i\,\mathbf{W}_{i-1} + \mathbf{Q}_i,\ i = 1, \dots, N_s - 1 \tag{13}$$

We calculate the final vector $\tilde{\mathbf{T}}_{N_s}$ using Eq. (12) and Eq. (13):

$$\tilde{\mathbf{T}}_{N_s} = \prod_{j=0}^{N_s-1}\tilde{\mathbf{K}}_j\,\tilde{\mathbf{T}}_0 + \mathbf{W}_{N_s-1}$$

[2] $\tilde{\mathbf{A}}^T = -(\mathbf{C}^{-\frac{1}{2}}\mathbf{G}\mathbf{C}^{-\frac{1}{2}})^T = -(\mathbf{C}^{-\frac{1}{2}})^T\mathbf{G}^T(\mathbf{C}^{-\frac{1}{2}})^T = \tilde{\mathbf{A}}$.

Taking into account the boundary condition given by Eq. (5), we obtain the following system of linear equations:

$$(\mathbf{I} - \prod_{j=0}^{N_s-1} \tilde{\mathbf{K}}_j)\, \tilde{\mathbf{T}}_0 = \mathbf{W}_{N_s-1} \tag{14}$$

We recall that $\tilde{\mathbf{K}}_i$ is the matrix exponential given by Eq. (9); therefore, the following simplification holds:

$$\prod_{j=0}^{N_s-1} \tilde{\mathbf{K}}_j = \mathbf{U}\ \mathrm{diag}\left(e^{\tau\lambda_0}, \ldots, e^{\tau\lambda_{N_n-1}}\right)\ \mathbf{U}^T$$

where $\tau$ is the application period. Substituting this product into Eq. (14), we obtain the following system:

$$(\mathbf{I} - \mathbf{U}\, e^{\tau\mathbf{\Lambda}}\, \mathbf{U}^T)\, \tilde{\mathbf{T}}_0 = \mathbf{W}_{N_s-1}$$

The identity matrix $\mathbf{I}$ can be split into $\mathbf{U}\mathbf{U}^T$, hence:

$$\tilde{\mathbf{T}}_0 = \mathbf{U}\,(\mathbf{I} - e^{\tau\mathbf{\Lambda}})^{-1}\, \mathbf{U}^T\, \mathbf{W}_{N_s-1} = \mathbf{Z}\, \mathbf{W}_{N_s-1} \tag{15}$$

where:

$$\mathbf{Z} = \mathbf{U}\ \mathrm{diag}\left(\frac{1}{1-e^{\tau\lambda_0}}, \ldots, \frac{1}{1-e^{\tau\lambda_{N_n-1}}}\right)\ \mathbf{U}^T$$

The equation gives the initial solution vector $\tilde{\mathbf{T}}_0$; the rest of the vectors $\tilde{\mathbf{T}}_i$ are successively found from Eq. (11).

Since the power profile is evenly sampled with the sampling interval $\Delta t$, the recurrence in Eq. (11) turns into:

$$\tilde{\mathbf{T}}_{i+1} = \tilde{\mathbf{K}}\, \tilde{\mathbf{T}}_i + \mathbf{Q}_i = \tilde{\mathbf{K}}\, \tilde{\mathbf{T}}_i + \tilde{\mathbf{B}}\, \mathbf{P}_i$$

where $\tilde{\mathbf{K}} = e^{\tilde{\mathbf{A}}\,\Delta t}$ and $\tilde{\mathbf{B}} = \tilde{\mathbf{A}}^{-1}(e^{\tilde{\mathbf{A}}\,\Delta t} - \mathbf{I})\mathbf{C}^{-\frac{1}{2}}$. Here $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{B}}$ are constants, since they depend only on the matrices $\tilde{\mathbf{A}}$, $\mathbf{C}$, and sampling interval $\Delta t$, which is fixed. In this case, the block diagonal of the matrix $\tilde{\mathbb{A}}$, similar to Eq. (6), is composed of the same repeating block $\tilde{\mathbf{K}}$ and the recurrent expressions take the following form:

$$\mathbf{W}_i = \tilde{\mathbf{K}}\, \mathbf{W}_{i-1} + \mathbf{Q}_i,\ i = 1, \ldots, N_s - 1 \tag{16}$$

$$\tilde{\mathbf{T}}_{i+1} = \tilde{\mathbf{K}}\, \tilde{\mathbf{T}}_i + \mathbf{Q}_i,\ i = 0, \ldots, N_s - 1 \tag{17}$$

where $\mathbf{Q}_i = \tilde{\mathbf{B}}\, \mathbf{P}_i$, $\mathbf{W}_0 = \mathbf{Q}_0$, and $\tilde{\mathbf{T}}_0$ is given by Eq. (15).

The last step of the solution is to return to temperature by performing the backward substitution opposite to Eq. (8):

$$\mathbf{T}_i = \mathbf{C}^{-\frac{1}{2}}\, \tilde{\mathbf{T}}_i,\ i = 0, \ldots, N_s - 1$$

As we see, the auxiliary substitution from Sec. 6.1 allows us to perform the single-time eigenvalue decomposition with orthogonal eigenvectors (Eq. (7)) that later eases the computational process at several stages. In Sec. 6.2 it can be observed that the solution of the system in Eq. (6) has been reduced to two successive recurrences in Eq. (16) and Eq. (17) over $N_s$ steps in the power profile, which implies a linear complexity on $N_s$ mentioned earlier.

It should be noted that the eigenvalue decomposition along with matrices $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{B}}$ are computed only once for a particular RC thermal circuit and can be considered as given together with the RC circuit. It has not to be recalculated when a SSDTP is generated, which significantly decreases the computation time.

# 7. LEAKAGE POWER

So far, we have assumed that power is independent of temperature. However, due to the leakage component, the power dissipation is a strong function of temperature that cannot be neglected (Sec. 2). Two techniques can be applied to include in our proposed solution temperature-dependent leakage modeling.

## 7.1 Iterative Computation

In this case, we have an iterative process, depicted in Fig. 2, where the temperature and power profiles are calculated in turns. With each new temperature profile we update the power profile by computing the leakage power and adding it to the dynamic power: $\mathbb{P}_i = \mathbb{P}_{\mathrm{dyn}} + \mathbb{P}_{\mathrm{leak}}(\mathbb{T}_i)$. The process continues until the temperature converges, i.e., the

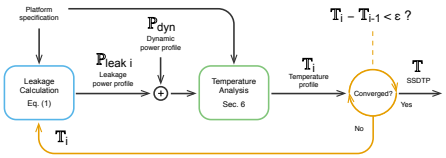



**Figure 2: SSDTP with leakage modeling.**

difference between two successive temperature profiles is below a predefined bound. In our experiments we used $0.5°C$ as the maximal acceptable difference and observed that the number of required iterations to converge is 4–7.

## 7.2 Linear Approximation

A linear approximation of the leakage power has the following matrix form: $\mathbf{P}_{\mathrm{leak}}(\mathbf{T}) = \mathbf{A}\,\mathbf{T}(t) + \mathbf{B}$ where $\mathbf{A}$ is a $N_n \times N_n$ diagonal matrix of the proportionality and $\mathbf{B}$ is a vector with $N_n$ elements of the intercept. Both characterize the leakage power for each of the $N_n$ thermal nodes in the system. It can be seen that the approximation keeps Eq. (2) untouched: $\mathbf{C}\,\frac{d\mathbf{T}(t)}{dt} + \bar{\mathbf{G}}\,(\mathbf{T}(t) - \mathbf{T}_{\mathrm{amb}}) = \bar{\mathbf{P}}$ where $\bar{\mathbf{G}} = \mathbf{G} - \mathbf{A}$ and $\bar{\mathbf{P}} = \mathbf{P}_{\mathrm{dyn}} + \mathbf{A}\,\mathbf{T}_{\mathrm{amb}} + \mathbf{B}$. Therefore, all solutions proposed in this paper are perfectly valid with the linearized model. Moreover, in spite of its simplicity, the model provides a good estimation, as shown in [6].

In order to evaluate the linearization, we have constructed a number of hypothetical platforms with 2–32 cores (other parameters are given in Tab. S1) and compared temperature profiles obtained with the linearization and the exponential model (Sec. 2), respectively. For the later, we use the iterative approach described in Sec. 7.1. For the linearization, the power curve fitting with the least squares regression [14] has been employed, targeted at the range between 40 and $80°C$. From the experiments we have observed that the NRMSE is bounded by 1–2%, indicating a good accuracy of the linear approximation.

# 8. RELIABILITY OPTIMIZATION

The proposed calculation of the SSDTA can be used in a wide range of system optimizations. One of them is reliability optimization that we discuss in this section. We perform a temperature-aware task mapping and scheduling in order to address the thermal cycling fatigue.

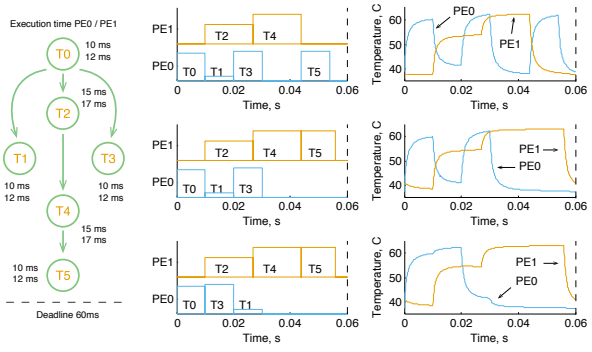
## 8.1 Application Model

The periodic application is modeled as a task graph $G = (V, E, \tau)$ where $V$ is a set of $N_t$ tasks (vertices of the graph), $E$ is a set of data dependencies between tasks (edges), and $\tau$ is the period of the application, which we assume to be equal to the deadline. Each pair of a task $v_i \in V$ and processing element $\pi_j \in \Pi$ is characterized by a tuple $(N_{\mathrm{clock}\,ij}, C_{\mathrm{eff}\,ij})$, where $N_{\mathrm{clock}\,ij}$ is the number of clock cycles and $C_{\mathrm{eff}\,ij}$ is the effective switched capacitance.

## 8.2 Temperature-Aware Reliability Model

We address temperature-driven failure mechanisms with the reliability model presented in [7, 8]. In this paper, our particular focus is on the thermal cycling (TC) fatigue, which is directly connected to the temperature variations. The derivation of the model is given in the appendix (Sec. S3).

Assuming the TC fatigue, the parameters affecting reliability are the amplitude and number of thermal cycles as




**(a) Graph. (b) Mappings, schedules, SSDTPs.**

**Figure 3: Motivational example.**

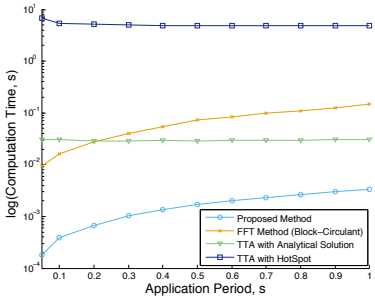



**Figure 4: Scaling with $\tau$.**

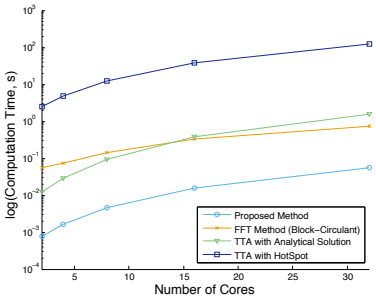



**Figure 5: Scaling with $N_p$.**

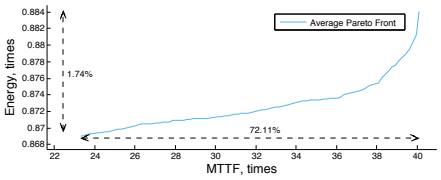



**Figure 6: Average Pareto front.**

well as the maximal temperature. A thermal cycle is a time interval in which the temperature starts from a certain value and, after reaching an extremum, returns back.

The mean time to failure (MTTF) of one processing element in the system can be estimated as the following:

$$\theta = \frac{\tau}{\sum_{i=0}^{N_m-1} \frac{1}{N_{c\,i}}} \tag{18}$$

where $N_m$ is the number of thermal cycles during the application period $\tau$. $N_{c\,i}$ characterizes the $i$th thermal cycle and is calculated according to the following expression:

$$N_c = A(\Delta T - \Delta T_0)^{-b} e^{\frac{E_a}{kT_{\max}}} \tag{19}$$

where $\Delta T$ is the thermal cycle excursion (the distance between the minimal and maximal temperatures) and $T_{\max}$ is the maximal temperature during the thermal cycle (more details in Sec. S3).

It can be seen that the computation requires the identification of the thermal cycles with their amplitudes and maximal temperatures. All these are captured by the SSDTP, which is needed as an input to the reliability optimization.

## 8.3 Motivational Example

Consider an application with six tasks, denoted "T0"–"T5", and a heterogeneous architecture with two cores, labeled "PE0" and "PE1". The task graph of the application is given in Fig. 3a along with the execution times for both cores. The period of the application is 0.06 $s$. A first alternative mapping and schedule, and the resulting SSDTP are shown at the top of Fig. 3b (where the height of a task represents its relative dynamic power consumption). It can be observed that initially PE0 is experiencing three thermal cycles. If we change the mapping of T5 and move it to PE1, we achieve two thermal cycles of PE0 instead of three. Finally, if we vary the schedule as well and change the order of T1 and T3, the number of cycles of PE0 becomes one. Using the reliability model from Sec. 8.2, we observe improvements in the lifetime of 44.69% and 54.53%, respectively, relative to the initial configuration.

## 8.4 Problem Formulation and Optimization

The problem formulation is the following:

Given:

- A multiprocessor system $\Pi$ (Sec. 2).
- A periodic application $G$ (Sec. 8.1).
- The floorplan of the chip at the desired level of details, configuration of the thermal package, and thermal parameters.
- The parameters of the reliability model (Sec. 8.2), i.e., the constants $A$, $\Delta T_0$, $b$, $E_a$ (see Eq. (19)).

Maximize:

$$\mathcal{F} = \min_{i=0}^{N_p-1} \theta_i \tag{20}$$

s.t.

$$t_{\text{end}\,i} \leq \tau,\ \forall i \tag{21}$$

$$T_{ij} \leq T_{\max},\ \forall i, j \tag{22}$$

where $\theta_i$ is the MTTF of the $i$th processing element given by Eq. (18), $t_{\text{end}\,i}$ denotes the end time of the $i$th task, $\tau$ is the period of the application, and $T_{ij}$ are temperature values in the SSDTP. Eq. (21) imposes the application deadline, which we assume to be equal to the period. Eq. (22) enforces the constraint on the maximal temperature in the temperature profile $\mathbb{T} = \{T_{ij}\}$.

The optimization procedure is based on a genetic algorithm (GA) [16] with the fitness function $\mathcal{F}$ given by Eq. (20). The algorithm is outlined in Sec. S3.2.

# 9. EXPERIMENTAL RESULTS

## 9.1 SSDTP Calculation

In this subsection we investigate the scaling properties of the proposed solution for the SSDTP calculation and compare it with the approach based on the TTA with HotSpot (Sec. 4.1)[3]. We also include in the comparison two additional techniques described in the appendix, namely the TTA with the analytical solution (Sec. S2.1) and the fast Fourier transform (FFT) (Sec. S2.2). In the cases of the TTA, the simulation over successive iterations is run until the NRMSE relative to the SSDTP obtained with the proposed method is less than 1%.

In the following experiments, the power sampling interval is set to 1 $ms$ and the thermal configuration of the die is the same as in Tab. S1. For the experiments in this subsection, the leakage power has not been considered. If considered according to the linearized model (Sec. 7.2), execution times remain unchanged; if considered according to the iterative model (Sec. 7.1), execution times increase proportionally for all the methods, which does not affect any of the conclusions.

First, we vary the application period $\tau$ keeping the architecture fixed, which is a quad-core platform with the core area of 4 $mm^2$. The comparison is depicted in Fig. 4 on a semilogarithmic scale. It can be seen that the proposed technique is roughly 5000 times faster than calculating the SSDTP by running the TTA with HotSpot and from 9 to 170 times faster than the TTA with the analytical solution.

In the second experiment we evaluate the scaling of the proposed method with regard to the number of processing elements. The application period is fixed to 0.5 $s$. The results are shown in Fig. 5. It can be observed that the proposed technique provides a significant performance improvement relative to the alternative solutions.

## 9.2 Reliability Optimization

In this section we evaluate the reliability optimization approach described in Sec. 8, first with a set of synthetic applications and, finally, using a real-life example.

The experimental setup is the following. Heterogeneous platforms and periodic applications are generated randomly [17] in such a way that the execution time of tasks is uniformly distributed between 1 and 10 $ms$ and the leakage power accounts for 30–60% of the total power dissipation[4]. The linear leakage model is used in the experiments, since, as discussed in Sec. 7.2, it provides a good approximation. The area of one core is 4 $mm^2$, other parameters of the die and thermal package are given in Tab. S1. The temperature constraint $T_{\max}$ (see Eq. (22)) is set to 100°$C$. In Eq. (19) the Coffin-Manson exponent $b$ is set to 6, the activation energy $E_a$ to 0.5, and the elastic temperature region $\Delta T_0$ to zero [13]. The coefficient of proportionality $A$ is not significant, since we are concerned about the relative improvement.

In each of the experiments, we compare the optimized solution with an initial temperature-aware solution proposed

[3]All the experiments are performed on a Linux machine with Intel® Core™ i7-2600 3.4GHz and 8Gb of RAM.

[4]The parameters of the applications and platforms (task graphs, floorplans, HotSpot configurations, etc.) used in our experiments are available online at [18].

**Table 1: Optimization results.**

**(a) Different numbers of cores.**

| $N_p$ | $N_t$ | $t, s$ | $\text{MTTF}_\times$ | $\text{E}_\times$ |
|---|---|---|---|---|
| 2 | 40 | 7.84 | 39.41 | 0.97 |
| 4 | 80 | 65.76 | 37.11 | 0.99 |
| 8 | 160 | 759.29 | 31.36 | 0.97 |
| 16 | 320 | 3484.59 | 13.51 | 0.98 |

**(b) Different application sizes.**

| $N_p$ | $N_t$ | $t, s$ | $\text{MTTF}_\times$ | $\text{E}_\times$ |
|---|---|---|---|---|
| 4 | 40 | 9.96 | 64.53 | 0.88 |
| 4 | 80 | 56.57 | 38.01 | 0.96 |
| 4 | 160 | 352.20 | 18.08 | 1.07 |
| 4 | 320 | 408.42 | 12.92 | 1.05 |

**(c) Different techniques.**

| $N_p$ | $N_t$ | $\text{MTTF}_\times^{PM}$ | $\text{MTTF}_\times^{HS}$ | $\text{MTTF}_\times^{SSA}$ |
|---|---|---|---|---|
| 4 | 40 | 64.53 | 1.29 | 25.10 |
| 4 | 80 | 38.01 | 1.67 | 13.87 |
| 4 | 160 | 18.08 | 2.02 | 5.33 |
| 4 | 320 | 12.92 | 1.72 | 3.82 |

in [19]. This solution consists of a task mapping and schedule that captures the spatial temperature behavior and tries to minimize the peak temperature while satisfying the real-time constraints. The deadline is set to the duration of the initial schedule extended by 5%.

In the first set of experiments, we change the number of cores $N_p$ while keeping the number of tasks $N_t$ per core constant and equal to 20. For each problem we have generated 20 random task graphs and found the average improvement of the MTTF over the initial solution ($\text{MTTF}_\times$). We also have measured the change in the consumed energy ($\text{E}_\times$). The results are given in Tab. 1a ($t$ indicates the optimization time in seconds). It can be seen that the reliability-aware optimization dramatically increases the MTTF by 13 up to 40 times. Even for large applications with, e.g., 320 tasks deployed onto 16 cores, a feasible mapping and schedule that significantly improve the lifetime of the system can be found in an affordable time. Moreover, our optimization does not impact the energy efficiency of the system.

For the second set of experiments, we keep the quad-core architecture and vary the size (number of tasks $N_t$) of the application. The number of randomly generated task graphs per application size is 20. The average improvement of the MTTF along with the change in the energy consumption are given in Tab. 1b. The observations are similar to those for the previous set of experiments.

The above experiments have confirmed that our proposed approach is able to effectively increase the MTTF of the system. The efficiency of this approach is due to the fast and accurate SSDTP calculation, which is at the heart of the optimization, and which, due to its speed, allows a huge portion of the design space to be explored. In order to prove this, we have replaced, inside our optimization framework, the proposed SSDTP calculation with the calculation based on HotSpot (Sec. 4.1) and based on the SSA (Sec. 4.2), respectively. The goal is to compare our results with the results produced using HotSpot and the SSA, after the same optimization time as needed with the proposed SSDTP calculation technique. The experimental setup is the same as for the experiments in Tab. 1b. The MTTF obtained with HotSpot and the SSA is evaluated and compared with the MTTF obtained by our proposed method. The results are summarized in Tab. 1c. For example, the lifetime of the platform running 160 tasks can be extended by more than 18 times, compared to the initial solution, using our approach, whereas, the best solutions found with HotSpot and the SSA, using the same optimization time, are only 2.02 and 5.33 times better, respectively. The reason for the poor results with HotSpot is the excessively long execution time of the SSDTP calculation. This allows for a much less thorough investigation of the solution space than with our proposed technique. In the case of the SSA, the reason is different. The SSA is fast but also very inaccurate (Sec. 4.2). The inaccuracy drives the optimization towards solutions that turn out to be of low quality.

We have seen that our reliability-targeted optimizations have significantly increased the MTTF without affecting the energy consumption. This is not surprising, since our optimization will search towards low temperature solutions, which implicitly means low leakage. In order to further explore this aspect, we have performed a multi-objective optimization[5] along the dimensions of energy and reliability. An example of the Pareto front averaged over 20 applications with 80 tasks deployed onto a quad-core platform is given in Fig. 6. It can be observed that the variation of energy is less than 2%. This means that solutions optimized for the MTTF have an energy consumption almost identical to those optimized for energy. At the same time, the difference along the MTTF is huge. This means that ignoring the reliability aspect one may end up with a significantly decreased MTTF, without any significant gain in energy.

Finally, we have applied our optimization technique to a real-life example, namely the MPEG2 video decoder [21] that is deployed onto a dual-core platform. The decoder was analyzed and split into 34 tasks. The parameters of each task were obtained through a system-level simulation using MPARM [22]. The deadline is set to 40 $ms$ assuming 25 video frames per second. The solution found with the proposed method improves the lifetime of the system by 23.59 times with a 5% energy saving, compared to the initial solution. The same optimization was solved using HotSpot and the SSA. The best found solutions are only 5.37 and 11.50 times better than the initial one, respectively.

[5]The multi-objective optimization is based on NSGA-II [20].

# 10. CONCLUSION

In this paper we have proposed an efficient and accurate technique to calculate the SSDTP of an embedded multiprocessor system. Using the proposed approach, we conducted a temperature-aware reliability optimization based on the thermal cycling failure mechanism and have shown that taking into consideration the temperature variations within a multicore platform can significantly prolong its lifetime without affecting its energy efficiency. The improvement, compared using the state of the art, is significant.

# 11. ACKNOWLEDGMENTS

We would like to thank Prof. Åke Björck from Linköping University for the valuable discussions and suggestions about the analytical solution.

# 12. REFERENCES

[1] F. Kreith. *CRC Handbook of Thermal Engineering*. CRC Press, 2000.
[2] K. Skadron et al. Temperature-aware microarchitecture. In *ISCA*, pages 2–13, 2003.
[3] J. Srinivasan et al. The impact of technology scaling on lifetime reliability. In *DSN*, pages 177–186, 2004.
[4] W. Liao et al. Temp. and supply voltage aware performance and power modeling at mictoarchitecture level. *IEEE Trans. CAD*, 24(7):1042–1053, 2005.
[5] A. K. Coskun et al. Analysis and optimization of MPSoC reliability. *J. of Low Power Electronics*, 2(1):56–69, 2006.
[6] Y. Liu et al. Accurate temperature-dependent IC leakage power estimation is easy. In *DATE'07*, 2007.
[7] L. Huang et al. Lifetime reliablity-aware task allocation and scheduling for MPSoCs. In *DATE'09*, 2009.
[8] Y. Xiang et al. System-level reliability modeling for MPSoCs. In *CODES+ISSS'10*, 2010.
[9] L. Thiele et al. Thermal-aware sys. analysis and software synthesis for embedded multi-processors. In *DAC'11*, 2011.
[10] D. Rai et al. Worst-case temperature analysis for real-time systems. In *DATE'11*, 2011.
[11] M. Bao et al. Temp.-aware idle time distribution for energy optim. with dynamic voltage scaling. In *DATE'10*, 2010.
[12] R. Rao et al. Fast and accurate prediction of the steady-state throughput of multicore processors under thermal constraints. *IEEE Trans. CAD of ICs and Systems*, 28(10):1559–1572, 2009.
[13] JEDEC. *Failure Mechanisms and Models for Semiconductor Devices*. JEDEC Publication, 2010.
[14] W. H. Press et al. *Numerical Recipes*. Cambridge University Press, 3rd edition, 2007.
[15] J. Stoer et al. *Introduction to Numerical Analysis*. Springer-Verlag, New York, 3rd edition, 2002.
[16] M. T. Schmitz et al. *Sys.-Level Design Techniques for Energy-Efficient Embed. Sys.* Kluwer Academ. Pub., 2004.
[17] R. P. Dick et al. TGFF: Task graphs for free. In *CODES/CASHE'98*, 1998.
[18] `http://www.ida.liu.se/~ivauk83/research/SSDTA`.
[19] Y. Xie et al. Temperature-aware task allocation and scheduling for embedded MPSoC design. *Journal of VLSI Signal Processing*, 45(3):177–189, 2006.
[20] K. Deb et al. A fast and elitist multiobjective GA: NSGA-II. *IEEE Trans. Evol. Comp.*, 6(2), 2002.
[21] `http://ffmpeg.mplayerhq.hu`.
[22] L. Benini et al. MPARM: Expl. MPSoC design space with SystemC. *J. of VLSI Sig. Proc. Sys.*, 41(2):169–182, 2005.

# APPENDIX

## S1. RC THERMAL CIRCUIT

The equivalent circuit of a multiprocessor system with a thermal package can be built in different ways depending on the intended level of details. Consequently, the number of nodes $N_n$ and structure of the matrices $\mathbf{C}$ and $\mathbf{G}$ in Eq. (2) depend on the particular model. Thermal nodes that belong to the package are called inactive, in the sense that their power dissipation is assumed to be zero.

Without loss of generality, in this paper we use thermal circuits where each of the $N_p$ processing elements is captured by one thermal node. Similar to [2], in the model, three cooling layers are present, namely the thermal interface material, heat spreader, and heat sink captured by $N_p$, $N_p + 4$, and $N_p + 8$ inactive thermal nodes, respectively. Therefore, the total number of thermal nodes $N_n$ is $4 \times N_p + 12$. The parameters of the die and thermal package, used throughout this paper, are given in Tab. S1.

A simplified example of such a thermal circuit for a dual-core architecture is depicted in Fig. S1. It can be seen that the inter-core thermal influence is taken into account by modeling the heat flux between the cores (the top two thermal nodes) with the corresponding thermal resistance.

Some or all cores can be also modeled at a finer level of granularity, where caches, ALUs, or registers will be captured as individual thermal nodes.

## S2. ANALYTICAL SOLUTION

In this section we further discuss the analytical solution from Sec. 5, its application for the TTA, and possible solution techniques in the case of the SSDTA.

### S2.1 Transient Temperature Analysis (TTA)

The recurrence obtained with the analytical solution in Eq. (3) is the following (Eq. (4)):

$$\mathbf{T}_{i+1} = \mathbf{K}_i \, \mathbf{T}_i + \mathbf{B}_i \, \mathbf{P}_i$$

Given the initial temperature $\mathbf{T}_0$, it can be applied to perform the TTA. Our experiments show that, since intervals $\Delta t_i$ have the same length and matrices $\mathbf{K}_i$ and $\mathbf{B}_i$ become constant, this approach produces a significant performance improvement compared to iterative solutions of ODEs, e.g., the fourth-order Runge-Kutta method used in HotSpot. The same observation is made in [9].

The TTA using the analytical technique given in Eq. (4) can be employed to approximate the SSDTP by applying it over successive application periods, as shown in Sec. 4.1. Since each iteration, with this approach, is much faster than with HotSpot, it will significantly speed up the SSDTP calculation. However, the number of required iterations is similar to the case when HotSpot is used (see Fig. 1a), still keeping the computational process slow (Sec. 9.1).

### S2.2 Straight-Forward Solutions (SSDTA)

The first straight-forward way to solve the system in Eq. (6) is to use dense solvers such as the LU decomposition [14]. However, a more advanced approach is to employ sparse solvers since the matrix of the system is a sparse matrix. Therefore, algorithms specially designed for such cases are preferable, e.g., the unsymmetric multifrontal method [S1]. The computational complexity of the solution is proportional to $N_s^3 N_n^3$ [14] where $N_n$ is the number of nodes and $N_s$ is the number of steps in the power profile. The problem here is that the systems to solve can be extremely large, in particular due to $N_s$. Our experiments have shown that direct solvers are extremely slow and consume a large amount of memory. Therefore, we do not consider them in the paper.

**Table S1: Parameters of the die and package.**

| Parameter | Value |
|---|---|
| Ambient temperature | 27 °C |
| Convection capacitance | 140.4 J/K |
| Convection resistance | 0.1 K/W |
| Die thickness | 0.15 *mm* |
| Thermal interface material thickness | 0.02 *mm* |
| Heat spreader side | 20 *mm* |
| Heat spreader thickness | 1 *mm* |
| Heat sink side | 30 *mm* |
| Heat sink thickness | 15 *mm* |

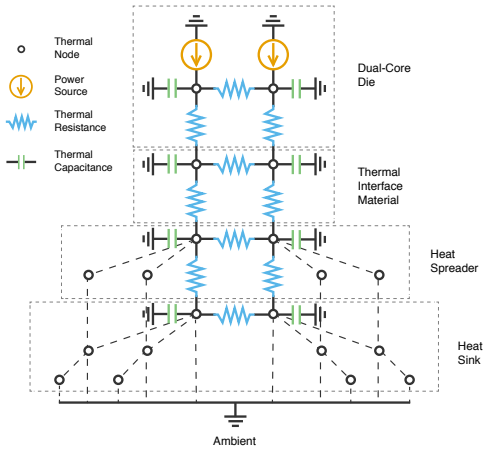



**Figure S1: Equivalent RC thermal circuit.**

The overall matrix of the system in Eq. (6) is, in fact, a block Toeplitz matrix. To be more specific, the matrix is a block-circulant matrix where each block row vector is rotated one block element to the right relative to the preceding block row vector. This leads to a wide range of possible techniques to solve the system, e.g., the fast Fourier transform (FFT) [S2] that we include in our experiments in Sec. 9.1.

Another possible technique is iterative methods for solving systems of linear equations (e.g., Jacobi, Gauss–Seidel, Successive Overrelaxation) [14]. These methods are designed to overcome problems of direct solvers and, consequently, they are applicable for very large systems. However, the most important issue with these methods is their convergence. In our experiments we did not observe any advantages of using these methods compared to the others considered in this paper. Therefore, they are excluded from the discussion.

## S3. RELIABILITY OPTIMIZATION

This section contains the derivation of the reliability model discussed in Sec. 8 and the description of the actual optimization procedure.

### S3.1 Temperature-Aware Reliability Model

In our analysis, we use the reliability model presented in [7, 8]. The model is based on the assumption that the time to failure $\mathcal{T}$ has a Weibull distribution, i.e., $\mathcal{T} \sim Weibull(\eta, \beta)$ where $\eta$ and $\beta$ are the scaling and shape parameters, respectively. The expectation of the distribution is the following:

$$\mathbb{E}[\mathcal{T}] = \eta \, \Gamma(1 + \frac{1}{\beta}) \tag{23}$$

where $\Gamma$ is the gamma function. $\mathbb{E}[\mathcal{T}]$ is the mean time to failure (MTTF) that we denote by $\theta$.

The shape parameter $\beta$ is independent of the temperature variation [S3], which, however, is not the case with the scaling parameter $\eta$. Therefore, the distribution varies with the temperature. We can split the overall period of the application $\tau$ into $N_m$ time intervals $\Delta t_i$, so that during each time interval $\Delta t_i$ the corresponding $\eta_i$ is a constant:

$$\eta_i = \frac{\theta_i}{\Gamma(1 + \frac{1}{\beta})} \tag{24}$$

where $\theta_i$ is the MTTF in the $i$th time interval as if we had the failure distribution of this interval all the time. For now the values $\theta_i$ are unknown and depend on the particular failure mechanism. As it is shown in [8], the reliability function $R(t)$, i.e., the probability of survival until an arbitrary time $t \geq 0$, can be approximated as the following:

$$R(t) = e^{-(\frac{t}{\tau} \sum_{i=0}^{N_m - 1} \frac{\Delta t_i}{\eta_i})^{\beta}}$$

The formula keeps the form of the Weibull distribution with the scaling parameter equal to:

$$\eta = \frac{\tau}{\sum_{i=0}^{N_m - 1} \frac{\Delta t_i}{\eta_i}} \tag{25}$$

The MTTF with respect to the whole application period can be obtained by combining Eq. (23), Eq. (24), and Eq. (25).

As mentioned previously, in order to compute the MTTF, we need to consider the particular failure mechanism and determine the values $\theta_i$ needed in Eq. (24). We focus on the thermal cycling fatigue (Sec. 8.2). Assuming this concrete failure model, the duration $\Delta t_i$, during which the corresponding scaling parameter $\eta_i$ is constant Eq. (24), is exactly a thermal cycle.

When the system is exposed to identical thermal cycles, the number of such cycles to failure can be estimated using a modified version of the well-known Coffin-Manson equation with the Arrhenius term [8, 13]:

$$N_c = A(\Delta T - \Delta T_0)^{-b} e^{\frac{E_a}{kT_{\max}}}$$

where $A$ is an empirically determined constant, $\Delta T$ is the thermal cycle excursion, $\Delta T_0$ is the portion of the temperature range in the elastic region which does not cause damage, $b$ is the Coffin-Manson exponent, which is also empirically determined, $E_a$ is the activation energy, $k$ is the Boltzmann constant, and $T_{\max}$ is the maximal temperature during the thermal cycle. Over the application period, the system undergoes a number of different thermal cycles each with its own duration $\Delta t_i$ and each cycle causes its own damage. Therefore, having $N_m$ thermal cycles characterized by the number of cycles to failure $N_{c\,i}$ and duration $\Delta t_i$, we can compute $\theta_i$:

$$\theta_i = N_{c\,i}\ \Delta t_i \tag{26}$$

Taking equations (23), (24), (25), and (26) together, we obtain the following expression to estimate the MTTF of one component in the system:

$$\theta = \frac{\tau}{\sum_{i=0}^{N_m-1} \frac{1}{N_{c\,i}}} \tag{27}$$

In order to identify thermal cycles in the temperature curve, we follow the approach given in [8] where the rainflow counting method is employed.

## S3.2 Optimization Procedure

The optimization procedure is based on a genetic algorithm [16] with the fitness function $\mathcal{F}$ given by Eq. (20). Each chromosome is a vector of $2 \times N_t$ elements, where the first half encodes priorities of the tasks and the second represents a mapping. The population contains $4 \times N_t$ individuals that are initialized partially randomly and partially based on the initial temperature-aware solution [19]. In each generation, a number of individuals, called parents, are chosen for breeding by the tournament selection with the number of competitors proportional to the population size. The parents undergo the 2-point crossover with 0.8 probability and uniform mutation with 0.01 probability. The evolution mechanism follows the elitism model where the best individual always survives. The stopping condition is an absence of improvement within 200 successive generations.

The fitness of a chromosome, Eq. (20), is evaluated in a number of steps. First, the decoded priorities and mapping are given to a list scheduler that produces schedules for each of the cores. If the application schedule does not satisfy the deadline, the solution is penalized proportionally to the delay and is not further evaluated; otherwise, based on the parameters of the architecture and tasks, a power profile is obtained and the corresponding SSDTP is computed by our proposed method. If the SSDTP violates the temperature constraint given by Eq. (22), the solution is penalized proportionally to the amount of violation and not further processed; otherwise, the MTTF of each core is estimated according to Eq. (18) and the fitness function $\mathcal{F}$ is computed.

# S4. REFERENCES


[S1] T. A. Davis. Algorithm 832: UMFPACK. *ACM Trans. Mathematical Software*, 30(2):196–199, 2004.

[S2] T. De Mazancourt et al. The inverse of a block-circulant matrix. *IEEE Trans. Anten. Prop.*, 31(5):808–810, 1983.

[S3] S.-C. Chang et al. Electrical characteristics and reliability properties of MOSFET with Dy2O3 gate dielectric. *Applied Physics Letters*, 89(5), 2006.