Energy Minimization for Multi-core Platforms through DVFS and VR Phase Scaling With Comprehensive Convex Model

Zuomin Zhu, Wei Zhang, Vivek Chaturvedi, and Amit Kumar Singh

***Abstract*—Energy management is a critical challenge in multi-core processors due to continuous technology scaling. Previous methods have mostly focused on the energy minimization of the processor cores. However, energy overhead of the off-chip voltage regulator (VR) has recently shown to be a non-trivial part of the total energy consumption and has been previously overlooked. In this paper, we propose an overall energy optimization method for the system that minimizes both per-core energy consumption and VR energy consumption using dynamic voltage frequency scaling (DVFS) and VR phase scaling by solving a comprehensive convex model. In order to improve the accuracy of the task latency model, a new task model considering both computation and memory access of the task is also developed. Furthermore, for better scalability and lower on-line overhead, we decompose our proposed convex method into two stages: an off-line stage and an on-line stage. During the off-line stage, we explore the convex model by assuming different numbers of active phases of the VR, various workload pressures and workload characteristics to collect the optimal frequency assignments under different scenarios. During the online stage, the specific frequency assignment for cores and optimal active phase number of the VR are selected and applied based on the actual workload pressure and its characteristics running on the cores . Experiments on real benchmarks show that when compared with state-of-the-art approaches, which are oblivious to VR overheads and exploit slack time to achieve energy minimization, our method can achieve a significant energy saving of up to 22.4% with negligible on-line overhead.**

***Index Terms*—Energy Minimization, DVFS, Voltage Regulator, Phase Scaling**

I. Introduction

As the energy consumption of modern computing systems increases every year, power and energy management have become a critical challenge [1]. Many works have proposed reducing the power or energy consumption of the processor cores in the multi-core platforms [2][3][4][5][6][7][8]. Among the available methods, dynamic voltage and frequency scaling (DVFS) is the most pervasive. Unfortunately, most of the previous DVFS schemes only consider the power consumption of on-chip cores and overlook the overhead of power delivery system. Moreover, most works target only compute-intensive applications while neglecting the memory-bound characteristics of applications [2][6][11].

In most existing energy minimization works using DVFS, the overhead of the power delivery system of the multi-core platform has been largely overlooked. The power delivery system delivers sufficient and stable power from the off-chip source to on-chip cores. Meanwhile, it also incurs a high energy overhead, especially for the multi-phase switching voltage regulators. A multi-phase voltage VR is comprised of multiple small VRs working in parallel, each of which is called a VR phase, and all VR phases operate in an interleaving mode to share the total burden of delivering output current. Recent studies suggest that a typical multi-phase off-chip VR can consume an overhead power of up to 20%-50% of its input power [13][14]. Hence, it is critical to optimize the power consumption of a VR considering its salient power overhead. Previous works have shown that disconnecting some phases at light load, denoted as phase scaling, can improve the conversion efficiency and reduce the power overhead of a VR [13][14][15]. Nevertheless, these works only focus on the power losses of a VR itself and determine the number of active phases according to the flow-in current at circuit level instead of considering the whole system power consumption. Few works have explored VR's phase scaling from a system perspective [16][17][18]. However, none of the methods consider the impact of phase scaling of a multi-phase VR on DVFS selection for energy optimization of the whole platform. All these works determine the DVFS level of the cores first, and then select the active phase of VR based on the flow-in current. The shortcoming of these methods is that during the decision of DVFS setting, the phase scaling of multi-phase VR is overlooked, which will lead to an inefficient DVFS level for the whole platform. In return, the phase scaling of VR is also affected by the inefficient DVFS level. The challenges to optimize energy consumption of the whole system lie in the interaction between DVFS levels and VR phase scaling.

In order to address aforementioned challenges, in this paper we propose a comprehensive convex-optimization-based approach for optimizing the per-core DVFS and phase scaling of the off-chip VR, such that the total energy consumption of a platform, including the on-chip cores and an off-chip multi-phase VR, can be minimized for independent execution of tasks with deadlines. In our proposed method, the DVFS setting is determined with an awareness of phase scaling of the off-chip VR. Meanwhile, phase scaling is set from a system level for energy minimization of the whole platform. In addition, our approach takes the CPU-bound and memory-bound characteristics of tasks into consideration to identify appropriate DVFS setting and VR phase number. Static power is also included in the power model of the cores. Based on the characteristics of tasks and the power models of the cores and VRs, we formulate a convex optimization problem that is able to provide an optimal setting of both DVFS and the off-chip VR within polynomial complexity [19][20]. Furthermore, to achieve better scalability and lower on-line

Zuomin Zhu and Wei Zhang are with the Department of Electrical and Computer Engineering, Hongkong University of Science and Technology, Clear Water Bay, Kowllon, Hong Kong. e-mail: {zzhuae, wei.zhang}@ust.hk

Vivek Chaturvedi are with Nanyang Technological University, P.O. Box 1212, Singapore.e-mail: vchaturvedi@ntu.edu.sg

Amit Kumar Singh are with School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, UK. Email: a.k.singh@essex.ac.uk

overhead, the proposed convex method is decomposed into two stages: an off-line stage and an on-line stage. During the off-line stage, we explore the convex model by assuming different numbers of active phases of the VR, and various workload pressures and workload characteristics to collect the optimal frequency assignments under different scenarios, and store them in a look-up table. During the on-line stage, based on actual workload pressure and its characteristics processed by the cores, the specific frequencies for cores and active phase number of the VR are selected from the table and applied to the system. To validate the efficiency and scalability of the two-step comprehensive convex optimization approach, we evaluate our algorithm on three different targeted platforms consisting of four cores, eight cores and sixteen cores, respectively. Experiments on real benchmarks show that comparing with previous approaches, our method can achieve significant energy savings of up to 22.4%.

In summary, the main contributions of this work include:

- To achieve the system level energy optimization, we consider the interaction between per-core DVFS and phase scaling of VR and develop a comprehensive convex optimization model.
- To reduce the on-line overhead and improve scalability, we decompose the proposed convex-optimization model into two stages, an off-line stage and an on-line stage, which significantly reduces the optimization time without incurring energy overhead.
- We investigate our proposed model on four-core, eight-core and sixteen-core platforms, respectively. The experimental results show that our method has reduced the system energy consumption by up to 22.4% with good scalability.

The rest of this paper is organized as follows. Section II discusses existing methods for energy consumption minimization on multi-core systems. Section III explains backgrounds and preliminaries for a multi-phase VR, and also introduces our task latency model as well as power model. A motivational example and the proposed convex formulation are presented in section IV, followed by the two-stage decomposition of the convex optimization approach. Section V describes the experimental results and analysis. Section VI concludes the paper.

II. Related Work

The energy minimization of multi-core processors using DVFS policies has been widely used, and many related approaches have been proposed [5][6][7][8][21][2]. Slack minimization, which selects the lowest possible frequency to extend the execution of a task to minimize energy consumption while catching the deadline, is most commonly adopted [5][9][10]. In [5], energy minimization is achieved by exploiting the execution slacks where static energy is not taken into consideration. Convex models have been formulated to achieve energy optimization in [8][6][21]. In [8], several DVFS strategies under bounded execution times are proposed. However, only dynamic power consumption is considered. In [6][21], the energy consumption of a core is modeled at cycle level, which cannot take the influence of task characteristics; for example, the memory access latency, into consideration. In [7], energy efficiency is formulated as a machine learning problem and much system data at runtime are collected for DVFS classification. This method incurs huge training overhead offline and high computational complexity at run-time. Marco et al. [22] introduces a memory-boundedness aware DVFS algorithm to exploit memory-bounded tasks for slack claim, yet it neglects the influence of static power on total energy. In [2], an analytical energy model is proposed to give the DVFS level for energy minimization through computing the first derivation. However, it only works for the compute-intensive workload, while applications with memory-bound characteristics are not considered. Moreover, all the above works do not consider the overhead of the off-chip VR, and their models cannot be directly extended to include the impact of the VR.

A power delivery system has a significant energy overhead due to its non-trivial parasitic resistance and capacitance of the VRs, which has been demonstrated in previous works [15][9][14][13][23]. However, most previous works on VRs have only concentrated on the components of the VR to improve the power efficiency of the VRs alone. In [23], a convex model is formulated to determine the parameters (channel width of MOS, phase number, switching frequency, etc.) of on-chip and off-chip VRs for exploring the tradeoff between the advantages and costs of employing on-chip VRs. In [13], hybrid power delivery system with both on-chip and off-chip VRs is shown to be more effective in maintaining high efficiency in a large range of output loads than the conventional paradigm with off-chip VRs.

In order to improve the energy efficiency of a multi-phase VR, there are also previous works proposed to explore the phase scaling at light loads [15][14][13][24]. In [24], a simple technique of dynamically changing the number of phases as a function of load is proposed for reducing fixed losses at light load. In [15], a time-optimal digital controller for the phase scaling, which is implemented in the FPGA, is introduced. In [14], a look-up table storing the maximum load current value of the highest efficiency indexed by the P-state value and the number of active phases, is formed offline and used at run-time to select the optimal active phase of the VR. Edward et al. [25] introduces a system control method for fully integrated voltage regulator on a 4th generation Intel core based on the current activity level of the domain. In [13], a quantized power management scheme is used to disconnect active phases based on the load. As just discussed, all these above works have only focused on reducing the power loss of VRs itself while ignoring the interaction between the power consumption of VRs and the cores.

Despite the few recent papers that have explored VRs from a system perspective as in [9][17][16], little attention has been paid to the question of how to maximize the energy saving of a multi-core platform from both VR and DVFS optimization. Kim et al. [9] explore the potential system-wide energy savings of implementing both off-chip and on-chip VRs in a 4-core CMP system. They apply an integer linear programming (ILP) to determine the DVFS levels for each core at offline. However,

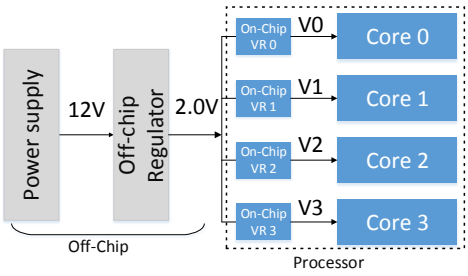


Fig. 1: System model of a four-core platform

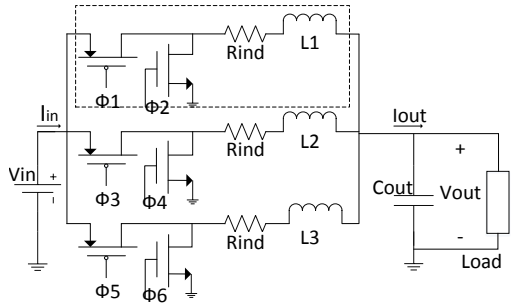


Fig. 2: Circuit diagram of a multi-phase VR

the ILP model only includes the power consumption of the cores while neglecting the power overhead of VRs. Choi et al. in [16] proposes a DVFS policy that is aware of the VR overhead characteristics for low-power embedded systems. They derive the optimal frequency of a core based on a power model to minimize the total energy consumption in both the core and the VR. However, the method only targets a single-phase VR for a single core. In [17] and [18], a VR consolidation method (VRCon), that combines cores of the same voltage level as well as relatively small amount of load current to be powered by a single VR, is proposed to reduce the VR power loss in the multi-core platform. However, the VRCon method assumes a VR-to-core distribution network, which incurs huge area and power overhead. Therefore, from a system perspective, how to efficiently modulate the phases of multi-phase VRs and adaptively adjusting DVFS levels of cores according to the system workload is worth exploring. In this paper, we propose an energy optimization method that combines phase scaling of VRs with DVFS to minimize the total energy consumption of a multi-core platform.

III. System Design

A. System Model

Fig.1 shows a targeted system that is composed of an off-chip multi-phase VR and four one-phase on-chip VRs supporting per-core DVFS [9]. For a bigger system containing eight or sixteen on-chip cores, the system model is similar to Fig.1. The off-chip VR performs the first step of converting the power supply voltage, assumed to be 12 V [13], to an intermediate voltage of 2.0 V [13], which is then shared by the on-chip components. The intermediate voltage, denoted as V_{int}, drives four on-chip regulators that further step down the voltage to different levels supplied to on-chip cores.

The off-chip VR is usually implemented as a multi-phase VR to deliver a high load current as increased number of phases help to reduce conduction losses and improve transient response [14][13][23]. On the other hand, the conversion efficiency for a multi-phase VR is usually quite low at a light load if all phases remain on-state, due to its fixed switching losses and control logic losses [13]. We consider phase scaling for off-chip VR as a control knob for energy minimization for the whole platform. For an on-chip VR supporting per-core DVFS, the load current is usually relatively smaller. The sizes of MOS transistors and inductors are smaller, while the switching frequency is usually much higher than that of an off-chip VR [9][13], which incurs a nonnegligible power overhead. Therefore, it is important to take the power overhead of an on-chip regulator into consideration when determining the DVFS level of its associated core for energy minimization. In this work, we derive the parameters of an off-chip and on-chip VR using PowerSoc considering a six phase off-chip VR and one-phase on-chip VR [13].

B. Voltage Regulator

A switching voltage regulator commonly consists of MOS power transistors, inductors and capacitors, as well as the feedback control circuit, as shown in Fig.2 [9]. The control circuit switches on/off MOS power transistors at a certain frequency to generate a pulse wave, which then goes through a low-pass filter composed of the inductor and capacitor, thereby providing a steady output voltage for its load. But this does not come free. The parasitic capacitance and resistance of the MOS and inductors incur non-negligible power overhead. Therefore it is important to model power losses of the VR for better power efficiency of the whole system. There are four main parts of power losses to discuss. We adopt the simple yet efficient model in [13]:

$$P_{driver} = C_{eff} V_{driver}^2 f_{sw} \tag{1}$$

$$P_{Ron} = (DR_{on,H} + (1-D)R_{on,L}) \cdot (I_{ind}^2 + \frac{\Delta I_{ind}^2}{12}) \tag{2}$$

$$\Delta I_{ind} = \frac{D(V_{in} - V_{out})}{f_{sw} L_{ind}} \tag{3}$$

$$P_{Rind} = R_{ind} \cdot (I_{ind}^2 + \frac{\Delta I_{ind}^2}{12}) \tag{4}$$

$$P_{ctrl} = I_{ctrl} V_{driver} \tag{5}$$

$P_{driver}, P_{Ron}, P_{Rind}$ and P_{ctrl} represent the switching loss of the MOS power bridge, resistive loss of the MOS power transistors, resistive loss of the inductor and the power loss of the control circuit, respectively. C_{eff} is the effective switching capacitance of the MOS transistors and f_{sw} is the switching frequency of the transistors. V_{driver} denotes the supply voltage of the drivers and control logic. $R_{on,H}$ and $R_{on,L}$ denote the on-state parasitic resistance of high/low-side MOS transistors and R_{ind} denotes the parasitic resistance of the inductor. I_{ind} and ΔI_{ind} are the average and peak-to-peak value of the inductor current in one phase, respectively. V_{in} and V_{out} are the input and output voltage of the regulator, respectively. D is the duty ratio of the gate signal and I_{ctrl} stands for the supply current of the control circuit of each phase. In addition to the power losses listed above, there are other negligible power losses, e.g. static power P_{stat} and short circuit power

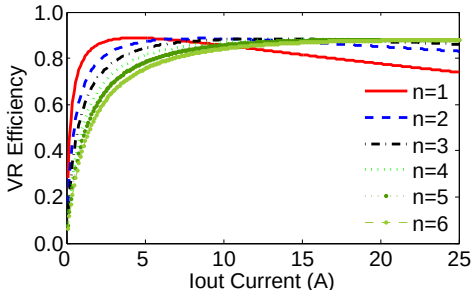


Fig. 3: Conversion efficiency of multi-phase VR (n: active phase number of a VR)

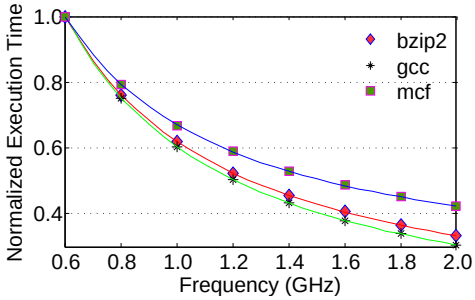


Fig. 4: Execution time of tasks at various frequency

P_{sc}. Note that for an off-chip VR and on-chip VR, the above parameters are totally different and we use the superscripts ^{off} and ^{on} to distinguish them. For an on-chip VR, the overhead can be expressed as

$$P_{vr}^{on} = P_{driver}^{on} + P_{Ron}^{on} + P_{Rind}^{on} + P_{ctrl}^{on} + P_{stat}^{on} + P_{sc}^{on}, \quad (6)$$

A multi-phase VR is comprised of multiple small VRs working in parallel, which provides several advantages in terms of output fluctuation reduction and faster response, but incurs high power losses. The components in the dashed box in Fig.2 form a phase for the multi-phase regulator. Thus, the total power loss of an off-chip VR with n active phases is given by

$$P_{vr}^{off} = n \cdot (P_{driver}^{off} + P_{Ron}^{off} + P_{Rind}^{off} + P_{ctrl}^{off} + P_{stat}^{off} + P_{sc}^{off}), \quad (7)$$

As shown in Fig.3, the conversion efficiency of a multi-phase VR is highly dependent on the active phase number n and its load. When the load is reduced, some phases can be disconnected to share the load current among a decreased number of phases to improve the power efficiency of the VR. In this paper, the number of active phases is one of the control knobs for energy optimization.

C. Characterization of Task Latency

A task is a sequence of instructions to be executed. Different kinds of instructions can incur different on-chip and off-chip latency, due to data dependency, cache miss etc. Based on the latency type, they can be classified into CPU-bound or memory-bound instructions [9][12]. Speeding up the processor helps to reduce the CPU-bound latency but it will not affect the time taken by memory-bound instructions. Thus, the execution time of a task j, t_j can be modeled as in [12]:

$$t_j = u_j + \frac{w_j}{f}, \quad (8)$$

where the first term u_j represents the memory-bound latency, which does not change with varying operating frequency, and the second term w_j/f represents the CPU-bound latency, which can be reduced by increasing the core's operating frequency. The parameters u_j, w_j are constants and depend on the characteristics of a task. $\langle t_j, f \rangle$ pairs can be collected by running a task with different operating frequencies on gem5 simulator [26] at offline stage, and then the parameters u_j, w_j can be derived from the linear regression model.

Fig.4 shows the normalized execution time of three benchmarks from SPEC2006 [27] running with different operating frequencies. We can see that the change of execution time with increasing frequency differs from task to task. Thus, it is not sufficient to just use a simple inverse linear model to describe the task execution time as in [6][5]. We have validated the model Eq.(8) by comparing the predicted execution time with the actual execution time collected on gem5, and the result demonstrates that model Eq.(8) incurs less than 1% error, as illustrated in Fig.4 where the fitting curves match the scattered dots very well.

To ease understanding of the paper, we define two necessary terminologies to indicate the characteristics of tasks as follows:

- Workload Pressure: If there is only one task j mapped on a core, the workload pressure of the core is defined as
$$\psi_j = \frac{u_j + w_j/f_{max}}{T_{int}}, \quad (9)$$
where f_{max} and T_{int} represent the maximum frequency and deadline, respectively. Higher workload pressure suggests that the core needs to run faster to finish all tasks before the deadline. $\psi > 1$ indicates that even when applying the highest frequency some tasks still can not catch the deadline.
- CPU-bound Ratio: The CPU-bound ratio of tasks is defined as
$$\phi_j = \frac{w_j}{u_j + w_j}, \quad (10)$$
A task with a high CPU-bound ratio means that w_j dominates over u_j in Eq.(10), which suggests that the processing time of the task t_j is dominated by the w_j/f part and can be obviously reduced by slightly increasing the operating frequency. On the contrary, a low CPU-bound ratio indicates that the processing time of the task is dominated by u_j, and t_j does not change visibly with varying operating frequency. In this paper, the CPU-bound ratio of task is also referred to as workload characteristics and these two terms are used interchangeably.

D. Power Model

For each on-chip core, the overall power consumption is composed of dynamic power and static power. Following the simple yet sufficiently accurate model in [6], we model the overall power consumption of core i as:

$$P_i = CV_i^2 f_i + V_i \cdot I_{leak}, \quad (11)$$

where the first term represents the dynamic power and the second term represents the static power. V_i is the supply voltage for core i, which is matched with its frequency f_i. I_{leak} is the leakage current, and C is the circuit effective capacitance. We adopt the leakage power model in [28][29] for

its simplicity and enough accuracy, where I_{leak} in this model is assumed to be a fixed parameter that accounts for 30% of the total current at the nominal frequency. For DVFS, the relationship between voltage and its corresponding frequency can be modeled as a linear function [9] as:

$$V_i = k \cdot f_i + V_0, \tag{12}$$

where k and V_0 are constants depending on the manufacturing technology and can be derived from the linear regression model based on the given available $\langle V, f \rangle$ pairs.

IV. Convex model based Energy Minimization

Different from the existing approaches for energy minimization, our convex formulation incorporates the impact of multi-phase VR overheads and the interaction between VRs and cores. Besides this, application characteristics are also considered. Our proposed methodology selects the optimal number of active phases of the off-chip VR and per-core DVFS setting based on convex optimization. The optimization objective is to minimize system level energy consumption comprising of energy consumed by the on-chip cores and off-chip multi-phase VR. As a result, we develop a comprehensive power management methodology for minimizing the total energy consumption of the complete system. In this section, we first present a motivational example to illustrate the necessity of the system level optimization with VR scaling. Then, a comprehensive convex model to achieve the energy minimization of the whole system is derived in detail. Finally, in order to achieve better scalability, we decompose the comprehensive convex model into a two-stage algorithm to significantly reduce the online running time.

A. Motivational Case Study

In this subsection, we explain why the comprehensive modeling of system energy consumption is important through a simple example. For ease of understanding, we adopt the four-core platform shown in Fig.1. The technology parameters of cores and VRs are presented in section V-A. We consider a processor with sixteen voltage settings, from 1.340 V down to 0.988 V with scaled frequencies from 2 GHz down to 600 MHz, similar to the $\langle V, f \rangle$ pairs in work [30]. To make the motivational example easy to follow without affecting generality, we assume four cores executing the same duplicated tasks. Two different cases are taken as an example here: $Case\,1$, running $Task\,1$, $\tau 1$ with parameter $u_{\tau 1} = 0$, $w_{\tau 1} = 0.3$ on all four cores; $Case\,2$, running $Task\,2$, $\tau 2$ with parameter $u_{\tau 2} = 0.85$, $w_{\tau 2} = 1.3$ on all four cores, with a deadline equal to 5 ms. We select tasks with different characteristics (workload pressure and CPU-bound ratio) to show the drawbacks of existing energy optimization methods.

For comparison, we implement two representative and effective energy minimization methods in [2] and [5] as our baseline algorithms. In [2], an analytical energy model of an interval including static power is proposed. It minimizes the energy model and derives the DVFS level off-line. This model gives the optimal DVFS level setting without incurring online overhead. However, as most previous works, this method only assumes the task as CPU-bounded. We denote this baseline algorithm as Analytical Model[2]. To model the power consumption of a core according to the model in [2], $p_{core}(f) = b \cdot f^a + s$, we use a curve-fitting and obtain $b = 1.699, a = 1.721$ and $s = 1.412$ based on Eq.(11) and Eq.(12) for our experimental setup. We also implement the representative slack minimization technique as proposed in [5] as another baseline algorithm for comparison. Slack minimization is a commonly used efficient method for minimizing energy consumption, and it selects the lowest operating frequency to execute a task while catching its deadline. However, this method cannot take the static energy into consideration, yet the static part has significantly contributed to the total energy consumption.

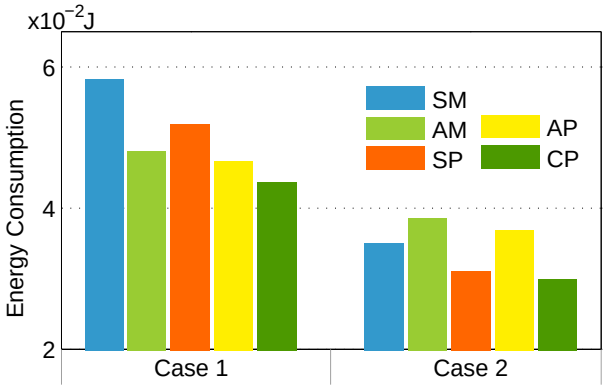


Fig. 5: Energy consumption of the whole platform for two cases. SM: Slack minimization only [5], AM: Analytical model for DVFS [2], SP: Slack minimization [5] + flow-in current based phase scaling, AP: Analytical model [2]+ flow-in current phase scaling, CP: Proposed comprehensive convex model combing DVFS and VR phase scaling

Since [2] and [5] only target the DVFS setting optimization without considering VR overhead, in order to make a fair comparison for the system level optimization, we construct another two representative baselines by applying the DVFS strategies in [2] and [5], followed by the state-of-art flow-in-current-based VR phase scaling method in [14][13]. In the following experiments, we will use the following four baselines to compare with our proposed system energy minimization model:

- Baseline 1 (SM): Slack minimization [5] without considering phase scaling.
- Baseline 2 (AM): Analytical model [2] without considering phase scaling.
- Baseline 3 (SP): Slack minimization [5] followed by flow-in-current-based VR phase scaling.
- Baseline 4 (AP): Analytical model [2] followed by flow-in-current-based VR phase scaling.

Fig.5 illustrates the energy consumption of the entire platform when applying different energy optimization strategies for the two cases. For both cases, slack minimization [5][9][10] selects 0.6 GHz which is the lowest possible frequency, and the analytical model in [2] selects 1.3 GHz which is derived as the optimal point from the energy model. However, from the histogram in Fig.5, SM and AM performs better in different case. It is because SM does not consider static power, and AM derives the frequency offline which cannot adapt to the variation of online tasks. This drawback

becomes more serious when the model does not consider the memory-bound task characteristics.

Moreover, comparing SP and AP to SM and AM, respectively, we can see that phase scaling of VRs can greatly reduce the overall energy consumption of the whole platform. However, all previous works for energy minimization determines the DVFS level of the cores first, and then selects the active phase of the VR based on the flow-in current. This means that in the first step of the DVFS setting, the overhead of the multi-phase VR is not considered, which will lead to a non-optimal DVFS level for the whole platform. In return, the phase scaling is also degraded by the non-optimal DVFS level setting. For example, in Fig.5, for $Case\,1$, SP determines DVFS level of 0.6 GHz and phase scaling n=1, while our proposed comprehensive algorithm determines the DVFS level of 1.1 GHz and n=2 phase for the VR. For $Case\,1$, CP outperforms the conventional SP by a 13.3% in terms of energy saving. Although our proposed algorithm increases the frequency and activates more phases of VR, it leads to lower energy consumption. The slightly higher frequency can greatly reduce the execution time of a fully compute-intensive task like $Case\,1$, which can substantially reduce the static energy consumption and offset the increase of dynamic energy. For $Case\,2$, AP determines the DVFS level of 1.3 GHz and phase scaling n=2 phases, while our proposed comprehensive algorithm determines the DVFS level of 0.7 GHz and n=1 phase for the VR. This decision achieves a 19.4% energy saving. Thus, it can be seen that, compared to the conventional methods of determining DVFS, which are unaware of the overhead of VRs followed by phase scaling based on the flow-in current, there is still great room to improve the energy saving in the multi-core platform with a multi-phase VR. Thus we propose a comprehensive convex-optimization-based approach that incorporates the per-core DVFS and phase scaling of VR in one convex model, with the CPU-bound and memory-bound characteristics of applications considered.

B. Problem Formulation

Given the system models and task model described in section III, a set of independent tasks $\boldsymbol{\tau} = \{\tau_1, \tau_2, \cdots \tau_j, \cdots\}$ are assigned to the on-chip cores $\boldsymbol{Cores} = \{C_0, C_1, \ldots, C_m\}$ at the beginning of every DVFS interval T_{int}, during which the DVFS level does not change. Assuming that during the task assignment and DVFS interval setting, all the assigned tasks must be finished in the current DVFS interval. The objective of our optimization problem is to set the frequency of on-chip cores and select the number of active phases for off-chip VR, such that the total energy consumed by the system is minimized.

Since task mapping to cores is not the focus of this work, we employ a representative load balancing algorithm [31] for initial task assignment. Once the tasks are mapped to cores, tasks assigned to the same core are clubbed together to form a hyper task. A hyper-task is the collection of all the tasks mapped to the same core. Within a hyper-task, intra-task scheduling is not required as these are independent of each other. The total execution time of all tasks assigned to C_i can be represented as

$$\begin{aligned} T_i &= \sum_{j \in C_i} u_j + \sum_{j \in C_i} w_j / f \\ &= U_i + \frac{W_i}{f}, \end{aligned} \tag{13}$$

where U_i and W_i represent the summation of u, w of all respective tasks assigned to core C_i. Thus the deadline constraint is expressed as

$$U_i + \frac{W_i}{f} \leq T_{int}, \tag{14}$$

According to Eq.(9) and (10), the workload pressure and CPU-bound ratio of the hyper task can be expressed as:

$$\Psi_i = \frac{U_i + W_i / f_{max}}{T_{int}}, \tag{15}$$

$$\Phi_i = \frac{W_i}{U_i + W_i}, \tag{16}$$

Higher workload pressure suggests that the core needs to run faster to finish all tasks before the deadline. $\Psi_i > 1$ indicates that even when applying the highest frequency some tasks still can not catch the deadline. Thus the number of tasks that can be assigned to a core must conform to the constraint $\Psi_i \leq 1$.

C. Comprehensive Convex Model

Based on the power model in section III-D, the phase current I_{ind}^{off} and the number of active phases n of the off-chip VR need to satisfy a constraint that the output power of the off-chip VR is equal to the power consumption of the on-chip components:

$$n \cdot I_{ind}^{off} \cdot V_{int} = \sum\nolimits_i (C V_i^2 f_i + V_i \cdot I_{leak} + P_{vr,i}^{on}) \tag{17}$$

In the same way, for the i-th on-chip VR , the phase current I_i^{on} satisfies:

$$I_i^{on} \cdot V_i = C V_i^2 f_i + V_i \cdot I_{leak} \tag{18}$$

Note that these sums of quadratic equality Eq.(17) and (18) do not conform to the rules of geometric programming (GP) [19]. We can relax it to an inequality, and it is shown in [20] that the relaxed problem is equivalent to the original problem and is able to derive the same optimal results.

From Eq.(1) to Eq.(7), P_{driver}, P_{ctrl} and P_{stat} for both an on-chip and off-chip VR are constants depending on the characteristics of the VR. P_{Ron}^{off}, P_{Rind}^{off} are directly proportional to square of I_{ind}^{off}, and $P_{Ron,i}^{on}$, $P_{Rind,i}^{on}$ are proportional to the square of I_i^{on}. Please note that I_{ind}^{off} and I_i^{on} are viewed as intermediate variables in the following convex model in Eq.(22), thus P_{Ron} and P_{Rind} in Eq.(2) and Eq.(4) also exhibit a quadratic equality constraint. Similar to the constraint in Eq.(17), the power loss model of the off-chip VR in Eq.(7) can also be relaxed to an inequality constraint.

$$P_{vr}^{off} \geq n \cdot (P_{driver}^{off} + P_{Ron}^{off} + P_{Rind}^{off} + P_{ctrl}^{off} . + P_{stat}^{off} + P_{sc}^{off}) \tag{19}$$

Energy consumption of the whole system is comprised of two parts: the energy consumed by on-chip components, and the energy consumed by an off-chip VR

$$E_{tot} = E_{on\_chip} + E_{off\_vr}. \tag{20}$$

E_{on_chip} denotes the energy consumed by all cores and its

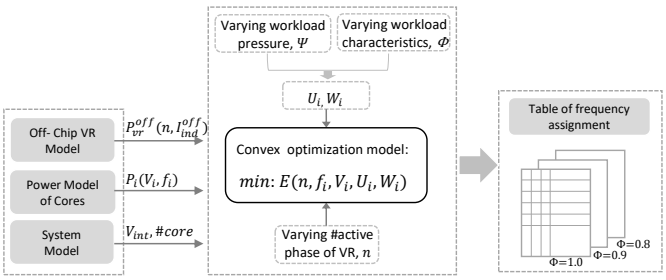


Fig. 6: Off-line stage of the comprehensive convex optimization method

associated on-chip VRs, which is the summation of the energy of the individual core and its associated on-chip VR. Thus,

$$E_{on\_chip} = \sum_i (CV_i^2 f_i + V_i \cdot I_{leak} + P_{vr,i}^{on})(U_i + \frac{W_i}{f_i}). \quad (21)$$

E_{off_vr} represents the energy overhead of the off-chip VR during a DVFS interval, which is the product of power consumption, P_{vr}^{off}, and activation time, t, of the off-chip VR.

Combining all the constraints listed above, the problem formulation is presented as follows:

$$\begin{aligned}
&\min_{\substack{f_i, n, \\ I_{ind}^{off}, t}} \sum_i (CV_i^2 f_i + V_i \cdot I_{leak} + P_{vr,i}^{on})(U_i + \frac{W_i}{f_i}) + P_{vr}^{off} \cdot t \\
&s.t.: P_{vr}^{off} \geq n \cdot (P_{driver}^{off} + P_{Ron}^{off} + P_{Rind}^{off} + P_{ctrl}^{off} \\
&\qquad + P_{stat}^{off} + P_{sc}^{off}) \\
&\quad P_{vr,i}^{on} \geq P_{driver}^{on} + P_{Ron,i}^{on} + P_{Rind,i}^{on} + P_{ctrl}^{on} \\
&\qquad + P_{stat}^{on} + P_{sc}^{on} \\
&\quad n \cdot I_{ind}^{off} \cdot V_{int} \geq \sum_i (CV_i^2 f_i + V_i \cdot I_{leak} + P_{vr,i}^{on}) \\
&\quad I_i^{on} \cdot V_i \geq CV_i^2 f_i + V_i \cdot I_{leak} \\
&\quad V_i = k \cdot f_i + V_0 \\
&\quad U_i + W_i / f_i \leq t \\
&\quad t \leq T_{int} \\
&\quad f_{min} \leq f_i \leq f_{max} \\
&\quad 1 \leq n \leq N.
\end{aligned} \quad (22)$$

where f_{min} and f_{max} represent the restrictions of the minimum and maximum frequency, and N represents the total number of phases of the off-chip VR.

The above problem formulation is actually a GP problem that can be converted to a convex optimization problem with polynomial complexity using logarithmic transformation [20]. In Eq.(22), the core's operating frequency f_i and the number of active phases n are continuous real variables. After f_{i_opt} and n_{opt} are found in the solver, which will be discussed later in the paper, we need to map these optimal values to an available frequency and an integer value of the number of active phases. To guarantee to catch the deadline, the core's frequency f_i is set to an available frequency that is closest to and not smaller than f_{i_opt}. The number of active phases of the off-chip VR, n, is set to the nearest integer to n_{opt}, i.e. $n = \lfloor (n_{opt} + 0.5) \rfloor$.

Incorporating the selection of the active phase number of the off-chip VR and DVFS into a comprehensive convex model leads to a bigger exploration space as compared to those considering the DVFS of the processor alone, and gives a globally optimal solution for energy minimization in polynomial time. Besides this, the latency characteristics of tasks and the overhead of the off-chip VR are precisely modeled. Compared to previous works, these advantages greatly help to optimize the total energy of the whole system.

TABLE I: Optimal frequency settings of the comprehensive convex optimization method when workload characteristics $\Phi = 0.9$

Workload Pressure		f_{opt}(GHz)					
		n=1	n=2	n=3	n=4	n=5	n=6
Ψ=0.11	U=0.1 W=0.9	0.94	0.99	1.02	1.05	1.08	1.11
Ψ=0.22	U=0.2 W=1.8	0.94	0.99	1.02	1.05	1.08	1.11
Ψ=0.33	U=0.3 W=2.7	0.94	0.99	1.02	1.05	1.08	1.11
Ψ=0.44	U=0.4 W=3.6	0.94	0.99	1.02	1.05	1.08	1.11
Ψ=0.55	U=0.5 W=4.5	1.00	1.00	1.02	1.05	1.08	1.11
Ψ=0.66	U=0.6 W=5.4	1.23	1.23	1.23	1.23	1.23	1.23

TABLE II: Optimal frequency settings of the comprehensive convex optimization method when workload characteristics $\Phi = 0.8$

Workload Pressure		f_{opt}(GHz)					
		n=1	n=2	n=3	n=4	n=5	n=6
Ψ=0.30	U=0.5 W=2.0	0.84	0.87	0.90	0.93	0.95	0.98
Ψ=0.36	U=0.6 W=2.4	0.84	0.87	0.90	0.93	0.95	0.98
Ψ=0.42	U=0.7 W=2.8	0.84	0.87	0.90	0.93	0.95	0.98
Ψ=0.48	U=0.8 W=3.2	0.84	0.87	0.90	0.93	0.95	0.98
Ψ=0.54	U=0.9 W=3.6	0.88	0.88	0.90	0.93	0.95	0.98
Ψ=0.60	U=1.0 W=4.0	1.00	1.00	1.00	1.00	1.00	1.00

D. Two-Stage Decomposition

Although the comprehensive convex model in Eq.(22) can give the globally optimal setting of per-core DVFS and optimal active phase number of the VR, the complexity of the convex model in Eq.(22) is polynomial in the number of variables and constraints [20][32]. This means that the running time to solve the convex problem increases sharply with the increasing number of cores since more cores lead to more variables f_i and constraints on f_i. For example, for a four-core platform, the specific GP solver, GGPLAB [33] takes around 3 ms to solve the convex model in Eq.(22) while for an eight-core platform, the solving time increases to 7 ms, as discussed in Sec.V-A. This makes it infeasible to find the optimal solution at the on-line stage within one DVFS interval.

The sharply increasing time overhead highly hinders the scalability of our convex model implementing at the on-line stage. Therefore, we decompose our comprehensive convex-optimization-based method into two stages: an off-line stage and an on-line stage, to reduce the on-line overhead.

Off-line Stage: The off-line stage of our method, which is performed at design time for a multi-core platform, is depicted in Fig.6. During the off-line stage, we solve the convex model in Eq.(22) with a convex optimization solver and collect the optimal frequency assignments corresponding to the different number of active phases of the VR for various workload pressures and workload characteristics, and store them in a look-up table. As shown in Fig.6, the off-chip VR model, power model of the cores and system model are obtained

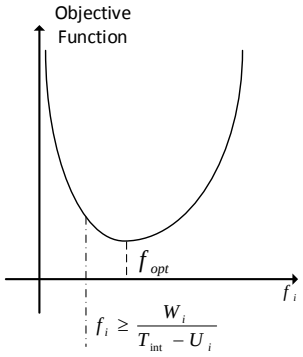


Fig. 7: Optimal solution to a convex problem with inactive constraints

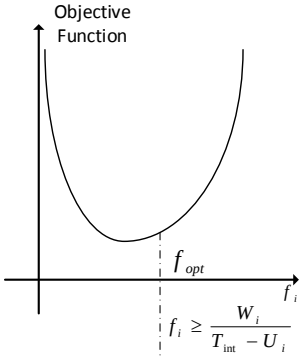


Fig. 8: Optimal solution to a convex problem with active constraints

as inputs. In the rounded rectangles, there are three varying values. When Ψ_i and Φ_i, which define the workload pressure and workload characteristics of the cores, respectively, are specified, U_i and W_i for tasks on cores can be derived from Eq.(15) and Eq.(16). The convex optimization procedure is solved for different workload pressures Ψ_i, different workload characteristics Φ_i, and n active phases of the VR. As repeating the procedure for all possible combinations of Ψ_i and Φ_i of the different cores leads to exponential complexity and infeasibility, we simplify the process by only iterating on one workload pressure value and one workload characteristics value for all cores. This is feasible since we assume a load balancing algorithm for task assignment, which leads to a relatively balanced workload distribution on every core. Furthermore, during the on-line stage, the different workload pressures and workload characteristics of every core are considered since we use the actual value of Ψ_i and Φ_i to determine the frequencies of cores, as discussed in the on-line stage in next subsection.

For each combination of these three variables, we substitute U_i, W_i and n with specific values in convex formulation in Eq.(22) and solve it at design time. Please note that we fix the active phase number n of the VR, as a specific integer value in every iteration of the convex optimization procedure. This is because the active phase number of the VR must be an integer and its available range is very limited due to the constraint of the total phase number of the VR [14].

To elucidate clearly the two-stage decomposition of our convex-optimization-based method, we present an example of applying our comprehensive convex model on a four-core platform as shown in Fig.1, where the off-chip VR and on-chip cores are modeled as in [13] and [30], respectively. The DVFS interval, T_{int}, is set to be 5 ms. Table I gives the optimal solutions of the off-line stage of our method for different workload pressures Ψ and different numbers of active phases of VR n when the workload characteristics Φ is fixed at 0.9. We can see that the optimal solutions in each column, corresponding to the same number of active phases of the VR, remain almost unchanged until Ψ increases up to 0.55 or 0.66, as shown in the blue cells. Similar to Table I, Table II, which gives the optimal solutions of the comprehensive convex model when the workload characteristics Φ is fixed at 0.8, also shows the same discipline that optimal solutions in each column remain unchanged until the workload pressure Ψ exceeds some threshold. Furthermore, it is easily found that the values in the blue cells are exactly the frequencies that catch the deadline constraint. For example, in the last row in Table I, where $U = 0.6, W = 5.4$, we can get the frequency lower bound $f \geq 1.23$ based on Eq.(14).

TABLE III: The look-up table created during off-line stage

Φ	f_{opt}(GHz)					
	n=1	n=2	n=3	n=4	n=5	n=6
Φ=1.0	1.1	1.1	1.2	1.2	1.3	1.3
Φ=0.9	0.9	1.0	1.0	1.1	1.1	1.1
Φ=0.8	0.8	0.9	0.9	0.9	1.0	1.0
Φ=0.7	0.7	0.8	0.8	0.8	0.8	0.9
Φ=0.6	0.7	0.7	0.7	0.7	0.8	0.8
Φ=0.5	0.6	0.6	0.6	0.7	0.7	0.7

From the optimal solutions in Table I and Table II, we conclude two facts:

1) In Table I and Table II, the workload characteristic Φ is fixed while varying Ψ and n. Based on the definition in Eq.(16), a fixed workload characteristic Φ means a fixed linear ratio between U_i and W_i, namely $U_i = \delta \cdot W_i$ where $\delta = (1-\Phi)/\Phi$. Thus, when the workload pressure increases (U_i, W_i increases), the objective function in the convex model (in Eq.(22)) actually does not change if we divide it with the coefficient W_i. As a result, the optimal solutions for the convex model remain unchanged if the deadline constraints are inactive, as shown in Fig.7 where an inactive constraint does not have any effect on the optimal solution.
2) When the workload pressure pushes the lower bound of the frequency to a value exceeding the original optimal solution, the deadline constraints become active. In this case, the optimal solution of the convex model turns out to be the value which exactly catches the deadline constraint, as shown in Fig.8 where an active constraint pushes the optimal solution to catch the constraint.

Based on the above facts, it is easily found that we do not need to store the whole content of Table I and Table II in the look-up table of the frequency assignment at design-time. What we need to store is only the optimal frequency assignments corresponding to the different number of active phases of the VR and various workload characteristics when the deadline constraint is inactive, namely when the workload pressure is low. During the on-line stage, the stored optimal frequency with the inactive deadline constraint is retrieved and compared with the lower-bound frequency derived from the deadline constraint. The bigger of the two values is exactly the optimal solution of the original convex model with the deadline constraint considered.

The look-up table we created at design-time is shown in Table III. The results in Table I and Table II correspond to the second and third row in Table III. Note that in this table we have mapped the optimal values in Table I and Table II to an available frequency that the on-chip cores support. Thus, fewer registers are needed for storing the look-up table since we can record the integer level number, which ranges from 1 to 15 (corresponding to 0.6 GHz to 2.0 GHz), instead

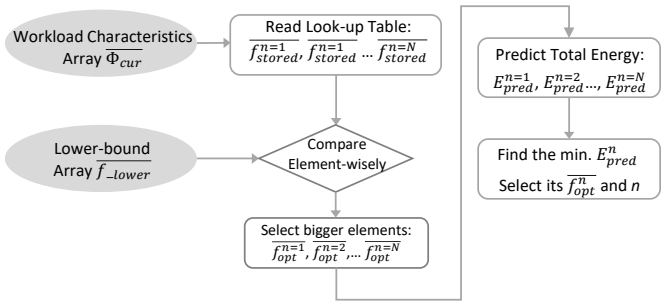


Fig. 9: Flow-chart of the on-line stage

of recording the floating-point frequency values. In this way, the 2-D array of Table III only takes 36 integers, namely 144 bytes if we assume an integer takes up 4 bytes. From Table III, we can see that the values of two adjacent rows, namely the optimal frequency assignments for two adjacent workload characteristics and the same number of active phases of VR, are very close. This means that the look-up table we create during off-line stage is fine-grained in terms of workload characteristics. Thus it can maintain a good optimality of the original convex model in Eq.(22).

On-line Stage: Fig.9 presents the overall flowchart of the on-line stage of our method. During the on-line stage, first, the workload characteristics array $\overline{\Phi_{cur}}$ of all cores are computed. Then, for active phase number $n = 1, 2, \ldots N$, the frequency assignments of all cores are retrieved from the look-up table indexed by n and the closest workload characteristics to $\overline{\Phi_{cur}}$. From Table III, we can see that when $\Phi = 0.5$, the frequency approaches to 0.6 GHz, which is the lowest available frequency. Thus, if $\Phi < 0.5$ for a hyper task on a core, we select 0.6 GHz as the frequency assignment for this core. In this way, we can get N frequency arrays $\overline{f^{n=1}_{stored}}, \overline{f^{n=2}_{stored}}, \ldots, \overline{f^{n=N}_{stored}}$ corresponding to active phase number $n = 1, 2, \ldots N$. Next, each of these N frequency arrays is element-wisely compared with the lower-bound frequency array $\overline{f_{\_lower}}$, where $\overline{f_{\_lower}}$ denotes the lower-bound frequency array of an individual core and is derived from Eq.(14). The bigger elements in each comparison form the candidate frequency assignments $\overline{f^{n=1}_{opt}}, \overline{f^{n=2}_{opt}}, \ldots, \overline{f^{n=N}_{opt}}$ for active phase number $n = 1, 2, \ldots N$. Thus, we have N local optimal settings of per-core frequency, and the active phase number of the VR, namely $\langle \overline{f^{n=1}_{opt}}, n{=}1 \rangle$, $\langle \overline{f^{n=2}_{opt}}, n{=}2 \rangle, \ldots$, $\langle \overline{f^{n=N}_{opt}}, n{=}N \rangle$. To find the most energy-efficient setting among these N choices, we use an exhaustive method since N is usually a small integer [14]. The total energy of the whole platform of these N local optimal settings, $E^{n=1}_{pred}, E^{n=2}_{pred}, \ldots E^{n=N}_{pred}$, can be predicted based on the objective function in Eq.(22). The minimum value in $E^{n=1}_{pred}, E^{n=2}_{pred}, \ldots, E^{n=N}_{pred}$ is easily found and the corresponding setting of $\langle \overline{f^{n}_{opt}}, n \rangle$ is exactly the most energy-efficient setting of per-core DVFS and active phase number.

From the flowchart in Fig.9, we can see that for a platform with M on-chip cores and the VR with N phases, the on-line stage performs NM integer reads and comparisons, followed by TM computations, where T is the number of terms in the energy model for each core in E_{on_chip} in Eq.(21). Thus, for a four-core platform with a total of 6 phases VR, the total number of cycles of the on-line stage of our decomposed algorithm only reaches up to several thousands of cycles. For a core running with 1 GHz, this only takes about several microsecond. Compared to the interval of 5 ms, the on-line overhead is negligible. Furthermore, compared to solving the comprehensive convex model in Eq.(22) at run-time, which takes around 3 ms for a four-core platform using the GGPLAB solver [33], our on-line stage has greatly reduced the on-line overhead and improved its scalability.

TABLE IV: Off-chip VR parameters for the four-core platform

Parameter	Value	Parameter	Value
$C_{eff}\,[F]$	1.40E-08	$R_{ds}\,[\Omega]$	4.35E-03
$V_{driver}\,[V]$	5	$R_{ind}\,[\Omega]$	9.09E-03
$f_{sw}\,[Hz]$	6.19E+05	D	0.175
$I_{ctrl}\,[A]$	2E-3	$L_{ind}\,[H]$	6.83E-07

TABLE V: On-chip VR parameters for the four-core platform

Parameter	Value	Parameter	Value
$C_{eff}\,[F]$	3.27E-10	$R_{ds}\,[\Omega]$	1.33E-02
$V_{driver}\,[V]$	2	$R_{ind}\,[\Omega]$	1.36E-02
$f_{sw}\,[Hz]$	8.2E+07	D	0.559
$I_{ctrl}\,[A]$	2E-3	$L_{ind}\,[H]$	1.36E-09

V. Experimental Results

In this part, we validate the efficiency of our approach through a series of experiments on real benchmarks. First, we present the experimental setup, followed by a thorough analysis of the results comparing our proposed methodology with the conventional DVFS technique.

A. Experimental Setup

System Configuration: We build our system using the processor and VR model described in section III-A and III-B. To show the scalability of our method, we use three different targeted platforms consisting of four, eight and sixteen cores respectively. We assume the Intel Haswell processors supporting per-core DVFS and the capacitance C for each core is derived from gem5, $C = 1.5nF$. The voltage-frequency pairs are based on the work in [30], which has been described in section IV-A. For the relationship between voltage and its corresponding frequency that is modeled in Eq.(12), we calculate the fitting constants using linear regression based on the available DVFS states and obtain that $k = 0.2467, V_0 = 0.8493$. The predicted voltages retrieved from Eq.(12) are compared against the given voltage levels in [30] and it is found that the model Eq.(12) only incurs a less than 1% error. The DVFS interval is set to be 5 ms and it can be increased by the OS scheduler if a longer interval is needed. For the off-chip and on-chip VR for a four-core platform, the parameters in Eq.(1) to Eq.(7) are obtained from work [13] and are listed in Table IV and Table V. For bigger platforms consisting of eight and sixteen cores, the parameters of the VR are obtained in the same way. For good performance and low complexity, the total number of phases of the off-chip VR is set to $N = 6$ according to work [14].

Convex Optimization Solver: During the off-line stage, we need to use a solver to find the global optimal solution for the comprehensive convex model. For a four-core platform as shown in Fig.1, the CVX [19], which is a general convex problem solver, takes around 2 seconds, while a specific GP solver, GGPLAB, only takes around 3 ms to determine the optimal solution [33]. As for a eight-core platform, the solving time increases to 6.5 s and 7 ms for CVX and GGPLAB, respectively. For a sixteen-core platform, the solving time increases to 8.5 s and 11 ms. In our experiment, we use the CVX solver for its acceptable running time and ease of programming. Note that the convex optimization solver is only applied at off-line stage to collect the optimal solutions for different inputs. Thus, the running time overhead of the solver lies in the off-line stage. With regard to on-line overhead, it only includes tens of accesses to the look-up table and comparisons between several entries, which incurs negligible overhead, as we have discussed in the on-line stage subsection in Section IV-D.

Task Benchmarks: Our task set comprises of independent tasks with a balanced mix of CPU-bound and memory-bound latency characteristics. To ensure the diversity, we use an extensive data set collected across all workloads in the SPEC 2006 benchmark suite [27]. The CPU-bound ratios of tasks in these benchmarks vary from 1.0 to 0.6. In the following experiments, we use the benchmark povray, soplex, dealII, gcc, mcf as the workload with CPU-bound ratios range from 1.0 to 0.6. The latency data for each benchmark is divided into 1 million instruction intervals, and these data are collected on gem5 using all available operating frequencies. Based on these latency data and their corresponding operating frequency, the curve fitting constants, u_j and w_j in Eq.(8), are calculated. As we have demonstrated in section III-C, the execution time model in Eq.(8) incurs a less than 1% error. Besides this, tasks with different CPU-bound ratios will be used to explore the effects of task characteristics on energy minimization methods.

Experiments: For comparison, we implement four baseline algorithms as discussed in section IV-A. Although there are some previous methods based on convex optimization for energy optimization [6][21], these methods cannot directly compare with our proposed model because their energy models are given at cycle level and only work for streaming applications. To fully evaluate our proposed model, we define two scenarios of applying our convex model:

- Scenario 1: The reduced convex model, which only considers per-core DVFS and is oblivious to VR phase scaling, where only $\overline{f_{stored}^{n=N}}$ is retrieved from the look-up table and active phase number is fixed at $n = N$ during the on-line stage.
- Scenario 2: The complete convex model (in Eq.(22)) combing per-core DVFS and VR phase scaling, where $\overline{f_{stored}^{n=1}}, \overline{f_{stored}^{n=2}}, \ldots \overline{f_{stored}^{n=N}}$ are all retrieved for active phase number $n = 1, 2 \ldots N$.

In all, we conduct four different experiments to assess the energy efficiency of our proposed method. First, we demonstrate the advantages of applying DVFS with the help of our convex model through comparing $Scenario\,1$ with $Baseline\,1$ and $Baseline\,2$. Next, to highlight the importance of considering the VR overhead, we conduct an experiment to compare $Scenario\,2$ with $Scenario\,1$. Then, $Scenario\,2$ is compared against $Baseline\,1$ and $Baseline\,2$ to show the overall advantage of our approach. Finally, we validate our proposed comprehensive method against the decoupling method through comparing $Scenario\,2$ with $Baseline\,3$ and $Baseline\,4$.

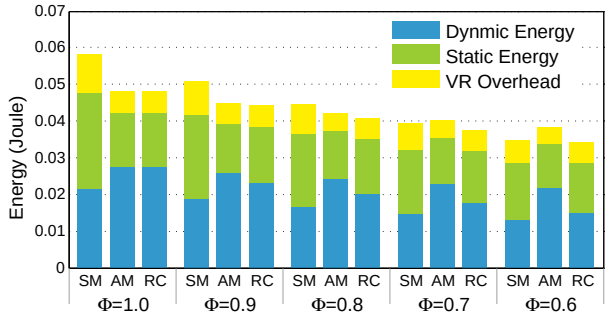


Fig. 10: Energy consumption of an interval of the whole platform at various workload characteristics when workload pressure Ψ=0.3; (SM: Slack minimization, AM: Analytical model [2], RC: Reduced convex model $Scenario\,1$).

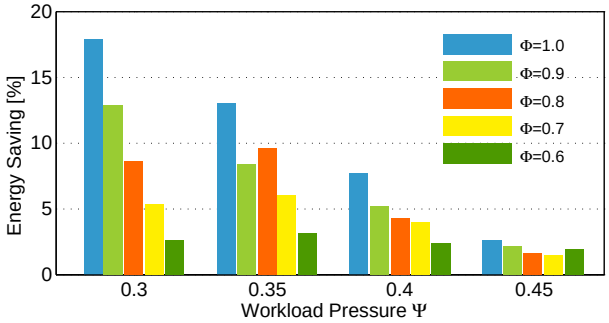


Fig. 11: Energy saving of the reduced convex model over slack minimization at various workload pressures for five different workload characteristics Φ.

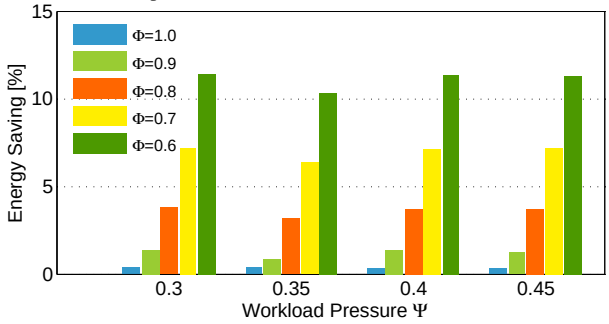


Fig. 12: Energy saving of the reduced convex model over the analytical model[2] at various workload pressures for five different workload characteristics Φ.

B. Results and Analysis for the Four-core Platform

1) Reduced convex model: We first compare the reduced convex model with the traditional $Baseline\,1$ and $Baseline\,2$ to demonstrate the advantage of determining DVFS level using a convex model with task characteristics considered.

Fig.10 illustrates the breakdown of average energy consumption of an interval of the whole platform at various workload characteristics Φ when Ψ=0.3. For a fair comparison, the active phase number n in the reduced convex model is set to be fixed at N through only retrieving $\overline{f_{stored}^{n=N}}$ during

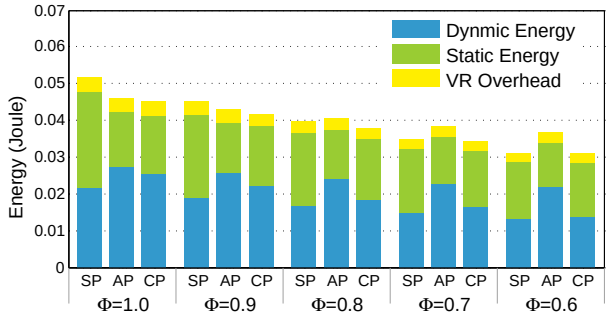


Fig. 13: Energy consumption of an interval of the whole platform at various workload characteristics when the workload pressure Ψ=0.3; (SP: Slack minimization + flow-in-current-based phase scaling, AM: Analytical model [2]+ flow-in current phase scaling, CP: Complete convex model $Scenario\,2$).

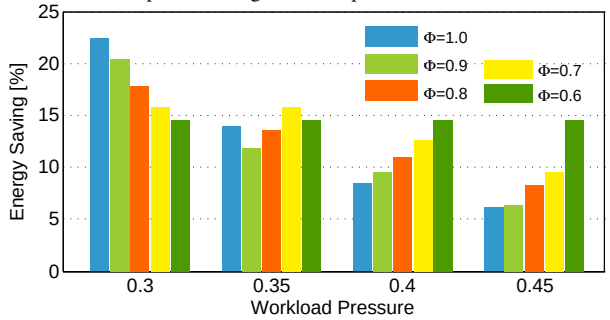


Fig. 14: Energy saving of the complete convex model over slack minimization at various workload pressures for four different CPU bounds.

the on-line stage. As you can see, the analytical model in [2] achieves better energy efficiency than slack minimization when Φ is close to 1.0, while it behaves in the opposite way when Φ decreases. However, in any case, the reduced convex model always outperforms these two baseline algorithms since our convex model can achieve a better trade-off between the dynamic energy, static energy and VR overhead.

Fig.11 shows the relative energy saving of the reduced convex model compared to slack minimization. We select the workload pressure range from 0.3 to 0.45 because during this range we can see clearly the energy saving varies with the workload pressure. When workload pressure is lower than 0.3, the energy saving of RC compared to SM does not change and remains the plateau. When workload pressure is higher than 0.5, the deadline constraint pushes higher frequency bound and shrinks the available DVFS space which leads to energy saving tending to zero. The histogram shows that determining the DVFS level using convex optimization can achieve up to a 17.6% energy saving compared to the conventional slack minimization method. This is due to the fact that our algorithm takes the static energy into consideration and uses precise task latency models. Hence, it can find the optimal point for total energy optimization. The trend of the histograms demonstrates two key properties of our convex algorithm: 1) The higher CPU-bound ratio gives a better energy saving when comparing RC to SM. This is because the high CPU-bound ratio provides an opportunity to greatly reduce the static energy by slightly increasing the operating frequency to greatly reduce execution time of tasks. 2) Lower workload pressure gives better energy saving. This is due to the fact that high workload pressure means that the core has to run very fast to catch the deadline. Lower workload pressure has lower frequency bound and gives

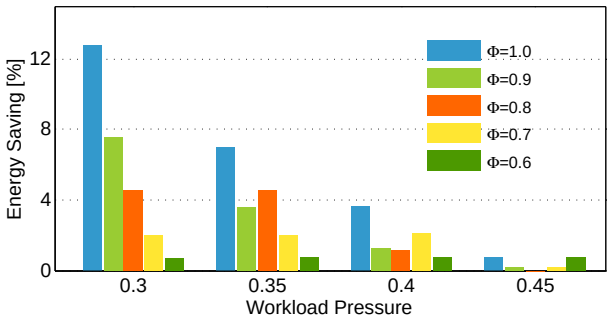


Fig. 15: Energy saving of the complete convex model over the slack minimization followed by flow-in-current-based phase scaling for a 4-core platform

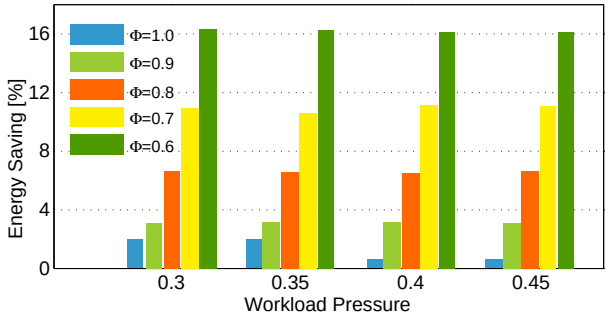


Fig. 16: Energy saving of the complete convex model over analytical model followed by flow-in-current-based phase scaling for a 4-core platform

higher flexibility of DVFS.

In Fig.12, the reduced convex model shows an up to 11.4% energy saving compared to the analytical model in [2]. In contrast to Fig.11, the reduced convex model achieves a better energy saving when workload characteristics Φ are lower. This is because the memory-bound characteristics of application are not considered in the analytical model [2], while our method accurately models the CPU-bound and memory-bound characteristics of different applications.

2) Complete convex model: In this subsection, we conduct experiments to compare the complete convex model with four baselines. These experiments prove the importance of optimizing DVFS level and phase scaling of VR in a comprehensive model, instead of determining DVFS and phase scaling in a decoupled manner. Fig.13 illustrates the breakdown of the average energy consumption of an interval of the whole platform at various workload characteristics Φ when Ψ=0.3 with overhead of the VR considered. Compared to Fig.10, we can see that the VR energy consumption has been greatly reduced through phase scaling. In our convex model, the VR overhead can also impact the decision of the DVFS setting.

Fig.14 illustrates the relative energy saving of the complete convex model compared to slack minimization. From the histogram, we can see that our algorithm can reduce the total energy by up to 22.4% compared to $Baseline1$. The trend of the histograms shows that at light workload pressure, a higher CPU-bound ratio is more beneficial for energy saving, while at heavy workload pressure, a lower CPU-bound ratio gives a better energy saving. This is because at a light load, the energy saving due to the convex model dominates, while at a heavy load, the energy saving due to phase scaling dominates.

Fig.15 and Fig.16 respectively illustrate the advantage of our proposed complete convex model compared to the decoupling

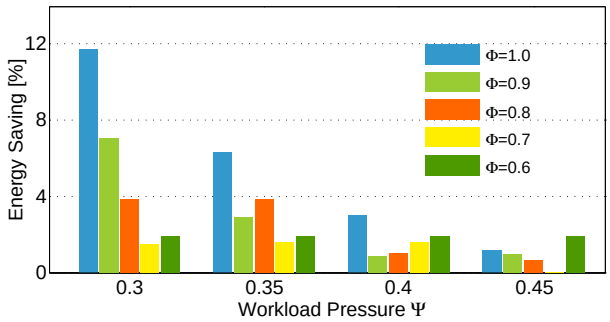


Fig. 17: Energy saving of the complete convex model over slack minimization followed by flow-in-current-based phase scaling for an 8-core platform.

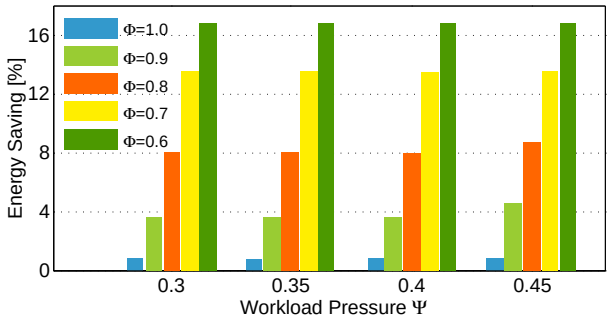


Fig. 18: Energy saving of the complete convex model over analytical model followed by flow-in-current-based phase scaling for an 8-core platform.

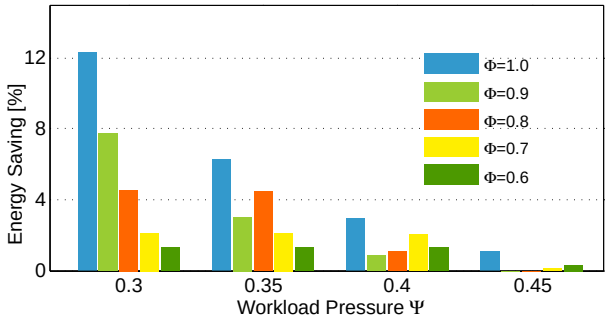


Fig. 19: Energy saving of the complete convex model over slack minimization followed by flow-in-current-based phase scaling for a 16-core platform.

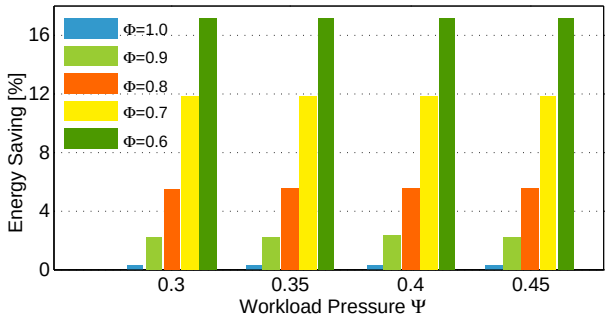


Fig. 20: Energy saving of the complete convex model over analytical model followed by flow-in-current-based phase scaling for a 16-core platform.

methods, $Baseline\,3$ and $Baseline\,4$. The histograms show that our proposed model can achieve an up to 12.7% energy saving compared to the decoupling method $Baseline\,3$, and an up to 16.3% energy saving compared to the decoupling method $Baseline\,4$. This is due to our comprehensive method incorporating phase scaling and DVFS into one comprehensive convex model with an accurate task characteristics model, which enlarges the exploration space. It gives a globally optimal selection of DVFS and phase scaling, as shown in Fig.5.

C. Platform Scalability

The previous analysis shows that our proposed two-stage comprehensive convex model achieves great energy savings for a 4-core platform. To validate the scalability and feasibility of our method, we implement the two-stage comprehensive convex model on larger platforms with eight cores and sixteen cores. Fig.17 and Fig.18 reveal the energy savings of our proposed comprehensive model compared to the decoupling method, $Baseline\,3$ and $Baseline\,4$, for an 8-core platform. Fig.19 and Fig.20 reveal the energy savings of our method for a 16-core platform. The results from Fig.17 to Fig.20 manifest an up to 12.34% and 17.11% energy saving compared to $Baseline\,3$ and $Baseline\,4$, respectively, and they display the same energy saving trend in Fig.15 and Fig.16. With the increasing number of on-chip cores, the time overhead at the on-line stage is $O(Num)$ where Num denotes the number of on-chip cores, which greatly guarantees the scalability of our two-stage method.

VI. Conclusion

This paper proposes a novel convex formulation to optimize the energy consumption of a multi-core platform. Based on workload characteristics, the VR overheads and static power of the processors, our proposed algorithm combines the DVFS setting and phase scaling of the off-chip VR into an integrated convex model. To achieve better scalability of our comprehensive convex model, we decompose our method into an off-line stage and an on-line stage. The experimental results show that our algorithm outperforms existing DVFS methods and achieves an up to 22.4% energy saving. Even compared to the conventional DVFS technique followed by the flow-in-current-based phase scaling method, our approach can achieve an up to 16.3% energy saving.

References

[1] Jonathan G Koomey. Worldwide electricity used in data centers. *Environmental research letters*, 3(3):034008, 2008.

[2] Patrick Cichowski, Jörg Keller, and Christoph Kessler. Energy-efficient mapping of task collections onto manycore processors. In *Proc. 5th Swedish Workshop on Mulitcore Computing (MCC 2012)*, volume 131, 2012.

[3] Hakan Aydin, Rami Melhem, Daniel Mossé, and Pedro Mejía-Alvarez. Power-aware scheduling for periodic real-time tasks. *IEEE Transactions on Computers*, 53(5):584–600, 2004.

[4] Jinfeng Liu, Pai H Chou, Nader Bagherzadeh, and Fadi Kurdahi. Power-aware scheduling under timing constraints for mission-critical embedded systems. In *Proceedings of the 38th annual Design Automation Conference*, pages 840–845. ACM, 2001.

[5] Amit Kumar Singh, Anup Das, and Akash Kumar. Energy optimization by exploiting execution slacks in streaming applications on multiprocessor systems. In *Proceedings of the 50th Annual Design Automation Conference*, page 115. ACM, 2013.

[6] Erwan Nogues, Maxime Pelcat, Daniel Menard, and Alexandre Mercat. Energy efficient scheduling of real time signal processing applications through combined dvfs and dpm. In *Parallel, Distributed, and Network-Based Processing (PDP), 2016 24th Euromicro International Conference on*, pages 622–626. IEEE, 2016.

[7] Ryan Cochran, Can Hankendi, Ayse Coskun, and Sherief Reda. Identifying the optimal energy-efficient operating points of parallel workloads. In *Computer-Aided Design (ICCAD), 2011 IEEE/ACM International Conference on*, pages 608–615. IEEE, 2011.

[8] Andrew Nelson, Orlando Moreira, et al. Reclaiming the energy of a schedule: models and algorithms. pages 117–124, 2011.

[9] Wonyoung Kim, Meeta S Gupta, Gu-Yeon Wei, and David Brooks. System level analysis of fast, per-core dvfs using on-chip switching regulators. In *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, pages 123–134. IEEE, 2008.

[10] Padmanabhan Pillai et al. Real-time dynamic voltage scaling for low-power embedded operating systems. In *SIGOPS OSR*. ACM, 2001.

[11] Rotem Efraim, Ran Ginosar, C Weiser, and Avi Mendelson. Energy aware race to halt: A down to earth approach for platform energy management. *IEEE Computer Architecture Letters*, 13(1):25–28, 2014.

[12] Kyungsu Kang, Giovanni De Micheli, Seunghan Lee, and Chong-Min Kyung. Temperature-aware runtime power management for chip-multiprocessors with 3-d stacked cache. In *Quality Electronic Design (ISQED), 2014 15th International Symposium on*, pages 163–170. IEEE, 2014.

[13] Haoran Li, Xuan Wang, Jiang Xu, et al. Energy-efficient power delivery system paradigms for many-core processors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(3):449–462, 2017.

[14] Hadi Asghari-Moghaddam, Ghasemi, et al. Vr-scale: runtime dynamic phase scaling of processor voltage regulators for improving power efficiency. In *Proceedings of the 53rd Annual Design Automation Conference*, page 151. ACM, 2016.

[15] A Costabeber, P Mattavelli, and S Saggini. Digital time-optimal phase shedding in multiphase buck converters. *IEEE Transactions on Power Electronics*, 25(9):2242–2247, 2010.

[16] Yongseok Choi, Naehyuck Chang, and Taewhan Kim. Dc–dc converter-aware power management for low-power embedded systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(8):1367–1381, 2007.

[17] Woojoo Lee, Yanzhi Wang, and Massoud Pedram. Vrcon: Dynamic reconfiguration of voltage regulators in a multicore platform. In *Proceedings of the conference on Design, Automation & Test in Europe*, page 365. European Design and Automation Association, 2014.

[18] Woojoo Lee, Yanzhi Wang, and Massoud Pedram. Optimizing a reconfigurable power distribution network in a multicore platform. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(7):1110–1123, 2015.

[19] Cvx: Matlab software for disciplined convex programming, http://cvxr.com/cvx/.

[20] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[21] Andrew Nelson et al. Power minimisation for real-time dataflow applications. In *DSD*. IEEE, 2011.

[22] Marco Spiga, Andrea Alimonda, Salvatore Carta, Francesco Aymerich, and Andrea Acquaviva. Exploiting memory-boundedness in energy-efficient hard real-time scheduling. In *Industrial Embedded Systems, 2006. IES'06. International Symposium on*, pages 1–10. IEEE, 2006.

[23] Xuan Wang, Jiang Xu, Zhe Wang, et al. An analytical study of power delivery systems for many-core processors using on-chip and off-chip voltage regulators. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 34(9):1401–1414, 2015.

[24] P Zumel, C Fernandez, A De Castro, and O Garcia. Efficiency improvement in multiphase converter by changing dynamically the number of phases. In *Power Electronics Specialists Conference*, pages 1–6, 2006.

[25] Edward A Burton, Gerhard Schrom, Fabrice Paillet, Jonathan Douglas, William J Lambert, Kaladhar Radhakrishnan, and Michael J Hill. Fivr - fully integrated voltage regulators on 4th generation intel® core™ socs. In *Applied Power Electronics Conference and Exposition (APEC), 2014 Twenty-Ninth Annual IEEE*, pages 432–439. IEEE, 2014.

[26] The gem5 simulator, http://gem5.org/Main_Page.

[27] Spec cpu® 2006, https://www.spec.org/cpu2006/.

[28] Jie Meng, Katsutoshi Kawakami, and Ayse K Coskun. Optimizing energy efficiency of 3-d multicore systems with stacked dram under power and thermal constraints. In *Proceedings of the 49th Annual Design Automation Conference*, pages 648–655. ACM, 2012.

[29] Jin Cui and Douglas L Maskell. A fast high-level event-driven thermal estimator for dynamic thermal aware scheduling. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(6):904–917, 2012.

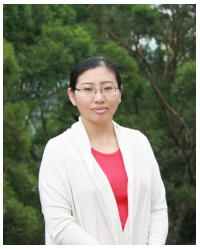
[30] Heather Hanson, Stephen W Keckler, et al. Thermal response to dvfs: Analysis with an intel pentium m. In *Proceedings of the 2007 international symposium on Low power electronics and design*, pages 219–224. ACM, 2007.

[31] Eep Sharma, Sarabjit Singh, and Meenakshi Sharma. M.: Performance analysis of load balancing algorithms. In *In: 38th World Academy of Science, Engineering and Technology*. Citeseer, 2008.

[32] Stephen Boyd, Seung-Jean Kim, Lieven Vandenberghe, and Arash Hassibi. A tutorial on geometric programming. *Optimization and engineering*, 8(1):67, 2007.

[33] Ggplab: A simple matlab toolbox for geometric programming, http://stanford.edu/~boyd/ggplab/.

Zuomin Zhu received his B.S degree in optical and electronic information from Huazhong University of Science and Technology (HUST), Wuhan, China in 2014. Since 2014, he has been a Ph.D student in the department of Electronic and Computer Engineering in Hong Kong University of Science and Technology (HKUST). His research interests include thermal management, low power design and energy minimization for multiprocessor systems.



Wei Zhang received her Ph.D. degree in Electrical Engineering from Princeton University with Wu Prize for research excellence. She joins Hong Kong University of Science and Technology in 2013 and establishes Reconfigurable Computing Systems Lab. She was an assistant professor in School of Computer Engineering at Nanyang Technological University, Singapore from 2010 to 2013. She is a co-investigator of Singapore-MIT Alliance for Research and Technology and works on low-power electronics. She is a collaborator of ASTAR-UIUC Advanced Digital Sciences Center and works on FPGA acceleration for multimedia applications. Prof. Zhang currently serves as the Associate Editor for IEEE Transactions on VLSI Systems and the Area Editor of Reconfigurable Computing for ACM Transactions on Embedded Computing Systems. She serves on many organization committees and technical program committees including CASES, ISLPED, ASP-DAC, FPT, FPL.

Vivek Chaturvedi (M09) is currently working as a Research Scientist in the School of computer science and engineering at Nanyang Technological University, Singapore. He received his M.S. from Syracuse University, NY in 2008 and PhD from Florida International University, Miami in 2013. He also worked in Sun Microsystems as a Student intern in the summer of 2007. Dr. Chaturvedi's Current research interest includes power and thermal optimization in multi/many core processors including both 2D and 3D architectures. He is also actively working on Hardware security and reliability. He is a reviewer and has served as TPC at several IEEE/ACM Journals and Conferences.

Amit Kumar Singh (M'09) received his B.Tech. degree in Electronics Engineering from Indian Institute of Technology (Indian School of Mines), Dhanbad, India, in 2006, and his Ph.D. degree from the School of Computer Engineering, Nanyang Technological University (NTU), Singapore, in 2013. He was with HCL Technologies, India for year and half before starting his PhD at NTU, Singapore, in 2008. He worked as a post-doctoral researcher at National University of Singapore (NUS) from 2012 to 2014, at University of York, UK from 2014 to 2016 and at University of Southampton, UK from 2016 to 2017. He is currently working as a Lecturer at University of Essex, UK. His current research interests include system level design-time and run-time optimizations of 2D and 3D multi-core systems with focus on performance, energy, temperature, and reliability.