Work-in-Progress: Towards a fine-grain thermal model for uniform multi-core processors

Javier Pérez Rodríguez
Patrick Meumeu Yomsi

CISTER-TR-201004

2020/12/01

Work-in-Progress: Towards a fine-grain thermal model for uniform multi-core processors

Javier Pérez Rodríguez, Patrick Meumeu Yomsi

CISTER Research Centre
Polytechnic Institute of Porto (ISEP P.Porto)
Rua Dr. António Bernardino de Almeida, 431
4200-072 Porto
Portugal
Tel.: +351.22.8340509, Fax: +351.22.8321159
E-mail: perez@isep.ipp.pt, pmy@isep.ipp.pt
https://www.cister-labs.pt

Abstract

On-chip power dissipation is recognized as one of the primary limiters, if not a show stopper, of performance for high-end safety-critical uniform multi-core processors. This paper proposes an efficient and simple thermal model for such a platform to be coupled with the large variety of schedulers designed to control the processor activity and the triggering of the cooling mechanism with as little impact on performance as possible.

Work-in-Progress: Towards a fine-grain thermal model for uniform multi-core processors

Javier Pérez Rodríguez and Patrick Meumeu Yomsi
CISTER Research Centre, ISEP, Polytechnic Institute of Porto, Portugal
Email: {perez,pmy}@isep.ipp.pt

Abstract—**On-chip power dissipation is recognized as one of the primary limiters, if not a show stopper, of performance for high-end safety-critical uniform multi-core processors. This paper proposes an efficient and simple thermal model for such a platform to be coupled with the large variety of schedulers designed to control the processor activity and the triggering of the cooling mechanism with as little impact on performance as possible.**

I. Introduction

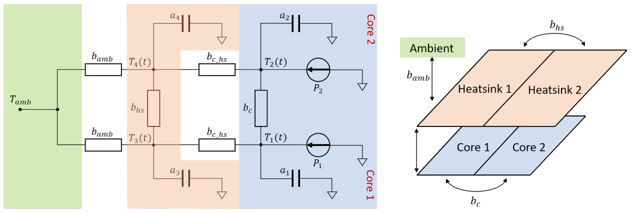
For several decades now, critical real-time applications have consistently been under the spotlights of experts because they exposed stringent functional and non-functional requirements that have to be met. In general, these applications are modeled by using a finite set of recurrent tasks to be executed on a targeted hardware platform. While the functional correctness of each task is important, the time it takes for a result to be produced is essential for these applications. Thus, several factors have to be considered at design time. Examples include the task interactions, concurrency, interference at the software level, and all the mechanisms that govern the execution of the tasks (preferably with a great level of detail) at the hardware level. To date, an entire body of knowledge and techniques has been proposed in the literature. However, new challenges arise almost on a daily basis due to the ever growing complexity and computational demand of the applications; and/or the Non-Disclosure Agreements signed on valuable information on the targeted platform by hardware vendors.

Despite these limitations, the integration of more and more processing elements in smaller silicon areas has become a reality [1], [2]. Hence, (i) the necessity for hardware miniaturization; and (ii) the ever increasing computational demand of the applications, together, have highlighted a serious problem: *the soaring power dissipation of the integrated circuits, which in turn translates in temperature dissipation*. Obviously, high temperatures create a number of problems, because transistors may fail to switch properly and therefore can lead to transient and/or permanent errors for the entire system. Consequently, it is important to build a robust and preferably simple thermal model that will allow us to predict beforehand the temperature of a critical real-time system upon the execution of a given workload. This will be the main focus of this paper. In the literature, the problem has mostly been addressed by using one of the following two strategies: (1) switching-off some core(s) [3], [4]; or (2) re-scaling the cores speed [5]–[7]. In either case, action is taken only when the reported temperature by the thermal sensor rises above a predefined threshold. Below the threshold no specific optimization and/or workload distribution strategy is used to maintain both the temporal and thermal behavior of the system. As a consequence, the time spent in cooling down the system at a specific time instant may cause temporal changes in the original tasks schedule and then jeopardize the entire system schedulability. To the best of our knowledge, existing thermal models (i) neglect the impact of lateral resistances between neighboring cores [8]; (ii) focus only on steady state conditions to control and/or reduce peak temperatures [6]; and (iii) consider a high number of thermal layers, which increases the model complexity [9]. In this paper, we advocate for a simple and "correct-by-construction" framework, wherein we model under the same umbrella both the temporal and thermal "on-core" and "un-core" activities for each processing element, i.e., we promote a bottom-up approach where each building block of our model of execution abstracts a processing element (e.g., a core; a memory, etc.), which in turn will be composed with the other building blocks in its vicinity. Our thermal model captures both the transient and peak temperatures at runtime. For single-core processors, such a framework that couples the thermal model and schedulers have been presented to control the processor activity and the triggering of the cooling mechanism [10]. From the comparison presented in [11] between the single-core thermal models HotSpot and TEMPEST, we concluded that HotSpot exposes better features for the design of an accurate thermal-aware management technique upon multi-cores. Therefore, we opt for an extension of the HotSpot thermal model which aims at being simple and efficient in order to build an RC thermal network model for multi-core platforms.

II. Multi-core thermal design

In our thermal network, the different parts of the chip and cooling solution are represented by N thermal nodes (electrical nodes in an electrical circuit), such that there are at least as many thermal nodes as blocks in the floorplan. Without any loss of generality, we will report our findings for uniform[1] dual-core platform (see Figure 1), where the number of thermal nodes corresponds to the number of blocks in the floorplan.

[1] Each core is characterized by a speed.



(a) Dual-core RC thermal network (b) Dual-core floorplan

Fig. 1. Dual-core representation

The temperature associated to each thermal node (with unit Kelvin $[K]$) is represented by the voltage on the node. Thermal nodes are interconnected between each other through thermal conductances (with units Watts per Kelvin [W/K]) and the heat transfer (or heat flow) among cores and other elements of the chip is represented by the currents flowing through the thermal conductances. There is a thermal capacitance associated to every thermal node which accounts for the transient thermal effects. The ambient temperature is represented by another thermal node denoted as T_{amb} and there is no thermal capacitance associated with it, as the ambient temperature is considered to be constant for long periods of time. The power consumption of the cores and other elements on the chip correspond to sources of heat (with unit Watt [W]). With these considerations, the temperatures throughout the chip are modeled as a function of the ambient temperature, the power consumption inside the chip, and the heat transfer among neighboring elements.

In Figure 1(a), $T_1(t)$ and $T_2(t)$ are the voltages representing the temperatures on Core_1 and Core_2. Then, voltages $T_3(t)$ and $T_4(t)$ represent the temperatures on the heatsink nodes directly above Core_1 and Core_2. The current supplies P_1 and P_2 represent the power consumptions on Core_1 and Core_2. For the heat transfer among thermal nodes, b_c is the thermal conductance between Core_1 and Core_2; b_{c_hs} is the thermal conductance between a core and the heatsink; b_{hs} is the thermal conductance between nodes of the heatsink; and b_{amb} is the thermal conductance between a heatsink node and the ambient temperature. Finally, the thermal capacitances of thermal node i is represented by capacitor a_i. The Kirchoff's first law states that: "The sum of currents flowing into a node is equivalent to the sum of currents flowing out of the node". By applying this law we derive the following system of first-order differential equations.

$$\begin{cases} P_1 + (T_3(t) - T_1(t)) \cdot b_{c\_hs} + (T_2(t) - T_1(t)) \cdot b_c - a_1 \cdot \frac{dT_1(t)}{dt} = 0 \\ P_2 + (T_4(t) - T_2(t)) \cdot b_{c\_hs} + (T_1(t) - T_2(t)) \cdot b_c - a_2 \cdot \frac{dT_2(t)}{dt} = 0 \\ (T_1(t) - T_3(t)) \cdot b_{c\_hs} + (T_4(t) - T_3(t)) \cdot b_c - a_3 \cdot \frac{dT_3(t)}{dt} + (T_{amb}(t) - T_3(t)) \cdot b_{amb} = 0 \\ (T_2(t) - T_4(t)) \cdot b_{c\_hs} + (T_3(t) - T_4(t)) \cdot b_c - a_4 \cdot \frac{dT_4(t)}{dt} + (T_{amb}(t) - T_4(t)) \cdot b_{amb} = 0 \end{cases}$$

By using matrices and vectors, this system leads to

$$\begin{bmatrix} a_1 & 0 & 0 & 0 \\ 0 & a_2 & 0 & 0 \\ 0 & 0 & a_3 & 0 \\ 0 & 0 & 0 & a_4 \end{bmatrix} \begin{bmatrix} T_1'(t) \\ T_2'(t) \\ T_3'(t) \\ T_4'(t) \end{bmatrix} + \begin{bmatrix} (b_{c\_hs} + b_c) & -b_c & -b_{c\_hs} & 0 \\ -b_c & (b_{c\_hs} + b_c) & 0 & -b_{c\_hs} \\ -b_{c\_hs} & 0 & (b_{c\_hs} + b_c + b_{amb}) & 0 \\ 0 & -b_{c\_hs} & 0 & (b_{c\_hs} + b_c + b_{amb}) \end{bmatrix} \begin{bmatrix} T_1(t) \\ T_2(t) \\ T_3(t) \\ T_4(t) \end{bmatrix} = \begin{bmatrix} P_1 \\ P_2 \\ 0 \\ 0 \end{bmatrix} + T_{amb} \begin{bmatrix} 0 \\ 0 \\ b_{amb} \\ b_{amb} \end{bmatrix}$$

which can be expressed as follows.

$$AT' + BT = P + T_{amb}G \qquad (1)$$

For a system with N thermal nodes, in Equation 1:

- Matrix $A = [a_{i,j}]_{N \times N}$ contains the thermal capacitance values (it is diagonal, since thermal capacitances are modeled to ground);
- Matrix $B = [b_{i,j}]_{N \times N}$ contains the thermal conductance values between vertical and lateral neighboring nodes;
- Column vector $T = [T_i(t)]_{N \times 1}$ represents the temperatures on the thermal nodes;
- Column vector $T' = [T_i'(t)]_{N \times 1}$ accounts for the first-order derivative of the temperature on each thermal node with respect to time;
- Column vector $P = [P_i]_{N \times 1}$ contains the values of the power consumption on every node. Assuming that $Node_i$ is operating at speed s_j then $P_i(s_j) = \beta_0 \cdot s_j^{\alpha} + \beta_1 \cdot s_j + \beta_2$, where α, β_0, β_1, β_2 are processor specific constants. This expression has proven to closely model the average power consumption on a core [12]. In this work we consider $\alpha = 3$, $\beta_0 = 1$, $\beta_1 = 0.002$, $\beta_2 = 0.1$ [13]; and
- Column vector $G = [b_{amb_i}]_{N \times 1}$ contains the values of the thermal conductances between each node and the ambient temperature.

Serway [14] pointed out that the thermal conductance $g_{hs}(m)$ of Core_m to the heatsink element h directly above it can be computed as in Equation 2.

$$b_{hs}(m) = \frac{A_m}{R_{chip} \cdot A_{chip}} \qquad (2)$$

In this equation, A_m denotes the area of Core_m; A_{chip} represents the area of the chip; and $R_{chip} = \frac{th_{si}}{K_{si} \cdot A_{chip}}$. In this expression, th_{si} is the thickness of the silicon and

K_{si} denotes its thermal conductivity. In our experiments, we used $th_{si} = 0.676mm$ and $K_{si} = 148W/mK$. The conductance g_{amb} of the heatsink element h to the ambient can be computed as in Equation 3 [9].

$$b_{amb}(m) = \frac{A_{hs} - A_{chip}}{R_{conv} \cdot A_{m_{hs}}} \tag{3}$$

Here, $R_{conv} \in [0.1, 2.0]$ is the convection resistance (in our experiments, we set it to $0.8K/W$); $A_{m_{hs}}$ is the area of the heatsink element under consideration; and A_{hs} is the area of the entire heatsink. We compute the conductance between core m and its neighboring core n by using Equation 4.

$$b_n(m, n) = \frac{w_{mn} \cdot th_{si} \cdot K_{si}}{L_{mn}} \tag{4}$$

In this equation, w_{mn} is the length of intersection between Core$_m$ and Core$_n$; and L_{mn} is the distance between the midpoint of Core$_m$ and that of Core$_n$. The lateral conductance between two heatsink elements can be computed in a similar fashion. In our experiments, we assumed that the heatsink is made of *copper*, with a thickness of $1.174mm$ and thermal conductivity of $400W/mK$.

Other key parameters used in the our proposed thermal model were collected from the i.MX8 chip data sheet, these are: $T_{amb} = 45°C$; $A_{chip} = 510.76mm^2$; $A_{hs} = 841.00mm^2$; and $P_{chip} \in [1.83, 17.68]W$ per core. If thermal node i is not in contact with the ambient temperature, then the corresponding value of g_i is set to zero. Assuming $a_i \neq 0$, $\forall i$, Equation 1 leads to

$$T' + A^{-1}BT = A^{-1}K \qquad \text{with} \quad K = P + T_{amb}G \tag{5}$$

In order to solve this system of first-order differential equations, we use the well-established *Laplace transform* technique. In its one-dimensional formulation, the Laplace transform of a function, say $f(t)$, defined for all real numbers $t \geq 0$, is the function $\check{f}(s) = \mathcal{L}(f(t))$, defined as

$$\check{f}(s) \overset{\text{def}}{=} \int_0^\infty f(t)e^{-st}dt \tag{6}$$

In this equation, parameter s is a complex number ($s = \sigma + i\omega$, and σ and $\omega \in \mathbb{R}$). The Laplace transform exhibits very interesting properties that are useful for solving our problem:

1) On the *linearity*: Assuming $c_1, c_2 \in \mathbb{R}$; and two functions $f(t)$ and $g(t)$, then
$$\mathcal{L}(c_1 \cdot f(t) + c_2 \cdot g(t)) = c_1 \cdot \check{f}(s) + c_2 \cdot \check{g}(s)$$
2) On the *derivative*: Assuming a function $f(t)$ and its derivative $f'(t)$, then
$$\mathcal{L}(f'(t)) = s \cdot \mathcal{L}(f(t)) - f(0) = s \cdot \check{f}(s) - f(0)$$

Going back to our system of differential equations, we denote the Laplace transform of the column vector T by $\check{T} = \left[\check{T}_i(s)\right]_{N \times 1}$ for the sake of readability. Then, by moving to the Laplace domain, we have:

$$s \cdot \check{T} - T_0 + A^{-1}B \cdot \check{T} = \frac{1}{s} \cdot A^{-1}K \tag{7}$$

In Equation 7, the column vector $T_0 = [T_{0_i}]_{N \times 1}$ contains the initial temperatures of all nodes at time $t = 0$. Thus, we have

$$(sI + A^{-1}B) \cdot \check{T} = \frac{1}{s} \cdot A^{-1}K + T_0 \tag{8}$$

where I is the identity matrix. By setting $\check{L} \overset{\text{def}}{=} (sI + A^{-1}B)$ and $\check{R} \overset{\text{def}}{=} \frac{1}{s} \cdot A^{-1}K + T_0$, we have $\check{L} \cdot \check{T} = \check{R}$, which, if matrix $\check{L}$ is inversible, means that:

$$\check{T} = \check{L}^{-1} \cdot \check{R} \tag{9}$$

Fortunately, this is the case for the inputs and type of matrices generated in this work. Indeed, all $b_{c\_hs} \neq 0$ and the determinant of the so-called "Schurr Complement Matrix" of $diag(b_{c\_hs})_{[N \times N]}$ is non-zero. Finally, by applying the "inverse Laplace transform" to Equation 9, we obtain the solution in the time domain. This is performed through a Python script by using the "`inverse_laplace_transform`" function from "`sympy`".

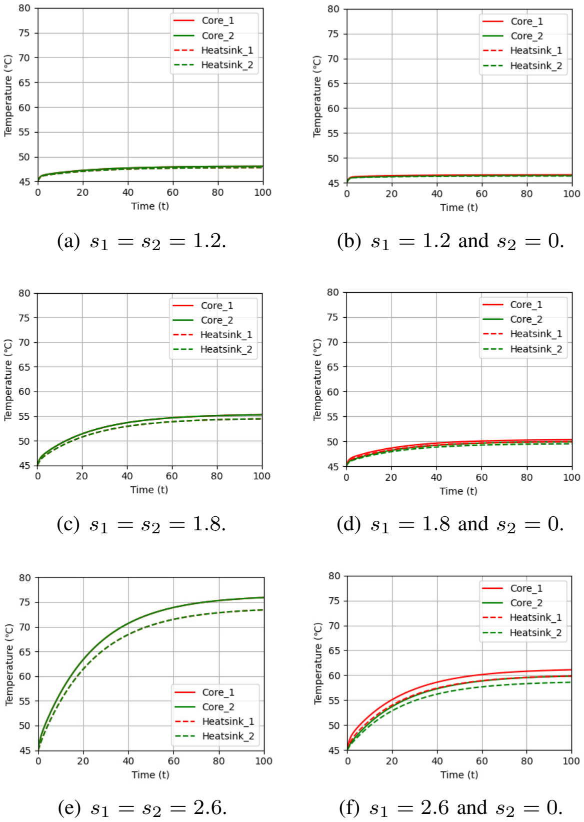
By assuming the above mentioned parameters and by assuming Core$_1$ and Core$_2$ operate at speeds s_1 and s_2 then the thermal behavior is governed by the following expressions.

$$\begin{aligned}
T_1(t) = 45.0 &+ 0.929s_1^3 - 0.017s_1^3e^{(-2.604t)} - 0.011s_1^3e^{(-1.683t)} - 0.029s_1^3e^{(-0.772t)} - 0.872s_1^3e^{(-0.041t)} \\
&+ 0.857s_2^3 + 0.013s_2^3e^{(-2.604t)} - 0.014s_2^3e^{(-1.683t)} + 0.032s_2^3e^{(-0.772t)} - 0.889s_2^3e^{(-0.041t)} \\
&+ 1.120e^{(-2.604t)} - 1.929e^{(-1.683t)} - 0.267e^{(-0.772t)} + 1.080e^{(-0.041t)} \\
T_2(t) = 45.0 &+ 0.857s_1^3 + 0.013s_1^3e^{(-2.604t)} - 0.014s_1^3e^{(-1.683t)} + 0.032s_1^3e^{(-0.772t)} - 0.889s_1^3e^{(-0.041t)} \\
&+ 0.929s_2^3 - 0.017s_2^3e^{(-2.604t)} - 0.011s_2^3e^{(-1.683t)} - 0.029s_2^3e^{(-0.772t)} - 0.872s_2^3e^{(-0.041t)} \\
&+ 1.120e^{(-2.604t)} - 1.929e^{(-1.683t)} - 0.267e^{(-0.772t)} + 1.080e^{(-0.041t)} \\
T_3(t) = 45.0 &+ 0.857s_1^3 + 0.009s_1^3e^{(-2.604t)} + 0.024s_1^3e^{(-1.683t)} - 0.040s_1^3e^{(-0.772t)} - 0.851s_1^3e^{(-0.041t)} \\
&+ 0.786s_2^3 - 0.011s_2^3e^{(-2.604t)} + 0.018s_2^3e^{(-1.683t)} + 0.043s_2^3e^{(-0.772t)} - 0.836s_2^3e^{(-0.041t)} \\
&+ 1.120e^{(-2.604t)} - 1.929e^{(-1.683t)} - 0.267e^{(-0.772t)} + 1.080e^{(-0.041t)} \\
T_4(t) = 45.0 &+ 0.786s_1^3 - 0.011s_1^3e^{(-2.604t)} + 0.018s_1^3e^{(-1.683t)} + 0.043s_1^3e^{(-0.772t)} - 0.836s_1^3e^{(-0.041t)} \\
&+ 0.857s_2^3 + 0.009s_2^3e^{(-2.604t)} + 0.024s_2^3e^{(-1.683t)} - 0.040s_2^3e^{(-0.772t)} - 0.851s_2^3e^{(-0.041t)} \\
&+ 1.120e^{(-2.604t)} - 1.929e^{(-1.683t)} - 0.267e^{(-0.772t)} + 1.080e^{(-0.041t)}
\end{aligned}$$

It is worth noticing that the thermal interference of each core on a neighboring element is materialized by its speed in the heating function of that element.

T_1 and T_2 govern the thermal behavior of the cores (which are *active* elements) and thus can be referred to as the *heating functions*; whereas T_3 and T_4 govern the thermal behavior of the heatsinks (which are *passive* elements) – see the figures below, all obtained from simulations.

In Figures 2(a), 2(c), and 2(e) the cores operate at the same speed $[1.2; 1.8; 2.6]$, respectively, and the maximum reachable temperatures when all elements originate from T_{amb} are $[48.1°C;\ 55.3°C;\ 75.8°C]$. This mean a non-linear increase of 13.01% from 1.2 to 1.8 of speed, and 26.01% from 1.8 to 2.6 of speed. When Core_2 is switched off (see Figures 2(b), 2(d), 2(f)) the maximum temperature of Core_1 drops to $[46.7°C;\ 50.4°C;\ 61.1°C]$, respectively. This mean a non-linear decrease of [2.91%; 8.86%; 19.39%].



(a) $s_1 = s_2 = 1.2$.

(b) $s_1 = 1.2$ and $s_2 = 0$.

(c) $s_1 = s_2 = 1.8$.

(d) $s_1 = 1.8$ and $s_2 = 0$.

(e) $s_1 = s_2 = 2.6$.

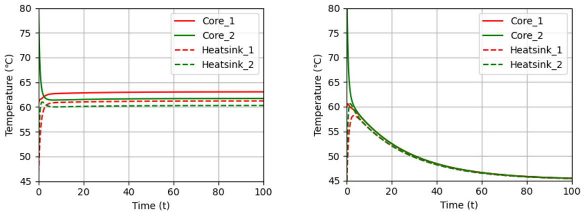
(f) $s_1 = 2.6$ and $s_2 = 0$.

Fig. 2. Thermal behavior when $T_1(0) = T_2(0) = T_3(0) = T_4(0) = 45°C$

Assuming that $T_1(0) = 80°C$, $T_2(0) = 60°C$ and $T_3(0) = T_4(0) = 45°C$, Figure 3(a) illustrates the scenario when both cores operate at different speeds; whereas Figure 3(b) shows the thermal behavior of the cores when they are both switched off. This latter display represents the cooling functions for that specific configuration.

III. Conclusion and Future Work

This paper discussed the work done towards the development of a robust thermal-aware model for uniform multi-core platforms. We provided a set of parameters; properties and a simple architectural/functional description of the hardware and software used to model the application and the platform. The next step is to evaluate efficient task-to-core mapping and scheduling strategies together with the associated analyses that will help us reduce the average temperature of the entire system as much as possible at run-time while keeping the performance as high as possible.



(a) Cores at different speeds.

(b) Both cores are switched off.

Fig. 3. Thermal behavior when $T_1(0) = 80°C, T_2(0) = 60°C$ and $T_3(0) = T_4(0) = 45°C$

Acknowledgment

This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UID/CEC/04234); by the European Union through the Clean Sky 2 Joint Undertaking, under the H2020 Framework Programme (H2020-CS2-CFP08-2018-01), grant agreement nr. 832011 (THERMAC).

References

[1] S. Borkar, “Design challenges of technology scaling,” *IEEE Micro*, vol. 19, no. 4, pp. 23–29, July 1999.

[2] R. Mahajan, “Thermal management of CPUs: A perspective on trends, needs and opportunities.” in *THERMINIC’02*, 2002.

[3] P. Kumar and L. Thiele, “Thermally optimal stop-go scheduling of task graphs with real-time constraints,” in *ASP-DAC’11*, 2011.

[4] Y. Chandarli, N. Fisher, and D. Masson, “Response time analysis for thermal-aware real-time systems under fixed-priority scheduling,” in *ISORC’15*, 2015.

[5] N. Bansal and K. Pruhs, “Speed scaling to manage temperature,” in *STACS’05*, 2005.

[6] S. Wang, Y. Ahn, and R. Bettati, “Schedulability analysis in hard real-time systems under thermal constraints,” *Real-Time Syst.*, vol. 46, 2010.

[7] Y. Fu, N. Kottenstette, C. Lu, and X. D. Koutsoukos, “Feedback thermal control of real-time systems on multicore processors,” in *EMSOFT’12*, 2012.

[8] R. Rao, S. Vrudhula, and C. Chakrabarti, “Throughput of multi-core processors under thermal constraints,” in *ISLPED’07*, 2007.

[9] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, “Temperature-aware microarchitecture: Extended discussion and results,” in *ISCA’03*, 2003.

[10] J. P. Rodríguez and P. M. Yomsi, “Thermal-aware schedulability analysis for fixed-priority non-preemptive real-time systems,” in *RTSS’2019*, 2019.

[11] ——, “Towards robust and cost-effective critical real-time systems under thermal-aware design,” in *ECRTS’19, (WiP Session)*, 2019.

[12] S. Pagani, “Power, energy, and thermal management for clustered many-cores,” Ph.D. dissertation, Karlsruher Institut für Technologie, 2016.

[13] N. Fisher, J. Chen, S. Wang, and L. Thiele, “Thermal-aware global real-time scheduling on multicore systems,” in *RTAS’09*, 2009.

[14] R. A. Serway, “Physics: For scientist and engineers with modern physics,” *Saunders, pp.(468-475, 264-265)*, 1990.