A Fuzzy Logic Based Dynamic Reconfiguration Scheme for Optimal Energy and Throughput in Symmetric Chip Multiprocessors

Muhammad Yasir Qadri AND Klaus D. McDonald-Maier
School of Computer Science and Electronic Engineering
University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK
yasirqadri@acm.org, kdm@essex.ac.uk

Abstract

Embedded systems architectures have traditionally often been investigated and designed in order to achieve a greater throughput combined with minimum energy consumption. With the advent of reconfigurable architectures it is now possible to support algorithms to find optimal solutions for an improved energy and throughput balance. As a result of ongoing research several online and offline techniques and algorithm have been proposed for hardware adaptation.

This paper presents a novel coarse-grained reconfigurable symmetric chip multiprocessor (SCMP) architecture managed by a fuzzy logic engine that balances performance and energy consumption. The architecture incorporates reconfigurable level 1 (L1) caches, power gated cores and adaptive on-chip network routers to allow minimizing leakage energy effects for inactive components. A coarse grained architecture was selected as to be a focus for this study as it typically allows for fast reconfiguration as compared to the fine-grained architectures, thus making it more feasible to be used for runtime adaption schemes. The presented architecture is analyzed using a set of OpenMP based parallel benchmarks and the results show significant improvements in performance while maintaining minimum energy consumption.

1. Introduction

Contemporary processor architectures do not leverage much flexibility for reconfiguration and have been designed for an overall average performance in terms of throughput and energy consumption for general applications. A large amount of research is now focusing on processors with micro-architectural reconfigurable units such as cache, pipelines etc. to make them more adaptive and energy efficient as per user workload requirements.

Field Programmable Gate Arrays (FPGAs) have been employed in most of the reconfigurable architectures proposed as they can allow reconfiguration at runtime. The foremost advantage of runtime reconfiguration is that it is possible to apply a set of configurations that would have a combined size larger than the available area on the reconfigurable hardware. However reconfigurable platforms such as FPGAs still face a few problems such as seamless adaptation of Operating Systems that are executing running multiple threads on the hardware changes [1], and large reconfiguration latencies which make them inefficient to meet deadlines in a real-time environment [2, 3]. Partially reconfigurable FPGAs offer an effective resource for fast reconfiguration making them suitable for real-time tasks [3]. On the other hand only a few partially reconfigurable devices are available to date and specialized software is required to support these features. A full reconfiguration model could be adapted to any FPGAs with large on chip SRAMs but such an approach suffers from large configuration latencies.

Dhodapkar et al. [4] have proposed a scheme to detect change in workload requirements by profiling working set signatures and use them to tune the architecture's parameters, and also saving and re-installing optimal configurations for recursive processes to avoid recalculation overheads. Exploiting FPGA reconfigurability Caspi et al. [5] proposed a scalable approach to virtualizes reconfigurable computing resources by dividing a computation up into fixed-size pages and time-multiplexing the virtual pages on available physical hardware. Klaus et al. [6] have demonstrated two pre-emptive scheduling algorithms based on earliest deadline first (EDF) for reconfigurable multiprocessor architectures. The algorithms are particularly useful for task deadlines greater than the hardware reconfiguration latency. An energy and throughput aware cache reconfiguration algorithm was presented by Balasubramonian et al. [7]. Their work cantered around a single core multilevel cache hierarchy, has shown significant amount of throughput improvements and energy reductions. Another work for reconfigurable single core architecture was presented by Yvan et al. [8], in which the authors have extended an RTOS for reconfiguration according to objectives such as quality of service (QoS), performance or power

978-1-4244-5889-9/10/$26.00 ©2010 IEEE

consumption. A camera based object tracking application was also implemented as proof of concept. Applying dynamic voltage and frequency scaling (DVFS) techniques Li [9] has proposed energy efficient task scheduling algorithm for real-time multiprocessor systems. The algorithm also offers to minimize schedule length. This work also includes a detailed analysis of the performance of various task scheduling algorithms in this scenario. A general perception of balanced thread scheduling (equal number of threads per core) being better than unbalanced thread scheduling was challenged by DeVuyst et al. [10] where the authors have demonstrated the power and performance advantages offered by unbalanced scheduling in multicore and multi-threaded architectures. Other examples of task scheduling approaches for reconfigurable or multicore architectures could be found in [11-15].

In this paper a novel coarse-grained reconfigurable symmetric chip multiprocessor (SCMP) architecture is presented. The platform is suitable for being targeted on FPGAs or custom System-on-Chips (SoCs), and provides an opportunity for research in the fields of Network-on-Chip (NoC), thread scheduling, and hardware adaption algorithms. This paper also proposes a novel fuzzy logic (FL) based reconfiguration scheme for finding an optimum balance between energy and performance of the system. The reconfigurable parameters of the system include Level 1 (L1) cache Associativity, L1 cache size, processors' operating frequency, and number of processor cores while the control variables of the fuzzy logic reconfiguration engine (FLRE) are aggregate cache miss rate, core utilization, and energy consumption of the SCMP.

The remainder of this paper is divided into four sections. The next section covers the related work in the field of hardware reconfiguration schemes, energy aware scheduling for multiprocessors and novel architectures for energy efficient Multiprocessor System on-chip (MPSoC) components. The section 2 visits the composition of the system covering the proposed SCMP architecture, the fuzzy logic reconfiguration system, benchmark applications, and the simulation environment. Sections 3 and 4 confer the results achieved from the proposed system and form a conclusion.

2. System Architecture and Experimental Setup

In this section an overview of the proposed approach is described in details: initially the reconfigurable SCMP architecture is introduced, which is then complemented by the proposed interconnect infrastructure, the fuzzy logic reconfiguration engine (FLRE), the benchmark software used and finally the experimental setup.

2.1. Reconfigurable SCMP Architecture

The SCMP platform presented in this paper is based on an oct-core x86 system. It follows a shared memory architecture with each core having its own L1 cache, and memory which is connected to common off-core L2 cache. The reconfigurable elements include the CMOS power gated cores, L1 cache size and Associativity, and finally the core frequency and voltage to support DVFS. Due to the variance in timing by changing cache size and Associativity L1 cache miss penalty was assumed to be fixed at 10 cycles.

Table I. Level 2 (L2) Cache Description

L2 Cache Parameters	
Size [Kbytes]	128
Associativity	8
Cache Line Size [bytes]	16
Block Size [bytes]	1024
Read Penalty [cycles]	30
Write Penalty [cycles]	30
Access Time [s]	3.73E-09
Cycle Time [s]	1.62E-09
Total Dynamic Read Energy [J]	4.68E-08
Total Dynamic Write Energy [J]	4.12E-09

The L2 cache was described as a fixed structure of 128 Kbytes, and its architecture is detailed in Table 1. The main memory is regarded as an off-chip module so its energy component is unaccounted. The cache energy and timing information was obtained from CACTI 4.0 [21] which is an open source standard tool for highly accurate cache analysis. However, it is not a trace driven simulator, so energy consumption resulting in number of hits or misses is not accounted for a particular application. The cache hit and miss rate profiling is discussed in details in section 3.5. Fig. 1 illustrates energy consumption of each core versus frequency and voltage. This information was referred from the performance statistics of Intel 486 GX embedded processor mentioned in its datasheet [22]. Fig. 2 shows the detailed architecture of the MPSoC and its various configurations.

Each core in the proposed SCMP architecture is coupled with a network adapter (NA) (see Fig. 3) and a network router which are finally connected via links. The role of network adapter is to handle the end-to-end flow control, encapsulate the messages generated by the cores and provide a transparent core to core interface. The NoC power data was obtained from Orion, a power-performance simulator for interconnection networks [16-18].

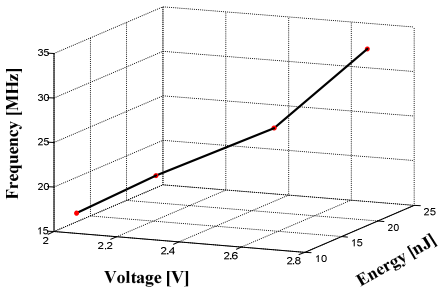


Figure 1. Processor Energy Consumption vs. Frequency and Voltage

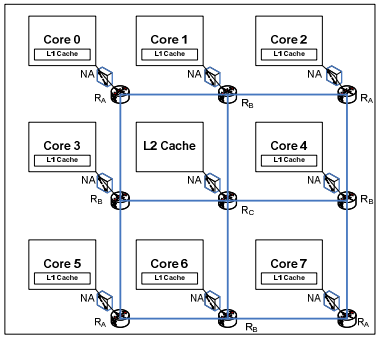


Figure 2. SCMP architecture

2.2. Fuzzy Logic Reconfiguration Engine (FLRE)

To find the optimal energy performance balance a Mamdani's Fuzzy Logic reconfiguration engine [19] was employed to control the target hardware reconfiguration. The fuzzy logic engine was implemented using fuzzy logic API presented in [20] conforming to IEC 61131-7 standard [21].

The reconfiguration parameters of the system are number of cores, L1 cache size and Associativity, and core operating frequency and voltage, while the input variables of the system include core utilization, total energy consumption of the SoC, and aggregate miss rate (see Fig. 3). In Fig. 3a, for L1 the cache miss rate varying from 0-100%, the membership function mf1 is qualifying a low miss rate which is defined from 0-40%, mf2 for a moderate miss rate from 30-70%, and mf3 for a high miss rate from 60– 100%. Similarly for Core Utilization, the lower bound is defined from 0-40%, moderate utilization is from 10-90%, and higher is from 60-100%. For Energy Consumption the limits for mf1, mf2, and mf3 are 0-40%, 10- 90%, and 60-100% respectively. For output variables, the cache size is allowed to be varied between 1-8KB, with mf1, mf2 and mf3 of 1-4KB, 1.5-7.5KB, and 5-8KB respectively. Cache Associativity, which is a discrete function, is to be varied between 0 to 16, with the lower bound defined as 0-2, moderate Associativity as 1-8 and higher Associativity as 4-16. The membership functions (mf1, mf2, mf3) for Processor Speed are defined as 16-20MHz, 20-25MHz, and 25-33MHz, whilst those for Number of Cores are defined as 1-3, 2-6, and 5-8 respectively (see Fig. 3b).

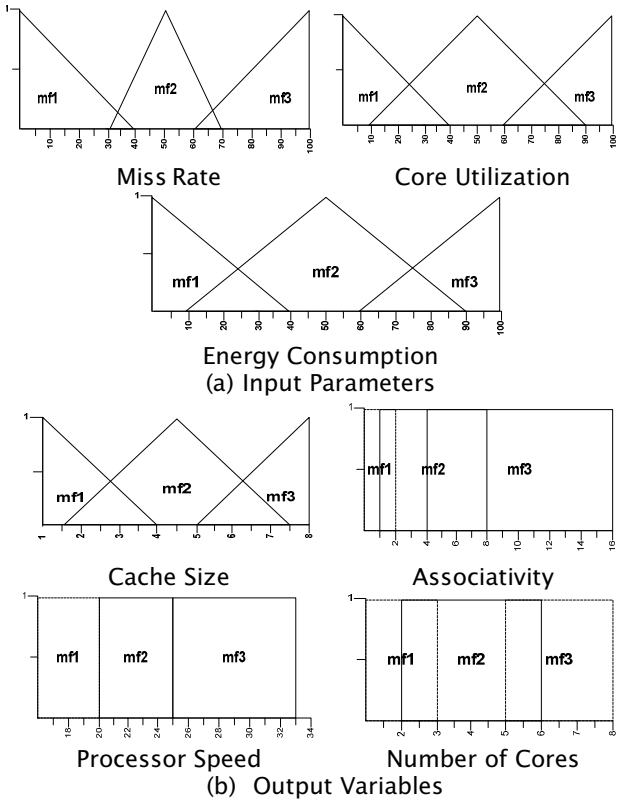


Figure 3. FLRE Membership Functions

In order to find an optimum solution, the FLRE varies the number of cores to maximize core utilization, but also monitors the throughput of the system. For example, a 100% core utilization with very low throughput would be considered an inefficient case. The criterion was defined to be a minimal of 50% throughput for a given configuration. A change of the miss rate of L1 cache effects both performance and energy consumption. A very large L1 cache with higher Associativity is an ideal solution towards minimum miss rate, but the cache throughput and cache energy consumption also affects the overall performance of the SoC. Therefore the FLRE strives to find a viable compromise between cache miss rate and SoC energy and throughput by adapting cache Associativity and size.

2.3. Benchmark Applications

As the proposed architecture comprises of a multicore setup, a set of NAS parallel benchmarks based on OpenMP [22] was selected to perform the target evaluation. The NAS parallel benchmarks are based on classical Computational Fluid Dynamics (CFD)

applications. The OpenMP is an Application Program Interface (API) designed to support programming for multicore shared memory architectures in C/C++ and Fortran [23]. The benchmarks were especially designed to exploit parallel architectures as it incorporates synchronization and communication of data between processing nodes, rearrangement of data stored in multiple nodes, and high-speed I/O operations between memory and mass storage [24]. Three benchmark applications namely Block Tridiagonal (BT), Conjugate Gradient (CG) and Integer Sort (IS) were used to evaluate the proposed architecture.

2.4. Simulation environment

The proposed architecture was simulated on a full system simulator Simics [25] which facilitates instruction-set level simulations and is capable to run unmodified OS' such as VxWorks, Solaris, Linux, Tru64, and Windows XP virtually on the target platforms. The simulator is targeted to provide fairly accurate timing profile, but at present does not support energy profiling of the target system. Simics provides a fairly accurate cache profiling utility, making it well suited for memory system research. An x86 based oct-core system was defined with each core having private L1 cache and coupled with a single shared L2 cache. Fedora Linux version 10 was chosen as target OS due to the inherent multicore support provided in Linux. Also Advanced Configuration and Power Interface (ACPI) enabled operating systems such as Linux supports hot-plugging (i.e. turning on/off) of a CPU core on the go which is a vital feature for reconfigurable MPSoC scenarios like the one presented here. The instruction execution, and cache hit/miss information was instrumented through Simics. Interconnect network energy information was gathered by using Orion, cache energy and timing information was gathered by using CACTI, the multilevel cache energy consumption and throughput was calculated based on mathematical models proposed by the authors in [26, 27]. The MPSoC total energy was calculated as the sum of interconnect energy (as calculated from Orion), cache energy (see Table 1), and each processor's core energy (see Fig. 1). The processor core energy information was obtained from Intel 486 GX embedded processor datasheet.

To profile thread execution statistics Intel Concurrency Checker [28] was used, which provided information such as core utilization, thread distribution, percentage of parallelism and timing of the applications. All the applications were sampled for first five seconds and then reconfiguration was carried out based on the decisions made by the FLRE. Simics provides facility of check-pointing through which each time the machine parameters such as cache size, and Associativity, and operating frequency were modified, and the number of cores were adjusted by using Linux hotplug feature. The applications were re-executed for each iteration, since, as the application proceeds in execution sampled average cache miss rate continue to vary, so a clear impact of cache reconfiguration cannot be analyzed, and the same is the case for thread based scheduling.

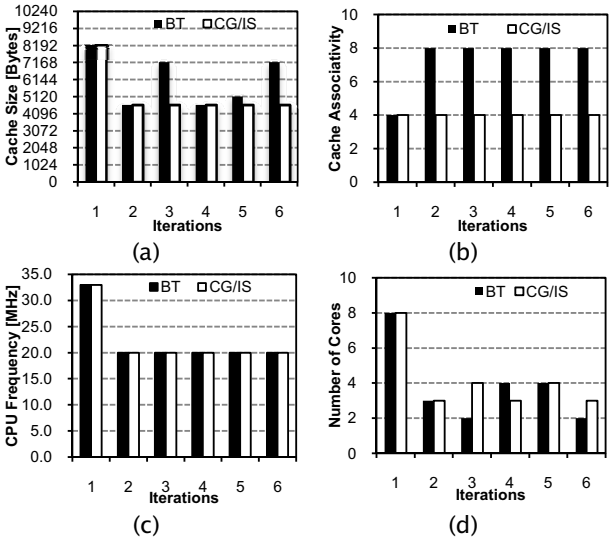
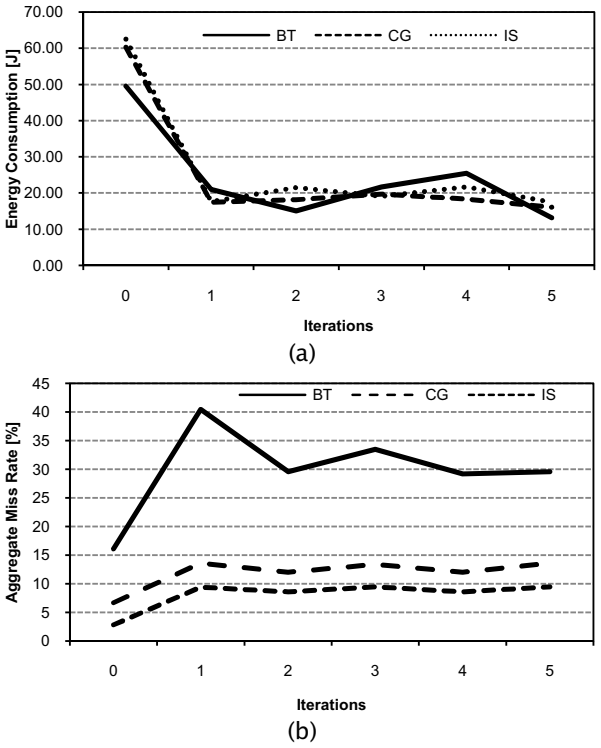
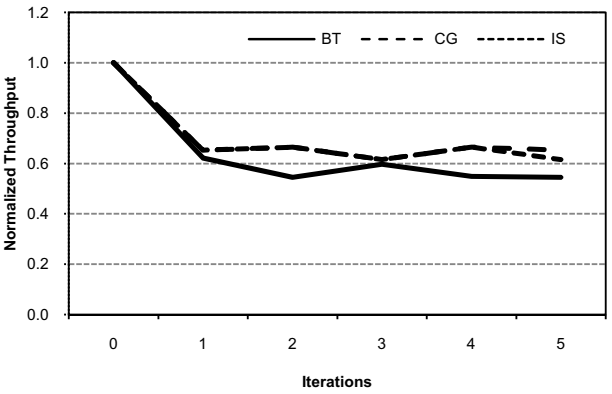
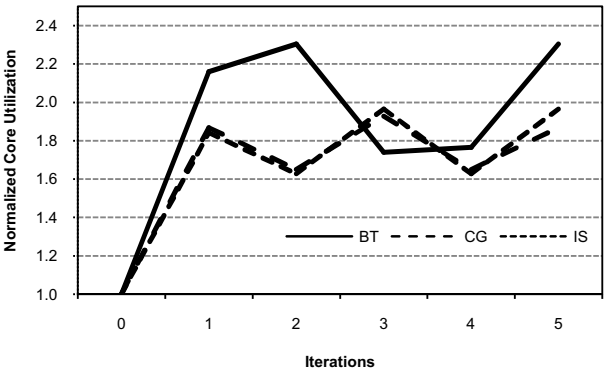


Figure 4. FLRE Results for (a) L1 Cache Sizing, (b) Associativity, (c) CPU Frequency, and (d) Number of Cores





(c)



(d)

Figure 5. Impact of optimizations on (a) energy consumption, (b) aggregate miss rate, (c) throughput, and (d) core utilization.

3. Results

As discussed in the previous section, three benchmark applications were executed to validate the design methodology. The operation of FLRE is shown in Fig. 4. The reconfiguration engine based on data of un-optimized core (iteration 0) starts modifying the reconfigurable parameters, i.e. Number of Cores, Operating Frequency, L1 Cache Size and Associativity (Fig. 4). The main objective of the system is to search for an optimum solution for performance and energy of the MPSoC. The reconfiguration engine completed the system configuration in five iterations and results were found unvarying for all the subsequent iterations. Fig. 5 shows the impact of each iteration of reconfiguration engine on the system's energy consumption (Fig. 5a), miss rate (Fig. 5b), throughput (Fig. 5c), and core utilization (Fig. 5d). On the average for all three applications, the overall core utilization of the system has increased by 2 times, and energy consumption has decreased by 3.7 times, while on the other hand the cost of such optimization was that the throughput has been decreased by 40%, and aggregate miss rate has increased by more than twice that of the original configuration. Fig. 6 shows a breakdown analysis of the total energy consumption of the MPSoC in terms of L2 cache energy, CPUs only energy and interconnect energy consumption. In general, the overall ratio of the Interconnect energy to the total energy varied from 4-10% and that for L2 cache energy was found to be between 3% to 12.5%.

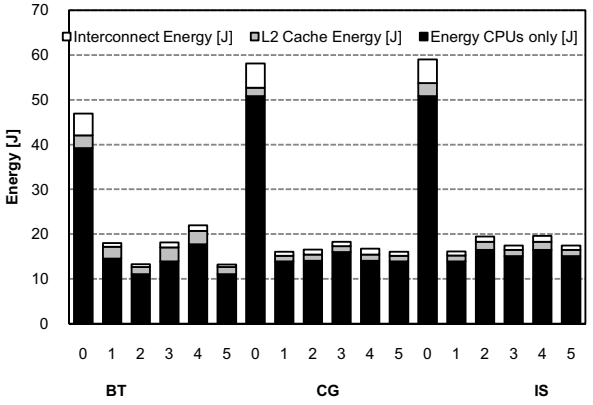


Figure 6. Total Energy consumption of MPSoC

4. Conclusions and Future Work

This paper presented a novel fuzzy logic based MPSoC reconfiguration scheme. The FLRE was used to find an optimal balance between energy consumption and performance of the system. To evaluate the proposed scheme an Intel x86 based multicore SoC with 8 processor cores based on a shared memory architecture, was simulated using the Simics full system simulator. The SoC architecture included power gated cores for minimum energy consumption whilst not in use. A detailed analysis of core, cache, and interconnect power consumption was conducted and on the average for all three applications, the overall core utilization of the system has increased by 2 times, and energy consumption has decreased by 3.7 times, on the other hand, the cost of such optimization was that the throughput has been decreased by 40%, and aggregate miss rate has increased by more than twice that of the original configuration.

The system in general validated the use of the proposed fuzzy based technique for MPSoC reconfiguration; therefore the same could be adapted for a variety of architectures to search a compromise for throughput and energy under user defined constraints. The proposed MPSoC architecture could be tailored to be used in variety of applications such as NoC research, dynamic thread scheduling, operating system development and high performance computing. In future a timing constraints based reconfigurable system is to be developed for real-time applications. It is also considered to investigate dynamic thread scheduling in the system to be able to reconfigure whilst executing a task.

References

[1] K. Compton and S. Hauck, "Reconfigurable computing: a survey of systems and software," *ACM Computing Surveys (CSUR),* vol. 34, pp. 171-210, 2002.

[2] T. Marescaux, A. Bartic, D. Verkest, S. Vernalde, and R. Lauwereins, "Interconnection networks enable fine-grain dynamic multi-tasking on FPGAs," in *Lecture Notes in Computer Science*. vol. 2438 Montpellier, France, 2002, pp. 795-805.

[3] J. Resano, D. Mozos, D. Verkest, and F. Catthoor, "A reconfigurable manager for dynamically reconfigurable hardware," *IEEE Design & Test of Computers,* vol. 22, pp. 452-460, 2005.

[4] A. S. Dhodapkar and J. E. Smith, "Managing multi-configuration hardware via dynamic working set analysis," *ACM SIGARCH Computer Architecture News,* vol. 30, pp. 233-244, 2002.

[5] E. Caspi, M. Chu, R. Huang, J. Yeh, J. Wawrzynek, and A. DeHon, "Stream computations organized for reconfigurable execution (SCORE)," *Lecture notes in computer science,* pp. 605-614, 2000.

[6] D. Klaus and P. Marco, "An EDF schedulability test for periodic tasks on reconfigurable hardware devices," in *Proceedings of the 2006 ACM SIGPLAN/SIGBED conference on Language, compilers, and tool support for embedded systems* Ottawa, Ontario, Canada: ACM, 2006.

[7] R. Balasubramonian, D. Albonesi, A. Buyuktosunoglu, and S. Dwarkadas, "Memory hierarchy reconfiguration for energy and performance in general-purpose processor architectures," 2000, pp. 245-257.

[8] E. Yvan and D. Jean-Philippe, "Specification and OS-based implementation of self-adaptive, hardware/software embedded systems," in *Proceedings of the 6th IEEE/ACM/IFIP international conference on Hardware/Software codesign and system synthesis* Atlanta, GA, USA: ACM, 2008.

[9] K. Li, "Performance Analysis of Power-Aware Task Scheduling Algorithms on Multiprocessor Computers with Dynamic Voltage and Speed," *IEEE Trans. Parallel Distrib. Syst.,* vol. 19, pp. 1484-1497, 2008.

[10] M. DeVuyst, R. Kumar, and D. M. Tullsen, "Exploiting unbalanced thread scheduling for energy and performance on a CMP of SMT processors," in *Proceedings of the 20th IEEE/ACM International Parallel and Distributed Processing Symposium* Rhodes Island, Greece: IEEE/ACM, 2006.

[11] P. Yang, C. Wong, P. Marchal, F. Catthoor, D. Desmet, D. Verkest, and R. Lauwereins, "Energy-aware runtime scheduling for embedded-multiprocessor socs," *IEEE Design & Test of Computers,* vol. 18, pp. 46-58, 2001.

[12] T. Li, D. Baumberger, D. A. Koufaty, and S. Hahn, "Efficient operating system scheduling for performance-asymmetric multi-core architectures," in *Proceedings of the 2007 ACM/IEEE conference on Supercomputing* Reno, Nevada: ACM, 2007, pp. 1-11.

[13] F. A. Bower, D. J. Sorin, and L. P. Cox, "The Impact of Dynamically Heterogeneous Multicore Processors on Thread Scheduling," *IEEE Micro,* vol. 28, pp. 17-25, 2008.

[14] K. Danne and M. Platzner, "A Heuristic Approach to Schedule Periodic Real-Time Tasks on Reconfigurable Hardware," in *Proceedings of the International Conference on Field Programmable Logic and Applications (FPL) 2005* Tempere, Finland, 2005, pp. 568-573.

[15] P. Saha and T. El-Ghazawi, "Extending Embedded Computing Scheduling Algorithms for Reconfigurable Computing Systems," in *Proceedings of 3rd Southern Conference on Programmable Logic, 2007. SPL '07. 2007* Mar del Plata, Argentina, 2007, pp. 87-92.

[16] X. Chen, L.-S. Peh, and S. Malik, "Leakage Power Modeling and Optimization in Interconnection Networks," in *Proceedings of the International Symposium on Low Power and Electronics Design (ISLPED)* Seoul, Korea, 2003.

[17] H. Wang, L.-S. Peh, and S. Malik, "Orion: A Power-Performance Simulator for Interconnection Networks," in *Proceedings of MICRO 35* Istanbul, Turkey, 2002.

[18] H. Wang, L.-S. Peh, and S. Malik, "A Power Model for Routers: Modeling Alpha 21364 and InfiniBand Routers," *IEEE Micro,* vol. 23, pp. 26-35, 2003.

[19] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies,* vol. 7, pp. 1-13, 1975.

[20] S. Rabin, *AI game programming wisdom*: Charles River Media, 2002.

[21] "International Standard: Programmable controllers - Part 7: Fuzzy control programming," International Electrotechnical Commission IEC 61131-7 ed1.0, 2000.

[22] H. Jin, M. Frumkin, and J. Yan, "The OpenMP Implementation of NAS Parallel Benchmarks and Its Performance," NASA Ames Research Center, 1999.

[23] L. Dagum, R. Menon, and S. G. Inc, "OpenMP: an industry standard API for shared-memory programming," *IEEE Computational Science & Engineering,* vol. 5, pp. 46-55, 1998.

[24] D. Bailey, E. Barszcz, J. Barton, D. Browning, R. Carter, L. Dagum, R. Fatoohi, S. Fineberg, and et

al., "The NAS Parallel Benchmarks," Moffett Field, CA: NASA Ames Research Center, 1994.
[25] P. S. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, G. Hallberg, J. Hogberg, F. Larsson, A. Moestedt, and B. Werner, "Simics: A full system simulation platform," *IEEE Computer,* vol. 35, pp. 50-58, 2002.
[26] M. Y. Qadri and K. D. McDonald-Maier, "Data Cache-Energy and Throughput Models: Design Exploration for Embedded Processors," *EURASIP Journal on Embedded Systems,* 2009.
[27] M. Y. Qadri and K. D. McDonald-Maier, "Analytical Evaluation of Energy and Throughput for Multilevel Caches," in *12th International Conference on Computer Modeling and Simulation*, Cambridge, UK, 2010.
[28] Intel, "Intel Concurrency Checker v2.1," Intel Corporation, 2008.