Chip temperature optimization for dark silicon many-core systems

Li, Mengquan; Liu, Weichen; Yang, Lei; Chen, Peng; Chen, Chao

2017

Li, M., Liu, W., Yang, L., Chen, P., & Chen, C. (2018). Chip temperature optimization for dark silicon many-core systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 37(5), 941-953. doi:10.1109/TCAD.2017.2740306

<https://hdl.handle.net/10356/90105>

<https://doi.org/10.1109/TCAD.2017.2740306>

© 2017 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. The published version is available at:
<https://doi.org/10.1109/TCAD.2017.2740306>.

Downloaded on 26 Apr 2023 16:59:57 SGT

Chip Temperature Optimization for Dark Silicon Many-Core Systems

Mengquan Li, *Student Member, IEEE*, Weichen Liu, *Member, IEEE*,
Lei Yang, *Student Member, IEEE*, Peng Chen, Chao Chen, *Member, IEEE*

Abstract—In the dark silicon era, a fundamental problem is: given a real-time computation demand, how to determine if an on-chip multiprocessor system is able to accept this demand and to maintain its reliability by keeping every core within a safe temperature range. In this paper, a practical thermal model is described for quick chip temperature prediction. Integrated with the thermal model, we present a Mixed Integer Linear Programming (MILP) model to find the optimal task-to-core assignment with the minimum chip peak temperature. For the worst case where even the minimum chip peak temperature exceeds the safe temperature, a heuristic algorithm, called temperature-constrained task selection (TCTS), is proposed to optimize the system performance within chip safe temperature. The optimality of the TCTS algorithm is formally proven. Extensive performance evaluations show that our thermal model achieves an average prediction accuracy of 0.0741°C within 0.2392 ms . The MILP model reduces chip peak temperature of $\sim 10^{\circ}\text{C}$ comparing with traditional techniques. The system performance is increased by 19.8% under safe temperature limitation. Due to the satisfying scalability of our MILP formulation, the chip peak temperature is further decreased by 5.06°C via the TCTS algorithm. The feasibility of this systematical technique is testified in a real case study as well.

Index Terms—Dark silicon, Thermal model, MILP model, Chip temperature optimization.

I. INTRODUCTION

MOORE’S Law, coupled with Dennard scaling, has resulted in an exponential increase in multi-core system performance for a long period. The ever-shrinking feature size of integrated circuits gives rise to the “Dark Silicon” phenomenon [1], [2]. Borrowed from [3], Table I shows the origin of dark silicon briefly, where S is the technology scaling factor. The chip power density stays constant in conventional transistors whose total power consumption is proportional to the chip area. However, in the leakage-limited regime, it costs ever-increasing leakage power to scale the threshold voltage (V_t). The operating voltage (V_{DD}) remains constant thereby. As a result, the chip power density increases with a factor of S^2 . To stay within an allowable power budget and a safe temperature limit, considering the limited device packaging and cooling technologies, only a fraction of cores can be simultaneously powered-on while others remain ‘dark’, which is technically known as “Dark Silicon”.

Chip temperature is not only a performance optimization metric but also a fundamental factor of safety. It has been

TABLE I
CLASSICAL VS. LEAKAGE-LIMITED SCALING

Transistor property	Classical scaling	Leakage-limited scaling
ΔV_t (threshold voltage)	$1/S$	1
ΔV_{DD} (supply voltage) $\approx V_t \times 3$	$1/S$	1
Δ Quantity	S^2	S^2
Δ Frequency	S	S
Δ Capacitance	$1/S$	$1/S$
Δ Power density = $\Delta(QFCV_{DD}^2)$	1	S^2
Δ Chip utilization = $1/\Delta$ Power density	1	$1/S^2$

a major concern for chip designers. Excessive power consumption in many-core chips introduces expensive packaging and cooling costs. Currently, limited by less-mature cooling techniques, the leakage power grows exponentially with the increasing chip temperature in the nanometer era [4]. In addition, high temperature, particularly *thermal hot spots* [5], increases the invalidation probability of integrated circuits (ICs) [6], reducing the lifetime reliability of systems. For instance, a 216% increase in soft error rate is observed on combinational circuits as the ambient temperature climbs from 25°C to 125°C [7], and a $10\text{-}15^{\circ}\text{C}$ temperature increase can lead to a $2\times$ reduction in the lifespan of devices [8]. To alleviate the overheating problem of many-core systems in the dark silicon era, researchers have proposed techniques from micro-architectural level [9], [10] down to physical and device level [11], [12].

The coming “Dark silicon” era introduces a new opportunity to optimize the thermal profile of many-core systems by selecting the optimal placement pattern of dark cores among all available ones. A TDP (Thermal Design Power) specification in conventional chips corresponds to only one mode where all cores are powered-on at full voltage/frequency, to a certain extent, a specific chip peak temperature [2]. In contrast, for the dark silicon many-core systems, different placement patterns of dark cores cause drastically various thermal profiles due to the different heat dissipation efficiency. As shown in Figure 1, inefficient placement patterns of dark cores (such as Pattern 1 and 2) may lead to thermal ‘hot spots’. The leakage power near the hot cores increases by an order of magnitude, which enormously limits the number of active cores and degrades system performance [2], [13]. Hence, we should grasp the opportunity offered by dark silicon to optimize chip temperature. Moreover, dynamic voltage and frequency scaling (DVFS) and power gating [14] techniques provide technical support to rearrange the location of ‘dark’

W. Liu (corresponding author) is with School of Computer Science and Engineering, Nanyang Technological University, Singapore. Email: liu@ntu.edu.sg.

M. Li, L. Yang, P. Chen and C. Chen are with College of Computer Science, Chongqing University, Chongqing, China. Email: mengquan@cqu.edu.cn.

cores for system-level chip temperature optimization.

Fig. 1. Impact of different dark silicon patterns on chip thermal profiles.

We address the following essential problem in the dark silicon era: *Given a real-time computation demand represented by a set of independent tasks with their own power consumption, how to determine if an on-chip multiprocessor system is able to respond to this demand and to keep every core within safe temperature range.* To solve this problem, there are two fundamental components to be explored.

- 1) *The first is to quickly determine the temperature of every core on the chip when the task-to-core mapping is given in prior.*
- 2) *By Utilizing the solution of the first problem, the second is to search for the optimal task-to-core mapping with the minimum chip peak temperature. It provides thermal headroom for further performance optimization.*

In this paper, we first present an Empirical Thermal Model (ETM) to predict the steady-state chip temperature for the given task-to-core mapping patterns. This thermal model, added with two empirical scaling factors, achieves high prediction accuracy and computation efficiency. We then propose a system-level temperature optimization technique. To optimize chip thermal hot spots, a Mixed Integer Linear Programming (MILP) model, which integrates the ETM, is proposed to obtain the optimal task assignment with the minimum chip peak temperature. If even the optimal chip temperature is still beyond the safe limit, a temperature-constrained task selection (TCTS) algorithm is further proposed to maximize the system performance within safe temperature limit. Experimental results show that the maximum prediction difference between ETM and *HotSpot* simulation is less than 0.3°C, and the ETM has high computation efficiency. Compared with traditional techniques, our MILP model reduces chip peak temperature as high as 9.1°C so that 19.8% more tasks can be executed in a multi-core system within safe thermal limit. The TCTS algorithm further decreases chip temperature by 5.06°C on average. We then test the scalability of the MILP formulation and formally prove the optimality of the TCTS algorithm. Finally, the feasibility of the entire chip temperature optimization technique is verified in a real case study.

The main contribution of this paper is that we propose a systematical technique for chip temperature prediction and optimization, by which **the new opportunity from the “dark silicon” phenomenon is utilized** to solve the overheating problem of on-chip multiprocessor systems. Extensive performance evaluations prove **the good practicality** of our technique.

The rest of this paper is organized as follows: Section II discusses the related work; Section III introduces the system model, the problem definition, and an overview of our technique; The proposed thermal model is described in Section IV; The MILP model is proposed in Section V and the TCTS algorithm is put forward in Section VI; Experimental results and concluding remarks are presented in Section VII and Section VIII, respectively.

II. RELATED WORK

Due to the rapid growth in power density and the limited advancement of cooling techniques, current on-chip multiprocessor systems suffer from overheating significantly [15], [16]. The dark silicon phenomenon is currently one of the foremost concerns to many researchers. Facing the coexistence of opportunity and challenge, the existing work addresses this problem from different design aspects, such as architecture heterogeneity and application-specific accelerators [17], On-chip Network Designs [18], [19], [20], run-time power and thermal management [21], [22], [23] and variability-aware techniques [13]. Analyzing these methods above, we find that the power and thermal management is *the essential solution* to address dark silicon phenomenon for on-chip many-core systems.

There have been many attempts at power budgeting techniques. Isci *et al.* [24] achieved maximum performance (energy efficiency) under a chip-level power budget. In view of the huge performance loss of many-core system caused by a constant chip-level power constraint, like TDP, pagani *et al.* [22] presented a novel power budgeting technique, called Thermal Safe Power (TSP), in which safe power budget is refined from a chip-level constant into a function of simultaneously operating cores' number. Besides core-level power budgeting techniques, many thread-level techniques are further proposed [25], [26]. Considering the gap between the power budget and the power capacity of the chip, given a conservative power budget in advance has the risk of performance underestimation, thereby causing more tasks to be suspended.

To avoid the risk resulted from conservative power budgets, one method is run-time temperature-aware management. Sheikh *et al.* [27] provided a comprehensive survey of thermal-aware task scheduling techniques for multi-core systems. To maximize the total system performance with the temperature and power constraints satisfied, Murali *et al.* [28] proposed a novel convex optimization based method to perform temperature-aware processor frequency assignment. Coskun *et al.* were devoted to achieving better temporal and spatial thermal profiles for multiprocessor systems-on-chips [4] and presented a proactive thermal-aware management approach to predict the future temperature and to adjust the job allocation

for chip thermal optimization [5]. Inspired by these prior studies, we realize the importance of alleviating chip thermal hot spots and the new chance lies in the dark silicon phenomenon. *In this paper, we provide a chip temperature optimization technique for many-core systems by taking advantage of the opportunity implied in the “dark silicon” phenomenon.*

Temperature estimation approaches are the foundation of thermal-aware techniques. Shafique *et al.* [13] employed *thermal sensors* for variability-aware dark silicon management (DaSiM) technique. Hardware thermal sensors have high accuracy and fast sensing speed while introducing additional overhead on chip area and hardware resources. Pagani *et al.* used the *HotSpot* simulation [29] for temperature prediction in Thermal Safe Power (TSP) technique [22]. *HotSpot* has been widely applied in architecture-level thermal modeling on MPSoCs. However, it is restricted to execute *HotSpot* simulation repeatedly during design space exploration due to the heavy computation overhead [30]. In addition, Chantem *et al.* modeled the heat dissipation process into equivalent RC circuit based on the Classical Fourier Heat Flow Model. Thus the chip temperature distribution is obtained by solving a set of equivalent equations in polynomial time [15]. This method simplifies the higher-order model of *HotSpot* and provides a practical way for chip temperature prediction. However, it is indicated by the experimental results shown in Section VII-A that the accuracy of this method is not perfectly promised due to the extreme uncertainty and complexity of heat dissipation process. Inaccurate temperature prediction techniques suffer from either underestimation or overestimation of chip temperatures, resulting in overheated processor cores or poor system reliability/reduced performance, respectively.

The steady-state temperature distribution of a many-core system is essential to thermal-aware task assignment and scheduling techniques because the execution of every task lasts for a period of time. We focus on the steady-state temperature in this paper. Pagani *et al.* [31] developed a new simulation called MatEx for transient peak temperature computation. Similarly, *it is necessary to develop a new thermal model for steady-state thermal prediction that can provide both measurement accuracy and computation efficiency.*

III. PROBLEM DEFINITION

In this section, the system configuration and the task model that are used in this paper are first described. We then formally define the problem to be addressed. And we give an overview of the chip temperature optimization technique.

A. System Model

We consider a homogeneous multi-core system with DVFS and the power-gating technologies, similar to the platforms in [32], [25]. The processors are architecturally identical. The only source of heterogeneity is that they can operate at different voltage and frequency levels or be power-gated off. A 4×4 2-D mesh network-on-chip (NoC) architecture is used for inter-core communications. We use the XY dimension-ordered routing algorithm and the wormhole routing switching technique for the NoC. The parameters of our system configuration are summarized in Table II.

TABLE II
SYSTEM CONFIGURATION PARAMETERS

Parameter	Value
Core type	ALPHA 21364
Core count	16
Topology	4×4 2-D Mesh
Routing algorithm	XY routing
Frequency range	1 GHz - 2.5 GHz
Technology node	22nm

The threshold voltage of transistors are reduced correspondingly with the decreasing operating voltage, consequently increasing the ‘off-leakage’ current [33]. The leakage power has become such a critical factor in design that cannot be ignored [34]. Without loss of generality, we take both dynamic and static power consumption into account in this paper. As shown in Table III, we use McPAT 1.0 [35] to estimate the average power consumption of each task (including both dynamic and leakage power) considering its operating frequency/voltage level and the system configuration presented in Table II. Not only dynamic power but also static power is modeled and addressed in McPAT. The power modeling in McPAT has utilized technology projections from ITRS [36] for static power, which is consumed due to the leakage current through the transistors. There are two types of leakage: the *subthreshold leakage* which occurs when a transistor that is supposed to be in the off state actually allows a small current to pass between its source and drain, and the *gate leakage* that leaks through the gate terminal. The unit leakage current is determined by using MASTAR [36], [37]. The temperature and leakage power are inter-dependent. The working temperature significantly affects leakage power, especially the subthreshold leakage. While the changes in the leakage power have an impact on the temperature in turn. In this paper, we focus on chip temperature prediction and optimization, and the power values of the processor cores are used as an input. Thus, we assume the working temperatures of processor cores are fixed. The average power values of a processor core at different frequency levels are obtained through McPAT, as shown in Table III. It is noteworthy that Table III is used as input in our experiments, and its values can be replaced by the actual power values in real systems. Without considering the power consumption of shared components for simplicity, the power consumption of a chip is the summation of the power consumption of individual cores. Without doubt, the power consumption from the shared components can be considered through some additional processing, for example, to treat the power consumption of individual cores as a power superposition of the shared components and the individual cores through a compensation function.

Formally, a MPSoC consists of a set of processor cores. Each core has n_f discrete frequency levels $\{L_1, \dots, L_{n_f}\}$. Each level is characterized by a triple $L_j = (V_j, F_j, P_j)$, $j \in \{1, \dots, n_f\}$, i.e., the voltage, frequency and the *average* power consumption, respectively. The actual power consumption of a

TABLE III
FREQUENCY LEVELS AND THE CORRESPONDING POWER CONSUMPTION

Level	1	2	3	4	5	6	7	8
Freq. (GHz)	1	1.2	1.4	1.6	1.8	2	2.2	2.5
Power (W)	2.95	3.14	3.32	3.51	3.7	3.89	4.08	4.37

processor core depends on not only its voltage and frequency (V/F) level but also the characteristics of the task that is executed on this core. However, it's reasonable to denote the frequency levels L as a triple of (V, F, P) according to the experiment results of the real case study in Section VII-C. In addition, we add a frequency level L_0 for each core to denote the 'dark' state, where the voltage, the frequency, and the power consumption of the core are all zero.

In this paper, we focus on the thermal profiles of applications running on many-core systems, which mainly depend on the power consumption of the applications. For simplicity, we call them tasks in this work. We abstract the execution of an application, or a task, into a power trace to highlight the power profile of the task. Therefore, in this paper, we define a real-time computation demand as a set of independent tasks, each of which occupies a core until it finishes. Each task is preassigned a frequency level according to its ordinary performance requirement, and the corresponding average power consumption is predetermined then. In DVFS and power-gating techniques, the state transition of processor cores introduce additional overhead in terms of time [38], energy [39], and functional reliability [40]. The evaluation on the state transition cost of Intel processors can be found in [41]. As each task is assumed to be preassigned a constant frequency and voltage based on its own performance requirement, no state transition is involved and thus no extra overhead is introduced for processor cores.

We define a computation demand D as the set of independent tasks with the preassigned running frequencies. In a relatively short period of time, a task running on a processor core can be considered to have a relatively stable power consumption, and a set of tasks running in parallel in a many-core system can be considered to form a constant power profile. In this perspective, the computation demand D is a snapshot of the set of tasks, or a scenario. The system running a set of applications can be seen as a continuous evolution of scenarios in short time periods. A constant power profile is kept for each scenario in the concerned period of time. Our technique can be applied for each scenario to determine the task allocation within the time period with the expectation of minimum chip peak temperature. All tasks in a demand form a computing requirement in current system. They should be executed simultaneously when the computation demand is newly launched. We assume that the tasks are allowed to be delayed in a soft real-time system if the safe temperature is violated. The computation demand D can include the 'delayed' tasks of the previous demands as long as they do not miss their own deadlines. The technique of selecting the tasks to be delayed is shown in Section VI. Formally, the set of n_t independent tasks are represented as $D = \{\tau_1, \dots, \tau_{n_t}\}$. Each

task is preassigned a frequency level, which is denoted as $\tau_k \leftarrow L_j = (V_j, F_j, P_j)$, $k \in \{1, \dots, n_t\}$, $j \in \{1, \dots, n_f\}$.

B. Problem Definition

We define a *Chip Power Distribution* (CPD) as a snapshot of the chip power consumption at a certain moment. A CPD contains the following information: the number of cores that are powered-on (the rest of cores are dark); their locations; their voltage and frequency settings. Formally, $CPD = (C, X)$, where C is the set of active cores, and X is the corresponding voltage and frequency levels of the active cores. Each task-to-core mapping has an equivalent CPD. And different task-to-core assignments of the same computation demand D would lead to various chip temperature profiles.

Figure 2 shows two different CPDs, in which all active cores run in the highest frequency in case (a) while in the lowest frequency in case (b). The two pictures below show their corresponding thermal conditions. Although both CPDs maintain the safe chip peak temperature which is determined by the physical parameters of the chip and the cooling system, for example, below 80°C , the maximum number of cores that can be powered on simultaneously and their locations are different. When running all active cores at the highest frequency, only eight cores can be simultaneously powered on, and the total power consumption is 34.96 W. When running active cores in the lowest frequency, twelve cores can be simultaneously powered on, and the total power consumption is 35.4 W.

Fig. 2. Two different CPDs and their corresponding thermal profiles (obtained through *HotSpot* simulations).

We formulate the problem addressed in this paper as follow: *Given the floorplan of a MPSoC containing a set of processor cores and a computation demand D , determine the CPD that satisfies the safe temperature constraint and maximizes the system performance with the minimum thermal profile.*

C. Technique Overview

Figure 3 illustrates the entire thermal optimization technique. In this paper, we focus on the chip thermal optimization for dark silicon many-core systems. As the foundation of thermal optimization, we first present an accurate yet efficient thermal model, Empirical Thermal Model (ETM), to quickly predict the chip thermal profile of a given task-to-core assignment. Two empirical scaling factors are added in the thermal model to achieve high prediction accuracy and running efficiency. Integrated with the ETM, we then present a Mixed Integer Linear Programming (MILP) model to find the optimal Chip Power Distribution (CPD) with the minimum chip peak temperature. For the worst case where the minimum chip peak temperature still exceeds the safe temperature, a heuristic algorithm, called temperature-constrained task selection (TCTS), is proposed to trade off the conflict between the safe thermal constraint and the real-time computation demand.

Fig. 3. An overview of the chip temperature optimization technique.

IV. THE THERMAL MODEL

As shown in Figure 4, a well-known duality between heat transfer and electrical phenomena exists [42]. The thermal transmission in a MPSoC can be modeled as an equivalent first-order RC circuit, similar to the approaches presented in [28], [15]. Heating/cooling process is complicated. The heat that a core dissipates in a certain area is not fixed due to the thermal transmission between components. Thermal conduction plays a key role in thermal transmission activities. Every core in a MPSoC, as a heat producer, is considered as an individual thermal element. The heatsink and heat spreader are considered as heat consumers since they consume little power and generate no heat. Heat spreader layer is so thin that we consider heatsink layer and heat spreader layer as a whole and call it heatsink for simplicity. We divide the heatsink into a set of heatsink elements according to the layout of the cores.

Let C_e be the thermal capacitance for any thermal element e , $e \in \mathcal{M} \cup \mathcal{H}$, where the set of processor cores and the set of heatsink elements are denoted by $\mathcal{M}$ and $\mathcal{H}$, respectively. Thermal capacitances model the transient behavior before a change in power results in the temperature's reaching steady state. Heat flow, similar to a 'current', passes through a thermal

Fig. 4. The equivalent first-order RC circuit diagram.

resistance and leads to a temperature difference analogous to a 'voltage'. Let $R_{m,l}$ be the lateral thermal resistance between core m and its neighboring core l , $m, l \in \mathcal{M}$. Let $R_{h,g}$ be the thermal resistance between two adjacent heatsink elements h and g , $h, g \in \mathcal{H}$. The set of neighboring cores of core m is denoted as N_m and the set of neighboring heatsink elements of heatsink element h is denoted as N_h . Let $R_{h,A}$ be the vertical resistance between the heatsink element h and the ambient. $Pw(t, m)$ indicates the power consumption of core m at time t . If the core is power-gated, then $Pw(t, m) = 0$. The temperature of thermal element $e \in \mathcal{M} \cup \mathcal{H}$ at time t is denoted by $T(t, e)$. The temperature of each thermal element can be expressed as a function of its power consumption, ambient temperature, and the temperature of neighboring thermal elements. Based on the classical Fourier's law of heat conduction, the heating/cooling process can be formulated as follows:

$$Pw(t, m) = \sum_{l \in N_m} \frac{T(t, m) - T(t, l)}{R_{m,l}} + C_m \cdot \frac{dT(t, m)}{dt} + f_c \cdot \frac{T(t, m) - T(t, h)}{R_{m,h}} \quad (1)$$

$$f_c \cdot \frac{T(t, m) - T(t, h)}{R_{m,h}} = \sum_{g \in N_h} \frac{T(t, h) - T(t, g)}{R_{h,g}} + C_h \cdot \frac{dT(t, h)}{dt} + f_h \cdot \frac{T(t, h) - T_A}{R_{h,A}} \quad (2)$$

where the heatsink element h is vertically on top of the core m . The thermal resistance of thermal elements can be calculated easily as described in [43]. Equation (1) and (2) can be used to calculate the temperature value of each individual core at any instant.

The heat dissipation of many-core systems mainly contains the lateral and the vertical heat transfer processes. The vertical thermal resistance, which indicates the heat dissipation rate, captures heat flow from one layer to the next, moving from the die through the package and eventually into the air. And the lateral thermal resistance captures the heat diffusion between adjacent thermal elements within a layer. We add two scaling

factors, f_c and f_h , to adjust the vertical heat dissipation rate on account of the considerable uncertainty of the vertical heat transfer process, as Equation (1) and (2) suggest. The scaling factors increase the accuracy of the thermal model in an efficient manner. These scaling factors are obtained through the MATLAB Curve Fitting Toolbox (Cftool) with a large set of random training samples.¹

Fig. 5. The generation of the thermal model.

Figure 5 illustrates the generation of the thermal model. It mainly contains two iterative loops: quantification loop and verification loop, which quantifies/trains the value of scaling factors and tests the accuracy of our thermal model, respectively. We assume the temperature value obtained from *HotSpot* simulation as the ground truth value since the *HotSpot* is widely recognized for temperature simulation in many-core systems.

It is obvious that the temperature of a core is non-decreasing if the core's execution speed is fixed. Moreover, if each core runs at a constant speed, the system will eventually reach a steady state where the temperature of all cores become steady. We can calculate the steady-state temperature of each cores in polynomial time when the Equation $C \cdot \frac{dT(t)}{dt} = 0$.

The main difference between the *HotSpot* simulation and our thermal model is the number of levels. The full 5-level RC network model from *HotSpot* simulation considers more thermal nodes in the network than our 2-level thermal model. Whether the full 5-level model from *HotSpot* simulation or the 2-level RC thermal network from [15], the key point of a thermal model is how practical it is. From this perspective, we highlight the benefit of our thermal model through the following two points: Firstly, our thermal model considers both the accuracy and the efficiency of chip temperature prediction in high-level modeling. Secondly, as a linearized formal method, our steady-state thermal model is able to be integrated into an MILP formulation (as shown in Section V)

¹Advanced methods, such as Linear Regression, Least square Method, Neural Network based methods and other machine learning techniques, are all alternative for adjusting the scaling factors and the further optimization of our thermal model. A lot of heuristic methods can be applied to address this problem in polynomial time with a given precision target.

to solve the optimization space exploration efficiently. All in all, our empirical thermal model has practical contributions for on-chip temperature modeling of many-core systems.

V. THE MILP-BASED OPTIMIZATION APPROACH

In this section, we present a MILP model to minimize chip temperature, to a certain extent, the thermal hot spots, of many-core systems. The safe threshold temperature (T_{safe}) is not taken into consideration in this section.

A. Inputs and Decision Variables

In Table IV, We summarized the input variables of the MILP formulation, with detailed definition.

TABLE IV
INPUT VARIABLES USED IN THE MILP FORMULATION

Notation	Definition
D	The computation demand.
$\mathcal{M}$	The set of individual cores, and the number of thermal elements is $2 * \mathcal{M} $.
$\mathcal{H}$	The set of heatsink elements.
n_f, n_t	The number of frequency levels and tasks.
v_k	The execution speed in the voltage/frequency level of V_k/F_k .
$Pw(V_k, v_k)$	The power consumption of the cores running with speed v_k .

In addition, the decision variables used in our MILP formulation are show in Table V.

TABLE V
DECISION VARIABLES USED IN THE MILP FORMULATION

Notation	Definition
$X_{i,k}$	Binary variable, equals to 1 if core i executes with speed v_k .
C_i	Binary variable, equals to 1 when core i is active.
T_i	The steady-state temperature of core i .

B. Optimization Objective

To minimize thermal hot spots of many-core systems, the major objective of our formulation is to minimize the chip peak temperature with the given computation demand D .

The objective formulation is shown as follow:

$$\min(\max(T_i)) \quad i \in \mathcal{M} \cup \mathcal{H} \quad (3)$$

C. Constraints

To satisfy the computation demand on the given many-core platform, the key constraints are described as follows:

- (i) *Constraint for target platform:* For each core $i \in \mathcal{M}$, it can be powered-off or powered-on in consideration of the

dark silicon phenomenon. The equation below guarantees that only one frequency and at most one active core are selected for each task. If core i is active, $C_i = 1$. Then $\sum_{k \in n_f} X_{i,k} = 1$, which means only one frequency is chosen for core i .

$$\sum_{k \in n_f} X_{i,k} = C_i \quad \forall i \in \mathcal{M} \quad (4)$$

- (ii) *Constraint for computation demand:* Equation (5) guarantees that the computation demand D is satisfied. D_k denotes the number of tasks in the voltage/frequency level of V_k/F_k .

$$\sum_{i \in \mathcal{M}} X_{i,k} = D_k \quad \forall k \in \mathcal{M} \quad (5)$$

- (iii) *Constraints for temperature:* In this paper, we consider a set of tasks running in relatively stable power profile, resulting in steady temperature states of the processor cores in the concerned period of time. We set Equation (1) and (2) with two capacitance values both equal to zero to describe the steady-state temperature. In addition, the differential part is eliminated, which greatly enhance the computation efficiency during design space exploration. The derived Equation (6) and (7) estimate the steady-state temperature of each core. And these linear constraints can be used in our MILP formulation.

$$\sum_{1 \leq k \leq n_f} X_{i,k} * Pw(V_k, v_k) = \sum_{l \in N_m} \frac{T_i - T_l}{R_{i,l}} + f_c \cdot \frac{T_i - T_h}{R_{i,h}} \quad (6)$$

$$f_c \cdot \frac{T_i - T_h}{R_{i,h}} = \sum_{g \in N_h} \frac{T_h - T_g}{R_{h,g}} + f_h \cdot \frac{T_h - T_A}{R_{h,A}} \quad (7)$$

The optimal $CPD_{opt} = (C, X)$ is obtained from the MILP formulation, which can be converted into a equivalent task assignment solution with the minimum chip peak temperature. When a set of tasks (i.e., a computation demand) is newly launched, we allocate the tasks in accordance with the assignment solution obtained by the MILP model. As shown in Section VII, our MILP-based thermal optimization model has satisfying flexibility and scalability.

VI. TEMPERATURE-CONSTRAINED TASK SELECTING ALGORITHM

The MILP model, which integrates with the thermal model for steady-state temperature prediction, can provide a globally optimal task assignment solution with minimum chip peak temperature. There is no available CPD that can meet the safe thermal constraint and satisfy the computation demand full set at the same time if the chip peak temperature of the optimal CPD still exceeds the chip safe temperature T_{safe} . The MILP model is still suitable for the tasks whose operating voltage/frequency levels can be reduced with their own deadline constraints satisfied. If it is not allow to slow down the execution speeds of tasks, only a subset of the

computation demand D can be executed simultaneously, i.e., some of tasks *have to* be selectively delayed, to guarantee the system reliability. We further propose a temperature-constrained task selection (TCTS) algorithm to maximize the system performance within safe temperature limit.

Fig. 6. The motivation of the TCTS algorithm.

As shown in Figure 6, the key objective of the TCTS algorithm is to maintain the optimality of the task allocation solution when coping with the conflict between the safe thermal constraint and the computation demand. To guarantee the maximum computation performance and take full advantage of the temperature slack, we implement the TCTS algorithm by greedily selecting the available tasks until the temperature threshold (T_{safe}) is satisfied. We assume that the tasks of a computation demand have identical priority. It achieves an optimized computation performance within safe temperature limit to maximize the number of selected tasks and their total power consumption. And from another perspective, we delay the tasks that have low frequency requests in Phase 2 alleviates the burden of many-core system when loading the next computation demand.

In Algorithm 1, we divide the selecting process into two phases and *phase* is initialized to 1. We sort all tasks according to their power/frequency from high to low (step 1-3). The peak temperature of the remaining tasks without τ_i is denoted as $T_{remain}(\tau_i)$ for convenience. In phase 1, we delay all τ_i whose $T_{remain}(\tau_i)$ exceeds T_{safe} until we meet the critical point τ_{key} (e.g. τ_4 in Figure 6). It's reasonable to refuse to handle these high-frequency tasks immediately because we can not even afford the remaining task subset. We move into phase 2 then. The critical point τ_{key} makes $T_{remain}(\tau_{key})$ higher than T_{safe} while $T_{remain}(\tau_{key+1})$ lower than T_{safe} . To fully utilize the temperature headroom, we continue the iteration rather than delay τ_{key+1} directly. We keep τ_{key+1} and test $T_{remain}(\tau_i)$ of the task τ_i , whose frequency level is lower than τ_{key+1} . The iteration will stop when we find the task τ_{out} , whose

$T_{remain}(\tau_{out})$ is lower than T_{safe} while $T_{remain}(\tau_{out-1})$ exceeds T_{safe} (e.g. τ_7 in Figure 6). Eventually, we obtain the maximum task subset that system can afford and the corresponding CPD within safe temperature. It is noteworthy that nothing stops the delayed tasks to be executed later in many-core systems when some of tasks are finished at a certain moment in the future, as long as the thermal constraint is satisfied. We define a function **MiBi_Search()** to gain τ_{key} and τ_{out} for efficiency (step 4, 7). **MiBi_Search()** is a combination of the **Binary Search** method and our MILP formulation. We can greatly reduce the time complexity via binary search rather than sequential search in Algorithm 1.

Algorithm 1 Temperature-Constrained Task Selection (TCTS)

Input: Computation demand D , MILP model;

output: Task subset and the corresponding CPD;

- 1: $AppList \leftarrow D$, and the index of $AppList$ is from min to max ;
- 2: Initialize $phase \leftarrow 1$;
- 3: Sort $\forall$ task $\tau_i \in AppList$ in descending power/frequency, high to low;
- 4: $\tau_{key} \leftarrow \text{MiBi_Search}(AppList, min, max, phase)$;
- 5: Remove all tasks who are on the left side of τ_{key+1} from $AppList$, update $AppList$;
- 6: $phase \leftarrow 2$;
- 7: $\tau_{out} \leftarrow \text{MiBi_Search}(AppList, max, key+1, phase)$;
- 8: Remove task τ_{out} from $AppList$, update $AppList$;
- 9: $CPD \leftarrow \text{MILP}(AppList)$;
- 10: **return** $AppList$ and CPD, which are the maximum affordable task subset and its optimized CPD pattern, respectively;

MiBi_Search($List, low, high, phase$)

$mid = low + (high - low) / 2$;

if $phase == 1$ **then**

$T_{remain} \leftarrow$ the peak temperature of $List$ without $\tau_{low}, \tau_{low+1}, \dots, \tau_{mid}$ via MILP formulation;

else if $phase == 2$ **then**

$T_{remain} \leftarrow$ the peak temperature of $List$ without τ_{mid} via the MILP formulation;

end if

if $T_{remain} > T_{safe}$ **then**

return **MiBi_Search**($List, mid + 1, high, phase$);

else if $T_{remain} < T_{safe}$ **then**

return **MiBi_Search**($List, low, mid - 1, phase$);

else

return mid ;

end if

Theorem 1. *The temperature-constrained task selection algorithm maximizes the affordable computation demand subset and its total power consumption within safe thermal limit.*

Proof. In phase 1, Algorithm 1 minimizes the number of the delayed tasks (i.e., maximizes the affordable computation demand subset). The TCTS algorithm is called as Algorithm $\mathcal{A}$ for simplicity hereinafter. Assume that the TCTS algorithm does not maximize the affordable task subset, there is

an Optimum Algorithm ‘ $\mathcal{O}$ ’ definitely which is most close to the TCTS algorithm while has the minimum delayed task subset. We denote the delayed task list of Algorithm $\mathcal{A}$ and Algorithm $\mathcal{O}$ as $\mathcal{A}_{t_1}, \mathcal{A}_{t_2}, \mathcal{A}_{t_3}, \dots, \mathcal{A}_{t_{k-1}}, \mathcal{A}_{t_k}, \dots$ and $\mathcal{O}_{t_1}, \mathcal{O}_{t_2}, \mathcal{O}_{t_3}, \dots, \mathcal{O}_{t_{k-1}}, \mathcal{O}_{t_k}, \dots$, respectively. Due to the similarity of two algorithms, we suppose that the first $k-1$ items, i.e., the first $k-1$ selected delayed tasks, of Algorithm $\mathcal{A}$ and Algorithm $\mathcal{O}$ are the same and the difference between them begin with the k -th task. Considering that the main policy of TCTS algorithm is to select the tasks from high to low frequency level, so it’s obvious that $F(\mathcal{O}_{t_k}) \leq F(\mathcal{A}_{t_k})$. However, we have to drop the task with higher frequency if we can not even afford a lower one. So the number of delayed tasks obtained from Algorithm $\mathcal{O}$ is more than that from Algorithm $\mathcal{A}$, which contradicts the assumption above. Consequently, the TCTS algorithm ensures the optimality of maximizing the affordable task subset within safe thermal constraint in phase 1. We obtain the maximum number of the affordable tasks in phase 1. Based on it, we delay a task with low frequency to obtain the maximum total computation power of the selected task subset within thermal threshold in phase 2. In conclusion, the temperature-constrained task selection (TCTS) algorithm maximizes the affordable computation demand subset and its total power consumption within safe temperature range. $\square$

Complexity Analysis: We denote the size of computation demand as N . The number of times that the MILP model is executed is down to $\mathcal{O}(\log N)$ due to a binary search. The scalability of the MILP formulation is tested in Section VII.

VII. PERFORMANCE EVALUATION

In this section, we verify the accuracy and efficiency of the proposed thermal model by comparing it with the existing thermal model in [15] and *HotSpot* simulation. We then test the effectiveness of the MILP model and the TCTS algorithm. Last but not least, we validate the feasibility of our temperature optimization technique in a real case study.

The parameters used in thermal models and *HotSpot* simulation are listed in Table VI. The threshold temperature (T_{safe}) is set to 80°C, which is considered as a high temperature value for silicon chips. The average power consumptions of a core under different frequency levels are listed in Table III. A summary of system configuration is presented in Table II.

TABLE VI
THE PARAMETERS USED IN THE THERMAL MODELS AND HOTSPOT

Parameter	Value
Silicon thermal conductivity [W/m·K]	100
Silicon specific heat [J/m ³ ·K]	$1.75 \cdot 10^6$
Thickness of the chip [mm]	3.5
Convection resistance [K/W]	0.1
Heatsink thickness [mm]	2
Heatsink thermal conductivity [W/m·K]	400
Heatsink specific heat [J/m ³ ·K]	$3.55 \cdot 10^6$
Ambient temperature [K]	318.15

A. Chip Temperature Prediction

We compare the ETM with the thermal model proposed in [15] and *HotSpot* in this paper. Differing from the training samples used in Section IV, the testing samples used in this experiment are randomly generated computation demands, based on our system configuration and DVFS levels. More precisely, we build a simulator with MATLAB. By using the RANDINT function to produce uniformly distributed random integers, 1000 sets of testing computation demands are generated to average the performance results. For each of the 1000 cases, we randomly generate the number of active processor cores and their locations (the rest of cores are dark). For every core that is powered-on, we assign it with a randomly generated voltage/frequency level and the projected average power consumption is obtained correspondingly by referring to Table III.

As shown in Figure 7, a large number of experimental results show a distinct difference on chip peak temperature prediction between the thermal model in [15] and *HotSpot* simulation. This huge temperature prediction error, nearly 10.05°C on average, is introduced by the facts that the heat spreader layer is omitted in [15] and the pure equivalent RC circuit cannot describe the complicated heat transmission accurately, especially the vertical heat transfer process. By improving the naive thermal model with the proper scaling factors, the temperature curves of ETM and *HotSpot* almost overlap together with a maximum of 0.27°C difference. We further present a box-and-whisker diagram to display the statistical distribution of the temperature inaccuracy of the ETM on the right side of Figure 7. The Box-and-Whisker Plot is a graphic way to display the median, quartiles, and extremes of a data set without making any assumptions of the underlying statistical distribution. It strongly verifies that our empirical thermal model has high accuracy for chip temperature prediction.

Fig. 7. Accuracy comparison among the thermal model in [15], our thermal model and *HotSpot* simulation.

Figure 7 indicates the situation on the peak temperature prediction for the chip. We further test the prediction accuracy of our thermal model for all the processor cores on the chip. We collect the temperature values of each individual core obtained by our thermal model and *HotSpot* simulation, respectively. The average temperature difference on each core in the 1000 test cases is summarized and is shown in Figure 8. For every

individual core, the temperature prediction difference between our thermal model and the *HotSpot* simulation maintains in a small range: the average difference is less than 0.18°C . The differences become more obvious for the cores at the center of the chip than the cores at the borders, while the maximum difference is 0.31°C only, which is negligible for chip-wide temperature prediction. The slightly increased difference is mainly due to the more complicated thermal transmission activities of the inner cores with their neighbors than those of the outer cores.

Fig. 8. The average temperature prediction accuracy for each processor core in all test cases.

Furthermore, we compare the execution time between our thermal model and *HotSpot* simulation. As shown in Figure 9, we omit the curve of the thermal model in [15] since it is almost the same as the curve of ETM. As shown in Figure 9(a), there is an obvious gap between the execution time of ETM and that of *HotSpot* simulation for a single run. The average execution time of ETM is 0.2392 ms while the average execution time of *HotSpot* is $70\times$ as much as the ETM's. According to the growth trend of two curves with the increasing number of iterations in Figure 9(b), we can observe that it takes a relatively long time for *HotSpot* simulation to perform temperature analysis, which restricts it to be invoked repeatedly to evaluate system design schemes during design space exploration. The gap between the execution time of ETM and that of *HotSpot* simulation also verifies the good efficiency of ETM for chip temperature prediction.

(a) Time for a single run of both algo- (b) Accumulated time for multiple runs of both algorithms.

Fig. 9. Efficiency comparison between our thermal model and *HotSpot* simulation.

The comparison is based on homogeneous multi-core systems with DVFS and power-gating technologies as presented in Table II, as well as the parameters listed in Table III and VI. While the proposed thermal model can be extended to support heterogeneous multi-core platforms through some additional efforts. With heterogeneous IP cores, the physical properties of cores and the partitioning of heatsink elements corresponding to the layout of the cores differ from homogeneous platforms. The average power consumptions of software tasks are more diverse when they execute on heterogeneous processor cores or accelerators. All of these factors need to be considered to obtain the thermal parameters in our thermal model, such as thermal resistances and capacitances. This can be done at design time by the consistent method as described in Section IV. The training process at design time to determine the values of the empirical scaling factors, f_c and f_h , are necessary for all the platforms before the thermal model can be applied.

B. Chip Peak Temperature Optimization

In this paper, we employ the Gurobi Optimization as the MILP solver, which is designed to be the fastest, most powerful solver available for MILP problems. Firstly, we test the optimization quality, the efficiency and the scalability of our MILP formulation. Based on it, a large number of experiments are conducted to show the optimization performance of the TCTS algorithm.

Our technique can optimize the on-chip temperature profile of many-core systems under arbitrary computation demands. Nevertheless, it has much more importance in high-load occasions. Based on this, we assume that all computation demand samples contain more than 8 tasks to achieve $>50\%$ load rate in our experiments. It is noteworthy that the MILP model and the TCTS algorithm have much better optimization quality under low load than the performance under high load. That is because when the number of tasks in a computation demand increases, the number of dark cores and the choices to position the dark cores will be reduced accordingly. Thus the heat dissipation becomes relatively harder and the benefit of our technique on temperature reduction will be decreased.

We randomly generate 1000 sets of computation demands to evaluate the temperature optimization quality of the MILP model. The multiple computation demands are sorted from lightweight to heavyweight for clarity. As shown in Figure 10(a) and (b), comparing with the random mapping method and the continuous mapping method which is adopted in [44], the MILP formulation reduces the chip peak temperature by an average of 2.98°C and 4.59°C , respectively. Therefore, 12.3% and 19.8% more tasks can be accepted by the many-core system under safe temperature limit, shown in Figure 10(c). Figure 10(a) shows a trend that the benefit of the MILP model reduces with the increasing number of tasks since more tasks, viz., less dark cores, provide less available CPD patterns to optimize the chip temperature profile. That's why the actual temperature reduction performance of the MILP model is much better than the average performance, e.g. a maximum of 9.1°C temperature reduction is achieved in lightweight computation demands shown in Figure 10(a).

Fig. 10. The optimization performance of our MILP formulation. For simplicity, we denote the Continuous Mapping method and the Random Mapping method as CM and RM, respectively. (b) The statistical distribution of the temperature differences between the MILP and the CM and that between the MILP and the RM. (c) The percentage of the additional tasks benefited from the MILP model comparing with the CM and the RM.

The number of thermal profile alternatives in many-core systems is exponential due to the possible placement pattern of dark cores. Indeed, the problem of finding the optimal placement pattern of dark cores to minimize the chip peak temperature is NP-hard. The MILP model is based on a formal method so that we can obtain a task assignment solution with the optimal chip temperature through it. In a word, the MILP model has excellent temperature optimization quality, especially in low-load condition.

Fig. 11. The efficiency of the MILP formulation.

To test the efficiency and scalability of the MILP formulation, we set 10^3 seconds as a time threshold and test how many tasks can be finished via the MILP formulation. Figure 11 indicates that we can solve more than 10^4 tasks in 10^3 Seconds, i.e., more than 10^4 tasks are assigned to the right cores on which the chip peak temperature is minimum. As shown on the right side of Figure 11, the MILP efficiency, which is defined as the number of finished tasks per second, is 16.55 tasks per second on average. The efficiency of the MILP formulation validates the feasibility of the MILP approach and provides a good foundation to the TCTS algorithm. And to some degree, the good efficiency of the MILP formulation is a sign of its satisfying scalability.

According to Figure 10, the optimal chip peak temperatures of 380 sets of computation demands still exceed the safe limit. The TCTS algorithm is used for these demands to maximize the system computation performance within safe temperature. According to Figure 12, the chip peak temperature can be further reduced by an average of 5.06°C via the temperature-constrained task selection (TCTS) algorithm. The curve of the resulting peak temperature roughly presents a serrated appearance since the frequency of delayed tasks changes approximately periodic, from L_{n_f} to L_1 . The final chip peak temperature, around 78.79°C on average, is very close to T_{safe} , which once again verifies that the TCTS algorithm takes full advantage of the temperature headroom to maximize the system computation performance.

Fig. 12. The temperature optimization benefits of the TCTS algorithm.

In this paper, we focus on the chip temperature optimization of dark silicon many-core systems. The power consumption of each task and the resulting temperature condition of the many-core system are the main concerns. In order to highlight the characteristic of the tasks in power consumption, we abstract our tasks as independent power traces without detailed descriptions to their inner processes/threads execution. Although no real benchmarks are explicitly used in the evaluation, the set of synthetic examples we used is quite large to cover most of the common cases of the power profiles of real applications.

C. Feasibility Verification of Our Temperature Optimization Technique in Real Cases

In this paper, we assume that each task is preassigned a constant power consumption according to its ordinary performance requirement. And we have verified the feasibility of our chip temperature optimization technique based on this assumption. However, the power consumption of a task is not only dependent on its voltage/frequency level but also on the specific characteristics of the task. To verify the feasibility of our technique in real cases, a vital issue is: *How's the feasibility of our temperature optimization technique when the real power consumption of tasks is considered?* Our thermal model remains suitable to measure the chip temperature profile in real cases. The key to testing the feasibility of the temperature

optimization technique in real cases is to determine whether this assumption, viz., replacing the real power consumption of a task with an average one, would affect the accuracy of the temperature prediction.

To reflect the actual execution of tasks, the real power consumption of every task follows the Gaussian distribution, $Power \sim N(\mu, \sigma^2)$, where μ is the average power consumption and σ^2 is the variance. The variance measures the deviation between the actual power values and the average one, which is determined by the number of frequency/power levels that is supported by a modern processor and the frequency/power gap between adjacent levels. Taking Table III for example,² there are 8 power levels and the variance of the power consumption of tasks is about 0.19^2 .

We generate 100 sets of random computation demands with real power consumption which is not constant and follows the Gaussian distribution. For each computation demand, we conduct multiple experiments and they are divided into two groups: the experiment group and the control group. The tasks in the experiment group are preassigned with the average power consumptions according to their frequencies, and in the control group, we generate 700 samples where the tasks are preassigned with the same frequencies while their own real power consumption. By using our thermal model, we compare the chip temperature of the experiment group and that of the control group. As shown on the left side of Figure 13, for clarity, we record the maximum (the green curve) and the minimum (the red curve) chip peak temperature obtained by the experimental samples in the control group for every computation demand. The chip peak temperature of the corresponding experiment group is indicated by the blue curve. It shows that the chip peak temperature resulted from the experiment group and from the control group are almost the same with a maximum difference of ± 0.7273 Celsius.

Fig. 13. Feasibility verification of our technique in real cases. (a) The maximum difference between the chip peak temperature from our power model and that in real cases. (b) The distribution of the temperature difference for a single computation demand.

Moreover, we zoom in and analyze the distribution of the temperature difference between the experiment group and the control group for one single computation demand. We calculate the Probability Density Function (PDF) of the temperature difference between the experiment group and 700

²The frequency/power levels are not entirely separated, but partly overlapping.

samples of the control group. As shown on the right side of Figure 13, we can find that the probability density of the temperature difference presented in the form of a histogram nearly follows the Normal distribution $\sim N(0.007, 0.19^2)$. It indicates that the temperature prediction has an inaccuracy of $[-0.182, 0.197]$ and $[-0.373, 0.387]$ in 68.26% and 95.44% of cases, respectively.

For a single computation demand, after analyzing the temperature difference between its experiment group and control group, we find that, in the vast majority of cases, the temperature prediction inaccuracy of the experiment group is less than 0.4°C . This results confirm the feasibility of modeling the real power consumption of a task as a constant according to its average power consumption for efficient temperature prediction and optimization. The feasibility of our temperature prediction and optimization technique in real cases is verified.

VIII. CONCLUSION

Given a real-time computation demand represented by a set of independent tasks with their own average power consumption and the floorplan of an on-chip multiprocessor system (MPSoC), how to determine an optimized CPD pattern to maximize the computation performance under safe temperature threshold. We first presented a practical thermal model for quick prediction of the chip temperature. Based on it, a MILP model was presented to find the optimal CPD with minimum chip peak temperature. In case the optimal peak temperature derived from the MILP formulation still exceeds the safe temperature, we presented a temperature-constrained task selection (TCTS) algorithm to maximize the system performance and to guarantee the thermal reliability. Experimental results validated the accuracy and the efficiency of our thermal model, as well as the excellent temperature optimization performance of the MILP formulation and the TCTS algorithm. In addition, the good efficiency of the MILP formulation provides our technique practicability contribution. Last but not least, the feasibility of our temperature optimization technique in real cases was verified.

ACKNOWLEDGMENT

This work is partially supported by NSFC No. 61402060, National 863 Program 2015AA015304, Chongqing High-Tech Research Program cstc2015jcyjA40042, and the Fundamental Research Funds for the Central Universities No. 106112017CDJQJ188829, China.

REFERENCES

- [1] H. Esmailzadeh, E. Blem, R. St Amant, K. Sankaralingam, and D. Burger, "Dark silicon and the end of multicore scaling," in *Computer Architecture (ISCA)*, 2011 38th Annual International Symposium on. IEEE, 2011, pp. 365–376.
- [2] M. Shafique, S. Garg, J. Henkel, and D. Marculescu, "The eda challenges in the dark silicon era: Temperature, reliability, and variability perspectives," in *Proceedings of the 51st Annual Design Automation Conference on Design Automation Conference*. ACM, 2014, pp. 1–6.
- [3] M. B. Taylor, "A landscape of the new dark silicon design regime," *IEEE Micro*, vol. 33, no. 5, pp. 8–19, 2013.
- [4] A. K. Coskun, T. S. Rosing, and K. Whisnant, "Temperature aware task scheduling in mpsoes," in *Proceedings of the conference on Design, automation and test in Europe*. EDA Consortium, 2007, pp. 1659–1664.
- [5] A. K. Coskun, T. S. Rosing, and K. C. Gross, "Proactive temperature balancing for low cost thermal management in mpsoes," in *Computer-Aided Design, 2008. ICCAD 2008. IEEE/ACM International Conference on*. IEEE, 2008, pp. 250–257.
- [6] J. S. S. T. Association *et al.*, "Failure mechanisms and models for semiconductor devices," *JEDEC Publication JEP122-B*, 2003.
- [7] S. S.-Y. Hsueh, R. H.-M. Huang, and C. H.-P. Wen, "Tasser: A temperature-aware statistical soft-error-rate analysis framework for combinational circuits," in *Quality Electronic Design (ISQED)*, 2014 15th International Symposium on. IEEE, 2014, pp. 529–534.
- [8] R. Viswanath, V. Wakharkar, A. Watwe, V. Lebonheur, M. Group, and I. Corp, "Thermal performance challenges from silicon to systems," 2000.
- [9] N. Goulding-Hotta, J. Sampson, G. Venkatesh, S. Garcia, J. Auricchio, P.-C. Huang, M. Arora, S. Nath, V. Bhatt, J. Babb *et al.*, "The greendroid mobile application processor: An architecture for silicon's dark future," *IEEE Micro*, vol. 31, no. 2, pp. 86–95, 2011.
- [10] Y. Turakhia, B. Raghunathan, S. Garg, and D. Marculescu, "Hades: architectural synthesis for heterogeneous dark silicon chip multiprocessors," in *Proceedings of the 50th Annual Design Automation Conference*. ACM, 2013, p. 173.
- [11] F. Paterna and S. Reda, "Mitigating dark-silicon problems using superlattice-based thermoelectric coolers," in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2013, pp. 1391–1394.
- [12] A. Raghavan, Y. Luo, A. Chandawalla, M. Papaefthymiou, K. P. Pipe, T. F. Wenisch, and M. M. Martin, "Computational sprinting," in *High Performance Computer Architecture (HPCA)*, 2012 IEEE 18th International Symposium on. IEEE, 2012, pp. 1–12.
- [13] M. Shafique, D. Gnad, S. Garg, and J. Henkel, "Variability-aware dark silicon management in on-chip many-core systems," in *Proceedings of the 18th Design, Automation and Test in Europe (DATE)*, March 2015.
- [14] K. Ma and X. Wang, "Pgcapping: exploiting power gating for power capping and core lifetime balancing in cmps," in *Proceedings of the 21st international conference on Parallel architectures and compilation techniques*. ACM, 2012, pp. 13–22.
- [15] T. Chantem, X. S. Hu, and R. P. Dick, "Temperature-aware scheduling and assignment for hard real-time applications on mpsoes," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 19, no. 10, pp. 1884–1897, 2011.
- [16] L. Yang, W. Liu, W. Jiang, C. Chen, M. Li, P. Chen, and H. Edwin, "Hardware-software collaboration for dark silicon heterogeneous many-core systems," *Future Generation Computer Systems*, vol. 68, pp. 234–247, 2017.
- [17] M. A. Aguilar, R. Leupers, G. Ascheid, and L. G. Murillo, "Automatic parallelization and accelerator offloading for embedded applications on heterogeneous mpsoes," in *Proceedings of the 53rd Annual Design Automation Conference*. ACM, 2016, p. 49.
- [18] H. Bokhari, H. Javaid, M. Shafique, J. Henkel, and S. Parameswaran, "darknoc: Designing energy-efficient network-on-chip with multi-vt cells for dark silicon," in *Proceedings of the 51st Annual Design Automation Conference*. ACM, 2014, pp. 1–6.
- [19] L. Yang, W. Liu, W. Jiang, M. Li, J. Yi, and E. H.-M. Sha, "Fotonoc: A hierarchical management strategy based on folded torus-like network-on-chip for dark silicon many-core systems," in *Design Automation Conference (ASP-DAC)*, 2016 21st Asia and South Pacific. IEEE, 2016, pp. 725–730.
- [20] L. Yang, W. Liu, W. Jiang, M. Li, P. Chen, and E. H.-M. Sha, "Fotonoc: A folded torus-like network-on-chip based many-core systems-on-chip in the dark silicon era," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 7, pp. 1905–1918, 2017.
- [21] A. Raghavan, L. Emurian, L. Shao, M. Papaefthymiou, K. P. Pipe, T. F. Wenisch, and M. M. Martin, "Utilizing dark silicon to save energy with computational sprinting," *IEEE Micro*, vol. 33, no. 5, pp. 20–28, 2013.
- [22] S. Pagani, H. Khdr, W. Munawar, J.-J. Chen, M. Shafique, M. Li, and J. Henkel, "Tsp: thermal safe power: efficient power budgeting for many-core systems in dark silicon," in *Proceedings of the 2014 International Conference on Hardware/Software Codesign and System Synthesis*. ACM, 2014, p. 10.
- [23] L. Yang, W. Liu, N. Guan, M. Li, P. Chen, and H. Edwin, "Dark silicon-aware hardware-software collaborated design for heterogeneous many-core systems," in *Design Automation Conference (ASP-DAC)*, 2017 22nd Asia and South Pacific. IEEE, 2017, pp. 494–499.
- [24] C. Isci, A. Buyuktosunoglu, C.-Y. Cher, P. Bose, and M. Martonosi, "An analysis of efficient multi-core global power management policies: Maximizing performance for a given power budget," in *Proceedings of the*

39th annual IEEE/ACM international symposium on microarchitecture. IEEE Computer Society, 2006, pp. 347–358.

- [25] K. K. Rangan, G.-Y. Wei, and D. Brooks, “Thread motion: fine-grained power management for multi-core systems,” in *ACM SIGARCH Computer Architecture News*, vol. 37, no. 3. ACM, 2009, pp. 302–313.
- [26] X. Hu, Y. Xu, J. Ma, G. Chen, Y. Hu, and Y. Xie, “Thermal-sustainable power budgeting for dynamic threading,” in *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*. IEEE, 2014, pp. 1–6.
- [27] H. F. Sheikh, I. Ahmad, Z. Wang, and S. Ranka, “An overview and classification of thermal-aware scheduling techniques for multi-core processing systems,” *Sustainable Computing: Informatics and Systems*, vol. 2, no. 3, pp. 151–169, 2012.
- [28] S. Murali, A. Mutapicic, D. Atienza, R. Gupta, S. Boyd, and G. De Micheli, “Temperature-aware processor frequency assignment for mpsoes using convex optimization,” in *Hardware/Software Codesign and System Synthesis (CODES+ ISSS), 2007 5th IEEE/ACM/IFIP International Conference on*. IEEE, 2007, pp. 111–116.
- [29] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. R. Stan, “Hotspot: A compact thermal modeling methodology for early-stage vlsi design,” *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 14, no. 5, pp. 501–513, 2006.
- [30] A. Ziabari, J.-H. Park, E. K. Ardestani, J. Renau, S.-M. Kang, and A. Shakouri, “Power blurring: Fast static and transient thermal analysis method for packaged integrated circuits and power devices,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 11, pp. 2366–2379, 2014.
- [31] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel, “MatEx: Efficient transient and peak temperature computation for compact thermal models,” in *Proceedings of the 18th Design, Automation and Test in Europe (DATE)*, March 2015.
- [32] S. Dighe, S. Vangal, P. Aseron, S. Kumar, T. Jacob, K. Bowman, J. Howard, J. Tschanz, V. Erraguntla, N. Borkar *et al.*, “Within-die variation-aware dynamic-voltage-frequency scaling core mapping and thread hopping for an 80-core processor,” in *Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2010 IEEE International*. IEEE, 2010, pp. 174–175.
- [33] H. Iwai, “Roadmap for 22nm and beyond,” *Microelectronic Engineering*, vol. 86, no. 7, pp. 1520–1528, 2009.
- [34] N. S. Kim, T. Austin, D. Baauw, T. Mudge, K. Flautner, J. S. Hu, M. J. Irwin, M. Kandemir, and V. Narayanan, “Leakage current: Moore’s law meets static power,” *computer*, vol. 36, no. 12, pp. 68–75, 2003.
- [35] S. Li, J. H. Ahn, R. D. Strong, J. B. Brockman, D. M. Tullsen, and N. P. Jouppi, “McPAT: An Integrated Power, Area, and Timing Modeling Framework for Multicore and Manycore Architectures,” in *MICRO 42: Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture*, 2009, pp. 469–480.
- [36] T. Skotnicki, “Model for assessment of cmos technologies and roadmaps (mastar),” <http://www.itrs2.net/itrs-models-and-papers.html>, 2007, [Online].
- [37] C. Auth, A. Cappellani, J.-S. Chun, A. Dalis, A. Davis, T. Ghani, G. Glass, T. Glassman, M. Harper, M. Hattendorf *et al.*, “45nm high-k+ metal gate strain-enhanced transistors,” in *VLSI Technology, 2008 Symposium on*. IEEE, 2008, pp. 128–129.
- [38] D. Hackenberg, R. Schöne, T. Ilsche, D. Molka, J. Schuchart, and R. Geyer, “An energy efficiency feature survey of the intel haswell processor,” in *Parallel and Distributed Processing Symposium Workshop (IPDPSW), 2015 IEEE International*. IEEE, 2015, pp. 896–904.
- [39] H. Xu, R. Vemuri, and W.-B. Jone, “Dynamic characteristics of power gating during mode transition,” *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 19, no. 2, pp. 237–249, 2011.
- [40] W. Liu, Y. Wang, X. Wang, J. Xu, and H. Yang, “On-chip sensor network for efficient management of power gating-induced power/ground noise in multiprocessor system on chip,” *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 24, no. 4, pp. 767–777, 2013.
- [41] D. Processor, “Power and thermal management in the intel® core™,” *Intel® Centrino® Duo Mobile Technology*, vol. 10, no. 2, p. 109, 2006.
- [42] K. Skadron, M. R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan, “Temperature-aware microarchitecture,” in *Computer Architecture, 2003. Proceedings. 30th Annual International Symposium on*. IEEE, 2003, pp. 2–13.
- [43] D. C. Giancoli, *Physics for scientists and engineers with modern physics*. Pearson Education, 2008, vol. 2.
- [44] M. Fattah, M. Palesi, P. Liljeberg, J. Plosila, and H. Tenhunen, “Shifa: System-level hierarchy in run-time fault-aware management of many-core systems,” in *Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE*. IEEE, 2014, pp. 1–6.

Mengquan Li received the B.E. degree from the School of Computer Science at Chongqing University, Chongqing, China, in 2015 and is currently pursuing the Ph.D degree from the Department of Computer Science at Chongqing University, Chongqing, China.

Her current research interests include temperature modeling and optimization, real-time embedded systems, fault-tolerant systems and high-performance parallel computing.

Weichen Liu (S07-M11) is an assistant professor at School of Computer Science and Engineering, Nanyang Technological University, Singapore. He received the PhD degree from the Hong Kong University of Science and Technology, Hong Kong, and the BEng and MEng degrees from Harbin Institute of Technology, China. Dr. Liu has authored and co-authored more than 70 research papers in peer-reviewed journals, conferences and books, and received Best Paper Candidate Awards from CODES+ISSS, CASES and ASP-DAC.

His research interests include embedded and real-time systems, multiprocessor systems and fault-tolerant systems.

Lei Yang received the B.E. degree from the School of Computer Science at Chongqing University, Chongqing, China, in 2013 and is currently pursuing the Ph.D degree from the Department of Computer Science at Chongqing University, Chongqing, China.

Her current research interests include Network-on-chip optimization, multi-core architecture optimization and high-performance parallel computing.

Peng Chen received the B.E. degree from the School of Software Engineering at Chongqing University, Chongqing, in 2015 and is currently pursuing the Ph.D. degree from the Department of Computer Science at Chongqing University, Chongqing, China.

His current research interests include embedded and real-time systems, multicore scheduling.

Chao Chen received the B.Sc. and M.Sc. degrees in control science and control engineering from Northwestern Polytechnical University, Xi’an, China, in 2007 and 2010, respectively, and the Ph.D. degree from Pierre and Marie Curie University and Institut Mines-TELECOM/TELECOM SudParis, Evry, France in 2014. In 2009, he was a Research Assistant with Hong Kong Polytechnic University, Kowloon, Hong Kong. He is an Associate Professor with the College of Computer Science, Chongqing University, Chongqing, China. His research interests

include pervasive computing, social network analysis, urban logistics, data mining from large-scale taxi data, and big data analytics for smart cities. Dr. Chen was a recipient of the Best Paper Runner-Up Award at the 2011 International ICST Conference on Mobile and Ubiquitous Systems.