OLTP On A Server-grade ARM: Power, Throughput and Latency Comparison

Utku Sirin* utku.sirin@epfl.ch
Raja Appuswamy* raja.appuswamy@epfl.ch
Anastasia Ailamaki*‡ anastasia.ailamaki@epfl.ch
*École Polytechnique Fédérale de Lausanne
‡RAW Labs SA

ABSTRACT

Although scaling out of low-power cores is an alternative to power-hungry Intel Xeon processors for reducing the power overheads, they have proven inadequate for complex, non-parallelizable workloads. On the other hand, by the introduction of the 64-bit ARMv8 architecture, traditionally low power ARM processors have become powerful enough to run computationally intensive server-class applications.

In this study, we compare a high-performance Intel x86 processor with a commercial implementation of the ARM Cortex-A57. We measure the power used, throughput delivered and latency quantified when running OLTP workloads. Our results show that the ARM processor consumes 3 to 15 times less power than the x86, while penalizing OLTP throughput by a much lower factor (1.7 to 3). As a result, the significant power savings deliver up to 9 times higher energy efficiency. The x86's heavily optimized power-hungry micro-architectural structures contribute to throughput only marginally. As a result, the x86 wastes power when utilization is low, while lightweight ARM processor consumes only as much power as it is utilized, achieving energy proportionality. On the other hand, ARM's quantified latency can be up to 11x higher than x86 towards to the tail of latency distribution, making x86 more suitable for certain type of service-level agreements.

1. INTRODUCTION

Intel's high performance heavily optimized Xeon processors are famous for their high power overheads [25]. This is mainly due to Intel's high base power consumption, i.e., the large amount of power the Intel processor consumes even if it is running only a single-threaded application. This large base power consumption is only compensated at very high utilization levels when all the available cores are actively running jobs. However, it has been shown that the server and data center applications mostly operate at between 10 and 50% of their maximum utilization levels [7], leaving the power-hungry heavily optimized Intel Xeon processors largely under-utilized, and therefore energy-inefficient [1]. Based on this fact, Barroso and Hölzle [7] propose energy-proportional computing as a primary design objective. Energy-proportional computing implies to consume power proportionally to the delivered throughput. Thereby, the servers consume only as much power as they utilize the machine, eliminating wastage during the low utilization periods.

Scaling out Intel's low-power Atom cores is a promising alternative for energy proportionality and energy-efficient cluster designs [6, 21]. While they are shown to deliver substantial performance and achieve high energy efficiency for computationally simple parallelizable workloads, they are largely inadequate for complex non-parallelizable workloads, e.g., OLTP [17]. Moreover, for many latency-critical applications, such as OLTP, low-end cores are not enough to meet service-level agreements (SLAs) requiring low-latency guarantees. Dean and Barroso [9] identify that, even though all the single servers finish 99% of the requests with low-latencies, the tail of 1% of latency distribution with high-latencies can cause a significant bottleneck in the overall system for high fan-out parallel requests. They refer to this problem as tail latency problem. Tail latency poses a serious challenge for energy-efficient computing since SLAs requiring low tail latency guarantees render Xeon-like heavily-optimized, power-hungry processors as the only choice.

On the other hand, by the introduction of 64-bit ARMv8 architecture [19], traditionally low-power low-end ARM Cortex-A* cores gradually advance to server-grade processors that are powerful enough to run computationally intensive server-class applications. Similar to Xeon, they integrate powerful micro-architectural structures, e.g., wide-issue out-of-order execution, deep pipeline stages and advanced branch prediction unit [2]. This renders ARM as a middle choice between heavily-optimized, power-hungry Xeon-like and low-power, low-end Atom-like processors.

In this study, we compare power, throughput and latency characteristics of a traditionally high performance Intel Xeon processor and a commercial implementation of ARM's recent server-grade processor Cortex-A57 when running OLTP, a complex server-class workload. Our goal is to understand if ARM's server-grade processor delivers satisfactorily high performance compared to Xeon while maintaining low power consumption characteristics. To do that, we firstly analyze the power and throughput characteristics of ARM and Xeon processors at different utilization levels. Then, we investigate the energy proportionality of the processors

[1]Here, we define utilization loosely as a measure of the application performance, i.e., throughput for OLTP, as in [7].

DaMoN'16, June 27, 2016, San Francisco, CA, USA

DOI: http://dx.doi.org/10.1145/2933349.2933359

Table 1: Server parameters

Processor	Intel(R) Xeon(R)	ARM Cortex-A57
#Sockets	2 (only one socket is active)	1
#Cores per Socket	8	8
Issue width	4	4
Clock Speed	2.00GHz	2.00GHz
Main memory	256GB	16GB
L1I / L1D	32KB / 32KB (per core) 16-cycle miss latency	32KB / 32KB (per core) 15-cycle miss latency
L2	256KB (per core) 40-cycle miss latency	256KB (per pair of cores) 80-cycle miss latency
LLC	20MB (shared) 170-cycle miss latency	8MB (shared) 175-cycle miss latency

based on how their energy efficiency varies as a function of utilization. Then, we analyze the impact of two complex micro-architectural structures of the Xeon processor, Hyper-Threading and Turbo Boost, on Xeon's energy-efficiency. Lastly, we analyze the latency characteristics of both processors at 50th, 99th, 99.9th, 99.99th, 99.999th percentiles. We use a traditional disk-based OLTP system, Shore-MT [16], and a new-generation main-memory optimized OLTP system, Silo [26], and run a community standard TPC-C benchmark [24].

Our analysis demonstrates the following:

- The ARM processor achieves up to 9 times higher energy efficiency than the Xeon processor by delivering only 1.7 to 3 times lower throughput with 3 to 15 times less power consumption.
- ARM's power consumption is commensurate to its utilization, and therefore achieves energy proportionality.
- Complex micro-architectural structures, such as Hyper-Threading and Turbo Boost, provide only marginal gains in energy efficiency for the Xeon processor.
- ARM's tail latency is 2 to 11 times higher than Xeon, making Xeon more suitable for high fan-out latency-critical workloads.

The rest of the paper is organized as follows. Section 2 discusses the related work, and Section 3 describes the experimental setup and methodology. Section 4 compares the Xeon and ARM processors in terms of their power and throughput characteristics. Section 5 examines energy proportionality of ARM and Xeon based on throughput, power and energy efficiency scalings. Section 6 discusses the advantages and disadvantages of Hyper-Threading and Turbo Boost technologies for Xeon in terms of energy efficiency. Section 7 examines tail latency of both processors. Lastly, Section 8 concludes our analysis.

2. RELATED WORK

There is a large body of work on scaling out low-power, low-end microprocessors. Andersen et al. [6] present FAWN, a scaling out solution of low-power microprocessors for computationally simple data-intensive workloads. Szalay et al. [23] presents another scaling out solution of low-power cores where each node in the cluster is an Amdahl-balanced blade server coupling variants of Intel Atom processor with energy-efficient solid-state drives. Harizopoulos and Papadimitriou [14] describe a cluster architecture of recycled smartphones suggesting to turn unused smartphones into micro-data centers for even better energy efficiency than scaling out of low-power cores. Schall and Härder [21] present an energy-proportional cluster architecture that can dynamically adjusts the cluster size based on the workload demands by using low-power microprocessors. While using low-power "wimpy" nodes indeed is a flexible solution for achieving energy proportionality, our analysis shows that a single server using ARM's high-performance server-grade processor can also be energy proportional, and therefore is a promising candidate for an alternative solution to scaling out low-power cores.

Although these studies demonstrate the energy and cost efficiency of low-power processors, Lang et al. [17] shows how scaling out low-power cores is less cost-efficient than Xeon-like high-performance single-server solutions for complex workloads. The reason is that the performance does not scale up linearly to the scale out properties of the application. Our analysis shows that ARM's server-grade processors can be a promising alternative to replace energy-inefficient high-end Xeon servers by energy-efficient high-end ARM servers. As the prices of ARM's server-grade processors are not publicly available, its cost efficiency is still ambiguous to us. However, coming from the low-power, low-cost mobile computing market, it is highly-likely to be much more cost-efficient than the high-end Xeon servers [20].

Tsirogiannis et al. [25] examine the energy efficiency of a single-node database server running a high-performance Intel Xeon processor concluding that the most energy-efficient configuration for a database server is the highest performing one. By contrast, our analysis of ARM shows that ARM processors, as a unit of computing subsystem, has constant energy efficiency regardless the performance. This contributes to the energy proportionality of the server, and renders ARM a good candidate for energy-efficient database server design. Furthermore, as CPU is the main source of power consumption in database servers (85% of active power consumption [25]), ARM's energy proportionality suggests re-considering energy efficiency of database servers when running on ARM, pointing towards a new line of research.

Rajovic et al. [20] argue that highest-volume computing platforms are taking over the high-performance market space simply because they are substantially cheaper than

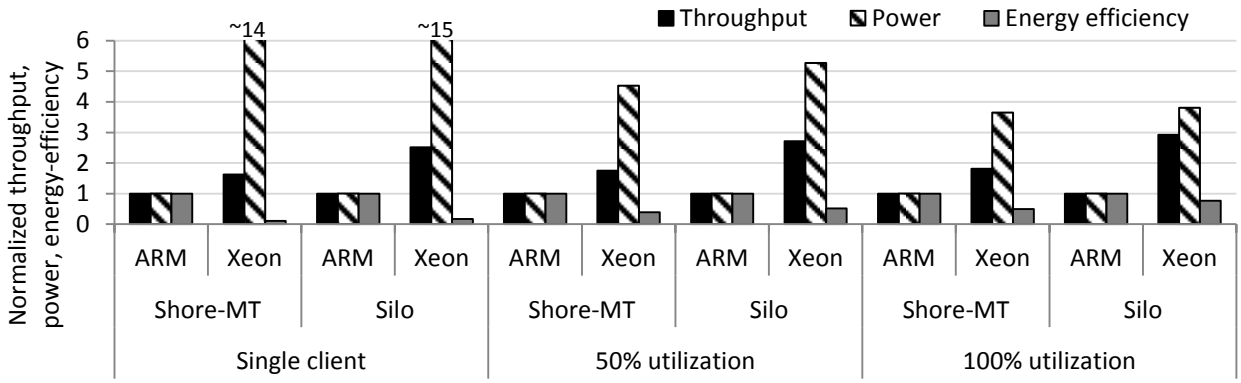

Figure 1: Normalized power, throughput and energy-efficiency values for different levels of utilization.

low-volume, high-performance servers. They argue that, as commodity desktop processors took over high-performance specialized computing platforms during the mid-2000s, mobile computing platforms are similarly likely to take over the high-performance Intel Xeon server market in the following decade. An example of this shift is BlueGene, a family of supercomputers using low-power embedded cores for high-performance computing [15]. Similarly, Calxeda's EnergyCore ECX-1000 [12] and Applied Micro's X-Gene [11] are server-class system-on-chips using ARM Cortex-A9 and ARM Cortex-A57 processors. Our results corroborate this trend. We show that a server-grade ARM processor can indeed serve as a promising alternative for server-class applications as it delivers substantial level of performance. Further improvements on ARM's micro-architecture, such as a better branch prediction unit, or an advanced prefetcher can potentially make the ARM an even better candidate for high-performance server market [5].

3. SETUP AND METHODOLOGY

In this section, we present the experimental setup and methodology for our study.

Hardware: We use a modern commodity Intel Xeon platform implementing Ivy Bridge micro-architecture, and a commercial implementation of ARM Cortex-A57 micro-architecture. Table 1 shows the details of the micro-architectural features of both platforms. We disable one of the two sockets of the Xeon server by using the available operating system mechanisms to have the equal number of active cores for both processors. We use Linux's `lmbench` to estimate the cache access latencies both for the ARM and Xeon processors.

Except for the experiments in Section 6 where we investigate the effect of Hyper-Threading (HT) and Turbo Boost (TB) technologies on energy efficiency, we disable the HT and TB for the Xeon server.

OS: We run the experiments using RHEL 6.5 with Linux kernel version 2.6.32 for the Xeon server, and using Ubuntu 14.04.3 with Linux kernel version 3.13.0-65-generic for the ARM server.

OLTP systems: We use one traditional disk-based OLTP system, Shore-MT [16], and one new-generation main-memory optimized OLTP system, Silo [26]. While running the OLTP systems, nothing else runs on the server, i.e., the software stack only contains the OLTP system.

Benchmarks: We run community standard TPC benchmark, TPC-C [24], with a database of size 5GB for all of our experiments.

Measurements: We populate the databases from scratch before each experiment and the data remains memory-resident throughout the experiments. We use memory-mapped I/O for log flushing. Both the worker threads executing the transactions and the client threads generating the transactions run on the same machine. We first start the server process, populate the database, and then start the experiment by simultaneously launching all clients that generate and submit transactional requests to the database server. For every client submitting transactional requests there is one OLTP worker thread satisfying the requests. Less than 1% of the execution time is spent for the client threads, whereas the remaining 99% of the time is spent for the server threads. We repeat every experiment three times and report the average result.

We measure the power by using ZigBee power plug (ZPLUG) and CleoBee software [8]. ZPLUG measures the wall socket power, i.e., the total power consumption of the server, whereas CleoBee provides the software interface to collect the power consumption numbers. We plug the power cable of the server into ZPLUG, and ZPLUG into the wall socket. To measure the power consumed by the processor, for each particular experiment, we measure the wall socket power when the server is idle, when the server is running the particular workload with the particular configuration, and use the difference as the power consumption of the processor. Energy efficiency is calculated by dividing the delivered throughput to the power consumption of the processor as in [25].

As shown by [25], 85% of the active power consumption of a database server belongs to the CPUs whereas the remaining 15% is equally shared by the SSDs and HDDs. DRAM accesses do not contribute to the active power consumption. Since in our experiments the data remains memory resident, and we use memory-mapped I/O, we attribute the entire active power consumption to the CPUs. On the other hand, the idle power consumption accounts $\sim$ 54% and 65% of the total power consumption of the Intel and ARM servers

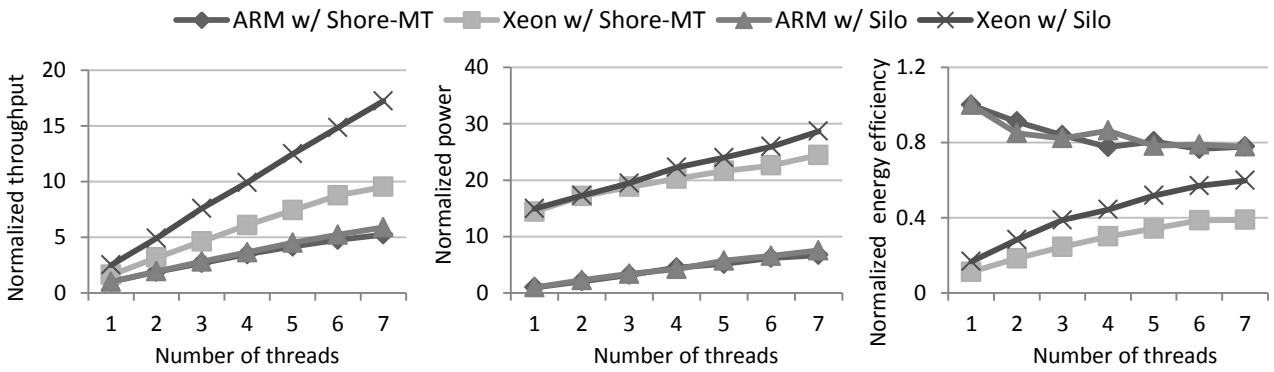

Figure 2: Throughput (left), power (middle), and energy efficiency curves (right) for energy proportionality experiments. Normalized to ARM's single-threaded results.

when all the eight cores are active. The idle power consumption includes the power consumed by CPUs when they are idle, DRAM DIMMs, SSDs/HDDs and system board. In this study, we only focus on the active power consumption due to the limitation of measuring the ARM processor's idle power consumption.

We use the term utilization as a measure of the application performance, i.e., throughput for OLTP, as in [7]. Since the OLTP systems we use scale well across multiple cores, throughput increases as the number of clients issuing transactions to the system increases. Therefore, utilization is proportional to the delivered throughput. For an 8-core processor, 50% utilization refers to executing four, and 100% utilization refers to executing seven clients issuing transactional requests. We spare one core for a background processing thread, e.g., logging for Shore-MT and garbage collecting for Silo.

4. POWER VS THROUGHPUT

In this section, we profile power and throughput characteristics of ARM and Xeon servers at different utilization levels. Figure 1 shows results normalized to ARM for three different configurations: single client, 50% utilization and 100% utilization.

We observe that Intel Xeon's throughput is only 1.7x higher than that of ARM for all the three configurations when running Shore-MT. Having 15x lower power consumption, this brings 9x higher energy efficiency for the ARM processor for the single-client configuration. For 50% and 100% utilization levels, ARM consumes 4.5x and 3.3x lower power, and achieves 2.5x and 2x higher energy-efficiency. Consequently, ARM's energy-efficient server-grade processor is indeed a powerful alternative to high-performance energy-inefficient Intel Xeon processors providing sufficient level of performance while maintaining low power consumption characteristics.

When running Silo, Intel Xeon delivers 2.5x higher throughput under single-client execution and 50% utilization, while consuming 15x and 5x higher power. This brings 6x and 2x higher energy efficiency for the ARM processor, once again showing that ARM delivers substantial level of performance while consuming dramatically lower power. For 100% utilization level, on the other hand, the difference in throughput increases to 3x, and the difference in power decreases to 4x, reducing the energy efficiency gain of the ARM processor to 25%. Silo is a new-generation main-memory optimized OLTP system that can deliver 15 to 30x higher throughput than the traditional disk-based OLTP system Shore-MT. While Shore-MT possesses significant amount of overheads, e.g., a large buffer pool component and a disk-oriented index structure [13], Silo eliminates most of the traditional overheads by using optimized data structures and protocols, e.g., Masstree, a scalable main-memory optimized index structure, and lightweight optimistic concurrency control mechanisms. Therefore, Silo can utilize better Xeon's heavily optimized micro-architectural structures, delivering higher normalized throughput than Shore-MT, and achieving closer energy efficiency to the ARM processor.

Overall, energy efficiency of the ARM processor is always higher than that of the Intel Xeon processor, having the difference ranging from 25% to 9x. We observe energy efficiency gains are higher at lower utilization levels. Assuming that servers running latency-critical workloads, such as OLTP, mostly operate at between 10 and 50% utilization levels [7], Intel Xeon servers are highly likely to waste large amount of power. Moreover, a big portion of the OLTP market is dominated by traditional disk-based OLTP systems [18]. Having at least 2x higher energy efficiency when running the disk-based Shore-MT, ARM's server-grade processors appear to be a promising choice for energy-efficient high-end server design.

5. ENERGY PROPORTIONALITY

In this section, we examine energy proportionality of the ARM and Xeon processors. To do that, we compare the throughput, power and energy efficiency curves of ARM and Xeon processors as we scale the OLTP system by increasing the number of worker threads from one to seven.

Figures 2 shows results normalized to the single-threaded ARM, separately for Shore-MT and Silo. Our first observation is that both Shore-MT and Silo scales linearly both on ARM and Xeon (Figure 2 (left)). This shows that the OLTP systems we use do not have inherent scalability bottlenecks.

Our second observation is that, ARM and Xeon follow similar trends in their power consumption, though Xeon and ARM have a constant raw power consumption difference (Figure 2 (middle)). This shows that the power overhead of activating an additional core is almost the same for both processors, and majority of Xeon's power consumption is due to the optimized micro-architectural structures rather

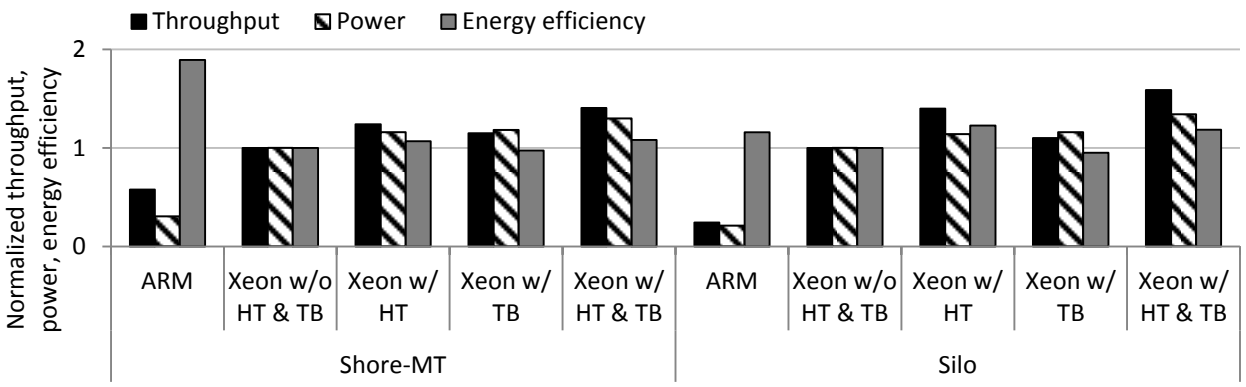

Figure 3: Effect of Hyper-Threading (HT) and Turbo Boost (TB) on energy efficiency of the Intel Xeon processor.

than activating the individual cores.

Lastly, as a net result of throughput vs power, ARM and Xeon exhibit different energy efficiency behavior as shown by Figure 2 (right). Under ARM, the increase in power consumption offsets the throughput gain resulting in almost constant energy efficiency curve. The energy efficiency decreases only 20% as the number of threads increases from one to seven. Under Xeon, however, the increase in throughput completely overshadows the fractional increase in power resulting in better energy efficiency at higher thread counts. The energy efficiency of Xeon is 3.5x higher when running seven threads compared to running one thread. Therefore, we observe that the ARM processor indeed achieves energy proportionality by consuming almost linear amount of power to its utilization. The throughput (Figure 2 (left)) and power (Figure 2 (middle)) curves of ARM go almost hand-in-hand as the number of thread increases. This results in constant energy efficiency curve regardless the utilization, providing large power-savings for non-peak, and lower-level utilizations. The Xeon processor, on the other hand, is far from being energy-proportional since its large power consumption is only compensated at peak load. This renders Xeon energy-inefficient especially for lower-level utilizations.

In our analysis, we exclude the idle power consumption of the processors. Adding the idle power consumption would introduce a base power consumption to the ARM, and increase the base power consumption of the Xeon. This would deteriorate the energy proportionality of both processors. However, ARM would still provide a significantly better energy proportionality than Xeon as Xeon already suffers from active base power consumption. Moreover, coming from low-power, low-cost mobile computing market, ARM is highly likely to have a low idle power consumption.

Xeon's energy efficiency seems to converging to a constant value as the the number of threads increases. This raises the question whether, for very large number of cores, Xeon would approximate the energy proportionality. Increasing the number of cores would require increasing the number of sockets. This would, however, increase the base power consumption, and therefore would leave Xeon energy-inefficient even for very large number of cores.

Energy proportionality of ARM and Xeon processors can vary for different types of workloads. For example, OLTP workloads mostly access the data randomly, and therefore largely under-utilize the shared last-level cache (LLC) [10]. Hence, concurrently running multiple threads on a shared LLC does not significantly affect the scalability pattern of the delivered throughput. A different workload having high data locality, however, might efficiently use the LLC, and therefore might have a different scalability pattern than OLTP workloads in terms of the delivered throughput, and therefore energy proportionality.

6. ACCELERATION FEATURES

The Intel Xeon processor we profile offers two acceleration features, Hyper-Threading (HT) and Turbo Boost (TB) [3, 4]. HT, also called simultaneous multi-threading, enables CPU cores to concurrently run multiple threads on each single core. This allows exploiting thread-level parallelism of the workload. Enabling HT provides the Xeon to have logically 2x more number of cores. TB allows CPU cores to automatically run at higher frequency rates based on dynamic load of the server. Both HT and TB technologies improve the throughput delivered by the Xeon processor, and also increases the power consumption.

Until this point, all of the experiments which were performed on Xeon have had HT and TB technologies disabled. In this section, we enable these two technologies for the Xeon processor and analyze their effect on Xeon's throughput, power and energy efficiency characteristics. Our goal is to understand how much complex micro-architectural structures of Xeon are useful for throughput, power and energy efficiency improvements. Figure 3 shows results normalized to Xeon without HT and TB, separately for Shore-MT and Silo. We use 100% utilization level. 100% utilization when having HT enabled refers to running 15 clients as Intel's HT provides 2 threads per core, and we spare one thread for background processing. We compare the configurations where HT is alone, TB is alone and HT and TB are together enabled.

We observe that HT provides 24% and 40% higher throughput for Shore-MT and Silo, respectively, which are far from the ideal improvement of 2x. Therefore, HT does not provide substantial improvement on throughput when running OLTP workloads. High-level of contention, pollution of private per-core and shared caches could be some of the reasons for that. Observe that Silo exploits HT better than Shore-MT as we argued in Sections 4. On the other hand, we

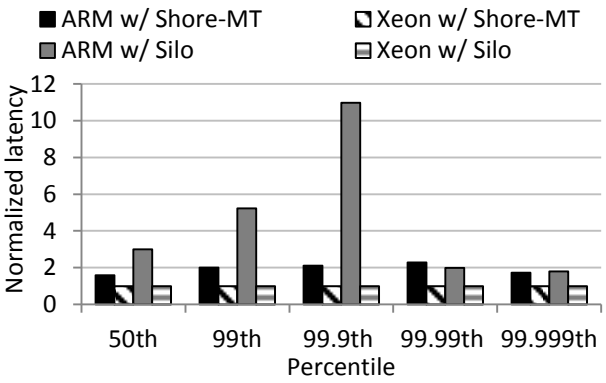

Figure 4: Tail latency.

observe that enabling HT increases power consumption by 15% for both OLTP systems. This results in only small energy efficiency gains: 6% and 23% for Shore-MT and Silo, respectively.

On the other hand, TB increases the throughput 15% for both OLTP systems, while causing 18% more power consumption, on average. Hence, enabling TB reduces the energy efficiency rather than improving. When we enable HT and TB together, both throughput and power increase, resulting in similar energy efficiency to having HT enabled alone.

Lastly, we observe that ARM's energy efficiency is significantly higher than all of the other configurations when running Shore-MT, whereas the energy efficiency gets closer when running Silo. This is once again due to that Silo is a more optimized system than Shore-MT, and is able to exploit micro-architectural structures better than Shore-MT. Being a main-memory optimized system, Silo suffers more from data cache misses than instruction cache misses [22]. Since data cache misses can be overlapped whereas instruction misses lock the pipeline, Silo provides more opportunities for Xeon to exploit HT. Therefore, Xeon having HT enabled delivers higher throughput and achieves better energy efficiency when running Silo than when running Shore-MT.

Overall, both HT and TB technologies provide modest improvements in throughput while increasing the total power consumption. This results in marginal energy efficiency gains.

7. TAIL LATENCY

In this section, we compare the tail latency of the Xeon and ARM at the 50th, 99th, 99.9th, 99.99th and 99.999th percentiles under 100% utilization.

Figure 4 shows results normalized to Xeon. We observe that, when running Shore-MT, ARM's latency values are only 2x higher than that of Xeon for all the percentiles. While the raw latency values increase towards the tail of latency distribution for both processors (not shown), the rates of the increase are more or less the same for both Xeon and ARM.

On the other hand, when running Silo, ARM's latency values are 3x, 5x and 11x higher than that of Xeon for 50th, 99th and 99.9th percentiles, respectively. Therefore, Xeon processor provides significantly lower latencies when running Silo towards the tail of latency distribution at 99.9th percentile. This represents a scenario where, despite its large power overhead, Xeon processors might be preferable against to ARM due to service-level agreements (SLAs) requiring low-latency guarantees at the 99.9th percentile. However, after 99.9th percentile, we observe that the difference decreases to 2x, similar to Shore-MT. This is because, towards to the very end of the tail, both processors are dominated by high-valued outliers. Hence, for SLAs requiring low latency guarantees beyond the 99.9th percentile of the tail, Xeon and ARM processors achieve close quantified latencies.

There can be several reasons for ARM's higher tail latency. The first reason could be that Xeon's more optimized out-of-order execution engine. This includes a more optimized instruction scheduler in terms of resolving the data dependencies, larger load/store buffers as well as a larger re-order buffer. Another reason could be Xeon's more optimized instruction fetch unit. This includes an advanced instruction prefetcher and a loop stream detector. Lastly, it can be Xeon's more sophisticated branch prediction algorithm [1].

8. CONCLUSION

In this paper, we compare power, throughput and latency characteristics of an high-performance Intel Xeon processor with a commercial implementation of ARM's recent server-grade processor, ARM Cortex-A57, when running OLTP workloads. We observe that, while ARM delivers 1.7 to 3 times lower throughput, it consumes 4 to 15 times lower power, achieving up to 9 times higher energy efficiency than Xeon. We observe that Hyper-Threading and Turbo Boost technologies contribute marginally to energy efficiency of the Xeon processor. Whereas the Xeon processor is far from being energy-proportional due to its large power overhead, the ARM processor achieves energy proportionality by consuming linear amount of power to its utilization. On the other hand, ARM's tail latency can be up to 11x higher than Xeon, making Xeon more suitable for certain type of service-level agreements for high fan-out latency-critical workloads.

Therefore, traditionally low power server-grade ARM processors are indeed a promising alternative for server-class applications by delivering substantial level of performance while maintaining low power consumption characteristics. Its energy proportionality renders it more suitable for alternative energy-efficient server designs.

9. ACKNOWLEDGMENTS

We thank the DaMoN reviewers, our colleagues at EPFL (Danica Porobic, Matthaios Alexandros Olma, Alexandros Daglis), and our collaborators at Huawei (Anthony Iliopoulos, Shay Goikhman and Eliezer Levy) for their comments. The research leading to these results is funded by the Swiss National Science Foundation (Project 200021_146407/1) and by an industrial-academic collaboration with Huawei's Central Software Institute (CSI).

10. REFERENCES

[1] Anand Lal Shimpi. ARM Challenging Intel in the Server Market: An Overview. *AnandTech*. N.p., 16 Dec. 2014. Web. 3 Apr. 2016.

[2] Anand Lal Shimpi. ARM's Cortex A57 and Cortex A53: The First 64-bit ARMv8 CPU Cores. *AnandTech*. N.p., 30 Oct. 2012. Web. 3 Apr. 2016.

[3] Anand Lal Shimpi. Homework: How Turbo Mode Works - Intel's Core I7 870 & I5 750, Lynnfield:

Harder, Better, Faster Stronger. *AnandTech*. N.p., 8 Sept. 2009. Web. 3 Apr. 2016.

[4] Anand Lal Shimpi. Intel's Hyper-Threading Technology: Free Performance? *AnandTech*. N.p., 14 Jan. 2002. Web. 3 Apr. 2016.

[5] AnandTech. ARM Reveals Cortex-A72 Architecture Details. *AnandTech*. N.p., 23 Apr. 2015. Web. 3 Apr. 2016.

[6] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan. FAWN: A Fast Array of Wimpy Nodes. In *SOSP*, pages 1–14, 2009.

[7] L. A. Barroso and U. Hölzle. The Case for Energy-Proportional Computing. *Computer*, 40, 2007.

[8] CLEODE. CLEODE: easy automation and wireless. http://cleode.fr/en/produits.php?page=cleobee.

[9] J. Dean and L. A. Barroso. The tail at scale. *Communications of the ACM*, 56:74–80, 2013.

[10] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A. D. Popescu, A. Ailamaki, and B. Falsafi. Clearing the Clouds: A Study of Emerging Scale-out Workloads on Modern Hardware. In *ASPLOS*, pages 37–48, 2012.

[11] A. Y. G. Singh, G. Favor. AppliedMicro X-Gene2. In *HotChips*, 2014.

[12] Gina Longoria. Calxeda EnergyCore-Based Servers Now Available. *ARM SERVERS*. N.p., 30 Oct. 2012. Web. 3 Apr. 2016.

[13] S. Harizopoulos, D. J. Abadi, S. Madden, and M. Stonebraker. OLTP Through the Looking Glass, and What We Found There. In *SIGMOD*, pages 981–992, 2008.

[14] S. Harizopoulos and S. Papadimitriou. A Case for Micro-cellstores: Energy-efficient Data Management on Recycled Smartphones. In *DaMoN*, 2011.

[15] IBM Systems and Technology. IBM SYStem Blue Gene/Q Data Sheet, 2011.

[16] R. Johnson, I. Pandis, N. Hardavellas, A. Ailamaki, and B. Falsafi. Shore-MT: a Scalable Storage Manager for the Multicore Era. In *EDBT*, pages 24–35, 2009.

[17] W. Lang, J. M. Patel, and S. Shankar. Wimpy Node Clusters: What about Non-wimpy Workloads? In *DaMoN*, pages 47–55, 2010.

[18] MySQL. MySQL Customers. https://www.mysql.com/customers/.

[19] R. Grisenthwaite. ARMv8 Technology Preview, 2011. http://www.arm.com/files/downloads/ARMv8_Architecture.pdf.

[20] N. Rajovic, P. M. Carpenter, I. Gelado, N. Puzovic, A. Ramirez, and M. Valero. Supercomputing with Commodity CPUs: Are Mobile SoCs Ready for HPC? In *SC*, pages 1–12, 2013.

[21] D. Schall and T. Härder. Energy-proportional Query Execution Using a Cluster of Wimpy Nodes. In *DaMoN*, pages 1–6, 2013.

[22] U. Sirin, P. Tozun, D. Porobic, and A. Ailamaki. Micro-architectural Analysis of In-memory OLTP. In *SIGMOD*, 2016.

[23] A. S. Szalay, G. C. Bell, H. H. Huang, A. Terzis, and A. White. Low-power Amdahl-balanced Blades for Data Intensive Computing. *SIGOPS Oper. Syst. Rev.*, 44(1):71–75, Mar. 2010.

[24] TPC. TPC Benchmark C Standard Specification, revision 5.11, 2010. http://www.tpc.org/tpcc.

[25] D. Tsirogiannis, S. Harizopoulos, and M. A. Shah. Analyzing the Energy Efficiency of a Database Server. In *SIGMOD*, pages 231–242, 2010.

[26] S. Tu, W. Zheng, E. Kohler, B. Liskov, and S. Madden. Speedy Transactions in Multicore In-memory Databases. In *SOSP*, pages 18–32, 2013.