Research on Multi-core Embedded Computer Architecture based on Cloud Computing

Yuwen Zheng

Shandong Women's University, Jinan, 250300, China

Keywords: Multi-core Computer Architecture, Cloud Computing, Embedded System, Queueing Network.

Abstract. Cloud computing is becoming one of the hottest research area in the computer science and technology community. With Moore's law supplying billions of transistors on-chip, embedded systems are undergoing a transition from single-core to multi-core to exploit this high transistor density for high performance. However, the optimal layout of these multiple cores along with the memory subsystem (caches and main memory) to satisfy power, area, and stringent real-time constraints is a challenging design endeavor. In this paper, we present a queueing theoretic and cloud computing based approach for modeling multi-core embedded systems that provides a quick and inexpensive performance evaluation both in terms of time and resources as compared to the development of multi-core simulators and running benchmarks on these simulators. We verify our queueing theoretic modeling approach by running SPLASH-2 benchmarks on the Super ESCalar simulator (SESC). Results reveal that our queueing theoretic model qualitatively evaluates multi-core architectures accurately with an average difference of 5.6% as compared to the architectures' evaluations from the SESC simulator. In the future, we plan to use some novel simulation techniques to modify the proposed framework.

Introduction

The Background Research

With Moore's law supplying billions of transistors on-chip, embedded systems are undergoing a paradigm shift from single-core to multi-core to exploit this high transistor density for high performance. This paradigm shift has led to the emergence of diverse multi-core embedded systems in a plethora of application domains. Many modern embedded systems integrate multiple cores (whether homogeneous or heterogeneous) on-chip to satisfy computing demand while maintaining design constraints (e.g., energy, power, performance, etc.). For example, a 3G mobile handset's signal processing requires 35–40 Giga operations per second (GOPS). Considering the limited energy of a mobile handset battery, these performance levels must be met with a power dissipation budget of approximately 1W, which translates to a performance efficiency of 25mW/GOP or 25pJ/operation for the 3G receiver [1]. These demands and competition power performance make challenging modern embedded system design. Increase customer forecast/demand functions, leading to the design of the embedded system complexity exponential rise. While industry focus is to increase the number of processor cores on the chip, to meet customer performance requirements, embedded system designers are faced with the new challenges of the processor core optimal layout, and memory subsystem (cache and memory), in order to meet the power, area and strict real-time constraints. Short listed time (time from product concept to market release) the design of the embedded system further challenges. Embedded systems architecture modeling helps to reduce time which make quick application-to-device mapping from determine an appropriate architecture as a set of the target application greatly reduces the time of embedded system design. To ensure timely completion of embedded system design have enough confidence in the product market, the design engineer must weigh between levels of abstraction of the system architecture model and implement precision.

We leverage for the first time, to the best of our knowledge, queueing network theory as an alternative approach for modeling multi-core embedded systems for performance analysis (though

queueing network models have been studied in the context of traditional computer systems [2-5]). Our queueing network model approach allows modeling the layout of processor cores (processor cores can be either homogeneous or heterogeneous) with caches of different capacities and configurations at different cache levels. Our modeling technique only requires a high-level workload characterization of an application (i.e., whether the application is processor-bound (requiring high processing resources), memory bound (requiring a large number of memory accesses), or mixed).

The Overview of Our Research

We present a novel, queueing theory-based modeling technique for evaluating multi-core embedded architectures that does not require architectural-level benchmark simulation. This modeling technique enables quick and inexpensive architectural evaluation, with respect to design time and resources, as compared to developing and/or using the existing multi-core simulators and running benchmarks on these simulators. Based on a preliminary evaluation using our models, architecture designers can run targeted benchmarks to further verify the performance characteristics of selected multi-core architectures. We also put forward a true benchmark probability method to quantify demand. Therefore, our modeling technology can provide performance evaluation and any calculation of load demand rather than simulation-driven architecture evaluation which can only provide specific benchmark performance results. We queue theory modeling method can be used for performance per watt per unit area and performance characteristics of multi-core embedded architecture, with different number of processor cores and the cache configuration, to provide a comparative analysis. Performance per watt per unit area and performance calculation is conducted by our approach through which we calculated different multicore chip area and power consumption of embedded system structure with different number of processor cores and the cache configuration.

Our queueing theoretic approach can be leveraged for early design space pruning by eliminating infeasible architectures in very early design stages, which reduces the number of lengthy architectural evaluations when running targeted benchmarks in later design stages. Specifically, our approach focuses on the qualitative comparison of architectures in the early design stage and not the quantitative comparison of architectures for different benchmarks. Our model is designed to operate using synthetic workloads that a designer can categorize for an expected behavior, such as processor memory-bound workloads, along with an estimate of the expected cache miss rates. The synthetic workloads preclude the need to obtain benchmark-specific statistics from an architecture level simulator. Furthermore, the cache miss rates are estimates, and thus are not required to be the exact miss rates for any specific benchmark. Our performance, strength and performance per watt results show that the multi-core embedded system structure, the use of Shared LLC, scalable and offer the best LLC performance per watt. However, sharing the company structure may introduce main memory response time and high throughput bottleneck cache miss rate. Using a mix of private and Shared the architecture of the LLC is a scalable, reduce main memory bottleneck at the cost of performance per watt. The architectures with private LLCs exhibit less scalability but do not introduce main memory bottlenecks at the expense of reduced performance per watt.

The Multi-Core Based Architecture Modelling

The Queueing Network Terminology

A queueing network consists of service centers (e.g., processor core, L1-I cache, L1-D cache, L2 cache, and main memory (MM)) and customers (e.g., jobs/tasks). A service center consists of one or more queues to hold jobs waiting for service. We use the term jobs instead of tasks (decomposed workload resulting from parallelizing a job) to be consistent with general queueing network terminology.

Our modeling approach is broadly applicable to multi-programmed workloads where multiple jobs run on the multi-core embedded architecture as well as for parallelized applications/jobs that run different tasks on the multi-core architectures. Arriving jobs enter the service center's queue and a scheduling/queueing discipline (e.g., first-come–first-served (FCFS), priority, round robin (RR), processor sharing (PS), etc.) selects the next job to be served when a service center becomes

idle. Queuing discipline is preemptive if a work can reach higher priority to suspend a lower priority service/execution work, otherwise no priority queue discipline. Non-preemptive queuing discipline first, work services into the order queue. Based on queuing disciplines can preemptive priority or not based on priority design work and services based on priority allocation. If the work not completed service time quantum, queue work placement in the subsequent service time quantum of the resume. After being serviced, a job either moves to another service center or leaves the network.

A queueing network is open if jobs arrive from an external source, spend time in the network, and then depart. If work can belong to different chain, network is more than a chain queuing network. Queuing network is an important class of the product form with the joint probability of queue size probability of network products for personal service center queue size. The queueing network performance metrics include response time, throughput, and utilization. The response time is the amount of time a job spends at the service center including the queueing delay (the amount of time a job waits in the queue) and the service time. The service time of a job depends on the amount of work (e.g., number of instructions) needed by that job. The throughput is defined as the number of jobs served per unit of time. Little's law governs the relationship between the number of jobs in the queueing network N and response time tr [6].

The Mathematical Modelling Steps

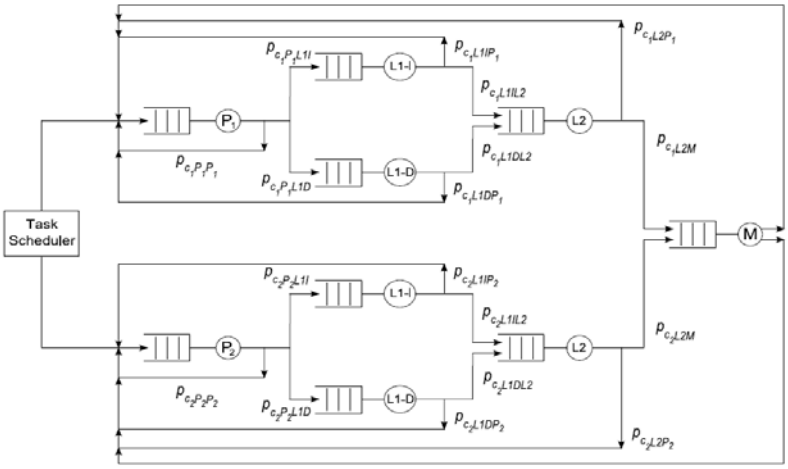

Fig.1: Queueing Network Model for the 2P-2L1ID-2L2-1M Multi-core Embedded Architecture

We consider the closed product-form queueing network for modeling multi-core embedded architectures because the closed product-form queueing network enables unequivocal modeling of workloads. We point out that additional applications can be added or updated in an embedded system (e.g., a smartphone) over time. However, these additional applications can be represented as synthetic workloads in our queueing-theoretic model. Furthermore, closed product-form queueing networks assume that a job leaving the network is replaced instantaneously by a statistically identical new job [7].

The performance metrics (e.g., throughput, response time, etc.) for a closed product-form queueing network can be calculated using a mean value analysis (MVA) iterative algorithm [8]. The basis of MVA is a theorem stating that when a job arrives at a service center in a closed network with N jobs, the distribution of the number of jobs already queued is the same as the steady state distribution of N −1 jobs in the queue [9]. We conduct the formulation through modelling methodology as the following steps:

$$r_i(k) = \frac{1}{\mu_i}\left(1 + l_i(k-1)\right) \qquad \lambda_i(k) = \upsilon_i \cdot T(k) \tag{1}$$

$$R(k) = \sum_{i=1}^{l} \upsilon i \cdot r_i(k) \tag{2}$$

$$T(k) = \frac{k}{R(k)} \qquad l_i(k) = \lambda_i(k) \cdot r_i(k) \tag{3}$$

To explain our modeling approach for multi-core embedded architectures, we describe a sample queueing model for the 2P-2L1ID-2L2-1M architecture in detail (other architecture models follow a

similar explanation). The figure 2 illustrates the revised model pattern.

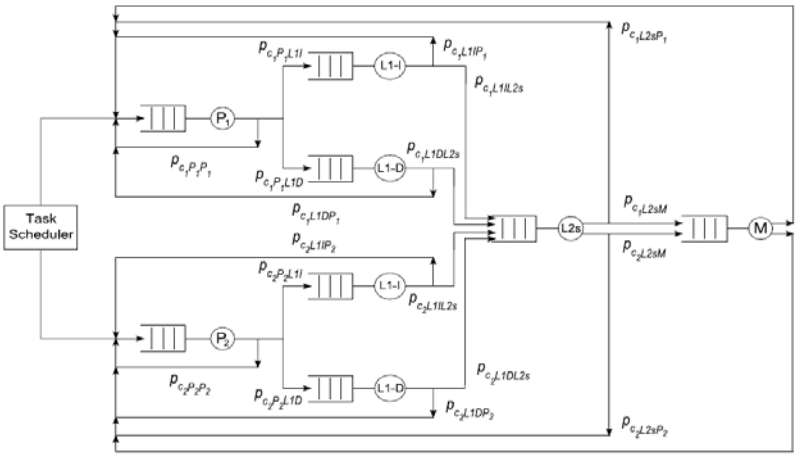

Fig.2: Queueing Network Model for the 2P-2L1ID-1L2-1M Multi-core Embedded Architecture

Our queueing theoretic models make some simplifying assumptions, which do not affect the general applicability of our approach. Our queueing network models assume cycle-level assignments of tokens (service time slices) for a given workload/job such that in each cycle, the tokens receive service from a particular service center with a given probability. For example, a job leaving the processor core either returns to the processor core's queue to wait for another time slice or goes to either the L1-I or L1-D cache for an instruction or data fetch, respectively. Completed jobs are replaced immediately by a statistically identical job, an assumption for closed product-form queueing networks, which holds true for embedded systems [10]. Since critical sections are effectively serialized, the response time of the workload containing critical sections will increase depending on the number of critical sections and the number of instructions in each critical section. Hence, additional time for executing critical sections can be calculated by the number of critical sections and the number of instructions in each critical section and added to the response time of the workload. We note that even though some of these assumptions may violate practical scenarios, such violations would not significantly impact the insights obtained from our queueing theoretic models because our models measure performance trends and focus on the relative performance of architectures for different benchmarks rather than the absolute performance.

The Experiment and Validation

The Theoretical Validation

We analyzed our queueing network models for different cache miss rates and workloads and find that the model's simulation results conform with expected queueing theoretical results. We present the average response time individually for the processor cores and the L1-I, L1-D, and L2 caches. For smaller L1-I, L1-D, and L2 cache miss rates, the processor core response time increases drastically as N increases because most of the time jobs are serviced by the processor core whereas for larger L1-I, L1-D, and L2 cache miss rates, the MM response time increases drastically because of a large number of MM accesses. These results along with our other observed results conform with the expected queueing theoretical results and validate our queueing network models for multi-core architectures. The figure 3 illustrates the simulation result.

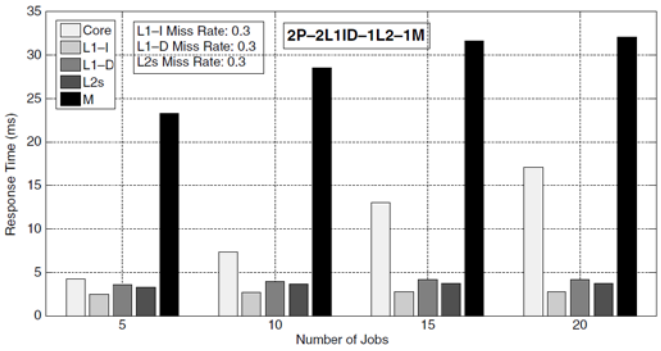

Fig.3: Queueing Network Model Validation of the Response Time in ms for Mixed Workloads for 2P-2L1ID-1L2-1M for a Varying Number of Jobs N

The Validation with a Multi-core Simulator

We further validate our queueing theoretic approach for modeling multi-core architectures using multi-threaded benchmarks executing on a multi-core simulator. We choose kernels/applications from the SPLASH-2 benchmark suite, which represent a range of computations in the scientific, engineering, and graphics domains. Our selected kernels/applications from the SPLASH-2 benchmark suite include fast Fourier transform (FFT), LU decomposition, radix, and water-spatial. We simulate the architectures in Table 1 using SESC [3]. To accurately capture our modeled architectures with SESC, our queueing theoretic models use the same processor and cache parameters (e.g., processor operating frequency, cache sizes and associativity, etc.) for the architectures as specified in the SESC configuration files. We consider single-issue processors with five pipeline stages and a 45 nm process technology. The execution times for the benchmarks on SESC are calculated from the number of cycles required to execute those benchmarks.

Table 1: The Simulation Result

Architecture	FFT	LU	Radix	Water-spatial
2P-2L-2L2	56.34	513.22	4.33	24.47
2P-2L-1L2	47.12	497.43	4.16	24.22
4P-4L-4L2	66.39	331.75	4.97	13.68
4P-4L-1L2	49.17	379.14	3.99	14.37
4P-4L-2L2	54.03	433.32	4.05	14.77

Conclusion

In this paper, we developed closed product-form queueing network models for performance evaluation of multi-core embedded architectures based on cloud computing for different workload characteristics. The simulation results for the SPLASH-2 benchmarks executing on the SESC simulator (an architecture-level cycle-accurate simulator) verified the architectural evaluation insights obtained from our queueing theoretic models. Results revealed that our queueing theoretic model qualitatively evaluated multi-core architectures accurately with an average difference of 5.6% as compared to the architectures' evaluations from the SESC simulator. The performance evaluation results indicated that the architectures with shared LLCs provided better cache response time and MFLOPS/W than the private LLCs for all cache miss rates especially as the number of cores increases. The results also revealed the disadvantage of shared LLCs indicating that the shared LLCs are more likely to cause a main memory response time bottleneck for larger cache miss rates as compared to the private LLCs. The memory bottleneck caused by shared LLCs may lead to increased response time for processor cores because of stalling or idle waiting. However, the results indicated that the main memory bottleneck created by shared LLCs can be mitigated by using a hybrid of private and shared LLCs (i.e., sharing LLCs by a fewer number of cores) though hybrid LLCs consume more power than the shared LLCs and deliver comparatively less MFLOPS/W. The performance per watt and performance per unit area results for the multi-core embedded architectures revealed that the multicore architectures with shared LLCs become more

area and power efficient as compared to the architectures with private LLCs as the number of processor cores in the architecture increases. In our future work, we plan to enhance our queueing theoretic models for performance evaluation of heterogeneous multi-core embedded architectures.

Acknowledgements

The research work was supported by Shandong Provincial Staff Education office No. 2013-324.

References

[1] Bistouni, Fathollah, and Mohsen Jahanshahi. "Pars network: A multistage interconnection network with fault-tolerance capability." Journal of Parallel and Distributed Computing 75 (2015): 168-183.

[2] Cao, Zheng, Roberto Proietti, and S. J. B. Yoo. "Hi-LION: Hierarchical Large-Scale Interconnection Optical Network With AWGRs [Invited]." Journal of Optical Communications and Networking 7.1 (2015): A97-A105.

[3] Liu, Xuejuan, et al. "Joint Lot-size and Preventive Maintenance Optimization for a Production System." International Journal of Performability Engineering 11.1 (2015): 91.

[4] El-Baky, MA Abd. "A tree-based algorithm for multicasting in 2D torus networks." Egyptian Informatics Journal (2015).

[5] CHAKRABORTY, SUPARNA, and NEERAJ KUMAR GOYAL. "Subset Cut Enumeration of Flow Networks with Imperfect Nodes." International Journal of Performability Engineering 11.1 (2015): 81.

[6] SRIVASTAVA, PREETI WANTI, and DEEPMALA SHARMA. "Optimum Time-Censored Step-Stress PALTSP with Competing Causes of Failure Using Tampered Failure Rate Model." International Journal of Performability Engineering 11.1 (2015): 71.

[7] Tsirkin, Michael S., and Gleb Natapov. "Systems and Methods for Providing Hypercall Interface for Virtual Machines." U.S. Patent No. 20,150,007,170. 1 Jan. 2015.

[8] Wang, Chao, et al. "Codem: software/hardware codesign for embedded multicore systems supporting hardware services." International Journal of Electronics 102.1 (2015): 32-47.

[9] Swanson, Robert C., et al. "MEMORY DUMP WITHOUT ERROR CONTAINMENT LOSS." U.S. Patent No. 20,150,006,962. 1 Jan. 2015.

[10] Benyamina, A. H., P. Boulet, and K. Benhaoua. "Static and Dynamic Mapping Heuristics for Multiprocessor Systems-on-Chip." Computing in Research and Development in Africa. Springer International Publishing, 2015. 229-247.