Journal scientific and applied research, vol. 22, 2021
International Journal

Original Contribution

ISSN 1314-6289

TESTING AMD RYZEN MICROPROCESSOR AND ITS ENERGY EFFICIENCY

Daniel R. Denev

KONSTANTIN PRESLAVSKI UNIVERSITY OF SHUMEN, 115 UNIVERSITETSKA, SHUMEN 9700,
E-mail: slimshady33@abv.bg

ABSTRACT: *In this paper, we show a case study of Ryzen processor energy efficiency using SPECjbb2015 as the workload. We analyze it in comparison to Phenom II, a CPU of previous generation from the same manufacturer AMD. In terms of maxi-mum transaction throughput, Ryzen performs 218% of Phenom. With the response time constraint, the relative performance of Ryzen is 288%. In the energy efficiency, Ryzen achieves 309% (at maximum throughput) and 389% (with response time constraints) of Phenom. Dynamic frequency scaling (DFS) is effective for reducing the power consumption of both processors, but the load levels at which DFS is most effective are quite different.*

KEY WORDS: *Workload Analysis, Performance Evaluation, Power Consumption, Energy Efficiency, Measurement, SPECjbb2015*

1. Introduction

Ryzen is a new series of CPUs from AMD that are targeted to the high-performance CPU market, which has been dominated by Intel for years. In this paper, we present a case study of performance and energy efficiency of Ryzen 5 1600, a six-core model of Ryzen. We evaluate Ryzen in comparison to a Phenom II 1065T, another six-core CPU from AMD which we have used in the past work extensively. For the workload of evaluation, SPECjbb2015, a benchmark suite from SPEC for Java application servers is used.

This paper is organized as follows. In the next section, the features of the Ryzen architecture are presented. In Section III, SPECjbb2015, the workload used in this study, is described. Section IV presents the measurement results and their analysis, including the performance metrics, power consumption, the

effectiveness of dynamic frequency scaling and energy efficiency. Section V concludes the paper with the topics of future work.

2. Ryzen processor

In this section, the key characteristics of Ryzen processors are described in contrast to Phenom II (used as the comparison base in Section IV). Ryzen is based on the Zen microarchitecture, which was presented at the HotChip conference in 2016. Its transistor size is 14nm, much smaller than 65nm of Phenom II. Both are capable of dynamic frequency scaling, with four (Phenom) and three (Ryzen) clock frequencies as shown in the table. Both processors can boost the clock frequencies to 3.4GHz (Phenom) and 3.6GHz (Ryzen) within the total power and thermal limits [1, 4].

There are two differences in the sizes of caches. First, L1 data cache is 32KB in Ryzen while it is 64KB in Phenom. In a Ryzen processor, four cores are grouped as a CPU complex (CCX) and share an L3 cache of 8MB. In the case of six-core models, including Ryzen 5 1600 used in this paper, a core in each of two CCXes is disabled. Phenom has a single L3 cache shared by all six cores. Both processors decode maximum four x86 instructions at a time, and decoded instructions (micro-ops) are dispatched to the execution units of 4 ALU, 2 address generation unit (AGU) and 4 FPU in Ryzen. The execution units of Phenom II consist of 3 ALU, 3 AGU and 3 FPU. The sizes of load/store queues are 72 and 44 entries in Ryzen, while those of Phenom are 44 and 24, respectively. Thermal design points (TDPs) of Ryzen and Phenom are 95 and 65Watt, respectively.

Table 1. Phenom II and Ryzen microarchitectures

	Phenom II X6 1065T	Ryzen 5 1600
Transistor Size	65nm	14nm
Clock Freq. (GHz)	0.8, 1.5, 2.2, 2.9	1.55, 2.8, 3.2
L/S Queue Entries	44/24	72/44
	Cache Hierarchy	
L1 (private) L2 (private) L3 (shared)	64KB (I) 64KB (D) 512KB 6MB	64KB (I) 32KB (D) 512KB 16MB (8MB/CCX)
Execution Units	3 ALU, 3 AGU, 3 FPU	4 ALU, 2 AGU, 4 FPU
TDP (W)	95	65

In additions to the specifications in Table 1, Ryzen has the following features:

— Simultaneous multi-threading (SMT), doubling the num-ber of logical cores as that of the physical cores
— Precision Boost, increasing the clock frequency in 25MHz granularity based on temperature, current and load level
— Neural net branch prediction1 and Smart Prefetch to reduce the instruction and data latency.

3. SPECjbb2015

In this section, brief descriptions of the target architecture, workload design, and performance metrics of SPECjbb2015 (jbb15) are presented. Please refer to the SPEC's site for the official information of the jbb15. Please also note that all measurement results using jbb15 in this paper are not audited by SPEC and fall into the academic research usage defined in.

jbb15 is a benchmark suite from the Standard Performance Evaluation Corporation for evaluating the performance of Java server applications. It models the IT infrastructure and business operations of the global supermarket chain. The benchmark consists of three types of components: Controller (Ctr), Transaction Injector (TxI) and Backend (BE). The Ctr conducts the overall operations of other components (TxI and BE), especially synchronizing the operation timing of other components and precedence of the execution phases, such as warming-up, measurement and validation.

jbb15 allows three configurations for the platform on which above three components are implemented (called "system under test," or SUT): Composite, Multi-JVM and Distribute. The Composite is the simplest form of the jbb15 SUT in which all components run within a single JVM on a single host. In the Multi-JVM configuration, BE can be spread over multiple instances of JVMs which run on a single host. In the Distributed configuration, TxI and Ctr run on a separate host than the BE. Also, BE can run on multiple JVMs on multiple hosts. All data structures are stored in the main memory, meaning the effect of the storage devices (e.g. HDD) on the performance is negligible.

jbb15 has two performance metrics: max-jOPS and critical-jOPS. The former is the number of transactions processed per second without response time constraint. jbb15 also measures the maximum throughput at five service level agreement (SLA) points based on the response times (10ms, 25ms, 50ms, 75ms and 100ms)2. The geometric mean of these five points is called the critical-jOPS.

The system scaling unit, which is proportional to the transactions throughput, is called the injection rate, or IR. The above performance metrics are also represented by the IR [2, 6].

4. Measurement and analysis

In this section, the results and analysis of the measurements are presented. We first describe the measurement platforms based on the two CPUs in Table I in the next subsection. In Section IV-B, both processors are compared in terms of the jbb15 metrics. We also evaluate the effects of updated Linux kernel on Phenom and SMT on Ryzen in Section IV-B. Section IV-C shows the effectiveness of dynamic frequency scaling on both processors. Section IV-D compares two pro-cessors in terms of energy efficiency at varying load levels.

4.1. Measurement Environment

Table 2 shows the specifications of the measurement plat-forms. The middle and the right columns of the table present the specifications of the Phenom and Ryzen-based platforms, respectively. The idle power consumption of the platforms is 73.0 (Phenom) and 41.9Watt (Ryzen). We use the dynamic power (total minus idle) as the metric of power consump-tion. While both platforms have 16GB of main memory, the DRAM specifications are PC3-10500 (Phenom) and PC4-19200 (Ryzen).

Table 2. Measurement platform specifications

CPU	Phenom II X6 1065T	Ryzen 5 1600
Memory	16 GB (PC3-10500)	16GB (PC4-19200)
Idle Power (W)	73.0	41.9
	Software	
OS	Ubuntu 16.04 (Kernel 4.4.0)	
Java version	1.8.0 111	
SPECjbb2015 version	1.0.0	
DFS Control	cpufreq utility	

Both platforms run jbb15 with the Composite configuration on Ubuntu 16.04 (Linux Kernel 4.4.0). To control the clock frequency, cpufreq utility is used. Watts up? Pro 99333 power meters are placed between the SUT and the power outlet for the measurements of the SUTs' power consumption. For each data point, an average of (minimum) ten measure-ment results is reported. While we were

working on this paper with jbb15 version 1.0.0, an updated version of jbb15, 1.0.1, was published. However, as mentioned in the results with Java 1.8 should be compatible between 1.0.0 and 1.0.1.

4.2 Performance Metrics

Up to we were using Oracle Linux 6.1 on Phenom, which was based on Linux kernel 2.6 for the compatibility with the prior work. For this paper, we use Ubuntu 16.04 which is based on kernel 4.4.0 for both Ryzen and Phenom platforms. First, we measured the effect of Kernel update on the Phenom platform. The left bars in Fig. 1 show the relative performance of Kernel 4.4.0 against 2.6.32 (left Y-axis). Please note that all results presented in this subsection are run without DFS (i.e. the clock frequency being fixed at 2.9GHz and 3.2GHz on Phenom and Ryzen, respectively). Kernel 4.4.0 improves the max-jOPS by 12% over the 2.6.32. However, when it comes to critical-jOPS, the advantage of the new kernel is reduced to 7%. By taking a look at each of SLA metrics, the performance gain comes mostly from SLA-75000 and SLA-100000. Middle bars in Fig. 1 compare the performance of Phenom and Ryzen (right Y-axis). Ryzen achieves 218% of relative performance of Phenom in terms of max-jOPS. For the critical-jOPS, Ryzen further improves the performance to 288%; especially, under the tightest response time limit of 10ms, Ryzen performs 304% of Phenom. From these results, Ryzen improves not only the throughput but also the response time of jbb15.

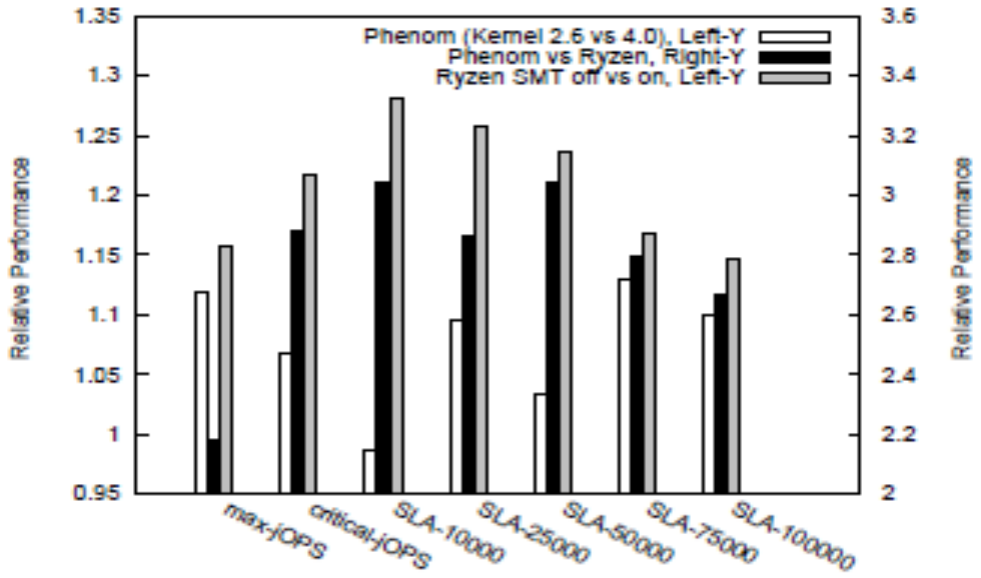

Fig. 1. Performance Comparisons in jbb15 Metrics: Kernel 2.6 vs 4.0 on Phenom (left), Phenom vs Ryzen (center) and SMT off and on on Ryzen (right).

One of the new features of Ryzen is the simultaneous multi-threading (SMT), which we have mostly seen as Intel's HyperThreading in the x86 market so far. We also evaluated the performance advantage of the SMT on Ryzen. Right bars in Fig. 1 compare the jbb15 metrics with SMT being enabled and disabled (left Y-axis). SMT improves the max-jOPS metric of Ryzen by 15%. SMT is more effective in terms of critical-jOPS; this performance metric is improved by 22%, ranging from 15% at response time of 100ms to 28% at 10ms. In we evaluated the effectiveness of the SMT (HyperThreading) on ATOM D525 using SPECjEnterprise2010 and observed 47% improvement with the SMT. There are numerous factors that prevent us from directly comparing the results in this work especially the difference between the in-order instruction issue of ATOM versus the out-of-order issue of Ryzen. In the latter case, the higher level of instruction-level parallelism of jbb15 already utilized available execution units and not much room for the thread level parallelism remained. The effectiveness of the SMT is one of the points we need further investigations [3, 5].

4.3 Effectiveness of DFS

Dynamic frequency scaling (DFS) is a technique to reduce the power consumption of the processor by lowering its clock frequency according to the load level. Both Ryzen and Phenom are capable of DFS, with the available clock frequencies in Table 1.

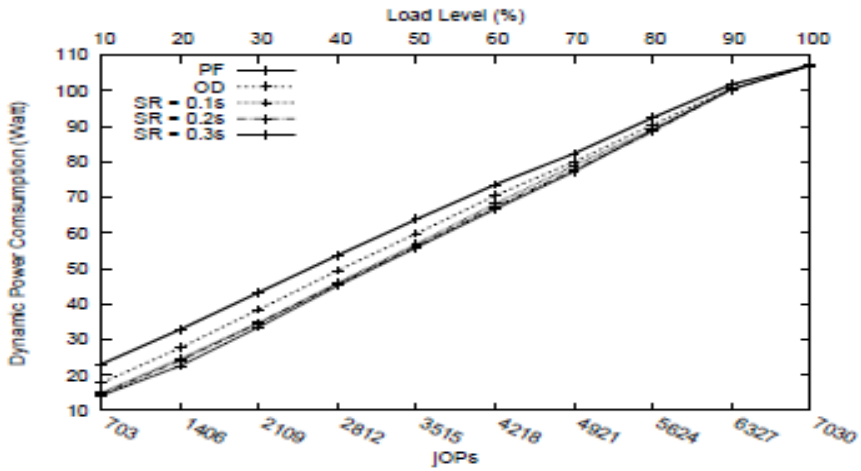

Fig. 2. Phenom Dynamic Power Consumption with DFS

From our past work the minimum (and default) sampling period 3 of Phenom, 10ms, which is the same for Ryzen, is too short and stretching it further reduces the power consumption. Phenom and Ryzen without DFS achieve max-jOPS of 7035 and 15340 jOPS, respectively. We take these throughput (rounded

to 7030 in the case of Phenom) as their 100% load levels. Figs. 2 and 3 show the dynamic power consumption of Phenom and Ryzen for the load levels in 10% increment with varying DFS options: PF (taken from the name of the DFS governor, performance, in cpufreq) is the case where DFS is disabled. OD (ondemand governor) is the case where DFS with the default sampling period (sampling_rate) of 10ms. SR = XS stands for the ondemand with the sampling period of X second.

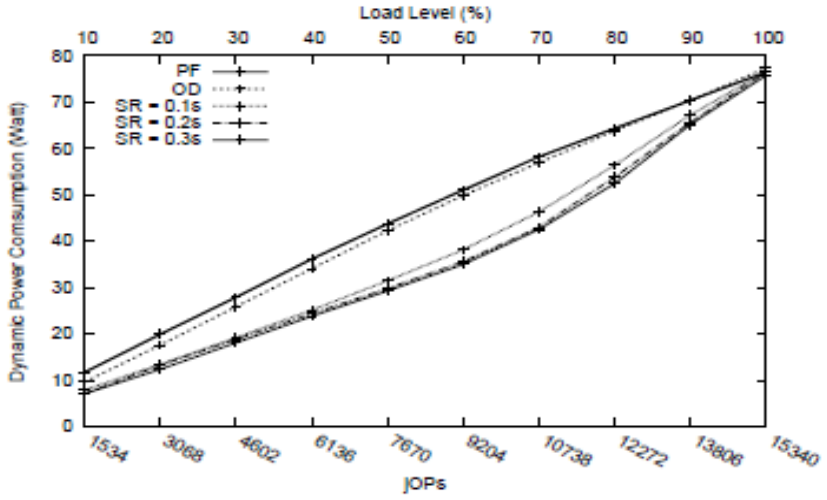

Fig. 3. Ryzen Dynamic Power Consumption with DFS

On Phenom, OD achieves the maximum power reduction of 5.3Watt at 10% load level and the power reduction gradually diminishes as the load level goes up (Fig. 2). SR = 0.1s further reduces the power consumption around 3Watt for the load level range of 10 to 50%, but SR ≥ 0.2S has little advantage on power reduction. We can see several differences in DFS behavior on Ryzen (Fig. 3). First, OD itself does not seem to be as effective as in the case of Phenom; the maximum reduc-tion is only 2.2Watt at 20% load level. However, SR = 0.1s significantly reduces the power consumption by up to 10.7Watt at 60%. Also, SR = 0.2s still reduces the power by up to 3.3Watt at 70% [3, 7].

Figs. 4 and 5 show the number of cores operating in each clock frequency at the load levels from 10% to 100%. As mentioned above, stretching SR from the default of 0.01s to 0.1s is effective in power reduction. In Fig. 4, we can see this phenomena most typically in the load level of 20%. While the number cores in the lowest frequency (0.8GHz) is decreased from 1.3 to 0.3, the number of cores in 1.55GHz is increased from 3.2 to 5.0. Similar behavior can be seen around 50% load level, where the number of 2.2GHz cores increases more than the decrease of the number of 1.55GHz cores.

Fig. 5 shows the clock frequency distribution of Ryzen. First, in relatively low load levels, most cores operate in the lowest frequency of 1.55GHz, which implies not much space for optimization with the DFS parameter. From the observation of the Phenom case in Fig. 4, if we had another lower frequency option (say, 1GHz), some of cores operating at 1.55GHz should migrate to that frequency, resulting in further power reduction. As seen in Fig. 3, the maximum power reduction was achieved around the load level of 60% and it was because more number of cores operate at 2.8GHz, replacing cores running at both 3.2 and 1.55GHz.

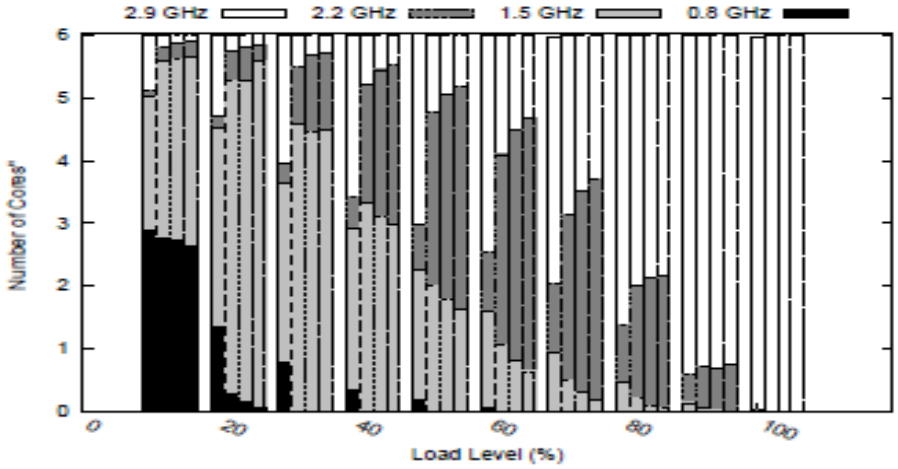

Fig. 4. Phenom Core Clock Frequency Distribution. For each load level, bars represent the values for OD (SR = 0.01s), 0.1s, 0.2s and 0.3s (left to right)

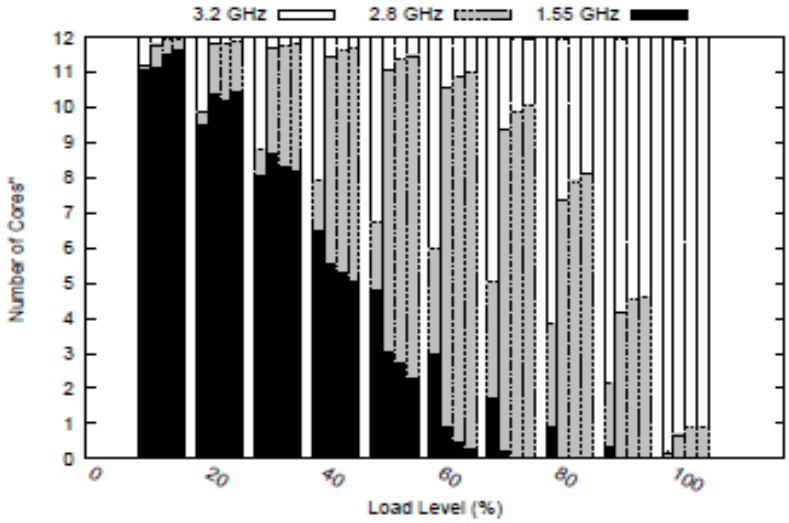

Fig. 5. Ryzen Core Clock Frequency Distribution. For each load level, bars represent the values for OD (SR = 0.01s), 0.1s, 0.2s and 0.3s (left to right)

4.4 Energy Efficiency

As seen in the previous section, the effectiveness of DFS differs not only between CPUs but also by the load levels. In addition, while DFS reduces the power consumption, it comes with the response time penalty (meaning lower critical-jOPS metrics). Figs. 6 and 7 plot the power consumption normalized by the throughput (which in turn indicates the relative energy per transaction). Please note that the data points in these figures include the load levels in 10% increments (corresponding to Figs. 2 and 3) and also the SLA points of response times from 10ms to 100ms (represented as dots). For example, + symbols in these graphs show the maximum jOPS for each DFS option under the response time constraint of 10ms and the normalized power consumption at these transaction throughputs.

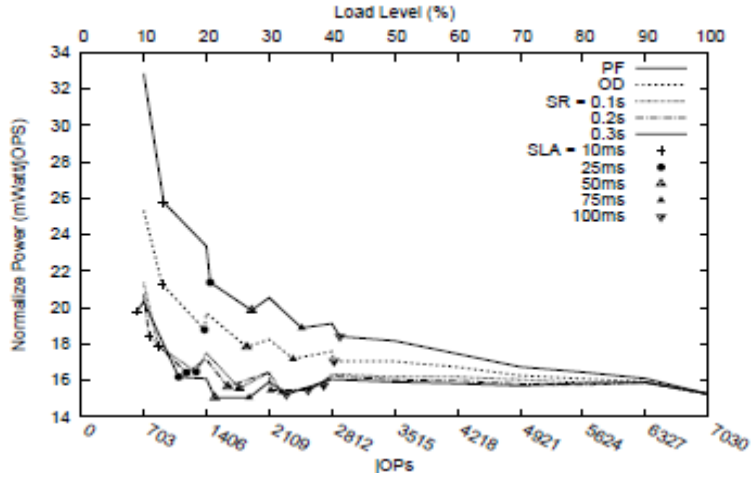

Fig. 6. Phenom Normalized Power Consumption

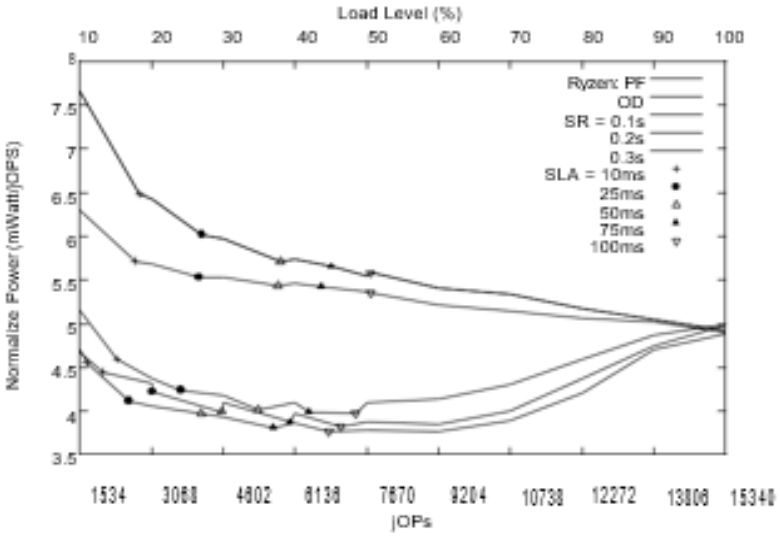

Fig. 7. Ryzen Normalized Power Consumption

On Phenom, the efficiency significantly improves for the load level range of 10% to 20%. From there, the efficiency still improves up to 40% at a lower rate. For SR ≥ 0.1S, the normalized power is almost constant at around 10mWatt per jOPS for the load level range of 40% to 90%. As we saw in Fig. 2, the improvements between P F → OD and OD → SR = 0.1S are similar, but graphs for SR ≥ 0.1S are mostly overlapped. On Ryzen, as we also saw in Fig. 3, the improvements for OD → SR = 0.1S is much larger than for P F → OD. A remarkable difference against Phenom is that the efficiency of SR ≥ 0.1 gets worse for the load level of 50% and beyond.

From the SLA points plotted in Figs. 6 and 7, we can see the trade-off between the energy efficiency and the performance penalty. From PF to OD and then to SR = 0.1s, the SLA points of the same response time are brought down almost vertically. However, beyond SR = 0.1s, SLA points are moved to left, meaning that the performance degradation due to the (overly) stretched sampling period for the DFS.

Fig. 8 shows the relative energy efficiency of Ryzen against Phenom (left) and Ryzen with SMT against without SMT (right). These bars indicate the ratios of the transactions processed by these two pairs with the same amount of energy. In all cases, DFS with sampling period of 0.1s is used. Ryzen achieves ×3 (max-jOPS) and ≈ ×3.9 (critical-jOPS) relative efficiency against Phenom. These ratios are larger than corre-sponding pairs in Fig. 1, showing that Ryzen not only improves the performance but also reduces the power consumption. The ratios of pairs are relatively small (≤ 4%), indicating that the SMT achieves the performance improvements in Fig. 1 with similar energy per transaction

5. Conclusion and future work

In this paper, we presented an energy efficiency study of Ryzen processor by comparing it to Phenom II processor under jbb15 workload. At the maximum throughput, it achieved more than ×2 performance and ×3 energy efficiency improvements. With the response time constraints, the improvements were around ×2.9 (performance) and ×3.9 (efficiency).

DFS and its parameter (sampling period) tuning were quite effective in reducing the power consumption of Ryzen running jbb15. However, we also found that there might be a room for further power reduction by adding a lower clock frequency than the current design (1.55GHz). We first need to learn how the current design choice on the Ryzen's DFS was made by AMD. With the current frequency options, further improvement could be obtained with the core

off-lining: for the feasibility study of core off-lining, we need to estimate the time to flushing the dirty cache lines when turning off cores and the effect of additional cache misses in the case resuming cores.

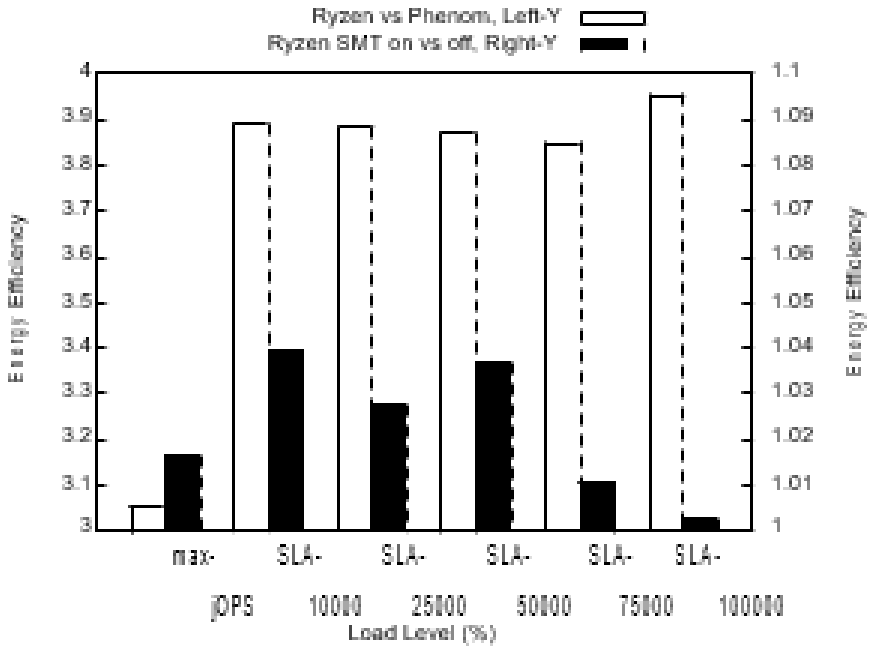

Fig. 8. Energy Efficiency Comparison. For all jbb15 metrics, left bars compare Phenom vs Ryzen and right bars compare Ryzen with SMT and without SMT. For both processors, DFS with sampling rate is used from 0.1 to 7.

One of the findings in this work was that the performance improvement by simultaneous multi-threading was relatively low (≈20%). It might be because that the instruction level parallelism of jbb15 was so high that almost all execution units were used and not much room for the thread level parallelism was remained. Analysis of this phenomena, for example, by means of the performance counters and tools like, is another topic of further investigation.

Reference:

[1] Hitoshi Oi and Sho Niboshi. Power Efficiency Study Using SPECjEnterprise2010. IEEE Systems Journal, Vol. 11, Issue 3, pp18670–1876, September 2017. DOI: 10.1109/JSYST.2015.2471798.

[2] Hitoshi Oi. Effectiveness of DFS Tuning on Java Server Workload. Journal of Circuits, Systems, and Computers. vol. 25, issue 01, January 2016.

[3] SPECjbb R 2015. https://www.spec.org/jbb2015/.

[4] Software Optimization Guide for the AMD Family 10h and 12h Processors. Rev.3.13, Feb. 2011, Advanced Micro Devices.
[5] The Zen Core Architecture. https://www.amd.com/en/technologies/.
[6] Michael Clark. A New, High Performance x86 Core Design from AMD. Hot Chips 28, Aug. 2016.
[7] SPEC Fair Use Rule. https://www.spec.org/fairuse.html.