

A Scalable and Decentral Approach to sustained System Monitoring

David Kramer*, Rainer Buchty*,
Wolfgang Karl*

* *Universität Karlsruhe (TH), Institut für Technische Informatik,
76131 Karlsruhe, Germany*

ABSTRACT

Future heterogeneous system are likely to employ reconfigurability. In order to manage the complexity of such systems, the use of so-called Self-X capabilities is required. Of these, Self-Awareness is the essential property, providing required information for autonomous system and application optimization. We therefore propose a scalable, hierarchical, and decentral infrastructure for sustained monitoring, applicable to heterogeneous multi-core systems. By analyzing the gathered data, the actual state of the (sub-)system is classified and appropriate actions may be triggered.

KEYWORDS: Adaptive Computing, Monitoring, Self-Awareness, Self-Optimization

1 Introduction and Motivation

Current forecasts and vendor road-maps indicate that future computer architectures will be not only massively parallel, but heterogeneous including potential use of reconfigurable logic. An example for such future heterogeneous multicore architectures is the Digital on-demand Computing Organism (DodOrg) [BBB⁺06]. DodOrg is heavily inspired by biological concepts, following a similar hierarchical design pattern as biological organisms, including hormone- and nerval-inspired communication concepts. Integral part of this architecture, depicted in Figure 1 is a dedicated monitoring infrastructure spanning all system layers providing a flexible and adaptive approach for continuous system introspection, information gathering, and data correlation [BKK08].

Due to the dynamic behavior of these architectures, certain optimizations – e.g. the minimization of the communication distances between two tasks – can only be performed at runtime. The use of conventional single, centralized optimization instances is impractical

¹E-mail: {kramer,buchty,karl}@ira.uka.de

due to scalability and dependability issues: such an approach does not scale, inferring a potential bottleneck due to required communication and data processing efforts, and also is a single point of failure [BK08]. Both issues may lead to situations where a classified state is no longer valid when the classification process is completed, and therefore the triggered optimization would be futile if not conflictive.

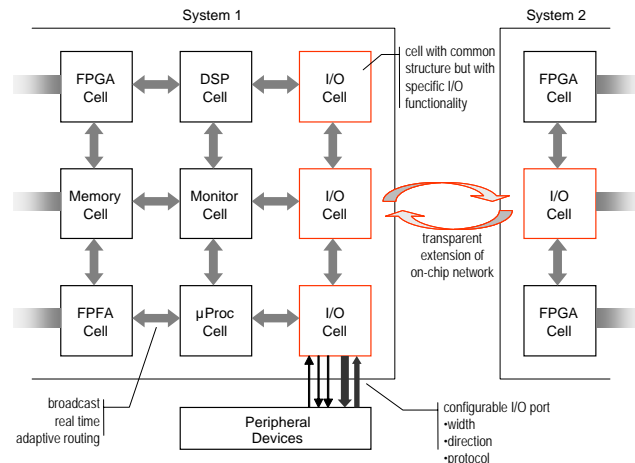


Figure 1: DodOrg: a heterogeneous multi-core architecture

The remainder of this abstract is organized as follows: In Section 2 we describe the scalable, hierarchical and decentral monitoring infrastructure for use in heterogeneous multi- and many-core systems. Section 3 contains an illustrative example, i.e. how the monitoring infrastructure can be used to minimize the communication distance between two tasks, therefore optimizing the overall performance.

2 Monitoring Infrastructure

In order to achieve required scalability and flexibility, our approach is organized hierarchically consisting of low- and high-level monitoring. Low-level monitors independently perform local (e.g. of one processing cell) data gathering and aggregation. The aggregated data is then forwarded to high-level monitors which correlate this individual state information, resulting in state classification and triggering of eventually required optimization processes.

By nature, low-level monitors are associative counter arrays (ACAs). Within DodOrg, such low-level monitors are integral part of each processing cell, monitoring status data like inter-cell communication or temperature. Optional software monitors may provide further information, e.g. cell utilization.

Unlike conventional performance counters available in modern CPUs the ACA is not limited to certain pre-defined events, but instead counts and accumulates *every* occurring event; DodOrg uses a hormone-inspired unique event coding scheme resulting in an individual, but easily parseable and processable *event tag* as shown in Figure 2: using a simple event mask, it is possible to narrow or widen the focus of event accumulation by specifying mandatory or “don’t care” fields of the event tag, providing increased monitoring flexibility.

This event is then counted by the ACA using a cache-like method: whenever an event occurs, its corresponding (meta) tag is compared with the individual tags stored in the ACA’s

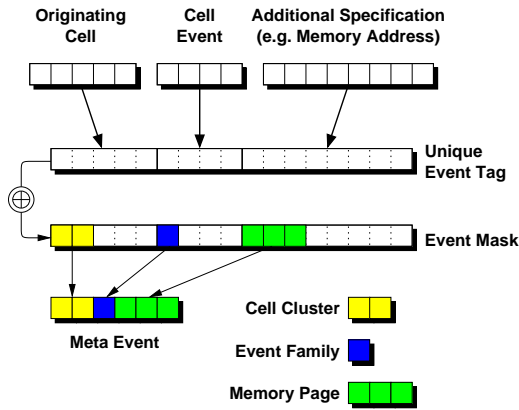


Figure 2: Event Tag Masking

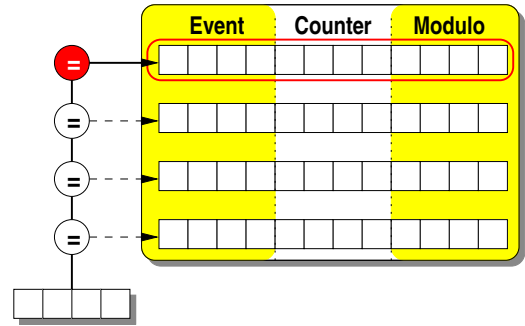


Figure 3: Basic Structure of the Associative Counter Array

tag fields. Upon a hit, the corresponding counter is increased; otherwise the tag is inserted into the array. If the ACA is completely filled, entries may be replaced according to suitable strategies such as LRU and LFU. Messages to high-level monitoring instances are generated when either an entry is replaced, a counter reaches its associated modulo value, or by forced reads, making it possible to easily vary the time interval of communication between low- and high-level monitoring. Figure 3 depicts the basic ACA structure.

Again inspired by the hormone-concept, messages from low-level monitoring broadcast into the communication network, ensuring that these messages will be visible to any high-level monitoring cell in communication range; following the hierarchical nature of biological organisms, this range may be limited, enabling the implicit partition of a whole system in individual, but coupled sub-systems.

High-level monitors then store incoming messages in a so-called *event list* of defined length in combination with a timestamp. It is hence assured that only the newest messages are used for classification. Using two or more lists enable high-level monitors to classify the state of the (sub-)system in a more fine-grained manner. Various classification algorithms can then be applied to events stored in the lists.

3 Monitoring-driven optimization: an application example

The performance of a P2P-based many-core architecture is directly related to its network performance. Huge latencies for data transfer between two cores directly affect performance eventually resulting in violated real-time constraints. Furthermore, long communication distances increase the overall network traffic and therefore the likelihood of congestion or buffer overflows. Hence, minimizing the distance between two communicating tasks is one optimization scenario.

With ACAs being part of each DodOrg processing cell, the network traffic is monitored on hardware level. If required, monitored traffic may be limited to certain message types or communication partners using event masking, so that e.g. only payload but not monitoring-related messages are counted, or only communication between certain cells or cell groups is tracked.

Based on the accumulation of events related to network traffic, regions of high traffic can be identified. For this, the related event counter values are accumulated and normalized as

shown in Equation 1.

$$\text{Traffic Utilization} = \frac{\sum \text{counter values}}{\text{max Messages Per Tick} * (\text{tick}_{\text{newest event}} - \text{tick}_{\text{oldest event}})} \quad (1)$$

Using more detailed analysis, possible candidates for task migration can be identified so that communicating tasks may be placed on nearby cells. For this, a histogram function is used for identifying frequently communicating tasks. Within a hierarchically organized architecture like DodOrg, two basic cases may occur depending on the fact that monitoring may be restricted to individual regions or sub-systems. If two communicating cells reside within the same monitoring region, then a more suitable task-to-cell mapping may be directly found within this region and according task migration occurs. If, however, the cells reside in different regions the affected tasks shall migrate to the region borders. Additionally, further, higher-level monitoring may be invoked providing a more global system view.

4 Conclusion and Outlook

In this abstract we presented a monitoring infrastructure for use in heterogeneous multi- and many-core architectures as demonstrated by the DodOrg architecture. We shortly introduced the scalable and decentral concept of this infrastructure, consisting of low-level monitoring for data gathering and aggregation, and high-level monitoring used for data analysis, classification, and triggering further optimization processes. Using an illustrative example, we showed how this monitoring infrastructure can be used to identify regions within a heterogeneous multi-core architecture with high network traffic and how the network traffic can be optimized by minimizing the distance between communicating tasks.

References

- [BBB⁺06] Jürgen Becker, Kurt Brändle, Uwe Brinkschulte, Jörg Henkel, Wolfgang Karl, Thorsten Köster, Michael Wenz, and Heinz Wörn. Digital On-Demand Computing Organism for Real-Time Systems. In Wolfgang Karl, Jürgen Becker, Karl-Erwin Großpietsch, Christian Hochberger, and Erik Maehle, editors, *Workshop Proceedings of the 19th International Conference on Architecture of Computing Systems (ARCS'06)*, volume P81 of *GI-Edition Lecture Notes in Informatics (LNI)*, pages 230–245, March 2006.
- [BK08] Rainer Buchty and Wolfgang Karl. Design Aspects of Self-Organizing Heterogeneous Multi-Core Architectures. In *Information Technology 5/2008 (Issue on Computer Architecture Challenges)*, pages 293–299. Oldenbourg Wissenschaftsverlag, October 2008.
- [BKK08] Rainer Buchty, David Kramer, and Wolfgang Karl. An Organic Computing Approach to Sustained Real-time Monitoring. In *Proceedings of WCC2008/BICC (IFIP Vol.268)*, pages 151–162. Springer, September 2008.