AgileRegulator: A Hybrid Voltage Regulator Scheme Redeeming Dark Silicon for Power Efficiency in a Multicore Architecture

Guihai Yan[†], Yingmin Li, Yinhe Han[†], Xiaowei Li[†], Minyi Guo[‡], Xiaoyao Liang[‡]

[†]State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences

[‡]Department of Computer Science and Engineering, Shanghai Jiao Tong University, China

{yan_guihai, yinhes, lxw}@ict.ac.cn, yingmin.li@gmail.com, {guo-my, liang-xy}@cs.sjtu.edu.cn

Abstract

The widening gap between the fast-increasing transistor budget but slow-growing power delivery and system cooling capability calls for novel architectural solutions to boost energy efficiency. Leveraging the fact of surging "dark silicon" area, we propose a hybrid scheme to use both on-chip and off-chip voltage regulators, called "AgileRegulator", for a multicore system to explore both coarse-grain and fine-grain power phases. We present two complementary algorithms: Sensitivity-Aware Application Scheduling (SAAS) and Responsiveness-Aware Application Scheduling (RAAS) to maximally achieve the energy saving potential of the hybrid regulator scheme. Experimental results show that the hybrid scheme achieves performance-energy efficiency close to per-core DVFS, without imposing much design cost. Meanwhile, the silicon overhead of this scheme is well contained into the "dark silicon". Unlike other application specific schemes based on accelerators, the proposed scheme itself is a simple and universal solution for chip area and energy trade-offs.

1 Introduction

Semiconductor road map shows power delivery and temperature will be more serious constraints to chip performance than the chip area as technology evolves. At 20nm technology node, it is estimated that roughly 20% chip area will become dark silicon [1], which cannot be turned on all the time to fully contribute to the chip performance. This projection indicates that putting more and more logics on a chip and expect higher performance is unsustainable if we do not pay attention to the energy efficiency.

As a major power management technique, Dynamic Voltage and Frequency Scaling (DVFS) promises to greatly save energy with only marginal performance loss, or significantly improve the performance with a given power budget. DVFS technique requires Voltage Regulators (VR) to provide variable voltages corresponding to different power states for each Voltage-Frequency Islands (VFI) in the microprocessor[2]. To achieve the best power efficiency, the DVFS operations should be able to respond to the transition of power phases in a timely way [3][4].

The response time of regulators virtually determines how fast the DVFS can operate. There are two kinds of regulators: the conventional off-chip VR and the more recently invented on-chip VR. Off-chip VR doesn't occupy chip area except power pins and the on-chip power network. It has higher power delivery efficiency, but is not as responsive as on-chip VR. By contrast, on-chip VR has much shorter latency to switch to a new voltage setting, but it has relatively lower power delivery efficiency and it dictates significant amount of chip area.

Unlike off-chip VRs which can only exploit the coarse-grain power phases, on-chip VRs are ideal to exploit fine-grain power phases. However, completely resorting to on-chip VRs are impractical due to two reasons: 1) Area overhead: It is estimated that to deliver 1 watt of power we need to pay $2mm^2$ chip area for the on-chip VRs [5]. Even though the regulator area can be partially amortized by the surge of dark silicon, it is still an overkill to pack per-core on-chip VRs which will occupy similar area as the cores on the chip. 2) Power delivery efficiency (DC-DC converter efficiency): Due to the physical constraints, the power transfer efficiency of the state-of-art on-chip VRs can hardly reach 80% [5] as compared to the 90% efficiency of off-chip VRs [6]. Therefore in some cases the energy benefit gained by using on-chip VRs can be totally offset by the loss of power delivery efficiency.

In this paper, we propose "AgileRegulator", a scheme leveraging dark silicon for a small number of on-chip VRs, rather than specialized computing logics or accelerators[7]. We might refer to the silicon taken as "Power Silicon" since the primary goal is for flexible power delivery and energy conservation. We demonstrate the hybrid scheme with mixed on-chip and off-chip VRs in a multicore architecture running multi-program applications. Since not all applications can benefit from fast DVFS, we only deploy a few on-chip VRs and other cores are still powered up by off-chip VRs. By combining the advantage of both on-chip and off-chip VRs through our smart DVFS and application scheduling algorithm, our scheme provides energy efficiency close to ideal per-core DVFS, with an increased area budget well fitted into the forecasted dark silicon.

[†]To whom correspondence should be addressed. The work was supported in part by National Basic Research Program of China (973) under grant No. 2011CB302503, in part by National Natural Science Foundation of China (NSFC) under grant No.(61100016, 61076037, 60921002).

978-1-4673-0826-7/12/$26.00 ©2011 IEEE

In particular, this paper makes the following contributions:

1. We found that different cores within a single VFI often exhibit unbalanced program activities. Since each VFI can only have one off-chip regulator and that regulator has to accommodate the program whose performance is most sensitive to power settings, such core-to-core behavior variation will result in unfairness and the degradation of the overall energy efficiency in that specific VFI. In this paper we propose a novel application scheduling algorithm, "Sensitivity-Aware Application Scheduling (SAAS)", to mitigate the unfairness by dynamically grouping the applications with similar energy behavior into the same VFI, under the constraint of other system limitations such as memory bandwidth. This scheme greatly improves the overall energy efficiency for a system powered up by off-chip VRs.

2. Beyond SAAS, we propose to use the dark silicon area for a very limited number of on-chip VRs to maximally explore the fine-grain power phases for the most benefited applications. We show a smart algorithm "Responsiveness-Aware Application Scheduling (RAAS)" to identify and schedule such applications to use on-chip VRs. We also find that an on-chip VR is only helpful for a limited number of applications after considering the lower power delivery efficiency. Given the hefty area overhead, we advise to adopt the on-chip VRs judiciously and selectively.

3. We demonstrate the importance of building a synergy between SAAS and RAAS by combining their advantages and limiting their overhead. In some cases, SAAS may cause memory congestion in a VFI. When the fairness optimization conflicts with the memory bandwidth, we can leverage RAAS to balance the bandwidth utilization, while still maintain the level of fairness required by the SAAS. We evaluate our scheme on a 16-core, a 36-core, and a 64-core systems with multi-program workloads. Experimental results show that such a hybrid scheme can achieve an energy efficiency close to the ideal per-core DVFS case.

The rest of this paper is organized as follows: Section 2 gives the background and clarifies key motivations. Section 3 introduces the principle and framework of SAAS, RAAS, and the synergy between them. Section 4 introduces the key implementation heuristic. Section 5 describes the experiment setup and the workloads we used for this experiment. We show results in Section 6 and introduce related work in Section 7. Finally we summarize the paper in Section 8.

2 Background

2.1 Application Sensitivity to DVFS

Unlike most prior researches focusing on fairness issues introduced by shared resources in a multicore processor[8][9][10][11][12], we find that DVFS operations can also degrade system fairness and therefore result in energy inefficiency. The sensitivity of application execution latency (or time) to power states varies widely across applications. For example, Figure 1 compares

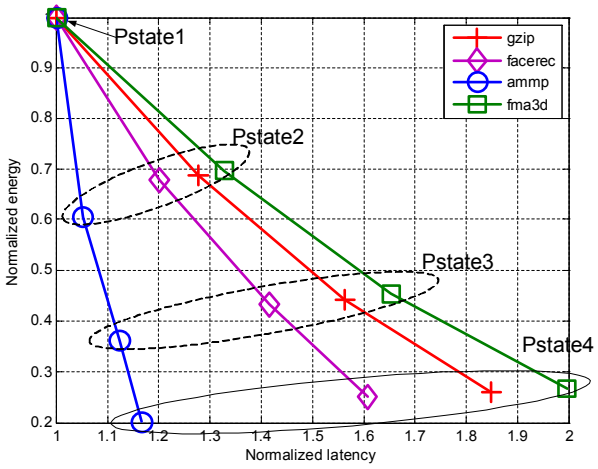

Figure 1. Energy vs. Latency within an execution epoch

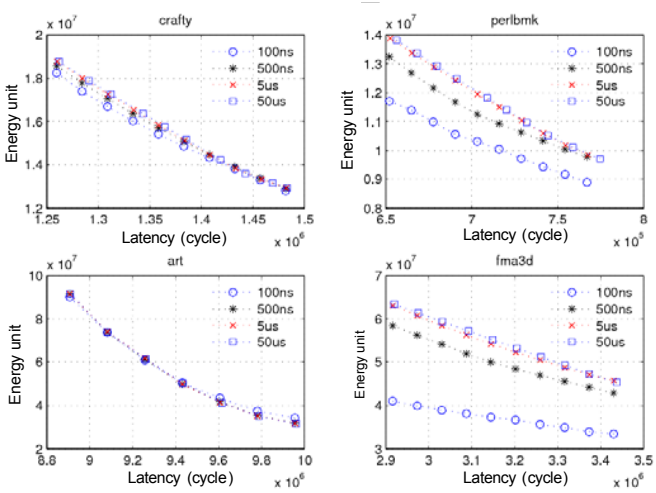

Figure 2. Energy vs. Latency pareto frontier under different DVFS intervals

the trade-off between execution latency and power states for different applications(`gzip`, `facerec`, `ammp`, `fma3d`). We have four power states with $Pstate1$ having the highest operating frequency and voltage. The execution latencies corresponding to all other power states ($Pstate2 - Pstate4$) are normalized to that of $Pstate1$. We find that the execution latency always varies with power states but to a different degree for different applications. The latency increase can vary from 17% for `ammp` to 100% for `fma3d`. Clearly, if the two applications reside in the same VFI powered up by the same voltage regulator, to meet the minimal system performance requirement, the power state has to be tuned based on `fma3d`, putting `ammp` into a sub-optimal operating mode. We call this as *DVFS-Induced Unfairness* in this paper.

2.2 Voltage Regulators

The key hardware support for DVFS operations relies on efficient voltage regulators, either off-chip or on-chip

[13][5]. The conventional regulator is either plugged onto the PCB board via a connector, or permanently embedded onto the board [14], referred as off-chip VR. The output voltage can be tuned according to processor operating mode/power state and typically ranges from 0.5V to 1.5V. A specific voltage is selected by programming a voltage identification register. The switching latency across power states is largely determined by the voltage tuning delay of the off-chip VRs. The latency is usually at the milliseconds range, which results in a relative slow DVFS response in practice. Each off-chip VR takes a large piece of board area and some amount of power supply pins of the chip. Because modern ICs are usually pin limited and the PCB boards are expensive, it is practical to pack only a few off-chip VRs onto the board.

By contrast, on-chip VR was invented for much faster response [13][5] with nano-seconds DVFS operations. However, this benefit does not come for free. Firstly, packing on-chip VRs into the silicon dictates huge area overhead. The area overhead is proportional to the required power delivery capacity. Empirically, delivering 1 watt power corresponds to $2mm^2$ chip area (the specific relationship varies with different technologies and types of regulators). That means up to $60mm^2$ silicon area is required to power up a 30 watt processor core in an Itanium® like processor where each core only takes up $75mm^2$. Clearly, the area of the on-chip VR is close to that of the core it powers. Since most of the regulator area is devoted to on-chip capacitors and inductors (metal), the area overhead is not likely to scale fast with advanced technology. However, the growth of dark silicon area provides us a perfect opportunity to use dark silicon as on-chip VRs for better voltage tuning. On the other hand, even with the dark silicon, the hefty area cost implies that on-chip VRs must be used very selectively to power up the most benefited parts of the microprocessor. Furthermore, the power delivery efficiency of on-chip VR is far from perfect. Recently, Kim et al. demonstrated a nanosecond-scale on-chip regulator with a peak efficiency staying at 77% [5]. On the other hand, the efficiency of off-chip VR is usually better than 90% [6].

2.3 Application Responsiveness to Fast DVFS

Not all applications require fast DVFS to achieve optimal energy efficiency. The application's preference to DVFS tuning latency is clearly demonstrated through analyzing the pareto optima of energy-latency tradeoffs. Figure 2 shows the energy-latency curve of four applications (`crafty`, `perlbmk`, `art`, `fma3d`) with different DVFS tuning latencies. The results show that the benefit of fast DVFS heavily depends on the characteristic of applications: for `crafty` and `art`, the fast DVFS brings little energy benefit. On the contrary, we can greatly improve the energy efficiency for `perlbmk` and `fma3d` with fast DVFS. In this paper, we call applications sensitive to DVFS latency as "sponge" applications, as they are like sponges releasing more water with more pressure (finer DVFS interval).

3 The Framework of "AgileRegulator"

"AgileRegulator" aims to build a synergy between two schemes, SAAS and RAAS, to combine the benefits of both off-chip and on-chip VRs. We assume that the target microprocessor consists of multiple clusters and each cluster consists of multiple cores. The hardware architecture is shown in Figure 3. Each core has a private L1 cache. The last-level cache can be logically shared by all of the cores within a cluster. Each cluster holds its own memory controller for the main memory access.

In our architecture, each cluster forms a VFI and is powered by an off-chip VR. Our scheme includes both on-chip and off-chip regulators. Given the hardware overhead of on-chip VR and the dark silicon area budget (around 20% chip area), we assume one on-chip VR is attached to each cluster. Only one core in a cluster can be opportunistically powered by the on-chip VR, depending on the characteristic of the application running on the core. When all the on-chip VRs are shut down, the target architecture degenerates to an off-chip VR only system.

In this section, we first describe the energy optimization problem in a multi-VFI processor and then explain how "AgileRegulator" can help boost the energy efficiency.

3.1 The Energy Optimization Problem for Multi-VFI Processors

In this paper, a *workload* is composed of multiple applications running on multiple cores. The applications are evenly divided into K consecutive epoches. At the beginning of each epoch, the applications will be rescheduled and re-assigned to VFIs. We assume a microprocessor has N VFIs and each VFI has M cores. $P(i,j)$ denotes the current power state of the ith VFI for the jth epoch. Given that each VFI hosts multiple applications, $D(i,j)$ is defined as the maximum execution latency across all the applications in the ith VFI for the jth epoch due to DVFS. $E(i,j)$ denotes the total energy consumption of all the applications in the ith VFI for the jth epoch. Equation (1) and (2) formulate these definitions.

$$D(i,j) = \max_{m \in \text{the } i\text{th VFI}} \{latency(m,j)\}, \tag{1}$$

$$E(i,j) = \sum_{m \in \text{the } i\text{th VFI}} \{energy(m,j)\}. \tag{2}$$

The energy optimization problem is to determine $P(i,j)$ for all VFIs over all epochs so that the total energy E_{total} is minimized, subject to the constraint that the maximum latency $D_{max} = \max\{\sum_{j=1}^{K} D(i,j)\}$ should not exceed the allowed maximum system latency ($Max_{latency}$). In summary, we have the following optimization problem:

$$\text{Minimize } E_{total} = \sum_{i=1}^{N}\sum_{j=1}^{K} E(i,j), \tag{3}$$

$$\text{Subject to } D_{max} \leqslant Max_{latency}.$$

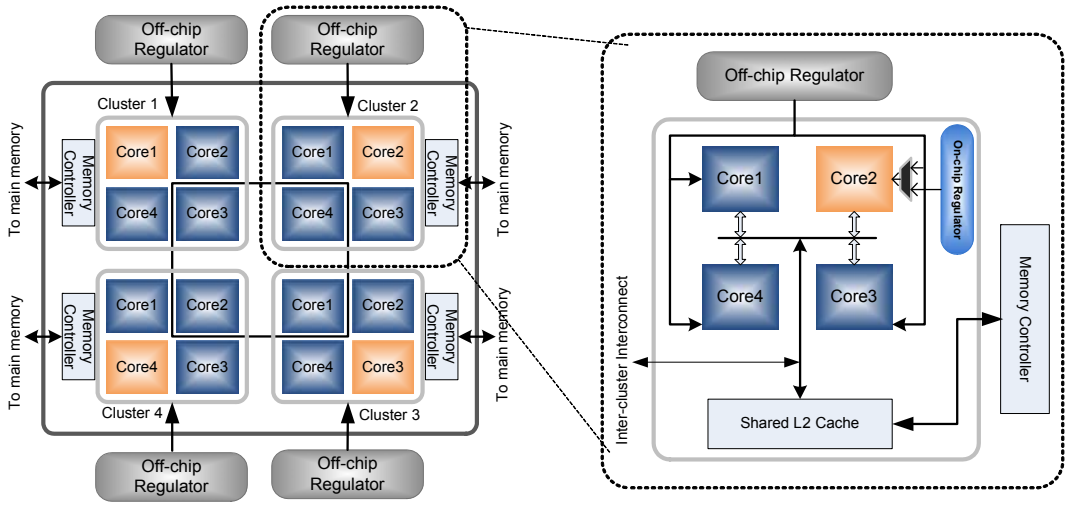

Figure 3. Top view of "AgileRegulator" architecture

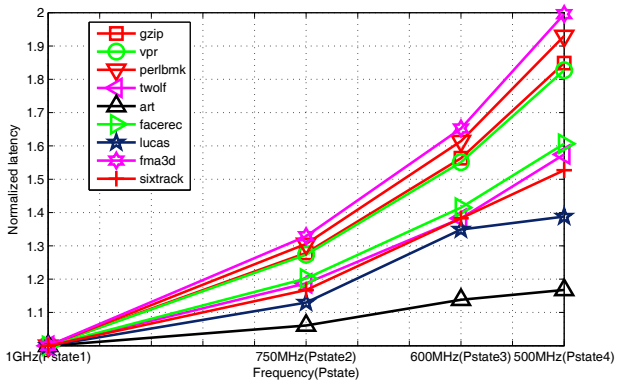

Figure 4. The S-factor for a $4-Pstate$ system

For simplicity, we use a relative $Max_{latency}$ definition, $D_{allowable}$, in the rest of this paper, which is defined as the ratio of actual execution time with DVFS to the running time at full-speed with the highest frequency/voltage setting. For example, $D_{allowable} = 105\%$ (or 1.05X) means the the system can accept 5% extra execution latency (or performance loss) after enabling DVFS for energy savings compared with the full-speed case.

3.2 SAAS: Sensitivity-Aware Application Scheduling

We define the power state with the highest voltage-frequency combination as $Pstate1$ and use it as the reference point. We then define the sensitivity of the execution latency associated with other power states, also called S-factor (S), for each epoch in any application:

$$S(Pstate1 \rightarrow PstateX) = \frac{Latency(PstateX) - Latency(Pstate1)}{Frequency(Pstate1) - Frequency(PstateX)}. \quad (4)$$

We use S-factor to characterize the fairness in a VFI. Figure 4 shows the execution latency with four power states for a bunch of SPEC benchmarks during an epoch of 10M instructions, where the latencies are normalized to the $Pstate1$. We observe that the latency change with power states varies greatly from benchmark to benchmark. For example, the latency of `art` only increases 17% from $Pstate1$ to $Pstate4$ while it varies 100% for `fma3d`. Such a big performance variation will seriously hurt the intra-VFI fairness if applications with big difference in S-factor are grouped into the same VFI. For a fair system with the optimal energy efficiency, the necessary condition is that there is as little intra-VFI imbalance on S-factor as possible. Otherwise, the selection of power state has to be bottlenecked by the application with the biggest S-factor, thereby wasting energy on other applications. This observation motivates the SAAS approach that is devoted to mitigate the intra-VFI imbalance based on the S-factors.

Figure 5 illustrates an example on how SAAS reduces the S-factor imbalance and in turn saves energy. In this example, we assume two VFIs, each hosting two applications during the kth epoch. The applications with dark color(`App2` and `App3`) have big S-factors, while applications with light color (`App1` and `App4`) have small S-factors. In the original case, the required frequency to meet $D_{allowable}$ is 1.6GHz for `App2` and `App3`. Without SAAS, the working frequency for both VFIs has to be set to 1.6GHz and the off-chip VRs have to select a high voltage (1.2V) to meet this frequency. If we take a further look at VFI1, this setting is inefficient because `App1` in that cluster is wasting power with the 1.6GHz/1.2V setting since its performance is insensitive to a lower power state. It can well meet the latency requirement with a much lower voltage and frequency setting. The same situation also applies to `App4` in VFI2. This is what we called S-factor imbalance in a VFI. Our proposed SAAS scheme can identify this situation and exchange the allocation of `App2` and `App4` to balance the intra-VFI S-factors. After the re-grouping process, the working frequency of VFI1 hosting `App1` and `App4` can be reduced to