ElasticCore: Enabling Dynamic Heterogeneity with Joint Core and Voltage/Frequency Scaling

Mohammad Khavari Tavana*, Mohammad Hossein Hajkazemi*, Divya Pathak†,
Ioannis Savidis†, Houman Homayoun*
*Department of Electrical and Computer Engineering, George Mason University, Fairfax, Virginia, USA
†Department of Electrical and Computer Engineering, Drexel University, Philadelphia, Pennsylvania, USA
Email: *{mkhavari, mhajkaze, hhomayou}@gmu.edu †{divya.pathak, ioannis.savidis }@drexel.edu

ABSTRACT

Heterogeneous architectures have emerged as a promising solution to enhance energy-efficiency by allowing each application to run on a core that matches resource needs more closely than a one-size-fits-all core. In this paper, an ElasticCore platform is described where core resources along with the operating voltage and frequency settings are scaled to match the application behavior at run-time. Furthermore, a linear regression model for power and performance prediction is used to guide the scaling of the core size and the operating voltage and frequency to maximize efficiency. Circuit considerations that further optimize the power efficiency of ElasticCore are also considered. Specifically, the efficiency of both off-chip and on-chip voltage regulators is analyzed for the heterogeneous architecture where the required load current changes dynamically at run-time. A distributed on-chip voltage regulator topology is proposed to accommodate the heterogeneous nature of the ElasticCore. The results indicate that ElasticCore on average achieves close to a 96% efficiency as compared to an architecture implementing the Oracle predictor where the application behavior is perfectly matched at run-time. Moreover, the proposed architecture is 30% more energy-efficient as compared to the BigLitte architecture.

Categories and Subject Descriptors

C.1.3 [**Processor Architectures**]: Other Architecture Styles – *Adaptable architectures.*

Keywords

Energy efficiency, voltage regulator, DVFS, regression model.

1. INTRODUCTION

Moore's Law scaling continues to provide increased transistor counts available to each processor generation. In recent generations, these transistors are primarily devoted to increased core counts. However, the increase in the number of transistors is not accompanied by an increased power budget. Therefore, the 'dark silicon era' [1] has arrived, where more transistors are available on an integrated circuit than are possibly turned on at any given time. Dark silicon requires a reevaluation of the tradeoffs between area (transistor count) and power. Transistors become all but free, while power is critically important. As a consequence, both academic and industrial research is focused on the development of new software and hardware paradigms and novel architectures to sustain Moore's law in the presence of dark silicon. One approach which is effective to render energy efficient computing is using heterogeneous processing cores. Heterogeneous multicore systems are comprised of cores with varying architectural features as well as power and performance characteristics. Each core is specialized and therefore efficient for executing one type of application [4], or an aggregation of different generations of cores with the same instruction set architecture is implemented [2][3].

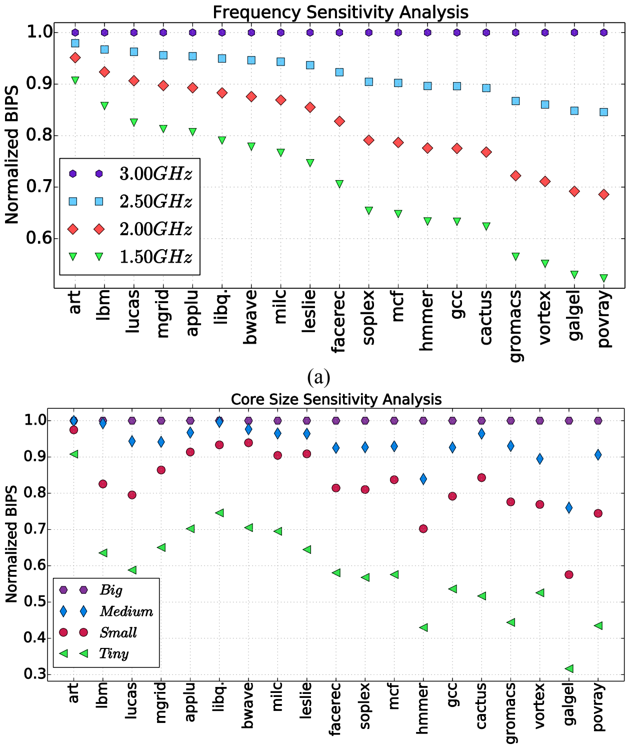
Multi-core systems in which the cores are diverse but the characteristics of each core are fixed at design time are described as statically heterogeneous. Examples of commercial products include the AMD Fusion APUs [4] in the high performance domain and the TI OMAP 5 [5] in the embedded system domain. The ARM big.LITTLE architecture [3], bridges the gap between these two domains by integrating high performance cores with smaller energy efficient cores.

A static heterogeneous architecture [7, 9] enables efficient thread-to-core mapping and permits a change in the mapping across phases of execution, through thread migration. Prior research has shown that the potential benefit of a static heterogeneous architecture is greater with fine-grained thread migration than with coarse-grain migration [6]. In [7] an Intel Xeon is integrated with an Atom processor. Code instrumentation is used to schedule different phases of the application on each processor. However, the separate core and memory subsystems in these architectures incur power and performance overheads for application migration, which makes dynamic mapping ineffective for fine-grained migration [6].

The Composite core proposed in [6] integrates heterogeneous cores to overcome the high migration overhead. This is realized by pairing an in-order and an out-of-order μEngine in the same core. The controller decides at run-time which phase of the application is more suited to execute on the out-of-order or the in-order portion of the core to reduce power with minor effect on performance.

All of the discussed architectures achieve heterogeneity through static or fixed cores, however, a particular heterogeneous design that is fixed may not match the needs of an arbitrary workload. In addition, the required resources for an application vary during different phases of the execution, but the core remains fixed in size and micro-architectural features. Dynamic heterogeneity is exploited in finer granularity in [11], where 3-D stacking is proposed as a means to share some of the structures that cause a performance bottleneck such as the register file or load store queue among the different stacked cores. 3-D stacking, however, suffers from the high cost of integrating dies vertically and elevated temperature of operation. Core Fusion [8] and TFlex [9] allow for the doubling or quadrupling of the size of the core by aggregating cores together to improve performance whenever the instruction level parallelism (ILP) is high. Major challenges with these architectures include data migration with changing core size and the additional pipeline latency imposed by the fused cores. Alternatively, large out-of-order cores are used in MorphCore [10] for workloads with high ILP while the

DAC '15, June 07 - 11, 2015, San Francisco, CA, USA
Copyright 2015 ACM 978-1-4503-3520-1/15/06$15.00
http://dx.doi.org/10.1145/2744769.2744833



(a)

(b)

Fig. 1: Normalized performance of different benchmarks with respect to (a) frequency, and (b) core size.

architecture is reconfigurable at run-time to many in-order cores for workloads with high thread-level parallelism (TLP). While MorphCore is an architecture that switches between out-of-order and wide in-order operation, the focus of this work is to dynamically adapt the out-of-order core window size (capacity), execution bandwidth (bandwidth), and frequency to the workload requirements at run-time to improve energy efficiency.

Based on the application requirements, the ElasticCore reconfigures resources dynamically at run-time. The primary enabling components that realize fine grain adaptation with high efficiency are the voltage regulators and power delivery system of the proposed platform. A suitable power delivery system based on available state of the art on-chip voltage regulator topologies is designed and the effect of on-chip and off-chip voltage regulators on the total energy efficiency of the system is evaluated. The proposed platform consists of dynamically scaling the bandwidth as well as the capacity. Different trade-offs in power and performance are achieved by reducing or expanding the size of various resources to construct cores of different sizes such as big, medium, little, or tiny. The trade-off for each core size is highly affected by the operating voltage and frequency, therefore, a joint core and dynamic voltage/frequency scaling (DVFS) optimization is required. This is particularly important for applications where performance is sensitive to frequency. Joint DVFS and core scaling are employed during different phases of the execution of an application. In particular, it is shown that increasing bandwidth and capacity reduces the sensitivity of an application to frequency. For example, while a core configuration with a higher bandwidth and capacity dissipates more power, there is a significant opportunity to use DVFS for reducing power with a minor performance loss. Energy efficiency optimization is examined based on core and voltage/frequency scaling. The main contributions of this work are as follows:

- The performance and power sensitivity of various standard benchmarks for different core sizes (e.g. big, medium, little, and tiny) and frequencies are investigated. The performance of standard benchmarks in terms of execution time is shown to be a linear function of frequency while no clear trend is observed as a function of core size. However, a linear regression model for power and performance prediction is shown to be as effective as an oracle predictor when used to guide the scaling of the core size and operating voltage/frequency to maximize efficiency. Moreover, the impact of core size on the frequency sensitivity of applications is explored. It is shown that modifying the core size changes the sensitivity of an application to the frequency.
- ElasticCore is proposed, which is a platform capable of scaling resources, i.e., bandwidth, capacity, voltage, and frequency, based on the application performance requirements at run-time while improving energy efficiency. The ElasticCore platform eliminates the need of migration that is required in BigLittle like architectures (i.e., two separate cores with diverse power and performance characteristics) and outperforms the BigLittle architecture by enabling fine-grained resource scaling.
- Circuit design challenges to realize fast core and voltage/ frequency scaling at run-time are examined. The ElasticCore is provisioned with multiple voltage regulators to not only scale the voltage quickly but also sustain high power conversion efficiency (PCE) over varying load requirements. The effect of both on-chip and off-chip voltage regulators on the energy efficiency of the system is evaluated. Based on these results, a two-tiered power delivery scheme is proposed.

2. MOTIVATION

In this section, the performance sensitivity of various standard benchmarks to frequency and core size is analyzed. The idea of dynamically scaling the core architecture with respect to the application behavior is described in this section.

2.1 Application Sensitivity to Frequency

A wide range of applications with diverse behavior from SPEC2000 and SPEC2006 benchmarks are selected and simulated on various core configurations and across various frequencies. The normalized billion instructions per second (BIPS), used as a performance metric, for the studied benchmarks when the operating frequency is swept from 1.5 GHz to 3GHz with a step of 500MHz is shown in Fig. 1 (a). Application sensitivity to the frequency increases when moving from the left to the right in Fig. 1 (a). For instance, the performance loss of *art* is less than 10% when reducing the frequency by half (far left). However, close to a 50% performance loss occurs when reducing the frequency by half when executing the *povray* benchmark (far right). Reducing the frequency slows the cpu-bound applications noticeably, while no significant impact is observed for memory-bound applications.

2.2 Application Sensitivity to Core Size

Applications have different performance behavior as a function of the size of the core resources. In this section, the performance sensitivity of the applications to the size of the core (capacity and bandwidth) is examined. For capacity, the *load store queue* (LSQ), *integer/floating point register file* (RF), *reorder buffer* (ROB), and *integer/floating point instruction queue* (IQ) are considered, as these four units define the instruction window size of the processor. For bandwidth, the *fetch, decode, issue, and commit* width are considered. Four balanced cores with different capacity and bandwidth are modeled to reduce the design exploration space. The detailed parameters for each of these configurations are listed in Table 1.

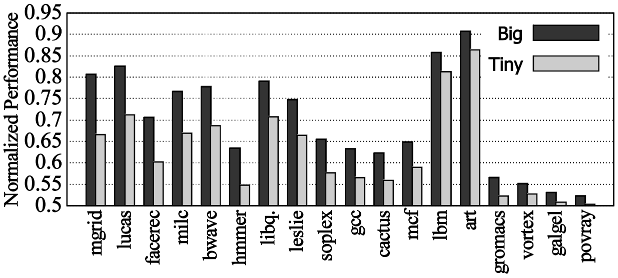


Fig. 2: Impact of core size on frequency sensitivity.

The normalized billion instruction per second (BIPS) for each application is shown in Fig. 1 (b). The BIPS for four configurations with different core sizes is reported. It is important to note that the sensitivity to core size is not uniform across applications. For instance, *lbm* shows less than 0.1% performance loss when comparing big to medium, whereas comparing big to small and tiny reveals a performance loss of 17% and 36%, respectively.

2.3 Impact of Core Size on Frequency Sensitivity

The simulations indicate that the application performance sensitivity to frequency changes noticeably, depending on the core size. The impact of a change in the bandwidth on the frequency sensitivity of the applications is shown in Fig. 2. Consider the case when the application is executed at half the maximum frequency. The normalized performance as compared to the case when the application is executed at the maximum frequency is plotted in Fig. 2. For a number of benchmarks when the core size varies from big to tiny, the impact on the application sensitivity to frequency is significant, however, for other benchmarks, less of an impact is observed. For instance, core size has the largest impact on frequency sensitivity for *mgrid* (the left corner) and the least impact on the *povray* benchmark.

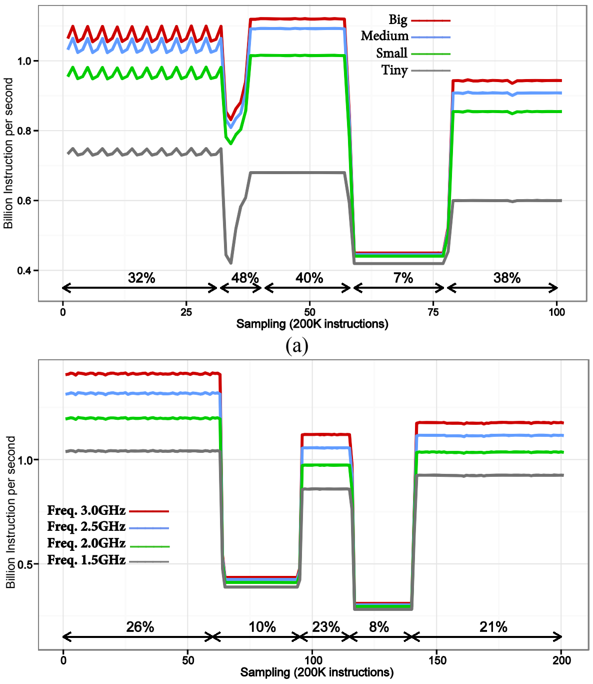
2.4 Application Sensitivity at Run-Time

As shown in Fig. 1, applications have different levels of sensitivity to frequency and core size. Over-provisioning resources to assure a core functions for the most demanding applications leads to inefficiency in energy usage. The many categories of applications that execute on different cores with various sizes highlights the need for a heterogeneous architecture to increase the energy efficiency. Heterogeneous designs that reduce over-provisioning without necessarily sacrificing performance enable significant gains in performance per Watt. However, even within an application, the behavior such as sensitivity to frequency and core size changes at run-time. The BIPS trace for the *applu* benchmark is plotted for various core size and frequency values in Fig. 3(a) and 3(b), respectively. Even for a range of 20 million instructions, the variation in performance changes significantly from 7% to 48% when the core size is changed from big to tiny. A similar dynamic behavior is also observed when changing the frequency. As shown in Fig. 3(b), the different phases of an application experience a variation in the performance when the frequency is reduced to half. The main goal of ElasticCore is to detect the upcoming phase of an application at run-time, expand or contract resources (capacity and bandwidth), and set the frequency/voltage to match the core with the currently executing workload.

3. ELASTIC CORE ARCHITECTURE

3.1 Microarchitecture Feasibility

Prior research have explored the micro-architecture and design modifications required to allow for resource adaptation in the pipeline, where components such as a register file, instruction queue, reorder buffer, load store queue, fetch width, issue width and commit width are dynamically resized [11], [17], [18], [19], [20]. The microarchitecture overhead as well as power and area overhead to implement resource resizing is minimal [17], [18], [19]. In this work, assuming a conservative implementation, the overhead to switch from one core configuration to another is considered as 1K cycles, which is much longer than the overhead to flush the pipeline (note that the cache subsystem remains unchanged when modifying the core configuration).



(a)

(b)

Fig. 3: Sensitivity of *applu* benchmark to (a) core size and (b) frequency at run-time over 20 and 40 million instructions, respectively. The maximum performance loss in percentage is included for each phase.

Redesigning core components such as the register files (RFs), reorder buffer (ROB), instruction queue (IQ) and load/store queue (LSQ) to realize adaptive resource resizing was studied in [18]. The RF and ROB are SRAM structures. Due to the circular FIFO nature of the ROB, two pointers are required to dynamically adjust the size. A modular architecture to dynamically resize both the ROB and RF is proposed in [17]. Structures such as the LSQ and IQ are designed as CAM+SRAM based architectures. These components hold the instructions until ready to issue. It is possible that two or more partitions of the IQ/LSQ are combined to form a larger partition without impacting cycle time [11]. Other prior research considers bandwidth adaptation based on the instruction level parallelism in

Table 1: Specifications for the ElasticCore platform

Core voltage/frequency	
Frequency (GHz)	1.5, 2.0, 2.5, 3.0
Vdd (V)	0.750, 0.914, 1.132, 1.350
Bandwidth *(tiny, small, medium, big)*	
Decode/Issue/Commit width	1, 2, 3 , 4
Fetch width	2, 4, 6, 8
Capacity *(tiny, small, medium, big)*	
IIQ/FIQ/LSQ size	16, 32, 48, 64
IREG/FREG size	24, 48, 64, 96
ROB size	32, 64, 96, 128
Memory subsystem	
L1 cache	32KB, 4-way
L2 cache	512KB, 8-way

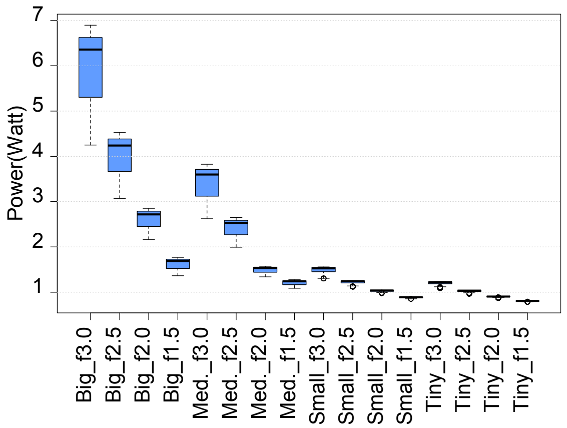


Fig. 4: Power dissipation of *gcc* benchmark at different voltage/ frequency pairs and core size.

the application [19], [20]. The ElasticCore exploits previous research to implement fine-grained resource adaptation at low cost. Note that ElasticCore is designed at maximum capacity and bandwidth (in this case, a pipeline width of 4 and window size of 128) to account for the worst-case scenario where all resources are needed. The ElasticCore specifications are listed in Table 1. The core includes four voltage/frequency settings. Note that in ElasticCore the memory subsystem remains unchanged across, however, the core size (capacity and bandwidth) is resized to four different configurations.

3.2 Efficiency Metric

The efficiency is measured in *BIPS^3/W* which is a metric for joint power and performance optimization for high performance processors [7] [11]. The *BIPS^3/W* is the inverse of the *energy×delay^2 product* (ED^{2}P). An estimation of the performance and power is required to dynamically adapt the core and the operating frequency to maximize *BIPS^3/W*. The average and variance of the power dissipation for the *gcc* benchmark for various frequency/voltage pairs and for each core size is shown in Fig. 4. For a phase of an application that is more sensitive to core size (rather than frequency), it is better to execute that phase with more bandwidth (larger core) at a lower frequency to optimize the *BIPS^3/W*. The goal is to accurately estimate the power and performance and find the best solution that maximizes *BIPS^3/W*.

3.3 Power and Performance Estimation

Recent work have proposed regression modeling to estimate the power [7] and performance [6, 23] of a processor at run-time. In this work, a regression model is used to predict the performance for various configurations. Correlation analysis is used to determine the best combination of performance counters which results in bounded accuracy. Even though the use of a non-linear regression or neural-network model provides a more accurate representation of the performance and power behavior of an application, for ease of implementation, a simple though effective linear regression model is used. The parameters for estimating power and performance are listed in Table 2.

Table 2: Performance data used for the regression model

Category	**Hardware performance counter**
Cache subsystem	L1 data access L1 data miss L2 cache access L2 cache miss D-TLB miss
Instructions	Integer instruction issue Integer floating point issue Load/Store instruction issue
Branch	Branch misprediction

The linear regression model (*LRM*) used for power and performance estimation is

$$LRM = \left(\beta_0 + \sum_{i=1}^{i=9} \beta_i Pc_i\right) + \beta_{10}CW + \beta_{11}f \tag{1}$$

where β_0 is the intercept, β_i are coefficients of the regression model, and Pc_i are target performance counters. The capacity and bandwidth of the core are given by *CW*, and *f* represents the frequency. The coefficients are calculated based on the first 200 million executed instructions of each benchmark. The power and performance are estimated at the end of each sampling interval (i.e., every 200k instructions). The first part of the *LRM* is calculated at the end of each interval and then summed for the different *CW* and *f* values listed in Table 1, which leads to a total of 25 addition and multiplication operations and 15 comparison operations to determine the best *CW*. Based on such a low computational cost, the overhead to calculate the *LRM* is fairly low.

3.4 Power Delivery to the ElasticCore

The current demand of the ElasticCore fluctuates significantly at run-time due to dynamic core scaling. The power delivery topology therefore plays a crucial role in the energy efficiency of the ElasticCore. A poor power delivery network design leads to dramatic power loss at the front-end. The ElasticCore, not only requires a power delivery system with high power conversion efficiency (PCE) but also a fine-grained, low-overhead voltage control for scaling the core and the voltage at run-time. The ElasticCore is therefore designed with on-chip voltage regulators (OCVR) which offer fast DVFS [12], a reduced PCB footprint, and a reduction in parasitic losses.

The fully integrated voltage regulator (FIVR) in the 4*th* generation Intel core microprocessors (Haswell) [13], [14] offers up to three times higher peak power as well as higher performance for a given power level as compared with previous generations of Intel core microprocessors. The power delivery system of the ElasticCore is modeled using a configuration similar to the Intel Haswell processor. The block diagram of the proposed two-tiered power delivery system is shown in Fig. 5. Instead of serving the ElasticCore with a single high rating FIVR, each partition (CW1 through CW4) is served with a dedicated OCVR of lower rating. The rating is defined in terms of the peak output current supplied by the voltage regulator. The PCE of a switching type OCVR is directly proportional to the load current and inversely proportional to the switching frequency [12], [15]. Implementing multiple OCVRs of lower rating, each serving a different partition (CW1 through CW4) of the ElasticCore instead of a single high rating OCVR provides multiple advantages:

- Variation in the power conversion efficiency is reduced as each OCVR supplies current to a smaller load variation.
- OCVRs with moderate peak current density (I_{max} per unit area) and low switching frequency are used to serve each partition if a

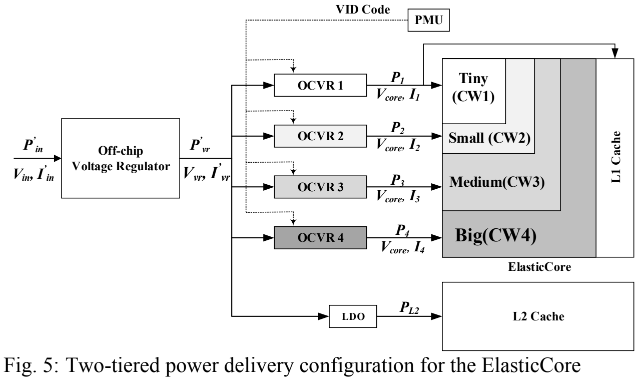


Fig. 5: Two-tiered power delivery configuration for the ElasticCore with multiple OCVRs.

higher weight is assigned to energy efficiency rather than area efficiency.

- Lower rating OCVRs provide faster voltage transitions and hence improved performance with DVFS.
- Reduction in the leakage power of the ElasticCore by selectively shutting down the OCVRs serving unused partitions (CW2 through CW4).

A multi-tiered power supply hierarchy is implemented where an off-chip voltage regulator (VR) first converts the power supply voltage (3.3V to 12V) or Li-Ion battery voltage (3.7V) to a fixed output voltage V_{VR} of 1.35V. The OCVRs then convert the V_{VR} to one of the four discrete voltage levels V_{core} (1.35V, 1.132V, 0.914V, and 0.75V) controlled by the power management unit (PMU). The L2 cache is on a separate voltage domain of 1.13V and is served by a dedicated on-chip low drop-out (LDO) voltage regulator. The LDO provides a high power conversion efficiency (η_{LDO}) of close to 90% [15] due to the small voltage drop from the input voltage V_{VR}. The peak load current consumed by the core across all four voltages, as obtained from McPAT [22], varies from 8A to 30A. Most VRs offer a PCE in the range of 85% to 95% to meet this load current requirement. For example, the PCE of a Texas Instruments LM27403 [16] (voltage mode synchronous buck controller) is 93% with an input voltage of 12V. The variation in the PCE for the LM27403 is less than 3% for a load current in the range of 5A to 40A. The LM27403 is selected as the VR for the ElasticCore. The PCE of the VR is assumed to remain constant across the different modes of operation of the ElasticCore. The peak current consumption for each partition of the ElasticCore is in the range of 4A to 13.5A. The ratings of the OCVRs are selected according to the load current, where a six-phase buck converter is required for low current demand and up to an eighteen-phase converter for high current demand. Each OCVR has a similar topology to the FIVR [14]. The peak PCE of each OCVR remains 90% for the given number of phases and load current. The PCE for the two-tiered power delivery configuration shown in Fig. 5 is given by (2). The energy efficiency of the power delivery system η_{system} is equal to the product of η_{VR} and η_{FIVR}, where η_{FIVR} is assumed equal to η_{LDO}. Based on the given PCEs of the VR and FIVR, the two tiered power delivery system of the ElasticCore offers a peak PCE of 84%.

$$\eta_{system} = \frac{P_{out}}{P_{in}} = \frac{\sum_{i=1}^{4} P_i + P_{L2}}{P_{VR}} = \eta_{VR} \times \eta_{FIVR} \qquad (2)$$

4. METHODOLOGY AND RESULTS

The SMTSIM simulator [21] and McPAT [22] are integrated to obtain the power dissipation of each component while simulating the target architectures. Each benchmark is simulated for 800M instructions after completing the first two billion instructions, and the performance counter data is captured every 200k instructions thereafter. DVFS and configuration adaptation is also performed during the same interval. The DVFS performance of the proposed two-tiered power delivery system for the ElasticCore is compared with a single tier off-chip voltage regulator (VR). A time overhead of 100ns and 100µs is assumed for the proposed power delivery scheme with, respectively, OCVRs and an off-chip voltage regulator [9]. In addition, the time overhead of implementing the *LRM* in hardware and calculating values at each interval is assumed negligible [6]. The power overhead of implementing the *LRM* is estimated as 5uW, which is further reduced by gating idle units during each interval [6]. The following schemes are evaluated for comparison:

- *ElasticCore-Oracle.* This is the ElasticCore architecture with a perfect application predictor, where all future behavior of the application as well as the power and performance for various configurations are known in advance. Therefore, the ElasticCore-Oracle adapts the core resources and frequency, and exploits all opportunities to maximize energy efficiency. The Oracle ElasticCore provides the upper bound for energy efficiency and is therefore used to normalize and compare all the other schemes.
- *ElasticCore-LRM.* This architecture is the ElasticCore architecture with the linear regression model described by (1) to estimate the *BIPS*^{3}*/W* for various core size and frequency/voltage pairs.
- *BigLittle-Oracle.* The BigLittle configuration is similar to the ElasticCore-Oracle architecture except that only two core configurations are used with the four frequency settings: the configuration with the highest performance (big) and the one with the greatest power efficiency (tiny). The selected configuration shows diverse power and performance trade-offs. Based on [3] a separate private L1 cache subsystem is modeled for the BigLittle architecture. Every task migration between cores is therefore costly and incurs an overhead of 20K cycles [3]. The subsequent cache misses are an additional drawback for this configuration. In the ElasticCore, the migration of states is no longer required and pipeline adaptation is done with low overhead.
- *Over-provisioned.* This configuration is based on the worst case behavior of the application. In other words, the core always remains at maximum size (capacity and bandwidth), for every application. The only technique to reduce energy consumption is voltage and frequency scaling which is set to achieve maximum efficiency. Over-provisioning was the primary approach used by industry prior to the development of heterogeneous architectures to respond to the demand of most computing intensive applications.

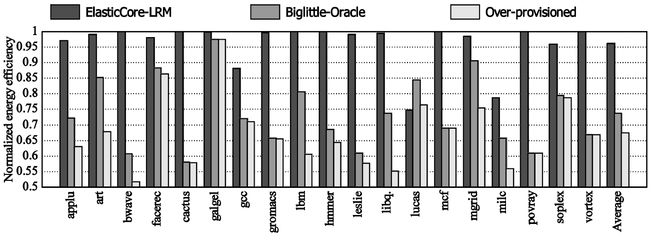


Fig. 6: BIPS^3/W of the applications on various architectures relative to ElasticCore-Oracle.

The results of all the benchmarks normalized to the ElasticCore-Oracle are shown in Fig. 6. The ElasticCore-*LRM* on average achieves close to a 96% efficiency as compared to the Oracle model. The ElasticCore-*LRM* also has an improved energy-efficiency as compared to the BigLittle architecture by an average of 30% across all benchmarks. The BigLittle and over-provisioned architectures provide 74% and 67% efficiency, respectively, as compared to the ElasticCore-Oracle. There are two reasons for the reduced efficiency of the BigLittle architecture. First, the overhead of migration whenever the application moves between the two cores. Second, although the architecture uses the most high performance and the most power efficient cores, the limitation in the core configurations bounds the maximum energy efficiency possible.

The distribution of executed instructions among various core configurations is shown in Fig. 7. The results indicate that the selected benchmarks require different core size and operating frequency. For instance, *art* mostly uses the tiny core configuration, whereas *galgel* mostly uses the big configuration. Moreover, *art* is strictly core size and frequency insensitive as compared to *galgel*, which is sensitive to both core size and frequency. As shown, the difference between ElasticCore-Oracle and ElasticCore-*LRM* is small, indicating that in most cases the ElasticCore-*LRM* accurately scales the core size and voltage/frequency together to maximize the energy-efficiency based on the behavior of the benchmarks. An

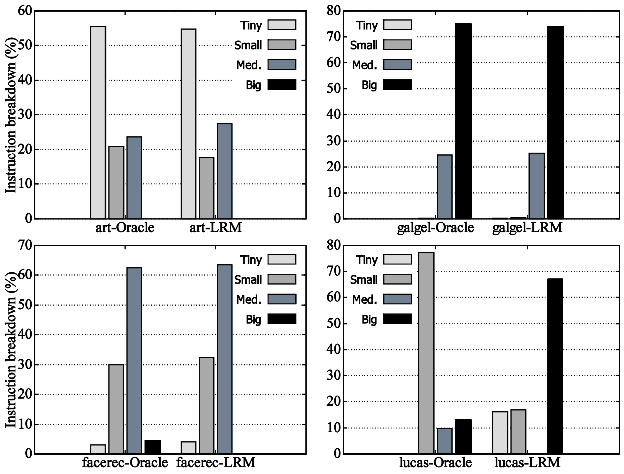


Fig. 7: Distribution of executed instructions on the various core sizes.

exception is for *lucas*, where the main source of error in selecting the most efficient core configuration and frequency lies with the linear regression model.

The impact of an on-chip (OCVR) and off-chip VR on the energy efficiency at the application level is shown in Fig. 8. Two factors reduce the energy efficiency at the front-end. The first factor is the PCE of the voltage regulators. As described in Section 3.4, 7% and 16% of the input power is lost in the off-chip VR and OCVR, respectively. The second factor is the time and energy overheads of scaling the core or the operating voltage/frequency. During voltage scaling and turning on/off the OCVRs, the ElasticCore is stalled. For the former case the voltage needs to be stable for reliable operation of the ElasticCore. For the latter case, the OCVRs turn on or off according to the required core size.

Even though the PCE of the OCVR is less than the off-chip VR, the fast DVFS and phase adaptation as well as more control on leakage power provide additional advantages when using OCVRs. On the other hand, for the off-chip VR, the use of frequent DVFS in the range of a few nanoseconds is not possible. There is therefore a trade-off such that for some applications where frequent core and frequency scaling are required, the OCVR is more beneficial, and in contrast, for other applications where a few changes are required, the off-chip VR is better suited. For instance, the time and energy overheads imposed by the off-chip VR for the *facerec* benchmark, deteriorates the total energy efficiency as compared to OCVR. The proposed two-tiered power delivery topology offers high energy efficiency even for applications that do not benefit from frequent frequency and core scaling by introducing high speed switches that *switch* the supply of current to each partion of the ElasticCore between the OCVRs and off-chip VR.

5. CONCLUSION

In this paper, ElasticCore is proposed, an architecture that is capable of expanding or contracting resources based on the application requirements that enhance the energy efficiency. The architecture concurrently scales the bandwidth, capacity, and voltage/frequency to increase the energy efficiency based on the behavior of the applications at run-time. A linear regression model for power and performance prediction is used to guide the adaptation of the core resources along with the operating voltage and frequency to improve the energy efficiency. In addition, the dynamically scalable partitions of the ElasticCore are powered with multiple on-chip voltage regulators with high power conversion efficiency that are able to realize fast DVFS. Both on-chip and off-chip voltage regulators are analyzed to determine the total energy efficiency at the application level. The ElasticCore was compared to the BigLittle architecture where two different cores with diverse power-performance characteristic are used. The results indicate that the ElasticCore achieves significant energy-efficiency gains and outperforms the BigLittle architecture by 30% on average across various standard applications.

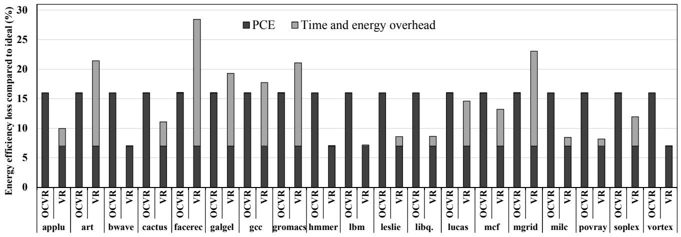


Fig. 8: Impact of the power conversion efficiency of the OCVR and off-chip VR on the energy efficiency.

6. REFERENCES

[1] H. Esmaeilzadeh *et al.*, "Dark silicon and the end of multicore scaling," *Computer Architecture (ISCA)*, pp. 365-376, 2011.

[2] R. Kumar *et al.*, "Single-ISA heterogeneous multi-core architectures", *MICRO-36*, pp. 81-92, 2003.

[3] P. Greenhalgh, "big.LITTLE processing with ARM Cortex-A15 & Cortex-A7," *ARM White Paper*, 2011.

[4] A. Branover *et al.*, "AMD fusion apu: Llano," *IEEE Micro*, no. 2, pp. 28-37, 2012.

[5] OMAP5432 Multimedia Device Data Manual [Public]. available at: http://www.ti.com/product/OMAP5432/technicaldocuments

[6] A. Lukefahr *et al.*, "Composite cores: Pushing heterogeneity into a core." *MICRO-45*, pp. 317-328, 2012.

[7] J. Cong and B. Yuan, "Energy-efficient scheduling on heterogeneous multi-core architectures," *ISLPED,* pp. 345-350, 2012.

[8] E. Ipek *et al.*, "Core fusion: accommodating software diversity in chip multiprocessors," *ACM SIGARCH Computer Architecture News*. vol. 35, no. 2, pp. 186-197, 2007.

[9] C. Kim *et al.*,"Composable lightweight processors," *MICRO-40*, pp. 381-394, 2007.

[10] K. Khubaib et al. "Morphcore: An energy-efficient microarchitecture for high performance ILP and high throughput TLP," *MICRO-45*, pp. 305-316, 2012.

[11] V. Kontorinis et al. "Enabling dynamic heterogeneity through core-on-core stacking." Design Automation Conference (*DAC*), pp. 1-6, 2014.

[12] W. Kim et al.,"System level analysis of fast, per-core DVFS using on-chip switching regulators," *HPCA-14*, pp. 123-134 , 2008.

[13] P. Hammarlund *et a*l., "Haswell: The Fourth-Generation Intel Core Processor," *IEEE Micro* 34, vol. 34, no. 2, pp. 6-20, 2014.

[14] E. A. Burton *et al.*, "FIVR—Fully integrated voltage regulators on 4th generation Intel® Core™ SoCs," *APEC,* pp. 432-439, 2014.

[15] I. Vaisband and E. G. Friedman, "Heterogeneous Methodology for Energy Efficient Distribution of On-Chip Power Supplies," *IEEE Transactions on Power Electronics,* vol. 28, no. 9, pp. 4267-4280, 2013.

[16] Texas Instruments, "LM27403 WEBENCH custom design," http://www.ti.com/product/lm27403.

[17] D. Ponomarev *et al.*, "Dynamic resizing of superscalar datapath components for energy efficiency," *Transactions on Computers,* vol. 55, no. 2, pp. 199-213, 2006.

[18] D. Albonesi *et al.*, "Dynamically tuning processor resources with adaptive processing," *IEEE Computer, Special Issue on Power-Aware Computing*, vol 36, no. 12, pp. 49-58, 2003.

[19] A. Buyuktosunoglu *et al.*, "Energy efficient co-adaptive instruction fetch and issue," *ISCA-30*, pp. 147-156, 2003.

[20] A. Baniasadi and A. Moshovos, "Instruction flow-based front-end throttling for power-aware high-performance processors," *ISLPED*, pp. 16-21, 2001.

[21] D. Tullsen, "Simulation and Modeling of a Simultaneous Multithreading Processor," *Computer Measurement Group Conference*, 1995.

[22] S. Li *et al.*, "McPAT: an integrated power, area, and timing modeling framework for multicore and manycore architectures," *MICRO-42*, pp. 469-480, 2009.

[23] B. Lee and D. Brooks, "Accurate and efficient regression modeling for microarchitectural performance and power prediction," *ACM SIGPLAN Notices*. vol. 41, no. 11, pp. 185-194, 2006.