Scalable Hardware-Based Power Management for Many-Core Systems

Bin Liu, Brent Bohnenstiehl and Bevan M. Baas
Department of Electrical and Computer Engineering
University of California, Davis

***Abstract*—Due to high levels integration, the design of many-core systems becomes increasingly challenging. Runtime dynamic voltage and frequency scaling (DVFS) is an effective method in managing the power based on performance requirements in the presence of workload variations. This paper presents an on-line scalable hardware-based dynamic voltage frequency selection algorithm, by using both FIFO occupancy and stall information between processors. To demonstrate the proposed solution, two real application benchmarks are tested on a many-core globally asynchronous locally synchronous (GALS) platform. The experimental results show that the proposed approach can achieve near-optimal power saving under performance constraints.**

***Index Terms*—Dynamic voltage and frequency scaling (DVFS), many-core processors, multi-processor systems-on-chip (MPSoCs), globally asynchronous locally synchronous (GALS)**

I. Introduction

With the continuous scaling of CMOS technology, a large number of processing elements (PE) are able to be integrated on a single silicon die, resulting in multi-processor systems-on-chip (MPSoCs). Many-core processors with network-on-chip (NoC) interconnects are promising architectures for high performance energy-efficient computing [1], [2]. However, power management is a critical challenge for the design of many-core platforms. Increasing power consumption not only decreases energy efficiency, but also causes high die temperature which jeopardizes the performance and reliability of chips.

In many-core systems, due to the diversity of tasks mapped on different cores, the workload and performance requirements of different cores are varied and also changing over time, as shown in Fig. 1. There are computation-intensive jobs that require processors to run at full speed; however, there are also non-performance critical jobs which can be computed at low speed while still meeting the performance requirements of the system. Dynamic voltage and frequency scaling (DVFS) exploits the fact that dynamic power is proportional to the cube of supply voltage, considering operating frequency has a linear dependence on supply voltage, to perform dynamic voltage scaling in order to provide "just-enough" processor speed to finish the workload under time and performance constraints, while reduce the power dissipation at meantime. Many-core systems with per-core DVFS have been proved to be capable of reducing energy dissipation significantly by adapting both voltage and frequency of the system with respect of changing workload [3], [4].

This paper proposes a scalable hardware-based per-core DVFS solution driven by workload variations for many-core systems. The remainder of this paper is organized as follows. Section II presents related work and the contributions of this paper. Section III discusses the preliminaries and assumptions used in this paper. Section III describes the proposed solution. Section V discusses the experimental results with real application benchmarks. Finally, Section VI concludes the paper.

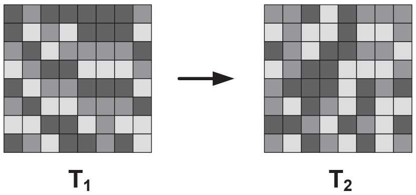

Fig. 1. Workload varies among cores, and also changes over time in many-core systems.

II. Related Work And Novel Contributions

A. Related Work

Various DVFS schemes based on FIFO occupancy for many-core processors with globally asynchronous locally synchronous (GALS) have been discussed in the literature. Wu et al. formally described a nonlinear model of the queue occupancy, and presented a proportional-integral-derivative (PID) controller, which requires detailed analysis of the queue behavior before actual hardware implementation [5]. Alimonda et al. analyzed more complex queue configurations, and developed a non-linear controller which offers better transient and steady-state performance, as compared with linear controllers [6]. Orgas et al. presented an adaptive feedback controller based on state-space models to determine the optimal voltage frequency island (VFI) for different PEs [7]. Garg et al. extended Orgas's work by adopting both local and global state feedback to balance the energy saving and the implementation complexity of the DVFS controller [8].

On the other hand, Choudhary and Marculescu proposed a method which adopts the stall information, rather than the occupancy, of FIFOs. The DVFS controller counts the stalling time from both the producer and the consumer of a communication link to determine the optimal VFI for PEs in the next control interval [9].

B. Novel Contributions

Compared to the previous work outlined here, we make the following novel contributions:

- We propose an online, scalable, hardware-based DVFS algorithm based on both FIFO occupancy and stall information between communication links. The proposed method not only takes advantage of the fast response to workload variation by monitoring FIFO occupancy, but also use FIFO stall information to cover the scenarios that are not solvable by monitoring FIFO occupancy only. In contrast, previous work either use FIFO occupancy or FIFO stall information only.
- We demonstrate the proposed DVFS scheme with real application benchmarks on a many-core GALS system. The experimental results shows the proposed DVFS controller can achieve near-optimal power saving under performance constraints.

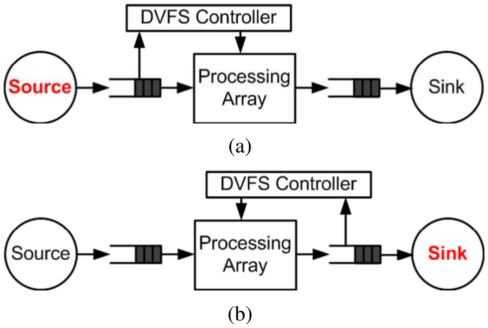

Fig. 2. DVFS controllers with constraints from (a) upstream (source) and (b) downstream (sink).

III. Preliminaries and Assumptions

A. Performance Constrained Systems

In general, there are two categories of performance constraints for real time systems, *upstream* constraints and *downstream* constraints. The constraints from upstream are given by the source, which means that the input data has to be processed at a certain rate to ensure correct operation, like analog-to-digital conversion. On the other hand, systems are constrained by downstream. In this case, a certain output rate is required to be satisfied for correct operation, like video processing, wireless communication and digital-to-analog conversion. Fig. 2 shows DVFS controllers are tuned by the inputs and outputs, for systems with upstream and downstream constraints, respectively. In the rest of this paper, input constrained systems are being considered, and all the results could be extended to output constrained systems as well.

B. Multiple Power Domains and Voltage Dithering

A common technique of supplying multiple voltage domains for many-core systems is to integrate on-chip DC-DC converter for each individual processor. However, the overhead of the approach is undesirable if the number of cores increases beyond a few. A alternative technique using limited number of parallel global power grids is adopted due to its efficiency and simplicity. Processors choose their supply voltage by connecting their local power grid to one of the parallel global power grids through power gates. The approach has been proved be simple, efficient and capable of switching between power grids in a few nanoseconds without introducing prominent voltage noise [3].

For systems with limited number of global power domains, there are two major voltage scaling approaches. The first one prioritizes frequency scaling over voltage scaling. The processor changes its operation frequency based on its workload, and chooses the lowest possible supply voltage from the global power grids,

$$VddCore = \begin{cases} Vdd_i, & \text{if } f_{max}(Vdd_{i-1}) < f_{req} \le f_{max}(Vdd_i) \\ Vdd_1, & \text{if } f_{req} \le f_{max}(Vdd_1) \end{cases} \qquad (1)$$

where f_{req} is the desired working frequency, and $Vdd_1 < Vdd_2 < ... < Vdd_{N-1} < Vdd_N$ are the available N global power grids.

In contrast of prioritizing frequency scaling, *voltage dithering* switches between the voltage and frequency pairs above and below the desired frequency to achieve the performance requirements [10]. For example, if the available normalized frequencies are 0.75 and 0.5, while the desired rate is 0.6, the core would spend 40% of processing time at the frequency of 0.75, and 60% at 0.5. In general, voltage dithering DVFS could be described as:

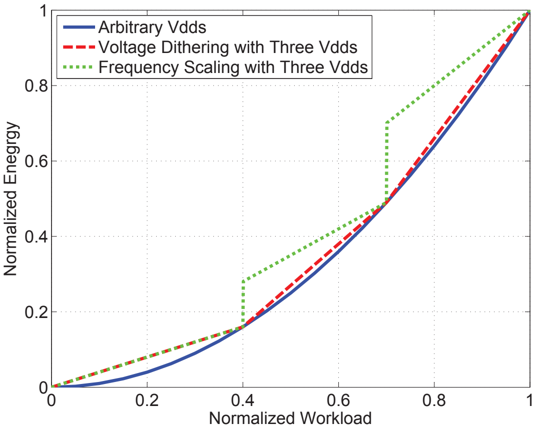

Fig. 3. Normalized energy versus workload of frequency scaling with three Vdds, voltage dithering with three Vdds and theoretically ideal case with arbitrary levels of voltage domain.

$$VddCore = \begin{cases} P \times Vdd_i + (1-P) \times Vdd_{i-1}, \\ \qquad \text{if } f_{max}(Vdd_{i-1}) < f_{req} \le f_{max}(Vdd_i) \\ Vdd_1, \text{if } f_{req} \le f_{max}(Vdd_1) \end{cases} \qquad (2)$$

where P and $(1-P)$ are the percentage of processing time at which the local $VddCore$ is assigned to Vdd_i and Vdd_{i-1}, respectively. P is defined according to the desired core's frequency as

$$P = \frac{f_{req} - f(Vdd_{i-1})}{f(Vdd_i) - f(Vdd_{i-1})} \qquad (3)$$

As illustrated in Fig. 3, voltage dithering achieves better energy efficiency ccompared to frequency scaling. Additionally, the energy efficiency obtained by voltage dithering with three Vdds is very close to ideal DVFS systems with infinite number of Vdds. Therefore, voltage dithering with three Vdds is used as the multiple voltage architecture in the following paper.

IV. Proposed Algorithm

The proposed DVFS algorithm applies both *FIFO occupancy* and *FIFO stall information* to determine the voltage and frequency for each individual core in the many-core system.

A. FIFO Occupancy

Since we are considering upstream constrained systems, the constraints come from the input ports of the system. When input FIFOs tend to be empty, it means that the processor runs too fast and could slow down by tying with a lower Vdd to save energy. On the other hand, if input FIFOs fill up, it means that the processor runs too slow and should speed up by tying with a higher Vdd to satisfy the performance requirements. Suppose there are N voltage domains, which are $V_1, V_2, ..., V_N$. Then each FIFO is split into $(2N-2)$ levels evenly, which ranges from $L_1, L_2, ..., L_{(2N-2)}$. If there are M input FIFOs for the core under test, and the input FIFO occupancy are represented by $O_1, O_2, ..., O_M$, the current input FIFO occupancy for the core under test is $O_{curr} = MAX(O_1, O_2, ..., O_M)$. Assume that the previous input FIFO occupancy of the core is O_{pre} and the core is

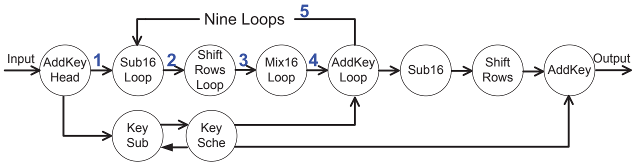

Fig. 4. 10-core AES engine dataflow diagram.

tied with V_{pre}, then the next voltage domain the core tied is

$$V_{occu} = \begin{cases} V_1, & \text{if } O_{curr} = L_1 \\ V_{pre+1}, & \text{if } O_{curr} > O_{pre+1} \\ V_{pre}, & \text{if } O_{pre-1} \le O_{curr} \le O_{pre+1} \\ V_{pre-1}, & \text{if } O_{curr} < O_{pre-1} \\ V_N, & \text{if } O_{curr} = L_{(2N-2)} \end{cases} \tag{4}$$

Then, the O_{pre} and V_{pre} are set to O_{curr} and V_{occu} for the next comparison, respectively.

B. FIFO Stall Information

Although FIFO occupancy is effective to help processors to choose appropriate voltage domains, it may provides wrong decisions in some scenarios. Fig. 4 shows a 10-core Advanced Encryption Standard (AES) engine on a many-core system [11]. Considering the AES engine has upstream constraints, and the cores' voltage and frequency pairs adjust only based on input FIFOs occupancy. When the performance requirement increases, the input FIFO for the whole engine would fill up and speed up *AddKeyHead*. Then, the FIFO 1 would tend to be full, and *Sub16-Loop* would be tied to a higher voltage and frequency pair. In the AES engine, only one 16-byte data block is allowed to be processed in the loop operation at anytime. Therefore, FIFO 2, 3, and 4 would never be filled up enough to speed up the corresponding cores. As a result, the AES engine fails to satisfy the desired throughput requirement. From the above example, it shows that choosing voltage and frequency only based on FIFO occupancy may cause failures.

In order to solve the problem, we propose a new voltage scaling technique called *workload inheritance*, which is based on FIFO stall information. Assume that one of the *coreA* input FIFOs connects with one of the *coreB*'s output FIFOs. If *coreA* is stalled on *coreB* due to empty on input (EOI), and *coreA* is not tied with the lowest voltage-frequency pair, then *coreB* is considered to inherit workload from *coreA*. Each core has a *stall counter*, C_{stall}, which counts up if it inherits workload from any of its output cores; otherwise counts down. Suppose there are N voltage domains, which are $V_1, V_2, ..., V_N$. Additionally, there are $(N-1)$ stall counter thresholds for switching between Vdds are defined as $T_1, T_2, ..., T_{N-1}$. The voltage of the core under test is determined as

$$V_{stall} = \begin{cases} V_1, & \text{if } C_{Stall} \le T_1 \\ V_i, & \text{if } T_{i-1} < C_{Stall} \le T_i \\ V_N, & \text{if } C_{Stall} > T_{N-1} \end{cases} \tag{5}$$

Considering the above example, when *Sub16-Loop* speeds up due to fill up of FIFO 1, while *ShiftRows-Loop*, *Mix16-Loop*, and *AddKey-Loop* run at lower frequencies than required, *Sub16-Loop* would be eventually stalled on *AddKey-Loop* due to read of empty on FIFO 5. Therefore, *AddKey-Loop* would inherit workload from *Sub16-Loop*, and speed up. Similarly, the voltage and frequency levels of *ShiftRows-Loop* and *Mix16-Loop* would be also tuned up until the performance requirement got satisfied.

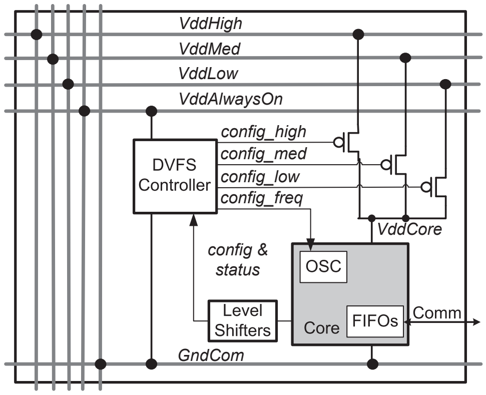

Fig. 5. Block diagram of a single core in the targeted many-core platform.

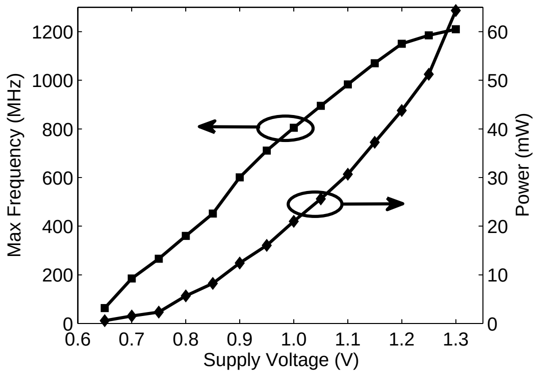

Fig. 6. Maximum operation frequency and 100% active power dissipation of one core versus supply voltage.

The algorithm of selecting voltage and frequency dynamically with combined FIFO occupancy and FIFO stall information is shown in Algorithm 1.

V. Experimental Results

To test the proposed DVFS algorithm, we use two AES engines on a many-core platform as benchmarks.

A. Targeted Many-Core Platform

Fig. 5 shows the block diagram of a single core in the targeted many-core system. Each core can run at one of the three power domains, which is controlled by the proposed DVFS algorithm. All the cores are clocked by local fully independent oscillators, and connected by a reconfigurable 2D-mesh network that supports both nearby and long-distance communication.

As shown in Fig. 6, each core can operate up to 1.2 GHz at 1.3 V [1]. The maximum frequency and power consumption of cores have a near-linear and quadratic dependence on the supply voltage, respectively. The measurement data of supply voltage, clock frequency and power are used in the case study.

B. Benchmark I – 137-core AES Engine

Fig. 7 shows mapping diagram of the 137-core AES engine on the targeted many-core platform. The throughput requirement of the AES cipher is set to 2.2 Gbps, which requires the processors on the critical path to run at 1.2 GHz with 1.3 V. The three voltage-frequency pairs in (V, MHz) are chosen as (1.3, 1210), (1.18, 1034) and (0.98, 700).

Algorithm 1 Dynamic Voltage and Frequency Selection

Inputs: Current voltage domain V_{curr}; N discrete voltage and frequency levels (V_1, f_1), ..., (V_N, f_N); M output FIFO links of the core under test, whether the sink core is stalled on it, and which voltage rail the sink core is using (S_1,V_{O1}), ..., (S_M,V_{OM}); Current FIFO occupancy of P input FIFO links of the core under test O_1, ..., O_P; $(N-1)$ voltage switching thresholds for stall counter T_1, ..., T_{N-1};
Outputs: New voltage and frequency level (V_{next},f_{next}) for the core under test.

```
# algorithm uses FIFO occupancy
$MaxOccupancy = O_1$
for $j = 2 : P$ do
  if $O_j > MaxOccupancy$ then
    $MaxOccupancy = O_j$
  end if
end for
if $MaxOccupancy == L_1$ then
  $V_{occu} = V_1$
else if $MaxOccupancy == L_{(2N-2)}$ then
  $V_{occu} = V_N$
else if $MaxOccupancy >$ (one level above $O_{pre}$) then
  $V_{occu} = V_{pre}$ plus one level
else if $MaxOccupancy <$ (one level below $O_{pre}$) then
  $V_{occu} = V_{pre}$ minus one level
else
  $V_{occu} = V_{pre}$
end if
if $V_{pre}! = V_{occu}$ then
  $V_{pre} = V_{occu}$
  $O_{pre} = MaxOccupancy$
end if
# algorithm uses FIFO stall information
$C_{inc}$ = FALSE
for $i = 1 : M$ do
  if $S_i$ and $V_{Oi}$ != $V_1$ then
    $C_{inc}$ = TRUE
    BREAK
  end if
end for
if $C_{inc}$ then
  $C_{stall}++$
else
  $C_{stall}--$
end if
$V_{stall} = V_1$
for $i = 2 : (N-1)$ do
  if $C_{stall} > T_i$ then
    $V_{stall} = V_i$
  end if
end for
# select the new voltage and frequency group
$V_{next} = MAX(V_{occu}, V_{stall})$
if $V_{next} > V_{curr}$ then
  Set Current Voltage to $V_{next}$
  Set Current Frequency to $f_{next}$
else if $V_{next} < V_{curr}$ then
  Set Current Frequency to $f_{next}$
  Set Current Voltage to $V_{next}$
end if
```

The optimal frequencies for different cores are obtained with static workload analysis. The cores on the critical path are configured to run at the highest voltage and frequency level to meet performance requirements. On the other hand, non-critical cores should run at the lowest possible voltage and frequency levels to minimize the power dissipation, as long as not violating performance requirements.

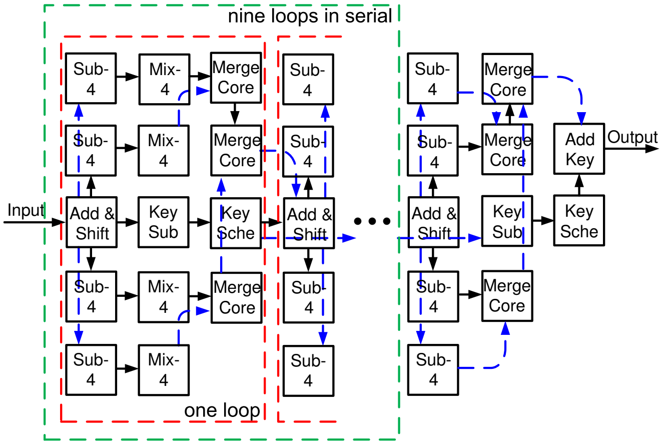

Fig. 7. Mapping diagram of the 137-core AES engine [11].

TABLE I
OPTIMAL FREQUENCIES AND WORKING FREQUENCIES SELECTED BY THE PROPOSED DVFS ALGORITHM OF ALL PROCESSORS IN THE 137-CORE AES ENGINE.

Processor Name	Optimal Freq. (MHz)	DVFS Selected Freq. (Mhz)
AddKeyShiftRows	1158	1198
SubByte-4	691	762
MixColumn-4	1210	1210
KeySub	898	906
KeySche	1037	1067
MergeCore	1210	1210
FinalRoundAddKey	545	584

The frequency selected by the proposed DVFS algorithm is defined by

$$Freq_{DVFS} = \sum Per_{Vdd_i} \cdot freq_{Vdd_i} \qquad (6)$$

where Per_{Vdd_i} is the percentage of processing time the core spends on Vdd_i, and $freq_{Vdd_i}$ is the frequency of the core runs at supply voltage Vdd_i. Table I shows the comparison between the optimal frequencies and the frequencies selected by the proposed DVFS algorithm.

Fig. 8 shows the power savings for each individual core by adopting three different solutions. IDEAL-3 represents the optimal static solution with three voltage frequency pairs, while IDEAL-Inf represents the optimal static solution with infinite voltage and frequency levels. IDEAL-Inf gives the theoretical best power savings. As shown in Fig. 8, *SubByte-4*, *KeySub* and *FinalRoundAddKey* obtain significant power saving as much as 60%. On the other hand, there is no power saving for *MixColumns-4* and *MergeCore*, since they are on the critical path and determine the performance of the system. Overall, the proposed DVFS algorithm shows a 18% power improvement under the throughput constraint, which is only 2% less than IDEAL-3 and 3% less than IDEAL-Inf.

C. Benchmark II – 10-core AES Engine

Fig. 4 shows the 10-core AES engine diagram on the targeted many-core platform. The throughput requirement is set to 43 Mbps, which is the maximum throughput of the 10-core AES engine. The three voltage and frequency pairs in (V, MHz) are chosen as (1.3, 1210), (0.95, 648) and (0.67, 50). As shown in Table II, all the four cores in the loop are tied with the highest voltage and run at the

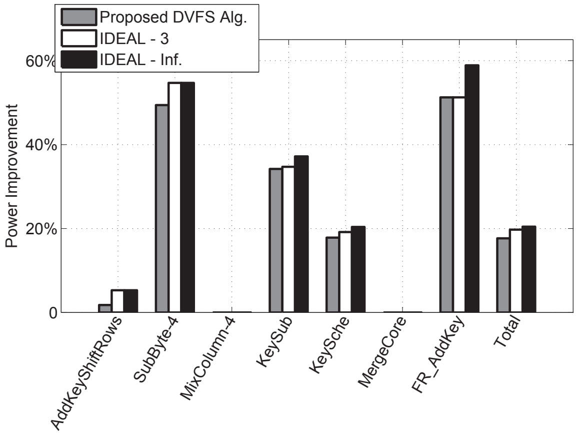

Fig. 8. Power improvement of all processors in the 137-core AES engine.

TABLE II
OPTIMAL FREQUENCIES AND WORKING FREQUENCIES SELECTED BY THE PROPOSED DVFS ALGORITHM OF ALL PROCESSORS IN THE 10-CORE AES ENGINE.

Processor Name	Optimal Freq. (MHz)	DVFS Selected Freq. (Mhz)
AddKey-Head	15	15
SubByte-4-Loop	1210	1210
MixColumn-4-Loop	1210	1210
ShiftRows-Loop	1210	1210
AddKey-Loop	1210	1210
KeySub	277	382
KeySche	277	339
SubByte-4	50	50
ShiftRows	12	12
AddKey	15	15

highest frequency all the time. All the other cores have great potential for power saving. Fig. 9 shows the proposed DVFS algorithm saves 40% to 90% power for each individual core, and approximate 28% for the whole system.

VI. CONCLUSION

In this paper, we propose a scalable online hardware based DVFS architecture for many-core systems. Compared to the previous work, both FIFO occupancy and FIFO stall information are considered to select the most energy efficient voltage and frequency level for each core depends on its workload. To demonstrate the proposed algorithm, two real application benchmarks are tested with a many-core system. The experimental results show that the proposed DVFS approach can achieve near-optimal power saving under performance constraints.

VII. ACKNOWLEDGMENTS

The authors gratefully acknowledge support from NSF Grant 1018972 and 0903549, SRC GRC Grant 1598, 1971 and 2321, CSR Grant 1659, C2S2 Grant 2047.002.014, UC Micro, Intel, and Intellasys. The authors would like to thank STMicroelectronics for donating the chip fabrication.

REFERENCES

[1] D. N. Truong, W. H. Cheng, T. Mohsenin, Z. Yu, A. T. Jacobson, G. Landge, M. J. Meeuwsen, A. T. Tran, Z. Xiao, E. W. Work, J. W. Webb, P. Mejia, and B. M. Baas, "A 167-processor computational platform in 65 nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 44, no. 4, pp. 1130–1144, Apr. 2009.

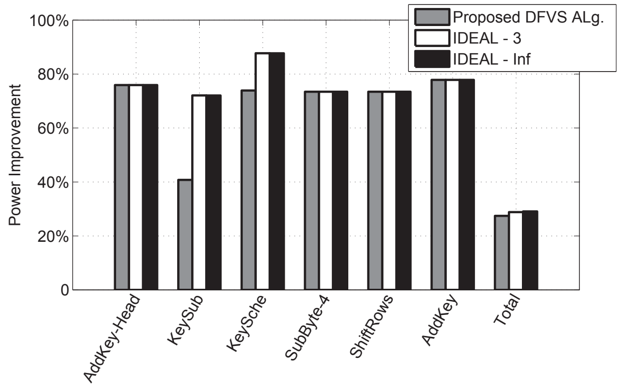

Fig. 9. Power improvement of all processors in the 10-core AES engine.

[2] S. R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar, "An 80-tile sub-100-w teraflops processor in 65-nm CMOS," *IEEE Journal of Solid-State Circuits*, vol. 43, no. 1, pp. 29–41, Jan. 2008.

[3] Wayne H. Cheng and Bevan M. Baas, "Dynamic voltage and frequency scaling circuits with two supply voltages," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2008, pp. 1236–1239.

[4] Wonyoung Kim, Meeta Sharma Gupta, Gu-Yeon Wei, and David Brooks, "System level analysis of fast, per-core DVFS using on-chip switching regulators," in *IEEE International Symposium on High Performance Computer Architecture (HPCA)*, Feb. 2008, pp. 123–134.

[5] Qiang Wu, Philo Juang, Margaret Martonosi, and Douglas W. Clark, "Formal online methods for voltage/frequency control in multiple clock domain microprocessors," in *Proceedings of the 11th International Conference on Architectural Support for Programming Languages and Operating Systems*, 2004, pp. 248–259.

[6] A. Alimonda, Salvatore Carta, A. Acquaviva, A. Pisano, and L. Benini, "A feedback-based approach to dvfs in data-flow applications," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 28, no. 11, pp. 1691–1704, Nov. 2009.

[7] U.Y. Ogras, R. Marculescu, and D. Marculescu, "Variation-adaptive feedback control for networks-on-chip with multiple clock domains," in *Design Automation Conference, 2008. DAC 2008. 45th ACM/IEEE*, June 2008, pp. 614–619.

[8] Siddharth Garg, D. Marculescu, and R. Marculescu, "Custom feedback control: Enabling truly scalable on-chip power management for MPSoCs," in *Low-Power Electronics and Design (ISLPED), 2010 ACM/IEEE International Symposium on*, Aug 2010, pp. 425–430.

[9] P. Choudhary and D. Marculescu, "Hardware based frequency/voltage control of voltage frequency island systems," in *International Conference Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, Oct. 2006, pp. 34–39.

[10] V. Gutnik and A. Chandrakasan, "Embedded power supply for low-power dsp," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 5, no. 4, pp. 425–435, 1997.

[11] Bin Liu and Bevan M. Baas, "Parallel AES encryption engines for many-core processor arrays," *Computers, IEEE Transactions on*, vol. 62, no. 3, pp. 536–547, march 2013.