Hierarchical Agent Based NoC with Dynamic Online Services

Alexander Wei Yin, Liang Guang, Pasi Liljeberg, Pekka Rantala, Ethiopia Nigussie, Jouni Isoaho, Hannu Tenhunen
Department of Information Technology, University of Turku, Finland
{yinwei, liagua, pakrli, peaura, ethnig, jisoaho, hatenhu}@utu.fi

***Abstract*—As the size of NoCs increases, power consumption and fault/variation tolerance have become two of the most crucial problems for system designers. To address these problems, we propose a NoC architecture based on a hierarchy of monitoring agents. By tracing the circuit properties at run time, the agents at different architectural levels are able to monitor and control over the whole NoC platform. This monitoring approach partitions various online diagnostic and management services onto hierarchical implementation levels so as to provide scalability and variability for large-scale NoC design. This paper explains the monitoring interaction between agent levels, and focuses on system optimization alternatives handled by different agent levels. It further quantitatively analyzes the feasibility and design alternatives in monitoring communication interconnection upon regular tile-based NoC layout.**

I. INTRODUCTION

Network-on-Chips (NoCs) have emerged as a promising approach to integrate a large number of processing elements (PEs) on a single chip, by introducing a network structure similar to that in parallel computers[1]. It has gained wide acceptance for the communication architecture in decentralized designs such as Intel 80-core Teraflop[2], Tilera 64-core[3], TRIPS[4], and RAW[5]. With the technology scaling, the size of NoCs also increases. However, problems such as power consumption [6] and fault/variation tolerance [7] pose tough, if not stronger, constrains on design and implementation methods. To achieve maximum power efficiency and variation tolerance, dynamic online services should be integrated in NoCs.

A few previous works have addressed system monitoring services on NoC platforms [8, 9, 10]. From them, several distinctive requirements for managing NoC structures in a scalable manner can be identified. Firstly, local circuits need to be provided with distributed monitoring modules. Distributed monitoring reduces the local operation delay and interconnect latency for urgent monitoring services, and it prevents the appearance of communication bottleneck. However, despite the system size, centralized monitoring is still an indispensable complement to localized monitoring schemes. Theoretically, a centralized monitor, with the knowledge of all on-chip resources, is able to coordinate and balance the functioning of all components with the aim of optimizing the overall system performance. In practice, as an example, [11] adopts a single processing unit for dynamic testing operations and a global-level scheduler. For either distributed or centralized monitoring scheme, the energy efficiency of monitoring services should be maximized.

In this paper, a hierarchical agent based NoC with dynamic online services is proposed. In the conventional NoC design methodology, there are two separate dimensions called communication and computation. The separation of these two dimensions is the key contribution of NoC, which allows lower power consumption and higher scalability and performance. In our design methodology, another dimension which we call autonomous dimension is added onto the NoC platform. It is implemented as the hierarchical monitoring agent architecture. The motivation of this dimension is to autonomously adjust the system in order to achieve low power consumption and fault/variation tolerance. Agents are functional units that monitor and control different architecture levels of the NoC platform depending on their hierarchical levels. This architecture aims at the enhancement of the system performance in both power consumption and fault tolerance aspects. It also provides a wide design and synthesis space for the realization of agents at each level.

This paper examines the functional partition of agent levels and a feasible implementation of the agents on a tile-based NoC platform (Section II). Upon the proposed NoC architecture, we demonstrate the flexible incorporation of system optimization techniques with agent monitoring architecture in terms of power consumption and fault/variation tolerance (Section III). As an extra communication layer upon existing interconnect, alternatives in realizing agent communications are examined quantitatively in Section IV, which reveals an optimal design trade-off for monitoring communication interconnects. Section V concludes the paper.

II. HIERARCHICAL MONITORING AGENT ARCHITECTURE

A. Agent Hierarchy

There are four levels of agents in the proposed NoC architecture, namely, application agent, platform agent, cluster agent and cell agent (Fig. 1). As the top level agent, the application agent is unique in a NoC platform. An application agent is a software module capturing the application functionalities and run time performance requirements and constraints. The platform agent is also unique in a NoC. Based on the specification from the application agent and resource availability, it (re)configures the network and PEs. The entire NoC is divided into a number of clusters, each of which is monitored and controlled by a cluster agent. A cluster is a group of processors with accompanying components (caches, scratchpad memories, switches, links, etc.). It is logically divided into cells which are the basic units in our architecture, consisting of a PE, a switch and the corresponding links. The cells are equipped with their own local monitors, the cell agents, which trace and adjust the local circuit conditions.

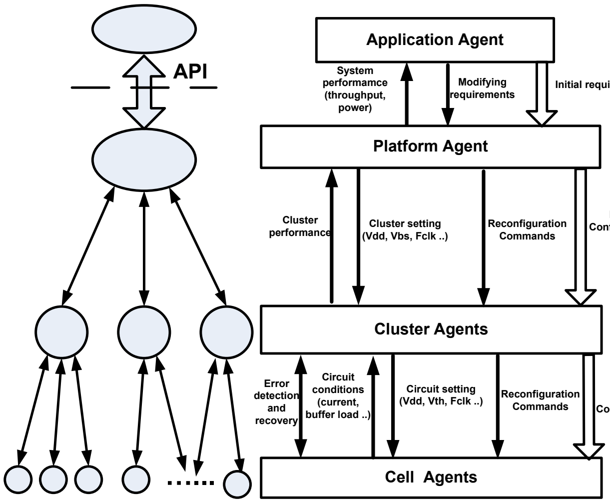


Figure 1. Hierarchical Agent Approach

B. Hierarchical Agents Implementation on NoCs

Figure 2 illustrates a feasible implementation of the hierarchical agents on a tile-based regular mesh NoC structure. As mentioned earlier, the basic monitored units are cells which comprise of a PE, a NI, a switch and the corresponding links. It is intuitive to allocate a cell agent for each cell by sharing the physical area of the PE. Other cell level monitoring units may be allocated at other particular places within a cell, such as a power-gating sleep-transistors on the links. The cluster agent is allocated at a fixed position during the design time. Since a cluster agent has more sophisticated functionalities and controlling algorithms than a cell agent, it requires more resources such as area, power, communication bandwidth and etc. Therefore, a cluster agent replaces one of the PEs in a cluster. To minimize the communication latency and balance the workload of the system, the platform agent and application agent which monitor and control over the whole system are located at the geographic center of the platform.

In order to offer scalability for extremely large scale NoC systems, clusters can be further divided into hierarchical sub-clusters and similar monitoring functional partition will be applied. It conceptually originates from the manner a bio-system or human society organizes its overwhelming amount of resources.

III. HIERARCHICAL AGENT BASED ONLINE SERVICES ON NOCS

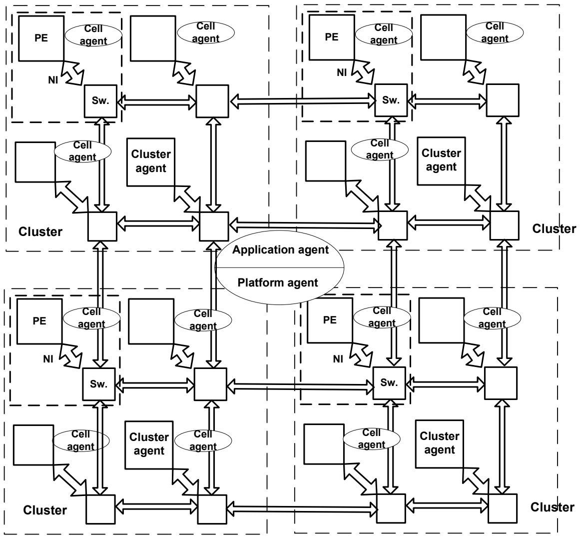


Figure 2. Hierarchical Agent Implementation on NoCs

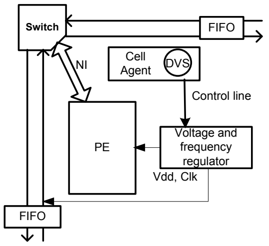
Our proposed monitoring agent based architecture is aimed at dynamically achieving low power consumption and fault/variation tolerance. In this section, we will highlight these online services on NoC systems.

A. Low-power Optimization with Agents

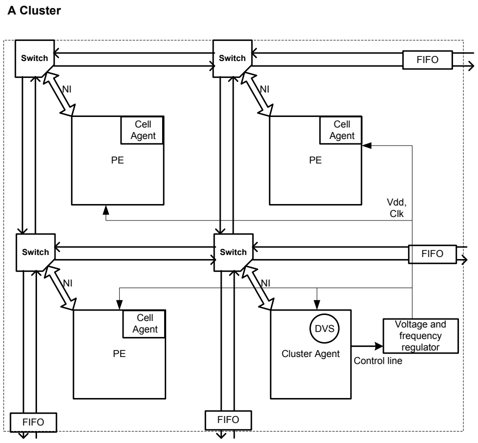
Although being under intensive research for decades, the power consumption is still the most critical constraint to be explored. The power consumption can be categorized into dynamic power consumption and leakage power consumption.

One of the major dynamic power saving techniques is DVFS (dynamic voltage and frequency scaling), which is traditionally provided on a chip-wide domain [12]. But chip-level single power domain is not able to utilize the local traffic variation in exploiting the supply scaling potential, thus per-core based DVFS is proposed [13]. In the cell-divided NoC platform, a cell can be conveniently set with a supply regulator with the cell agent in charge of the voltage and frequency adjustment (Fig. 3(a)). The overhead for per-cell based DVFS is significant. [14] reports $0.14mm^2$ area overhead and 83.2% peak efficiency of a DC-DC converter in $90nm$ technology. Each time the voltage is converted, extra energy will be consumed for the power regulation.

To alleviate the per-core-based DVFS overhead, the concept of voltage islands [15, 16] has been proposed. A voltage island is a physical entity on the chip that has its own internal power distribution network which is isolated from the primary chip level power distribution network. Up-to-date, voltage islands are statically determined at design time. To incorporate multiple voltage islands on the NoC platform, each cluster agent determines the voltage and frequency for its own cluster (Fig. 3(b)). The area and energy overhead is reduced proportional to the number of cells in a cluster. Per-cluster-based power optimization, however, does not support the reconfiguration of cells into different clusters at the run-time, though assigning spares into clusters initially still provides cell replacement possibilities against component failures.



(a) Cell-level DVFS (showing one cell)



(b) Cluster-level DVFS (showing one cluster)

Figure 3. Power Optimization Services by Different Agent Levels

The granularity of monitoring services is a design choice dependent on the size of the actual platform, the workload and constraints of the application. In terms of power optimization, per-cluster-based monitoring with lower implementation overhead seems to be more feasible in the long term with smaller-sized processing cores. In general, any monitoring service can be configured at the design time or execution time (with the support of reconfigurable platform) to be handled by different level of agents, correspondingly in various granularities.

Besides dynamic power consumption, hierarchical monitoring architecture aims to minimize the leakage consumption, which grows exponentially with decrease of transistor size. One process generation increases leakage by a factor of 6 to 10x. According to the current leakage trend, a microprocessor in 100*nm* technology may dissipate up to 50% leakage power [17]. To reduce the leakage power, dynamic power management methods, such as turning off the non communication intensive links, should be applied in the NoC platform by the usage of hierarchical monitoring agents.

B. Fault/Variation Tolerance with Agents

The major causes affecting the reliability of the NoC systems are the shrinking of the feature size and decreasing of the supply voltage, which expose them to different faults of permanent, transient or intermittent nature. Among the failure mechanisms, we can enumerate factors such as crosstalk, electromigration, electromagnetic interference, alpha particle hits, and cosmic radiation [18]. These phenomena and system variations can change the timing and functionalities of the NoC fabrics and thus degrade their QoS or, eventually, lead to failures of the whole NoC-based system. Providing resilience from such faults and other PVT (process, voltage, temperature) variations is mandatory for the NoCs.

The proposed monitoring agent based architecture can provide the fault/variation tolerance for NoC based systems, by the joint effort of all levels of agents.

Before execution, the platform agent utilizes a number of resources and configures the network based on the initial application requirements with power and performance awareness [19]. A number of resources are reserved as spares in case of component failures. The initial configuration is enforced from the platform agent to the cluster and then cell agents. After the application starts running, the cell agents trace their local circuit conditions such as failures and PVT variations. They first attempt to fix the errors if feasible (for example by retransmission in case of transient crosstalk-induced error [20]). If the errors cannot be solved by the cell agent, they have to be reported to the cluster agents. The cluster agents allocate the spares to take the places of the faulty cells and re-run the faulty instructions. In case that the errors cannot be solved within a cluster, they have to resort to the platform agents to re-map the application or reconfigure the system if necessary.

The application agent and platform agent are also responsible for the reconfiguration of the system to balance the workload in order to maintain the circuit under relative low temperature. This is of crucial importance due to the fact that the circuit is more error-prone under high temperature. Moreover, leakage currents also increase exponentially with temperature.

Figure 4 shows a study case where we focus on the functions of agent hierarchy to flexibly provide the trade-off on a fault-tolerant NoC platform with a pool of DSP processors divided into clusters. We simulate 64-point FFT/IFFT computation on 2-D mesh NoC, each processing element as a DSP unit running at the same frequency, and initially with 30% redundancy. The platform agent has two architecture alternatives, one exploring more parallelism (thus finishing faster) while using more processors [21] than the other [22]. The study case is simulated by Matlab/SimuLink.

We assume that every DSP works at 600MHz with 16-bit wide data, and one complex multiplication takes 6 cycles. Fig. 4(a) shows that when the system is configured with the architecture described in [21], the computation takes 6 processors and 8ms. If some components fail (Fig. 4(b)), the platform agent will replace them with spare processors and reconfigure the network. If the application agent specifies tougher timing constraints, the platform agent may utilize more available resources to achieve another performance/cost tradeoff. In Fig. 4(c), with architecture alternative as in [22], the computation time is reduced to 3 ms with the cost of another 10 processors used (spare ones in the data flow are only bypassed, not used in computation).

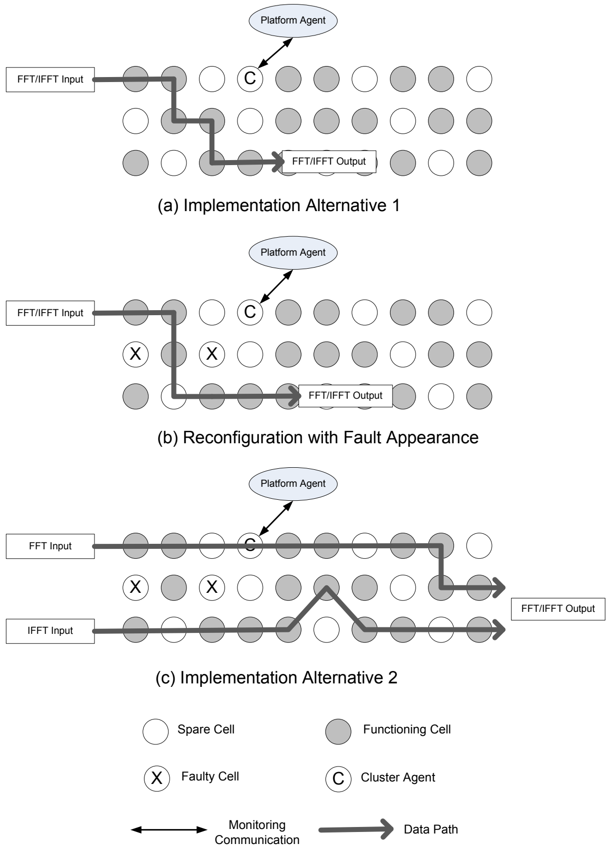


Figure 4. Study case on fault tolerance with hierarchical agents

IV. DESIGN TRADE-OFFS FOR AGENT COMMUNICATION

A. Monitoring Communication Interconnect Alternatives

Agents exchange monitoring information with their higher or lower counterparts as illustrated in Fig. 1. The monitoring communication needs to be reconfigurable so new cells can be incorporated to certain clusters at the run-time. Some conventional interconnection does not support reconfiguration (for instance, the star-like network). Instead, we consider three interconnect alternatives which all support run-time reconfiguration but have different area, energy and latency overheads. Throughput is not a prioritized design constraint, since the monitoring communication is low in data volume ([23] reports 8% and 5% debugging monitoring traffic overhead for two streaming applications).

The first alternative is to realize monitoring communication as TDM (Time-Division-Multiplexing)-based virtual channel upon existing links. This option incurs design complexity in virtual channel arbitration and allocation, increases the switch latency of both monitoring interconnect and data communication. The virtual channel arbitration and allocation also incur energy overhead. Wiring overhead, however, is kept to the minimum though the switch area is moderately increased.

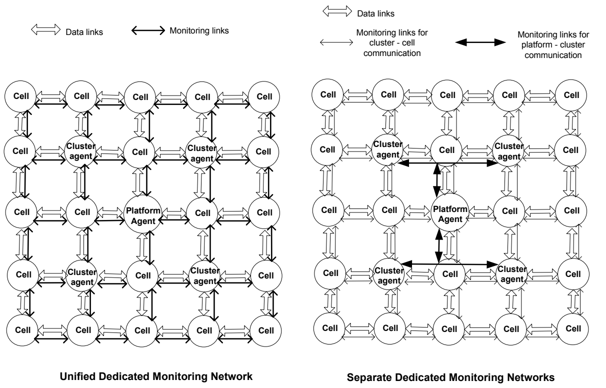


Figure 5. Alternative Dedicated Monitoring Interconnect Architectures

The second alternative is to adopt a “unified dedicated monitoring network” for monitoring communication (Fig. 5 on the left side). It is called “unified” as monitoring communication between both cluster-cell agents and platform-cluster agents is transmitted on the same dedicated network. This option utilizes more wiring resources but simplifies the switch arbitration between data and monitoring communication, thus reducing the communication energy and latency.

The third alternative is to adopt “separate dedicated monitoring networks” for monitoring communication (Fig. 5 on the right side). Compared to the unified monitoring network, this option adds another network connecting the single platform agent to a small number of cluster agents. As a result, the communication between platform and the cluster agents is simplified with very limited wiring overhead.

B. Quantitative Analysis of Monitoring Interconnects

We simulate the energy and latency of the agent communication on an 8*8 network simulator. The locations of the platform agent, cluster agents and cells (with cell agents) are illustrated in Fig. 6. The switch is input-buffered with matrix crossbar. Each link is 2mm long, and is modeled as segmented wires with drivers and evenly inserted repeaters[1]. Data links are 32 bits and monitoring links are 8-bit wide. The whole NoC system is assumed to be mesochronous with network frequency as 1GHz and the supply voltage as 1V.

We estimate the area and energy overhead of switches by simulating with Orion [24], a widely-used on-chip switch power simulator. The switch latency is estimated based on [25]. The wires are modeled and simulated by Cadence. The Orion simulator does not produce result for 65nm technology directly, thus we apply scaling factors (based on [26]) to the result of 70nm technology simulation using Orion. The scaling factors for energy, area, and latency are 0.86, 0.86 and 0.93 respectively. The energy of wires are simulated by Cadence. The latency in the switch buffer assumes an average 50% occupancy ratio.

---

[1] wire width: 210nm; spacing: 210nm; repeater interval: 0.25mm; repeater size: 10x minimal inverter size; driver size: 12x.

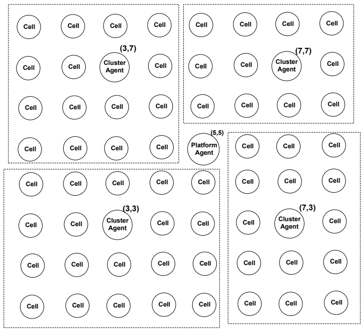


Figure 6. Locations of Platform, Cluster and Cell Agents in the Experimental Platform (with initial cluster boundary labeled)

Table I summarizes the average per-flit energy consumption for monitoring communication in each interconnect architecture. Table II summarizes the latency comparison for monitoring communication in each interconnect architecture.

Table I
ONE-FLIT MONITORING COMMUNICATION ENERGY OF THREE MONITORING INTERCONNECT ARCHITECTURES (NETWORK WORKING AT 1GHZ)

Interconnect Architecture	Energy (cluster <-> cell agents)	Energy (platform <-> cluster agents)
TDM-based	12.92 pJ	12.92 pJ
Unified Dedicated Network	5.40 pJ	5.40 pJ
Separate Dedicated Networks	5.40 pJ	2.31 pJ

Table II
WORST-CASE MONITORING COMMUNICATION DELAY IN THREE INTERCONNECT ARCHITECTURES

Interconnect Architecture	Delay (cluster <-> cell agents)	Delay (platform <-> cluster agents)
TDM-based	24 cycles	24 cycles
Unified Dedicated Network	16 cycles	16 cycles
Separate Dedicated Networks	16 cycles	8 cycles

C. Optimal Design Trade-off for Future NoCs

The estimated figures show that separate dedicated monitoring networks are the most energy-efficient and low-latency interconnection for monitoring communication. Compared to TDM-based interconnection, it reduces the latency by 66.7% and energy consumption 82.1% for the communication between the platform and cluster agents, while achieving the same latency and energy efficiency as unified dedicated network for the communication between the cluster and cell agents. However there is area penalty involved (Table III; the chip area ia assumed to be 275mm^2 [2]): the area overhead is increased from 2.71% to 3.32%. Nonetheless the wiring area overhead has become less of a design constraint as multi-layer fabrication process provides quite abundant wiring potential for on-chip systems ([10]; TILE64 processors incorporate 5 physically separate networks, each of them being 64-bit wide). With transistor feature size and wire dimension continue to decrease in the foreseeable future, the separate monitoring networks will provide the most optimal trade-off exploiting the on-chip wiring resources while minimizing the more critical power consumption and global interconnect latency.

Table III
AREA OVERHEAD OF THREE MONITORING INTERCONNECT ARCHITECTURES

Interconnect Architecture	Area (mm^2)	Percentage (of a chip area)
TDM-based	7.44	2.71%
Unified Dedicated Network	8.95	3.26%
Separate Dedicated Networks	9.11	3.32%

V. CONCLUSIONS

Hierarchical agent monitoring architecture provides great scalability and design flexibility for future large-scale NoC systems. With an extra monitoring layer comprised of four levels of agents, the system is potentially able to achieve maximized efficiency with online monitoring services. This paper elaborately explains the hierarchical monitoring approaches enabled by the interactions of all levels of agents, and examines the design alternatives for low-power optimization of different granularities as an example of flexible functional partitions among agent levels. Quantitative analysis for agent interconnection alternatives suggests reasonable trade-offs between area, energy and latency overhead, and motivates separate dedicated monitoring networks for inter-agent communication. This work demonstrates the potential and feasibility of multi-level online monitoring layer upon the overwhelming amount of on-chip resources, which provides a great diversity of design options in a scalable manner.

At present, specific monitoring services on regular NoC platform with the proposed architecture is under intensive study and analysis.

REFERENCES

[1] L. Benini and G. D. Micheli. *Networks on Chips: Technology And Tools*. Morgan Kaufmann, 2006.

[2] S.R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar. An 80-tile sub-100-w teraflops processor in 65-nm cmos. *JSSC*, 43(1):29–41, 2008.

[3] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, and M. Reif. Tile64tm processor: A 64-core soc with mesh interconnect. In *Solid-State Circuits Conference, 2008. ISSCC 2008. Digest of Technical Papers. IEEE International*, 2008.

[4] Karthikeyan Sankaralingam, Ramadass Nagarajan, Haiming Liu, Changkyu Kim, Jaehyuk Huh, Doug Burger, Stephen W. Keckler, and Charles R. Moore. Exploiting ilp, tlp, and dlp with the polymorphous

---

[2] the size of a TeraFLOPS chip

trips architecture. In *Proc. 30th Annual International Symposium on Computer Architecture (ISCA 2003), 2003.*, pages 422–433.

[5] Michael Bedford Taylor, Walter Lee, Saman Amarasinghe, and Anant Agarwal. Scalar operand networks: on-chip interconnect for ilp in partitioned architectures. In *Proc. the Ninth International Symposium on High-Performance Computer Architecture, 2003. HPCA-9 2003.*, pages 341–353.

[6] Jan M. Rabaey. Scaling the power wall: Revisiting the low-power design rules. Keynote speech at SoC 07 Symposium, Tampere, November 2007.

[7] K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer. High-performance cmos variability in the 65-nm regime and beyond. *IBM Journal of Research and Development*, 50(4/5):433–449, 2006.

[8] C. Ciordas, T. Basten, A. Radulescu, K. Goossens, and J. Meerbergen. An event-based network-on-chip monitoring service. In *Proc. of the 9th IEEE International High-Level Design Validation and Test Workshop*, pages 149–154, 2004.

[9] C. Ciordas, K. Goossens, A. Radulescu, and T. Basten. Noc monitoring: impact on the design flow. In *Proc. IEEE ISCAS 2006*, pages 1981–1984, 2006.

[10] D. Wentzlaff, P. Griffin, H. Hoffmann, Liewei Bao, B. Edwards, C. Ramey, M. Mattina, Chyi-Chang Miao, J.F. Brown, and A. Agarwal. On-chip interconnection architecture of the tile processor. *IEEE MICRO*, 27(5):15–31, 2007.

[11] D. Sylvester, D. Blaauw, and E. Karl. Elastic: An adaptive self-healing architecture for unpredictable silicon. *IEEE Design & Test of Computers*, 23(6):484–490, 2006.

[12] Canturk Isci, Alper Buyuktosunoglu, Chen-Yong Cher, Pradip Bose, and Margaret Martonosi. An analysis of efficient multi-core global power management policies: Maximizing performance for a given power budget. In *Proc. of 39th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO-39)*, pages 347–358, 2006.

[13] W. Kim, M. S. Gupta, G.-Y. Wei, and D. Brooks. System level analysis of fast, per-core dvfs using on-chip switching regulators. In *International symposium on high-performance computer architecture*, Feb. 2008.

[14] P. Hazucha, G. Schrom, Jaehong Hahn, B.A. Bloechel, P. Hack, G.E. Dermer, S. Narendra, D. Gardner, T. Karnik, V. De, and S. Borkar. A 233-mhz 80%-87% efficient four-phase dc-dc converter utilizing air-core inductors on package. *IEEE Journal of Solid-State Circuits*, 40(4):838–845, 2005.

[15] Lap-Fai Leung and Chi-Ying Tsui. Energy-aware synthesis of networks-on-chip implemented with voltage islands. In *Proc. 44th ACM/IEEE DAC '07*, pages 128–131, 2007.

[16] D.E. Lackey, P.S. Zuchowski, T.R. Bednar, D.W. Stout, S.W. Gould, and J.M. Cohn. Managing power and performance for system-on-chip designs using voltage islands. In *Proc. IEEE/ACM International Conference on Computer Aided Design ICCAD 2002*, pages 195–202, 2002.

[17] Desmond Kirkpatrick Rabindra (Rob) Roy Gregory S. Spirakis Naveed Sherwani Craig Peterson Timothy Kam, Shishpal Rawat. Eda challenges facing future microprocessor design. *IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems*, 19:1498–1506, 2000.

[18] C. Grecu, L. Anghel, P.P. Pande, A. Ivanov, and R. Saleh. Essential fault-tolerance metrics for noc infrastructures. In *Proc. 13th IEEE International On-Line Testing Symposium IOLTS 07*, pages 37–42, 2007.

[19] Jingcao Hu and R. Marculescu. Energy and performance-aware mapping for regular noc architectures. *IEEE Transactions on COMPUTER-AIDED DESIGN of Integrated Circuits and Systems*, 24(4):551–562, 2005.

[20] Teijo Lehtonen, Pasi Liljeberg, and Juha Plosila. Online reconfigurable self-timed links for fault tolerant noc. *VLSI Design*, 2007:13, 2007.

[21] Hsin-Lei Lin, Hongchin Lin, Yu-Chuan Chen, and R.C. Chang. A novel pipelined fast fourier transform architecture for double rate ofdm systems. In *Proc. IEEE Workshop on Signal Processing Systems SIPS 2004*, pages 7–11, 2004.

[22] Shousheng He and M. Torkelson. A new approach to pipeline fft processor. In *Proc. 10th International Parallel Processing Symposium, IPPS '96*, pages 766–770, 1996.

[23] C. Ciordas, K. Goossens, T. Basten, A. Radulescu, and A. Boon. Transaction monitoring in networks on chip: The on-chip run-time perspective. In *Proc. International Symposium on Industrial Embedded Systems IES '06*, pages 1–10, 2006.

[24] Hang-Sheng Wang, Xinping Zhu, Li-Shiuan Peh, and S. Malik. Orion: a power-performance simulator for interconnection networks. In *Proc. 35th Annual IEEE/ACM International Symposium on (MICRO-35) Microarchitecture*, pages 294–305, 2002.

[25] L.-S. Peh and W.J. Dally. A delay model and speculative architecture for pipelined routers. In *Proc. of The Seventh International Symposium on High-Performance Computer Architecture*, pages 255–266, 19–24 Jan. 2001.

[26] W. Haensch, E. J. Nowak, R. H. Dennard, P. M. Solomon, A. Bryant, O. H. Dokumaci, A. Kumar, X. Wang, J. B. Johnson, and M. V. Fischetti. Silicon cmos devices beyond scaling. *IBM Journal of Research and Development*, 50(4/5):339–361, 2006.