Transregional Collaborative Research Centre 89

Invasive Computing

Friedrich-Alexander-Universität Erlangen-Nürnberg
Karlsruher Institut für Technologie
Technische Universität München

Annual Report 2014/2

Coordinator
Prof. Dr.-Ing. Jürgen Teich
Lehrstuhl für Informatik 12
Universität Erlangen-Nürnberg
Cauerstraße 11, 91058 Erlangen

Managing Director
Dr.-Ing. Jürgen Kleinöder
Lehrstuhl für Informatik 4
Universität Erlangen-Nürnberg
Martensstr. 1, 91058 Erlangen

Deputy Managing Director
Dr. Katja Lohmann
Lehrstuhl für Informatik 12
Universität Erlangen-Nürnberg
Cauerstraße 11, 91058 Erlangen

Public Relations
Dr. Katja Lohmann
Lehrstuhl für Informatik 12
Universität Erlangen-Nürnberg
Cauerstraße 11, 91058 Erlangen

Dr. Sandra Mattauch
Lehrstuhl für Informatik 12
Universität Erlangen-Nürnberg
Cauerstraße 11, 91058 Erlangen

Preface

This report summarises the activities and scientific progress of the Transregional Collaborative Research Centre 89 “Invasive Computing” (InvasIC) in 2014.

The main idea of InvasIC is to develop and investigate a completely novel paradigm for designing and programming future parallel computing systems. To support its major ideas of self-adaptive and resource-aware programming, not only new programming concepts, languages, compilers and operating systems are necessary but also revolutionary architectural changes in the design of Multi-Processor Systems-on-a-Chip (MPSoCs).

InvasIC is funded by the Deutsche Forschungsgemeinschaft in its second funding phase from July 2014 – July 2018. The research association aggregates about 60 of the best researchers from three excellent sites in Germany (Friedrich-Alexander-Universität Erlangen-Nürnberg, Karlsruher Institut für Technologie, Technische Universität München). This scientific team includes specialists in algorithm engineering for parallel algorithm design, hardware architects for reconfigurable MPSoC development as well as language, tool and application, and operating-system designers.

This year’s highlight was the successful two day review event in February, where representatives of the DFG and about 15 peer reviewers evaluated our joint results of the first funding period and the research programme of the second funding phase. We would like to thank all members of the CRC/Transregio InvasIC for their outstanding effort during the review process, which has made this success possible!

At this point, we would also like to thank all of our partners from industry and academia for the fruitful collaborations and inspiring discussions. We do hope that you will enjoy reading about the first results of our second funding phase achieved in 2014, as well as about our planned research for the years 2015 – 2018.

Jürgen Teich
Coordinator

Contents

I

Invasive Computing

1 About InvasIC

The Idea of Invasive Computing

The main idea and novelty of *invasive computing* is to introduce resource-aware programming support in the sense that a given program gets the ability to explore and dynamically spread its computations to processors similar to a phase of invasion, then to execute portions of code of high parallelism degree in parallel based on the available (invasible) region on a given multi-processor architecture. Afterwards, once the program terminates or if the degree of parallelism should be lower again, the program may enter a *retreat phase*, deallocate resources and resume execution again, for example, sequentially on a single processor. To support this idea of self-adaptive and resource-aware programming, new programming concepts, languages, compilers and operating systems are necessary as well as architectural changes in the design of MPSoCs (Multi-Processor Systems-on-a-Chip) to efficiently support invasion, infection and retreat operations by involving concepts for dynamic processor, interconnect and memory reconfiguration. Decreasing feature sizes have also led to a rethinking in the design of multi-million transistor system-on-chip (SoC) architectures, envisioning dramatically increasing rates of temporary and permanent faults and feature variations.

As we can foresee SoCs with 1000 or more processors on a single chip in the year 2020, static and central management concepts to control the execution of all resources might have met their limits long before and are therefore not appropriate. Invasion might provide the required *self-organising* behaviour to conventional programs for being able to provide scalability, higher resource utilisation, required fault-tolerance and, of course, also performance gains by adjusting the amount of allocated resources to the temporal needs of a running application. This thought opens a new way of thinking about *parallel algorithm design*. Based on algorithms utilising invasion and negotiating resources with others, we can imagine that corresponding programs become *personalised* objects, competing with other applications running simultaneously on an MPSoC.

First Achievements

A Transregional Collaborative Research Centre aggregating the best researchers from three excellent sites in Germany provides an ideal base to investigate the above revolutionary ideas. Starting off at basically zero in terms of invasive processor hardware, language, compiler, and operating-system availability, we have truly fostered the *fundamentals of invasive computing* in our first funding phase: These include the definition of required programming language elements for the specification of invasion operations as well as a set of constraints to argue about number, types, and state of resources that may be invaded defining the *invasive command space* (project area A). A first invasive language based on the language X10 by IBM as well as a compiler for translation of invasive X10 programs (project area C) onto a heterogeneous invasive multi-tile architecture that has also been successfully jointly architected (project area B) is meanwhile ready for experimentation on an FPGA-based prototype (project Z2). The compiler interfaces to the invasive runtime support system *i*RTSS that provides for dedicated operating-system support for invasive computing. First invasive applications exploiting different types of processor and communication resources of an invasive network-on-chip (*i*NoC) are running successfully and have shown considerable gains in resource utilisation and computational efficiencies in comparison with their non-invasive counterparts.

Next Scientific Goals

A unique jewel of invasive computing, however, has not been exploited at all so far: By the fact that resources are temporally claimed (by default) in an exclusive manner, interferences due to multiple applications sharing the same resources being the reality on today's multicore systems may be reduced if not avoided completely. Moreover, *run-to-completion* is the default mode of thread execution. Finally, memory reconfiguration and isolation as well as bandwidth guarantees on the designed network-on-chip allow us also to provide predictable QoS also for communication.

In the second funding phase, we want to play out this ace systematically by tackling (a) *predictability* of (b) *multi-objective execution qualities* of parallel invasive programs and including their (c) *optimisation* and *exploration of design space*. Our joint investigations include new language constructs to define so-called *requirements* on desired, respectively amended qualities of execution. Application-specified qualities will not only be of type performance (e.g. execution time, throughput,

etc.), but also include aspects of *security* and *fault tolerance*. Through analysis of application requirements from different domains including stream processing and malleable task applications, not only efficiency but also predictable execution qualities shall be demonstrated for applications stemming from robotics, imaging, as well as HPC. As another new yet very important facet of invasive computing, a special focus in the second funding phase is devoted to the problem of *dark silicon* and *energy-efficient computing*.

Long Term Vision

With the aforementioned fundamental investigations in mind, we intend to demonstrate that invasive computing will be a – if not the – vehicle for solving many current problems of multicore computing today by providing *resource awareness* for a mixture of *best-effort* applications and applications with *predictable quality*. We do expect that a huge application and business field in embedded system applications might be accessed through the foundations of invasive computing.

2 Participating University Groups

Friedrich-Alexander-Universität Erlangen-Nürnberg

Lehrstuhl für Hardware-Software-Co-Design

- Prof. Dr.-Ing. Jürgen Teich
- Prof. Dr.-Ing. Michael Glaß
- Dr.-Ing. Frank Hannig
- Dr.-Ing. Stefan Wildermann

Lehrstuhl für IT-Sicherheitsinfrastrukturen

- Prof. Dr.-Ing. Felix Freiling

Lehrstuhl für Verteilte Systeme und Betriebssysteme

- Prof. Dr.-Ing. Wolfgang Schröder-Preikschat
- Dr.-Ing. Daniel Lohmann

Karlsruher Institut für Technologie

Institut für Anthropomatik und Robotik

- Prof. Dr.-Ing. Tamim Asfour

Institut für Programmstrukturen und Datenorganisation

- Prof. Dr.-Ing. Gregor Snelting

Institut für Technik der Informationsverarbeitung

- Prof. Dr.-Ing. Jürgen Becker

Institut für Technische Informatik

- Prof. Dr.-Ing. Jörg Henkel
- Dr.-Ing. Lars Bauer

Technische Universität München

Lehrstuhl für Entwurfsautomatisierung

- Prof. Dr.-Ing. Ulf Schlichtmann

Lehrstuhl für integrierte Systeme

- Prof. Dr. sc. techn. Andreas Herkersdorf
- Prof. Dr.-Ing. Walter Stechele

Lehrstuhl für Rechnertechnik und Rechnerorganisation

- Prof. Dr. Michael Gerndt

Lehrstuhl für Technische Elektronik

- Prof. Dr. rer. nat. Doris Schmitt-Landsiedel

Lehrstuhl für Wissenschaftliches Rechnen

- Prof. Dr. Hans-Joachim Bungartz
- Prof. Dr. Michael Bader

II

Research Program

3 Overview of Research Program

To investigate the main aspects of invasive computing, the TRR is organised in 5 project areas:

Area A: Fundamentals, Language and Algorithm Research
Research in project area A focuses on the basic concepts of invasion and resource-aware programming as well as on language issues, algorithmic theory of invasion and on analysis and optimisation techniques for application characterisation and hybrid (mixed static/dynamic) core allocation.

Area B: Architectural Research
Project area B investigates micro- and macroarchitectural requirements, techniques and hardware concepts to enable invasive computing in future MPSoCs.

Area C: Compiler, Simulation and Run-Time Support
The focus of project area C is on software support for invasive computing including compiler, simulation and operating-system functionality as well as on design space exploration with a special focus on run-time management.

Area D: Applications
Applications serve as demonstrators for the diverse and efficient deployment of invasive computing. The applications have been chosen carefully from the domains of robotics and scientific computing in order to demonstrate distinct and complementary features of invasive computing, for example its capability to provide quality-predictable execution of parallel programs.

Z2: Validation and Demonstrator
A hardware demonstrator will serve again as the key concept for validation of invasive computing principles. It will allow for co-validation and demonstration of invasive computing through tight integration of hardware and software research results at the end of the second project phase and to decide on the further roadmap of specific hardware for invasive computing.

The four working groups **Predictability**, **Memory Hierarchy**, **Benchmarking and Evaluation** and **Power Efficiency and Dark Silicon** defined on top of these project areas support the interdisciplinary research.

Research Area	Project	
A: Fundamentals, Language and Algorithm Research	Basics of Invasive Computing *Prof. Dr.-Ing. J. Teich, Prof. Dr.-Ing. G. Snelting, Dr.-Ing. S. Wildermann*	A1
	Design-Time Characterisation and Analysis of Invasive Algorithmic Patterns *Prof. Dr.-Ing. M. Glaß, Prof. Dr. M. Bader*	A4
B: Architectural Research	Adaptive Application-Specific Invasive Microarchitecture *Prof. Dr.-Ing. J. Henkel, Dr.-Ing. L. Bauer, Prof. Dr.-Ing. J. Becker*	B1
	Invasive Tightly-coupled Processor Arrays *Prof. Dr.-Ing. J. Teich*	B2
	Power-Efficient Invasive Loosely-Coupled MPSoCs *Prof. Dr.-Ing. J. Henkel, Prof. Dr. sc. techn. A. Herkersdorf*	B3
	Hardware Monitoring System and Design Optimisation for Invasive Architectures *Prof. Dr. D. Schmitt-Landsiedel, Prof. Dr.-Ing. U. Schlichtmann*	B4
	Invasive NoCs — Autonomous, Self-Optimising Communication Infrastructures for MPSoCs *Prof. Dr.-Ing. J. Becker, Prof. Dr. sc. techn. A. Herkersdorf, Prof. Dr.-Ing. J. Teich*	B5
C: Compiler, Simulation and Run-Time Support	Invasive Run-Time Support System (*i*RTSS) *Prof. Dr.-Ing. W. Schröder-Preikschat, Dr.-Ing. D. Lohmann, Prof. Dr.-Ing. J. Henkel, Dr.-Ing. L. Bauer*	C1
	Simulative Design Space Exploration *Dr.-Ing. F. Hannig*	C2
	Compilation and Code Generation for Invasive Programs *Prof. Dr.-Ing. G. Snelting, Prof. Dr.-Ing. J. Teich*	C3
	Security in Invasive Computing Systems *Prof. Dr.-Ing. F. Freiling, Prof. Dr.-Ing. W. Schröder-Preikschat*	C5
D: Applications	Invasive Software-Hardware Architectures for Robotics *Prof. Dr.-Ing. T. Asfour, Prof. Dr.-Ing. W. Stechele*	D1
	Invasion for High-Performance Computing *Prof. Dr. H.-J. Bungartz, Prof. Dr. M. Bader, Prof. Dr. M. Gerndt*	D3
Z: Administration	Validation and Demonstrator *Prof. Dr.-Ing. J. Becker, Dr.-Ing. F. Hannig, Dr.-Ing. T. Wild*	Z2
	Central Services *Prof. Dr.-Ing. J. Teich*	Z
WG: Working Groups	Predictability *Prof. Dr. M. Gerndt, Prof. Dr.-Ing. M. Glaß*	WG1
	Memory Hierarchy *Dr.-Ing. L. Bauer, Prof. Dr.-Ing. G. Snelting*	WG2
	Benchmarking and Evaluation *Prof. Dr. M. Bader, Prof. Dr.-Ing. W. Stechele*	WG3
	Power Efficiency and Dark Silicon *Dr.-Ing. F. Hannig, Prof. Dr.-Ing. J. Henkel*	WG4

4 Research Projects

A1: Basics of Invasive Computing

Jürgen Teich, Gregor Snelting, Stefan Wildermann

Andreas Zwinkau, Andreas Weichslgartner

The goal of A1 is to develop a programming model and the theoretical foundations for enforcing *predictability* of invasive program execution with multiple non-functional requirements. Research focuses on (a) a formal semantics and nonstandard (dependent) type system of the invasive core language to provide resource usage guarantees and a memory model for invasive architectures, (b) programming extensions to express typical invasive programming patterns and non-functional requirements and to alleviate formal program analysis, (c) run-time management strategies for feasible and optimal program execution.

In the following, first results are presented wrt. the topics of programming extensions, design methodology, theoretical results of run-time management strategies, as well as invasion of network resources.

Programming Extensions and Language Design

For enhancing the predictability of invasive programs on heterogeneous MPSoCs, we focus on streaming applications. Besides the reservation of processing elements (aka compute cores) for the execution of such applications, it is also necessary to invade memory and network resources for exchanging data between parallel tasks. Our publication on *communication-aware programming* [Hei+14a] presents an X10 framework, which enables the programmer to invade resources for data transmission over the network-on-chip while obtaining guarantees for communication latency and/or throughput—this a prerequisite for predictability. The paper received a HiPEAC paper award.

Based on these results, Project A1, in cooperation with projects A4 and C2, works on an *actor programming model* (introduced in [RHT14]) that

```
1  /* Code for task A */
2  val codeA = (task:Task) => {
3    val c = task.getWriteChannel("B");
4    c.write("Hello World");
5  };
6  /* Code for task B */
7  val codeB = (task:Task) => {
8    val c = task.getReadChannel("A");
9    val str = c.read() as String;
10   Console.OUT.println(str);
11 };
12 /* Create tasks */
13 val taskA = Task.make("A", codeA);
14 val taskB = Task.make("B", codeB);
15 /* Connect tasks */
16 taskA.connect(taskB);
17 /* Run tasks */
18 taskA.run();
19 taskB.run();
```

Figure 4.1: Example X10 program consisting of two actors, task A and task B. A directed channel is established between both actors, and a test string is sent from task A to task B.

allows the programmer to specify concurrent program segments (actors) which perform parallel data processing based on messages exchanged over directed channels. Channels represent data dependencies and have to be mapped to memory and network resources, according to the results of [Hei+14a]. Figure 4.1 shows a source code example. This model makes it possible to (a) analyse code partitioned into actors and (b) modularise naturally the invasion of different resource configurations by different actors (i.e., the number and types of resources claimed for executing the actor network).

As a main idea of invasive computing, the resource configuration may change during the execution of a program. However, this dynamic nature of invasive programming is not entirely compatible with the current design of the X10 language, which assumes a static number of places (shared-memory domains) corresponding to the available hardware. In contrast, invasive computing assumes that *i*-lets may only access their claim. To expose this safely to the programmer, tiles not covered by the current claim are not available as places. Upon reinvasion, the claim might gain processing elements on another tile, which corresponds to an additional place. We adapted the X10 run-time system to use a more dynamic approach [BBMZ14]. For example, the list of all places is not a static list anymore, but is dynamically computed

from the current claim. the interface of distributed arrays is extended, because the data can be redistributed when additional places appear. While removing a place is no problem for the agent system, it also implicitly frees the application's data there and denies access. So by default, the agent system will not remove tiles from the claim of an application on reinvade unless explicitly allowed by the application.

Next Steps in Language Design

We investigate mechanisms that enable dynamic resource adaptation in a safe way for the programmer and in a predictable fashion. For the first year of the second funding phase, language extensions for *invasive programming patterns* are planned. For example, the concept of malleable applications has proofed to be efficient [FSS13], but is not supported yet. The difficulty lies in the fact that a malleable application must be able react to external adaptations of the set of assigned resources at any time (e.g., triggered by the run-time system). In contrast, non-malleable applications can pick specific points where adaptation is possible. A malleable application must provide a handler function, which the resource manager calls whenever the claim is changed. The handler must have access to resources which are about to be removed as well as the freshly gained resources. This implies that malleable applications cannot trade resources directly, because they cannot own the same resource at the same time.

Additionally, we plan further extensions of the set of constraints. For example, high-level requirements must be mapped to lower-level constraints. The agent system can vary its efforts to optimise resource distribution, so there could be a hint from the application how to enforce execution time and other quality of execution requirements. We also want to provide more high-level framework opportunities to ease and unify the programmer's work. Also, we plan a distributed work queue and improved support for divide-and-conquer algorithms.

Furthermore, we will provide extensions for specifying non-functional user requirements for individual actors as well as a complete actor network, e.g., regarding execution time, security, and reliability.

Analysis of Predictable Invasive Computing

For enforcing predictability of program execution on heterogeneous multicore/manycore systems, it becomes necessary to anticipate how non-functional properties depend on the resources claimed for program execution and how user requirements are affected. In this realm, we

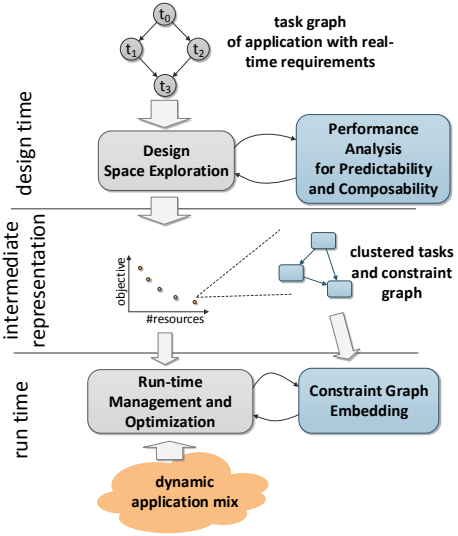


Figure 4.2: Design flow for hybrid application mapping from [Wei+14a]. DSE generates a set of design points according to which run-time management can execute a program with guaranteed execution qualities.

have developed a novel *hybrid application mapping* approach presented in [Wei+14a], which introduces a design flow according to Figure 4.2. The flow requires a formal representation of the program's data flow as input (denoted as *task graph*), which can be directly derived from the actor programming model. Then, *design space exploration (DSE)* coupled with a formal performance analysis delivers resource allocations and task mappings that are (i) verified with respect to real-time requirements and (ii) optimised with respect to additional objectives specified by the programmer (e. g., regarding execution time, power consumption, resource utilisation).

The outcome of DSE is a set of design points, each defined by its objectives, resource allocation, and a so-called *constraint graph*; the latter being illustrated in Figure 4.3. This graph captures (i) the sets of tasks that should be mapped to the same resource for this specific design point (denoted as *task clusters*), (ii) a set of *binding constraints* for each task cluster, and (iii) a set of *routing constraints* for each NoC connection that has to be established for transmitting messages between actors of one task cluster to another. This information can be handed over to the run-time system to perform run-time management and optimisation.

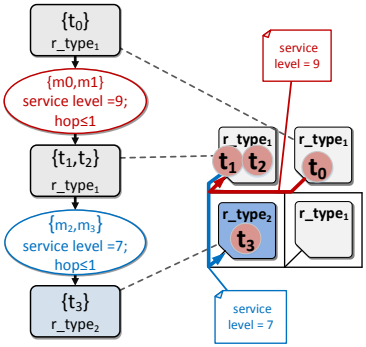


Figure 4.3: Example of a constraint graph containing three task clusters, each with binding constraints specifying the required resource type. Furthermore, two NoC connections are required, each with routing constraints specifying a minimal service level and the maximal hop distance on the NoC. A feasible binding according to this constraint graph is illustrated on the right.

Optimality, Feasibility, and Convergence Guarantees

For providing predictable program execution when the set of active applications varies over time, the run-time management has to be able to deal with multiple conflicting application and system requirements. In [WGT14], we explain how this run-time management problem may be expanded as an optimisation problem that deals with such (conflicting) application and system objectives and constraints. This problem is solved by proposing a heuristic based on non-linear programming techniques. The experiments in [WGT14] show that this approach may significantly outperform related heuristics for dynamic core and memory allocation for MPSoCs in execution time and the obtainable objective values (we tested power consumption and throughput requirements). Besides this, also theoretical results were obtained. First, *convergence guarantees* exist for the applied non-linear programming technique, which also directly apply for the presented heuristic. Second, the technique calculates an upper bound for the optimal value, which makes it possible to give *optimality guarantees*, i. e. how close the result's objective value is to the optimal value of the optimisation problem.

However, a disadvantage of our current solution—and of related work in general—is that the routing constraints associated with each design point's constraint graph are neglected. As communication latencies have an impact on the end-to-end latency of an application, neglecting the routing during run-time management could violate predictability, as

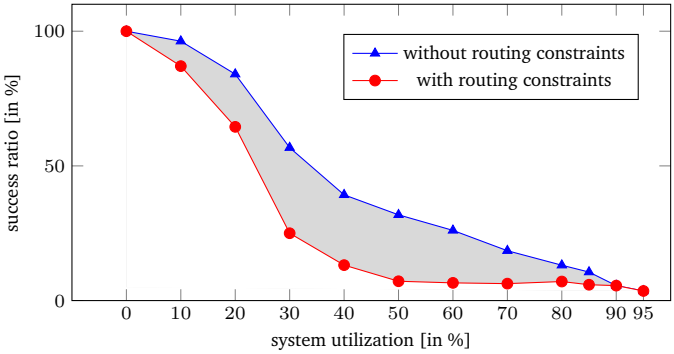


Figure 4.4: Plot illustrating the success ratio (in %) of adding programs to a 5×5 NoC depending on the utilisation of the system (given in %), based on the overall result of 6,000 test cases. The results are given for run-time management (RM) without considering routing constraints (blue) and with routing constraints (red). The divergence between both approaches is highlighted by the grey area. It indicates the serious optimism when neglecting routing. E. g., for a system utilisation of 50 %: RM without routing constraints gets an optimistic success ratio of ≈ 32 %. In fact, our approach from [Wei+14a] reveals that the actual success ratio falls below 10 % as soon as RM also considers the routing constraints, meaning that a feasible program execution is actually only possible in less than 10 % of the cases.

illustrated in Figure 4.4. As a remedy, we presented a constraint solving algorithm in [Wei+14a] that uses a design point's constraint graph to determine a set of (computation and communication) resources in a NoC-based manycore system for executing a program according to these design points. We also demonstrated how to apply this technique to repair infeasible outcomes of the run-time optimisation heuristic. Interestingly, the theoretical results regarding the optimality guarantee still apply for this combination of run-time optimisation and repair technique, which we will exploit in future work to provide even more efficient run-time optimisation techniques.

Scientific Activities

Project A1 also contributed a lot in spreading the ideas of invasive computing in invited talks and keynotes [GBTW14; Tei14a; Tei14c; Tei14b; Tei14f; Tei14d] and a DAAD winter school lecture in Tunisia [Tei14e].

Publications

[BBMZ14] M. Braun, S. Buchwald, M. Mohr, and A. Zwinkau. *Dynamic X10: Resource-Aware Programming for Higher Efficiency*. Tech. rep. 8. X10 '14. Karlsruhe Institute of Technology, 2014. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000041061.

[FSS13] P. Flick, P. Sanders, and J. Speck. "Malleable Sorting". In: *Parallel Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*. ISSN 1530-2075. 2013, pp. 418–426. DOI: 10.1109/IPDPS.2013.90.

[GBTW14] M. Glaß, M. Bader, J. Teich, and S. Wildermann. "Assisting Run-time Optimization of Many-Core Systems by Design-time Characterization". HiPEAC Spring Computing Systems Week, Barcelona, Invited Talk. May 13, 2014.

[Hei+14a] J. Heisswolf, A. Zaib, A. Zwinkau, S. Kobbe, A. Weichslgartner, J. Teich, J. Henkel, G. Snelting, A. Herkersdorf, and J. Becker. "CAP: Communication Aware Programming". In: *Design Automation Conference (DAC), 2014 51th ACM / EDAC / IEEE*. 2014, 105:1–105:6.

[RHT14] S. Roloff, F. Hannig, and J. Teich. "Towards Actor-oriented Programming on PGAS-based Multicore Architectures". In: *Workshop Proceedings of the 27th International Conference on Architecture of Computing Systems (ARCS)*. Lübeck, Germany: VDE Verlag, Feb. 25–28, 2014. ISBN: 978-3-8007-3579-2.

[Tei14a] J. Teich. "Foundations and Benefits of Invasive Computing". University of Bologna, Italy, Invited Talk in the Seminar Series Trends in Electronics. May 23, 2014.

[Tei14b] J. Teich. "Foundations and Benefits of Invasive Computing". Seminar, Mc Gill University, Montreal. July 29, 2014.

[Tei14c] J. Teich. "Introduction to Invasive Computing". Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014), Paderborn, Germany, Tutorial Talk. May 29, 2014.

[Tei14d] J. Teich. "Invasive Computing – Concepts and Benefits". Keynote, DASIP 2014, Conference on Design and Architectures for Signal and Image Processing, Madrid, Spain. Oct. 8, 2014.

[Tei14e] J. Teich. "Reconfigurable Computing for MPSoC". Invited Lecture, Winter School Design and Applications of Multi Processor System on Chip, Tunis, Tunesia. Nov. 26, 2014.

[Tei14f] J. Teich. "System-Level Design Automation of Embedded Systems". Invited Talk, Tagung Deutsche Forschungsgesellschaft für Automatisierung und Mikroelektronik e.V. (DFAM). Sept. 25, 2014.

[Wei+14a] A. Weichslgartner, D. Gangadharan, S. Wildermann, M. Glaß, and J. Teich. “DAARM: Design-Time Application Analysis and Run-Time Mapping for Predictable Execution in Many-Core Systems”. In: *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2014)*. Oct. 12, 2014, 34:1–34:10. DOI: 10.1145/2656075.2656083.

[WGT14] S. Wildermann, M. Glaß, and J. Teich. “Multi-Objective Distributed Run-time Resource Management for Many-Cores”. In: *Proceedings of Design, Automation and Test in Europe (DATE 2014)*. Dresden, Germany, Mar. 2014, pp. 1–6. DOI: 10.7873/DATE.2014.234.

A4: Design-Time Characterisation and Analysis of Invasive Algorithmic Patterns

Michael Glaß, Michael Bader

Tobias Schwarzer, Alexander Pöppl

The new Project A4 investigates existing, develops novel, and analyses invasive algorithmic patterns wrt. qualities of execution to exploit the resource awareness of invasive computing. The research focuses on (a) stencil computations and non-regular tree traversals as invasive algorithmic patterns inspired by the invasive applications from Projects D1 and D3 and (b) design-time characterisation techniques for the derivation of sets of optimised and diverse operating points by considering symmetries and system services of a given heterogeneous invasive manycore architecture. In the following, the key ideas and research directions of A4 are briefly outlined and first research results are presented.

Key Ideas and Research Directions

The management and exploitation of dynamically-adapting and heterogeneous resources substantially increases the complexity of invasive algorithm design and invasive scheduling decisions: not just a certain amount of resources, but the best suitable combination of computation and communication resources needs to be selected and invaded at run time. To enforce a predictable quality of execution, applications need to carefully define their quality requirements and provide a precise characterisation of achievable qualities, which depend on the available resources (on a given manycore platform). Moreover, exposing requirement definitions and scheduling strategies to application developers opens new opportunities for the design of invasive algorithms in the sense of a co-design of algorithms, hardware, and resource management.

By investigating and characterising novel *invasive algorithmic patterns*, Project A4 addresses the following main research questions: (a) How to derive and pass knowledge about the algorithms' and respective applications' performance characteristics on heterogeneous resources to a

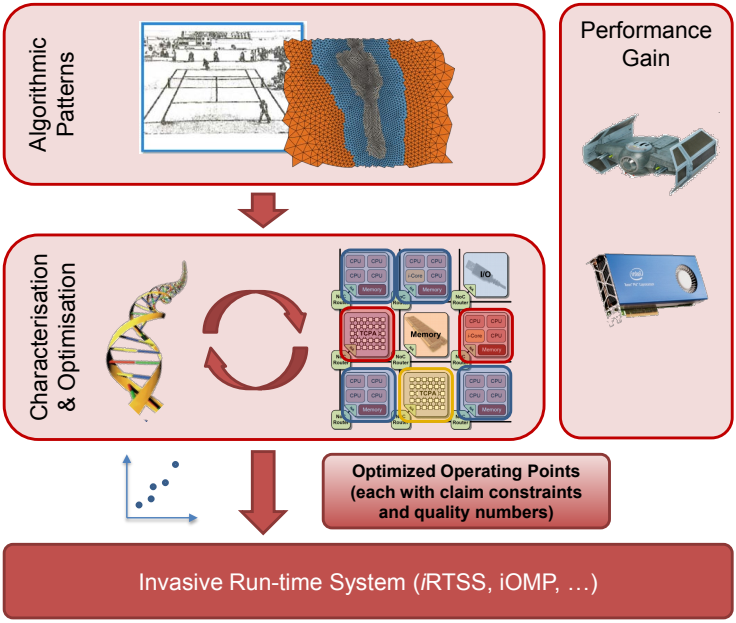


Figure 4.5: Project A4 in a nutshell: The co-design of invasive algorithmic patterns and a design-time characterisation as novel contribution to the "invasion chain" shall result in performance gains and enabling predictability by providing statically analysed operating points (tuples of claim constraints and quality numbers) to the invasive run-time system for dynamic resource allocation.

run-time system? (b) How to support online scheduling decisions based on this knowledge to improve (and/or enforce) quality requirements and system constraints? (c) How to develop invasion-aware algorithms such that they may benefit from invasion on heterogeneous platforms?

Project A4 starts from the understanding that a co-design of applications, algorithms, and invasive platforms is required to exploit the full benefits achievable by invasive computing. It therefore contributes to the entire "invasion chain" by developing invasive algorithmic patterns, by a design-time characterisation of these patterns on available platforms, by derivation of sets of optimised operating points, and finally by evaluating the performance gain on concrete invasive computing platforms. At its core, see Figure 4.5, A4 focuses on the interplay of algorithm development and invasive scheduling and mapping decisions that shall be guided by application characteristics automatically determined at design time. This characterisation that delivers which quality

numbers may be achieved by a certain claim assembly and mapping (specified by *claim constraints*) is also indispensable for program execution at predictable quality.

First Results

The big picture of A4 as well as key research questions have been presented in [GBTW14], particularly including the interplay of design-time characterisation and run-time management investigated together with Project A1. The idea of providing application characterisations which include multiple quality numbers for each operating point contributed to the theoretical investigation of multi-objective run-time management techniques presented in [WGT14]. The design space exploration techniques employed by Project A4 as part of the characterisation are integrated in a first complete flow of design-time characterisation and run-time management proposed in [Wei+14a]. There, not only multiple quality numbers but also run-time mapping requirements are already extracted automatically during characterisation and passed to the run-time management strategy.

In [GRGT14], two novel symbolic routing approaches as a core part of design-time characterisation are proposed: Using a graph-based representation of the architecture, feasible routes are encoded based on the links (*edges*) in the network, thus, avoiding the complex hop-based encoding used in literature, cf. Figure 4.6. A second approach further improves the scalability in mesh-like MPSoC architectures by pre-calculating the set of feasible—depending on the respective routing algorithm like XY—*paths* in the network. The latter approach is particularly interesting when characterising applications on heterogeneous MPSoCs with a regular communication infrastructure such as the *i*NoC (Project B5) due to the enormous design space that results from routing alternatives alone.

The potential impact of A4 for invasive applications was studied in a setup motivated by tsunami simulation, as investigated by Project D3. We started from SWE, a solver for the shallow water equations required for tsunami modelling that acts as a proxy application to evaluate approaches for performance optimisation on regular Cartesian grids. Via explicit vectorisation using intrinsics, the SWE code was optimised for modern CPU architectures, including the Intel Xeon Phi coprocessor [BBHR14]. To combine hardware performance with adaptive simulations in space and time, SWE was embedded into the adaptive mesh refinement framework Peano (`http://www.peano-framework.`

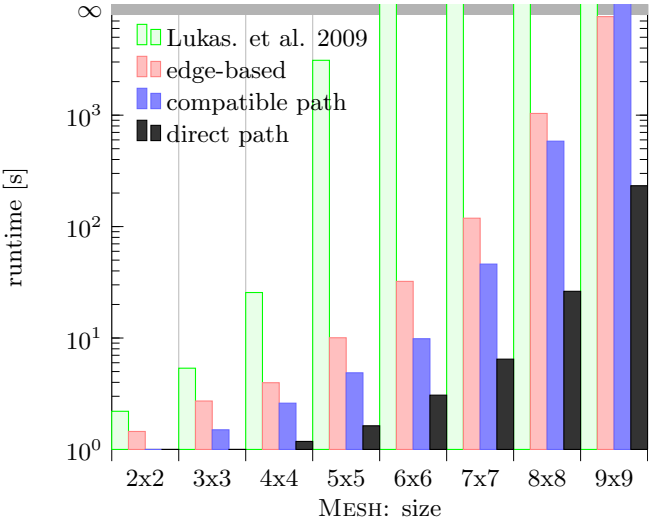


Figure 4.6: Novel symbolic routing encodings [GRGT14] significantly enhance the scalability of design-time characterisation, particularly for manycore systems with regular communication infrastructures such as the *i*NoC. Note that the run-time is given in log scale.

`org/`), which provides tree-structured meshes for discretisation and supports so-called *local time stepping*, i.e., the option to advance grid cells in time at individual time step sizes and in an asynchronous way. Embedding SWE grid patches into the leaf cells of the Peano spacetree not only leads to strongly improved performance, but offers a wide range of options for parallelisation and optimisation: parallelisation can happen within SWE patches (loop parallelism), use SWE patches as parallel work units (task model) or apply both paradigms; varying the patch size has substantial impact on the speed of computation (large patches compensate overhead of adaptivity), on memory requirement (small patches lead to better adaptivity and thus smaller memory footprint), and on parallel load balancing (large atomic patches lead to stronger imbalances). Achieving the best possible performance requires a careful tuning of all these parameters, as already the run-time of the simulation depends on the parameters in a highly non-regular way that is not easily predictable [WBUW14; Wei+14b]. For example, Figure 4.7 shows the performance (measured in cell updates per second) of an adaptive simulation depending on varying parallelisation strategy and patch size. Note that neither the performance is easily predictable nor the best

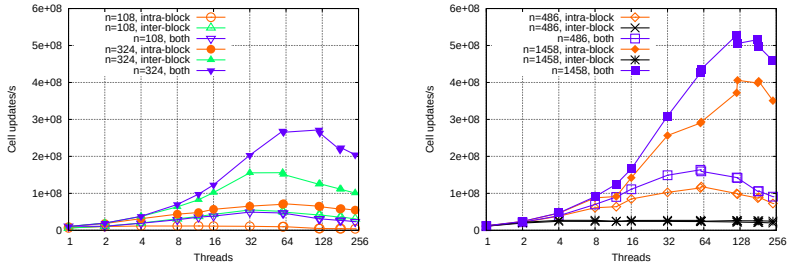


Figure 4.7: Performance of adaptive shallow water equations based on Peano and SWE with varying patch size (n) and parallelisation approach (intra-block loop parallelism vs. inter-block task parallelism vs. both/hybrid) using 1–240 threads on a 60-core Intel Xeon Phi coprocessor [WBUW14]

parameter combination (e.g., how many threads per core lead to the best performance).

A4 addresses the problem that similar to the Peano-SWE example, execution time, energy consumption, hardware efficiency or any other quality number will strongly and non-regularly depend on a large number of optimisation parameters. Characterisation of algorithms and applications is thus necessary to provide the invasive run-time system with the necessary information (and optimised operating points) to improve predictability and efficiency.

Publications

[BBHR14] M. Bader, A. Breuer, W. Hölzl, and S. Rettenberger. "Vectorization of an Augmented Riemann Solver for the Shallow Water Equations". In: *Proceedings of the 2014 International Conference on High Performance Computing and Simulation (HPCS 2014)*. Ed. by W. W. Smari and V. Zeljkovic. IEEE, 2014, pp. 193–201.

[GBTW14] M. Glaß, M. Bader, J. Teich, and S. Wildermann. "Assisting Run-time Optimization of Many-Core Systems by Design-time Characterization". HiPEAC Spring Computing Systems Week, Barcelona, Invited Talk. May 13, 2014.

[GRGT14] S. Graf, F. Reimann, M. Glaß, and J. Teich. "Towards Scalable Symbolic Routing for Multi-Objective Networked Embedded System Design and Optimization". In: *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2014)*. Oct. 12, 2014, 2:1–2:10. DOI: 10.1145/2656075.2656102.

[Wei+14a] A. Weichslgartner, D. Gangadharan, S. Wildermann, M. Glaß, and J. Teich. “DAARM: Design-Time Application Analysis and Run-Time Mapping for Predictable Execution in Many-Core Systems”. In: *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2014)*. Oct. 12, 2014, 34:1–34:10. DOI: 10.1145/2656075.2656083.

[WBUW14] T. Weinzierl, M. Bader, K. Unterweger, and R. Wittmann. “Block Fusion on Dynamically Adaptive Spacetree Grids for Shallow Water Waves”. In: *Parallel Processing Letters* 24.3 (2014), p. 1441006.

[Wei+14b] T. Weinzierl, R. Wittmann, K. Unterweger, M. Bader, A. Breuer, and S. Rettenberger. “Hardware-aware block size tailoring on adaptive spacetree grids for shallow water waves”. In: *HiStencils 2014 – 1st International Workshop on High-Performance Stencil Computations*. 2014.

[WGT14] S. Wildermann, M. Glaß, and J. Teich. “Multi-Objective Distributed Run-time Resource Management for Many-Cores”. In: *Proceedings of Design, Automation and Test in Europe (DATE 2014)*. Dresden, Germany, Mar. 2014, pp. 1–6. DOI: 10.7873/DATE.2014.234.

B1: Adaptive Application-Specific Invasive Microarchitecture

Jörg Henkel, Jürgen Becker, Lars Bauer

Artjom Grudnitsky, Carsten Tradowsky, Martin Haaß, Marvin Damschen, Srinivas Rao Kerekare

Project B1 investigates mechanisms that provide run-time adaptivity in the microarchitecture (μArch) and by using a run-time–reconfigurable fabric. We propose concepts and methods that allow invading the reconfigurable fabric and μArch within the invasive core (*i*-Core). In the following, we briefly describe our Phase II activities and results for our adaptive μArch simulation, intra-tile cache adaptation, concurrent reconfigurable fabric utilisation, and compiler-assisted identification and generation of accelerators.

Adaptive μArch

This year, we introduced a novel integrated approach for the development and evaluation of adaptive processor architectures. The contribution of this work is to transfer this concept towards adaptive simulations to enable the application developer to evaluate the necessary features of the adaptive hardware very early during the development cycle in a high abstraction layer [THMB15]. To promote these novel adaptive simulations, an intuitive framework is presented and a transparent user model is introduced. Additionally, we developed an adaptive localisation processor by using our intuitive flow [Tra+14a].

To be able to cope with the agility of dynamically adaptive systems like invasive computing systems, we introduced an adaptive cache architecture [Tra+14b]. The cache model is an HDL realisation that allows for dynamic run-time adaptations of various cache parameters and settings. Different design trade-offs are weighed against each other and a modular implementation is presented. This hardware representation makes it possible to deeply integrate the adaptive cache into the *i*-Core microarchitecture. In Phase II, we focus on intra-tile cache adaptation, i.e., the available cache memory resources can be re-allocated between the RISC cores within a tile. To reduce the cost of such a cache archi-

tecture, each RISC core is able to acquire half of the cache resources of each neighbouring RISC core. Thus, each RISC core could double its cache capacity while only marginally increasing the multiplexers and wires. By executing two benchmarks (ADPCM (MiBench) and Coremark), we are able to demonstrate the potential of the cache reallocation. Coremark executes with 4 cache tiles on $Core\,2$. ADPCM is running less cache-efficient with 4 cache tiles on $Core\,1$. Thus we reallocate 2 cache tiles from $Core\,1$ to $Core\,2$. As shown in Table 4.1, we can see a relative speedup of nearly $3\,\%$ including $880\,ns$ adaptation time. The speedup will increase, the longer the application runs using the selected cache set up. This trade-off, including the power and energy impact of the adaptation will be investigated in more detail in the next year.

Table 4.1: Relative run-time after cache re-allocation.

	Benchmark	Relative Speedup
Core 1	ADPCM (MiBench)	$-0.5\,\%$
Core 2	Coremark	$+2.8\,\%$

Reconfigurable Fabric

During the previous phase of the project, we have shown that a significant speedup can be achieved by using the reconfigurable fabric of the *i*-Core, e.g. $10\times$ for SIFT matching (used in the object recognition application by Project D1) compared to a LEON-3 with an FPU. We have observed that even when implementing complex kernels on the fabric, a part of the fabric resources was idle. Additionally, the performance improvement was available only to *i*-lets running on the *i*-Core. Therefore, one of the Phase II goals is to grant other cores in an *i*-Core tile access to the fabric, allowing concurrent execution of multiple kernels (issued by different cores on the tile) through the use of idle fabric resources.

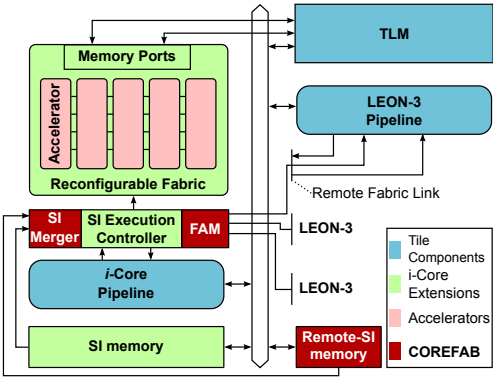


Figure 4.8: *i*-Core tile with COREFAB extensions.

To achieve this objective, we proposed the COREFAB (COncurrent

REconfigurable FABric utilization) technique in [GBH14a]. COREFAB describes hardware components (Figure 4.8) and a protocol to allow GPPs in a reconfigurable shared-memory multicore (which corresponds to a Tile in the invasive architecture) accessing the fabric of a reconfigurable core. Additionally, it introduces the concept of *on-the-fly* merging of fabric accesses from different cores at run time, to allow execution of multiple kernels concurrently. Compared to state-of-the art, COREFAB improves performance of the LEON cores in an *i*-Core tile by 1.3× on average, without reducing the performance of the *i*-Core itself.

B1

Automatic Identification of Application-Specific Accelerators

We developed a first approach to automatically identify application-specific accelerators for the *i*-Core at compile time [HBH14]. It focuses on code regions (basic block) with memory access, as these can benefit from the high-bandwidth connection from the *i*-Core to the Tile-local memory (TLM). Suitable basic blocks are obtained by identifying and unrolling inner loops of an algorithmic kernel. Within a basic block all memory accesses are extracted, analysed and then mapped to those patterns that are supported by the Address Generation Unit (AGU) of the *i*-Core.

The resulting basic block is free of direct memory accesses as those are now handled by the AGU together with the Load-Store Unit (LSU) of the *i*-Core. The remaining operations are checked to be suitable for hardware (e.g. whether or not floating-point operations are supported). The compiler then emits the machine code of the application that uses the new accelerator, the VHDL description of the accelerator, as well as multiple further files for the run-time system and tool chain.

Publications

[GBH14a] A. Grudnitsky, L. Bauer, and J. Henkel. “COREFAB: Concurrent Reconfigurable Fabric Utilization in Heterogeneous Multi-Core Systems”. In: *International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656106.2656119.

[GBH14b] A. Grudnitsky, L. Bauer, and J. Henkel. “MORP: Makespan Optimization for Processors with an Embedded Reconfigurable Fabric”. In: *Proceedings of the 22nd ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA)*. Mon-

terey, CA, USA, Feb. 26–28, 2014, pp. 127–136. DOI: 10.1145/2554688.2554782.

[HBH14] M. Haaß, L. Bauer, and J. Henkel. “Automatic Custom Instruction Identification in Memory Streaming Algorithms”. In: *International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656106.2656114.

[Hen14a] J. Henkel. “Adaptive Embedded Computing with i-Core”. Keynote Talk, 6th Workshop on Adaptive and Reconfigurable Embedded Systems, CPSWeek (APRES). Berlin, Germany, Apr. 14, 2014.

[HBGZ14] J. Henkel, L. Bauer, A. Grudnitsky, and H. Zhang. “Adaptive Embedded Computing with i-Core”. In: *ACM SIGBED Review – Special Issue on the 6th Workshop on Adaptive and Reconfigurable Embedded Systems*. Vol. 11. Extended Abstract for Keynote Talk. Oct. 2014, pp. 20–21. DOI: 10.1145/2692385.2692387.

[SBH14] M. Shafique, L. Bauer, and J. Henkel. “Adaptive Energy Management for Dynamically Reconfigurable Processors”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 33.1 (Jan. 2014), pp. 50–63. DOI: 10.1109/TCAD.2013.2282265.

[Tra+14a] C. Tradowsky, T. Gädeke, T. Bruckschlögl, W. Stork, K.-D. Müller-Glaser, and J. Becker. “SmartLoCore: A Concept for an Adaptive Power-Aware Localization Processor”. In: *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*. Feb. 2014, pp. 478–481. DOI: 10.1109/PDP.2014.118.

[THMB15] C. Tradowsky, T. Harbaum, L. Masing, and J. Becker. “A Novel ADL-based Approach to Design Adaptive Application-Specific Processors”. In: *Best of IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 2015. forthcoming.

[Tra+14b] C. Tradowsky, M. Schreiber, M. Vesper, I. Domladovec, M. Braun, H.-J. Bungartz, and J. Becker. “Towards Dynamic Cache and Bandwidth Invasion”. In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Vol. 8405. Lecture Notes in Computer Science. Springer International Publishing, Apr. 2014, pp. 97–107. DOI: 10.1007/978-3-319-05960-0_9.

B2: Invasive Tightly-coupled Processor Arrays

Jürgen Teich

Srinivas Boppu, Vahid Lari, Éricles Sousa, Frank Hannig

Project B2 investigates invasive computing on tightly-coupled processor arrays (TCPAs). These have been shown to provide a highly energy-efficient [Lar+13] and, at the same time, timing-predictable acceleration for many computationally-intensive applications that may be expressed by nested loops from diverse areas such as scientific computing and image and signal processing, to name a few.

In the first funding phase, concepts for hardware-controlled invasion through a cycle-wise propagation of invasion control signals between neighbouring processing elements (PEs) have been investigated. Not only may such decentralised parallel invasion strategies reduce the invasion overhead by two orders of magnitude w. r. t. a centralised software-based approach. Even bounds on the invasion time of invading N processing elements in $\mathcal{O}(N)$ clock cycles have been shown to be achievable. For invasion control, two variants, namely *finite state machine based (FSM-based)* and *programmable* variants have been proposed, and different 1D and 2D invasion strategies were evaluated [Han+14]. Moreover, the self-adaptive nature of invasive computing was also exploited for the purpose of dynamic power management by controlling the wake up as well as the power down of regions of processors directly by the invade and retreat signals, respectively. As invades and retreats are initiated application-driven, a TCPA may therefore nicely adopt itself to the application requirements in terms of power needs. Finally, an invasive robotic demonstration on a multi-tile architecture was developed in collaboration with other projects on the CHIPit prototyping system. As a case study, an invasive implementation of a robotic application, i. e., Harris Corner Detection, has been developed that enjoys the resource awareness provided by invasive computing by adapting its workload to the type and the number of resources available on the system [Pau+14b; Sou+14].

Predictable multicore computing is the major focus of research for

Project B2 in the 2nd funding phase. Whereas it is shown in [TTH13] that the execution time (latency, throughput) of an invasive loop program may still be determined and optimised statically and schedules may be described by parametric expressions depending only on problem size and claimed array size as run-time parameters (thanks to recent results on *symbolic loop scheduling* (Project C3)), major architectural innovations to support guarantees for multiple non-functional properties such as *fault tolerance* and *energy consumption* will drive our research. Here, in order to provide fault tolerance as a requirement of parallel execution, we will investigate and lift concepts of dual (DMR) and triple modular redundancy (TMR) known for the single processor case to the level of *replicated array processing*. In this realm, the architecture of TCPAs must be enabled to reconfigure itself for either single, double, or triple array replicated processing. In order to support also the required voting efficiently, special hardware structures and the local connectivity of TCPAs shall be investigated, respectively exploited to be able to switch on and off a replication scheme based on the *requirements* of an application or upon observation of a certain failure behaviour. Also in this context, efficient *fail-safe* techniques such as halting, application restart, and parallel recovery strategies shall be investigated for TCPAs.

Code efficiency (density) is another important goal driven by tight memory constraints within processing elements of a TCPA and effecting also the time needed to infect a claimed array with proper instruction sequences in parallel. Here, we will propose a new processing element architecture for VLIW processors called *orthogonal instruction processor (OIP)* that shall enable to run loop nests on TCPAs at a much higher code density, better energy efficiency and, as a resulting effect, also much lower expected configuration (infect) time. The main idea for avoiding an explosion in code size for complex applications is to provide a separate instruction memory and decoder basically for each functional unit inside each VLIW core.

Finally, in view of an increased importance of the problem of dark silicon, we will investigate also new *fine-grained power control* techniques for TCPAs such that the decision for a TCPA tile is not just a means to have a smaller affected area of dark silicon on an MPSoC through higher energy efficiency than on standard cores, but also a trade-off becomes possible how to fine-tune the power consumption for each TCPA application running individually at run time, e. g., based on latency of throughput slack in the applications' requirements. In the following, we report on first investigations and results in these three directions.

Safe(r) loops – fault-tolerant parallel loop processing

The high integration density of future multicore systems will inevitably lead to more and more vulnerability of the circuits to malfunction due to thermal effects, circuitry wear-outs, or cosmic radiation. However, instead of analysing error and fault effects on single cores, lifting well-known fault tolerance schemes such as DMR and TMR to the level of loop programs and their parallel processing on multicores has not been investigated yet. Starting in the second half of 2014, we are investigating approaches in which based on application *requirements*, an invasive application may request to switch on and off fault tolerance schemes for error detection and/or correction of certain parts or a parallel loop application as a whole [LTHT14]. For this purpose, the regular structure of TCPAs may either offer an application to claim (a) a non-redundant, (b) a dual-replicated (see Figure 4.9(a)), or even (c) a triple-replicated array instance for computing the parallel program in lock-step mode (see Figure 4.9(b)-(d)). This involves basic architectural research on how to adaptively provide (a) signal replication of input and output signal streams, (b) voting (hard-wired vs. dedicated functional units), and (c) error detection and recovery techniques (memory and communication error protection hardware) in a reconfigurable as well as efficient manner. Project C3 will provide the necessary compiler support for loop program transformation for fault-tolerant loop computation. The on-demand enablement of fault-tolerant loop execution on TCPAs is going to be tested for applications from Project D1 and Project D3.

Figure 4.9 illustrates first ideas of our approach: Rather than claiming a single array of processors, we propose to claim double or triple times the number of processors in a contiguous region to allow for the detection or correction, respectively, of soft errors, e. g., single event upsets (SEUs) automatically. Our proposed approach for providing on-demand DMR or TMR implementation on such processor arrays works as follows: Using the principles of hardware/software co-design, a safety-critical loop program or individual variables therein such as found in digital media, linear algebra and signal processing applications, is transformed first so to execute a *lock-step parallel schedule* of two (for DMR) or three (for TMR) identical copies of a parallel loop. Based on run-time quantitative analysis for the execution time and reliability gains in terms of probability of failure in dependence of experienced soft error rate during operation, we envision a redundancy scheme might be selected by the run-time system (Project C1) automatically to enforce a desired safety-level. Moreover, an implementation of the voting instructions in hardware is foreseen as well as an extension of the provided analy-

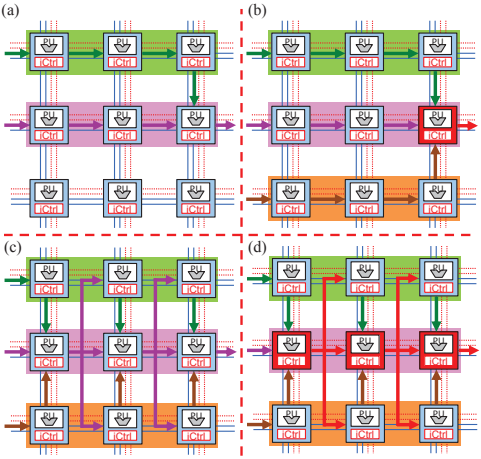


Figure 4.9: Redundancy is allocated on-demand by claiming identical subarrays to realise (a) (DMR), and (b)-(d) TMR. Comparison and voting are performed at the array boundary in (a) and (b) respectively. In (c) and (d), voting is performed in software, respectively in hardware, inside the processor array for earlier detection/correction of potential errors.

sis of reliability of on-demand voting structures for loop programs for the DMR implementation with software voting inside the PEs under the assumption also of their vulnerability against soft errors. We also plan to investigate proper techniques for fault recovery for parallel loop executions on TCPAs.

Timing Predictability

In [SGHT14], we proposed a run-time reconfigurable bus arbitration technique for providing predictable execution of concurrent applications on heterogeneous MPSoC architectures. There, a hardware/software approach has been introduced as part of a run-time framework that enables selecting and adapting different policies (i. e., fixed-priority, TDMA, and Round-Robin) such that the performance goals of concurrent applications on a tile of processors may be satisfied. To evaluate the hardware overhead, we compared our proposed solution with respect to a well-known SPARC V8 architecture supporting fixed-priority arbitration. Notably, even providing the flexibility for selecting up to three different policies, our reconfigurable arbiter needs only 25% and 7% more LUTs and slices registers, respectively. The reconfiguration overhead for changing between different policies is 56 cycles. For

programming new time slots, only 28 cycles are necessary. For demonstrating the benefits of this reconfiguration framework, we have setup a mixed hard/soft real-time scenario by considering four applications with different timeliness requirements. The experimental results show that by reconfiguring the arbiter, less processing elements are required for achieving a specific target frame rate. Moreover, by adjusting the time slots in case of TDMA bus arbitration, we can speed up soft real-time applications while still being able to satisfy the communication deadlines of hard real-time applications, if this is possible. This is elaborated in [Gan+14], where a *Dynamic Bus Reconfiguration Policy (DBRP)* is proposed that decides when to reconfigure a shared bus between Non-Preemptive Fixed Priority (NP-FP) and TDMA scheduling. The required TDMA slot sizes are computed on-the-fly before NP-FP to TDMA reconfiguration such that communication deadlines of a maximal number of active Hard Real-Time (HRT) applications may be satisfied and Soft Real-Time (SRT) applications are serviced evenly. Our proposed DBRP has been implemented on a real MPSoC platform consisting of cores connected by an AMBA AHB. The case studies have demonstrated nicely that reconfiguration of bus arbitration ensures that communication deadline constraints of HRT applications are maximally satisfied while hardware and reconfiguration overheads (in time and overhead) may be kept negligibly low.

Outlook

Our current work and next activities include a quantitative reliability and probability of failure analysis for applications that are mapped with different levels of redundancy on TCPAs. Based on application execution characteristics and a soft error rate being experienced on a processor array, the system would decide for a suitable redundancy scheme to be configured. Once detecting errors, especially in case of DMR execution, a fault is signalled to the application level by throwing an exception. We will work on the full flow to notify the presence of an error in a program mapped to a TCPA, over the run-time system up to the application level, by triggering fault exceptions. Moreover, our architecture research will be guided by the analysis and realisation of orthogonal instruction processing within the PEs of a TCPA.

Publications

[Gan+14] D. Gangadharan, É. Sousa, V. Lari, F. Hannig, and J. Teich. “Application-driven Reconfiguration of Shared Resources for

Timing Predictability of MPSoC Platforms". In: *Proceedings of Asilomar Conference on Signals, Systems, and Computers (ASILOMAR)*. (Pacific Grove, CA, USA). IEEE, Nov. 2–5, 2014. Forthcoming.

[Han+14] F. Hannig, V. Lari, S. Boppu, A. Tanase, and O. Reiche. "Invasive Tightly-Coupled Processor Arrays: A Domain-Specific Architecture/Compiler Co-Design Approach". In: *ACM Transactions on Embedded Computing Systems (TECS)* 13.4s (Apr. 2014), 133:1–133:29. ISSN: 1539-9087. DOI: 10.1145/2584660.

[LTHT14] V. Lari, A. Tanase, F. Hannig, and J. Teich. "Massively Parallel Processor Architectures for Resource-aware Computing". In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 1–7. arXiv:1405.2907 [cs.AR].

[Lar+13] V. Lari, S. Muddasani, S. Boppu, F. Hannig, M. Schmid, and J. Teich. "Hierarchical Power Management for Adaptive Tightly-Coupled Processor Arrays". In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 18.1 (Jan. 2013), 2:1–2:25. DOI: 10.1145/2390191.2390193.

[Pau+14b] J. Paul, W. Stechele, É. Sousa, V. Lari, F. Hannig, J. Teich, M. Kröhnert, and T. Asfour. "Self Adaptive Harris Corner Detection on Heterogeneous Many-core Processor". In: *Proceedings of the Conference on Design & Architectures for Signal & Image Processing (DASIP)*. Oct. 2014, pp. 1–8.

[SGHT14] É. Sousa, D. Gangadharan, F. Hannig, and J. Teich. "Runtime Reconfigurable Bus Arbitration for Concurrent Applications on Heterogeneous MPSoC Architectures". In: *Proceedings of the EUROMICRO Digital System Design Conference (DSD)*. Verona, Italy: IEEE, Aug. 27–29, 2014, pp. 74–81. DOI: 10.1109/DSD.2014.105.

[Sou+14] É. Sousa, V. Lari, J. Paul, F. Hannig, J. Teich, and W. Stechele. "Resource-Aware Computer Vision Application on Heterogeneous Multi-Tile Architecture". Hardware and Software Demo at the University Booth at Design, Automation and Test in Europe (DATE), Dresden, Germany. Mar. 24–28, 2014.

[TTH13] J. Teich, A. Tanase, and F. Hannig. "Symbolic Parallelization of Loop Programs for Massively Parallel Processor Arrays". In: *Proceedings of the 24th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. Best Paper Award. Washington, DC, USA: IEEE, June 5–7, 2013, pp. 1–9. ISBN: 978-1-4799-0493-8. DOI: 10.1109/ASAP.2013.6567543.

B3: Power-Efficient Invasive Loosely-Coupled MPSoCs

Andreas Herkersdorf, Jörg Henkel

Santiago Pagani, Anuj Pathania, Philipp Wagner, Muhammad Shafique, Thomas Wild

The overall goal of Project B3 is to optimise power/energy efficiency considering the dark silicon problem. Within the paradigm of invasive computing, the goal is to ensure that invaded *claims* remain thermally reliable while providing the *teams* for invading and executing *i*-lets for *infecting*. The pursued objectives are:

Objective 1: Improving power efficiency under dark silicon constraints.

Objective 2: Developing an adaptive system for dark silicon and energy management.

Objective 3: Modeling and online estimation of dark silicon for invasive computing systems.

The related scientific challenges include the maximisation of the performance under a given power budget or under a given energy budget, and minimising the energy consumption or peak power under a certain performance requirement. These goals require deep insight into the system. The intelligent collection and aggregation of individual data points from all over the SoC is another challenge that we address by means of powerful on-chip on-the-fly data analysis infrastructure.

Dark Silicon Management

Dark silicon brings new challenges as well as opportunities for the design community, particularly in the context of the interaction of dark silicon with thermal, reliability and variability concerns. In [SGHM14], we describe these new challenges and opportunities, and provide preliminary experimental evidence in their support.

A major step towards dealing with the dark silicon problem is through *efficient* power budgeting techniques. Usually, system designers use TDP as power budget. However, using a *single* and *constant* value as power budget, e.g., TDP, is a pessimistic approach for manycore systems. Therefore, we proposed a new power budget concept, called Thermal

Safe Power (TSP) [Pag+14], which is an abstraction that provides safe power constraints as a function of the number of active cores. Executing cores at power values below TSP results in a higher total system performance than state-of-the-art solutions, while the maximum temperature among all cores remains below the threshold level that triggers DTM. TSP is a fundamental new step and advancement towards dealing with the dark silicon problem as it alleviates the pessimistic bounds of TDP and enables system designers and architects to explore new avenues for performance improvements in the dark silicon era.

Application-Driven Power Management

B3

Hardware based closed-loop safeguards, e.g., DTM, pose a big challenge for real-time tasks. Managing the worst-case peak power usage of a chip can help toward resolving this issue. In [Mun+14], we present a scheme to minimise the peak power usage for frame-based and periodic real-time tasks on manycore processors by scheduling the sleep cycles for each active core and introduce the concept of a sufficient test for peak power consumption for task feasibility. We consider both inter-task and inter-core diversity in terms of power usage and present computationally efficient algorithms for peak power minimisation for these cases, i.e., a special case of "homogeneous tasks on homogeneous cores" to the general case of "heterogeneous tasks on heterogeneous cores".

We provide [SH14] comprehensive analysis of the computational complexity, power consumption, temperature, and memory access behaviour for next-generation High Efficiency Video Coding (HEVC). Moreover, we developed techniques for power-efficient workload balancing for multi-threaded HEVC, using DVFS [KSH14] and core-gating [SKH14].

Energy-Efficient On-Chip Memory

An energy-efficient hybrid on-chip video memory architecture (enHyV) that combines private and shared memories using a hybrid design (SRAM and STT-RAM) is presented in [Sam+14c]. The key is to leverage the application-specific properties to efficiently design and manage the enHyV. To increase STT-RAM lifetime, we propose a data management technique that alleviates the bit-toggling write occurrences. An adaptive power management is proposed for static-energy savings.

Furthermore, for the next-generation parallel HEVC, we present a novel content-driven memory pressure balancing and video memory power management scheme [Sam+14b], and an energy-efficient distributed Scratchpad Video Memory Architecture (dSVM) [Sam+14a].

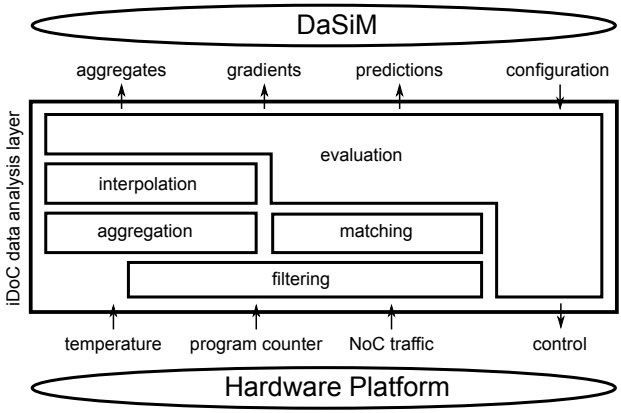


Figure 4.10: A look into iDoC with exemplary building blocks and use cases

B3

iDoC: On-Chip Data Analysis for Better Dark Silicon Management

An effective power management depends on accurate and insightful data to base its decisions on. With iDoC, we are rethinking which data can be utilised and how it can be collected and analysed.

Data is abundant in a modern SoC such as InvasIC. Hence, the challenge is to filter out what's relevant (and first, to answer the question what relevant data looks like), to aggregate it in reasonable ways, and to deliver it to the right place in the system to be consumed by DaSiM.

In addition to the normally for power management used classes of data, temperature and power consumption measurements, we utilise data traditionally used in performance diagnostics (profiling) and debugging. Figure 4.10 shows an example of that. Some data, like temperature measurements, may traverse iDoC essentially unmodified. Other data, such as the program counter of a CPU, can be used to calculate the number of executed instructions per clock cycle (IPC) or to determine which *i*-let is currently running. Data from different sources can also be combined to create even more valuable pieces of information, such as a prediction of how many *i*-lets will be executed at a future point in time.

Currently, we are in the process of defining which pieces of information are useful to guide Dark Silicon management decisions, and how this information can be obtained. In parallel, we are building a library of data analysis components which allow us to do the data analysis on-chip. A major research focus is on resource-efficient programmable units which are specialised in processing diagnostic data streams into useful information.

Workshop on "A Roadmap for EDA Research in the Dark Silicon Era" at IEEE/ACM International Conference on Computer-Aided Design (ICCAD) 2014: This workshop was intended to provide a common platform for EDA experts to discuss their vision and perspectives on the dark silicon problem, and to define a research roadmap for the next decade. More information at http://ces.itec.kit.edu/EDA4DS/.

Special Session [Sha+14] at International Conference on Hardware/-Software Codesign and System Synthesis (CODES+ISSS) 2014 at ESWeek: The goal of this special session was to expose dark silicon challenges for hardware-software co-design along with an overview of some of the early research efforts that are attempting to shape the design and run-time management of future generation heterogeneous dark silicon processors.

Prof. Henkel gave two keynotes about dark silicon [Hen14b; Hen14c].

Publications

[Hen14b] J. Henkel. "Dependability of On-Chip Systems in the Dark Silicon Era". Keynote Talk, 32nd IEEE International Conference on Computer Design (ICCD). Seoul, Korea, Oct. 19–22, 2014.

[Hen14c] J. Henkel. "The Dark Silicon Problem in Multi-Core Systems – Invasive Computing as a Solution". Keynote Talk, Thematic Session at HiPEAC Computer Systems Week. Barcelona, Spain, May 13, 2014.

[KSH14] M. U. K. Khan, M. Shafique, and J. Henkel. "Software Architecture of High Efficiency Video Coding for Many-core Systems with Power-efficient Workload Balancing". In: *Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Dresden, Germany, Mar. 24–28, 2014. DOI: 10.7873/DATE.2014.232.

[Mun+14] W. Munawar, H. Khdr, S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel. "Peak Power Management for Scheduling Real-time Tasks on Heterogeneous Many-Core Systems". In: *20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*. Hsinchu, Taiwan, Dec. 16–19, 2014.

[Pag+14] S. Pagani, H. Khdr, W. Munawar, J.-J. Chen, M. Shafique, M. Li, and J. Henkel. "TSP: Thermal Safe Power - Efficient power budgeting for Many-Core Systems in Dark Silicon". In: *IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. Best Paper Award. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656075.2656103.

[Sam+14a] F. Sampaio, M. Shafique, B. Zatt, S. Bampi, and J. Henkel. “dSVM: Energy-efficient distributed Scratchpad Video Memory Architecture for the next-generation High Efficiency Video Coding”. In: *Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Dresden, Germany, Mar. 24–28, 2014. DOI: 10.7873/DATE.2014.033.

[Sam+14b] F. Sampaio, M. Shafique, B. Zatt, S. Bampi, and J. Henkel. “Content-Driven Memory Pressure Balancing and Video Memory Power Management for Parallel High Efficiency Video Coding”. In: *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. La Jolla, CA, USA, Aug. 11–13, 2014. DOI: 10.1145/2627369.2627615.

[Sam+14c] F. Sampaio, M. Shafique, B. Zatt, S. Bampi, and J. Henkel. “Energy-Efficient Architecture for Advanced Video Memory”. In: *IEEE/ACM 33rd International Conference on Computer-Aided Design (ICCAD)*. San Jose, CA, USA, Nov. 3–6, 2014.

[SGHM14] M. Shafique, S. Garg, J. Henkel, and D. Marculescu. “The EDA Challenges in the Dark Silicon Era: Temperature, Reliability, and Variability Perspectives”. In: *Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference (DAC)*. San Francisco, CA, USA, June 1–5, 2014. DOI: 10.1145/2593069.2593229.

[SH14] M. Shafique and J. Henkel. “Low Power Design of the Next-Generation High Efficiency Video Coding”. In: *Proceedings of the 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*. Singapore, Jan. 20–23, 2014, pp. 274–281. DOI: 10.1109/ASPDAC.2014.6742902.

[SKH14] M. Shafique, M. U. K. Khan, and J. Henkel. “Power Efficient and Workload Balanced Tiling for Parallelized High Efficiency Video Coding”. In: *21st IEEE International Conference on Image Processing (ICIP)*. Paris, France, Oct. 27–30, 2014.

[Sha+14] M. Shafique, S. Garg, T. Mitra, S. Parameswaran, and J. Henkel. “Dark Silicon As a Challenge for Hardware/Software Co-design: Invited Special Session Paper”. In: *IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656075.2661645.

B4: Hardware Monitoring System and Design Optimisation for Invasive Architectures

Doris Schmitt-Landsiedel, Ulf Schlichtmann

Qingqing Chen, Elisabeth Glocker, Shushanik Karapetyan,
Daniel Müller-Gritschneder, Bing Li, Christoph Werner, Cenk Yilmaz

In the second funding phase, the overarching guideline is a focus on the satisfaction of non-functional requirements on invasive computing systems. The goal of Project B4 is to provide support to invasive computing systems with a flexible resource allocation to dynamically adapt to changing environmental conditions (like changes in supply voltage, temperature and aging) and to deal with manufacturing variations. One focus will be "fix it before it breaks": With monitoring data available, we will be able to predict that a component is approaching a hardware failure (catastrophic failures as well as parametric failures such as not meeting frequency requirements or exceeding power limitations) and give the invasive computing system a chance to react and take corrective actions in order to prevent the failure.

Another focus is on the optimisation of the entire monitoring system and therefore to analyse the types, quantities and duty cycle required of the monitors as well as the needed accuracy of monitoring data. The goal is an optimal trade-off between accurate monitoring data supplied to the invasive computing system and the used resources (primarily chip area and power consumption) to obtain those data. A special emphasis is on the power optimisation of the monitoring system in order to alleviate problems on dark silicon.

In [Glo+14], [Glo+14b] and [Glo+14a], we presented an approach (Fig. 4.11) for emulating an ASIC power and temperature monitoring system by FPGA prototyping for MPSoC computing architectures based on LEON3 processors as well as for tightly-coupled processor arrays (TCPAs).

Aging of integrated circuits can no longer be neglected in advanced process technologies. With the focus on reliability and system reliability prediction based on monitor data, we develop strategies in monitoring, analysing and later on preventing aging in the invasive computing

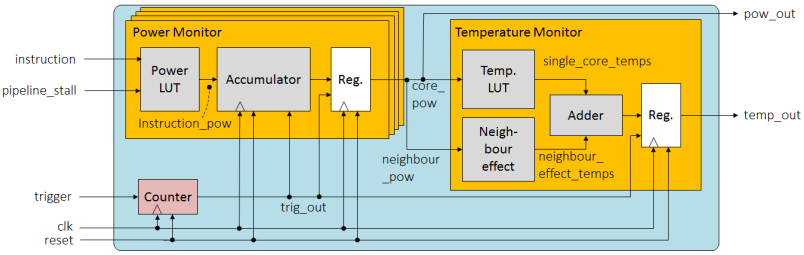


Figure 4.11: Implementation of the ASIC power and temperature monitor system (TPMon) for FPGA prototyping for MPSoC architectures based on LEON3 processors.

system. We will integrate those strategies into an emulated aging monitor for the invasive hardware demonstrator platform.

The strong dependence of the delay degradation of digital circuits on the workload is still an unsolved problem. If this workload is not known exactly, only a worst-case design guarantees correct functioning. In [LBS14], we proposed a method (Fig. 4.12) with a better-than-worst-case design that monitors the circuit periodically and takes countermeasures once the circuit degrades too much. We presented an algorithm that identifies circuit paths that might become critical (possible critical paths, PCPs) during the specified lifetime. In this approach, also local process variations are considered, otherwise it is not guaranteed that all possible critical paths are found.

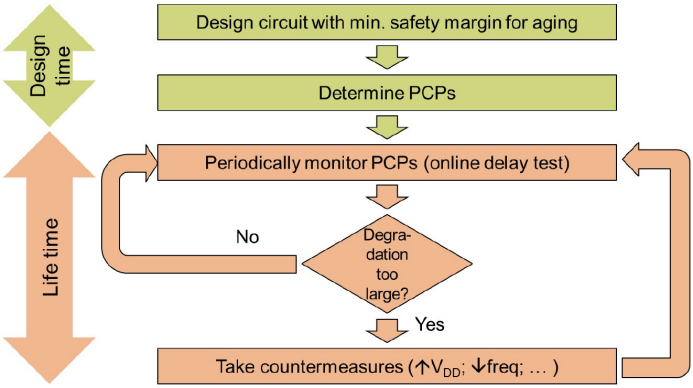


Figure 4.12: Overview of the proposed approach to monitor the aging of integrated circuits.

[Kle+14] presented a model for NBTI degradation and recovery based on trapping/detrapping that accurately describes the relaxation during detrapping and the quasi-permanent degradation. It shows good agreement with measurements from a 65 nanometer technology. This model is utilised by an aging analysis to consider variations in chip environment and workload. We showed that the analysis can be used for efficient system-level design decisions and reduces substantially estimated degradation.

We are currently trying to apply the methodology proposed in [LBS14] to the invasive computing system. We will build an aging monitoring concept for the invasive computing system, as well as an emulating approach for prototyping systems. Another focus of 2015 will be the study of in-situ delay monitoring concepts for invasive computing systems. Also the current power and temperature monitors will be improved with power mode and transient features, respectively.

Publications

[Glo+14] E. Glocker, S. Boppu, Q. Chen, U. Schlichtmann, J. Teich, and D. Schmitt-Landsiedel. “Temperature modeling and emulation of an ASIC temperature monitor system for Tightly-Coupled Processor Arrays (TCPAs)”. In: *Advances in Radio Science* 12 (2014), pp. 103–109. DOI: 10.5194/ars-12-103-2014. URL: http://www.adv-radio-sci.net/12/103/2014/.

[Glo+14a] E. Glocker, Q. Chen, A. Zaidi, U. Schlichtmann, and D. Schmitt-Landsiedel. “Emulated ASIC Power and Temperature Monitor System for FPGA Prototyping of an Invasive MPSoC Computing Architecture”. In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 14–15. arXiv:1405.2909 [cs.AR].

[Glo+14b] E. Glocker, Q. Chen, A. Zaidi, U. Schlichtmann, and D. Schmitt-Landsiedel. “Emulierung eines ASIC-Leistungsverbrauchs- und Temperaturmonitorsystems für FPGA-Prototyping eines ressourcengewahren Computersystems”. In: *16. Workshop Analogschaltungen, Wien, Österreich*. 2014.

[Kle+14] V. B. Kleeberger, M. Barke, C. Werner, D. Schmitt-Landsiedel, and U. Schlichtmann. “A Compact Model for NBTI Degradation and Recovery under Use-Profile Variations and its Application to Aging Analysis of Digital Integrated Circuits”. In: *Microelectronics Reliability* 54.6-7 (2014), pp. 1083–1089. DOI: 10.1016/j.microrel.2013.12.002. URL: http://www.sciencedirect.com/science/article/pii/S0026271413004368.

[LBS14] D. Lorenz, M. Barke, and U. Schlichtmann. “Monitoring of aging in integrated circuits by identifying possible critical paths”. In: *Journal of Microelectronics Reliability* 54 (2014), pp. 1075 – 1082. DOI: http://dx.doi.org/10.1016/j.microrel.2014.01.013. URL: http://www.sciencedirect.com/science/article/pii/S0026271414000298.

B5: Invasive NoCs – Autonomous, Self-Optimising Communication Infrastructures for MPSoCs

Jürgen Becker, Andreas Herkersdorf, Jürgen Teich

Jan Heißwolf, Andreas Weichslgartner, Aurang Zaib

Networks-on-Chip (NoCs) have emerged as the interconnect of preference for the scalability of manycore systems. Also in invasive computing, a NoC offers the needed communication infrastructure for the proposed heterogeneous tiled architectures (see Figure 4.13).

In the first funding phase, a NoC architecture consisting of hardware modules and protocols was developed and implemented. This invasive NoC (*i*NoC) [Hei+14b] provides the possibility to invade communication resources [Hei14]. Through invasion of communication links, certain Quality-of-Service (QoS) guarantees can be given in terms of upper bounds for bandwidth and latency. Besides this elementary invasion capability, various decentralised and self-optimising mechanisms were investigated. *Self-embedding*, for instance, enables the decentralised hardware-assisted mapping of communication topologies. *Rerouting* helps to reduce congestion through remapping of connections and through monitoring of the traffic inside the network adapter links can be invaded autonomously (*Auto-GS*) [Zai+13].

In the second funding phase, the capabilities of application-initiated network invasion of *i*NoC communication links shall be investigated under the common goal of increasing the *predictability* of the execution properties of concurrent applications. Therefore, hardware support inside the *i*NoC for actor-oriented task models, defined by Project A1 is needed and tighter bounds for communication latency shall be enabled through circuit switching. Apart from communication-time guarantees through virtual link invasion, other non-functional properties such as *security*, *fault tolerance*, and *energy consumption* will be the aspects under investigation. Other research tackles the NoC topology (including 3D NoCs) and cache coherence as well as novel synchronisation mechanisms.

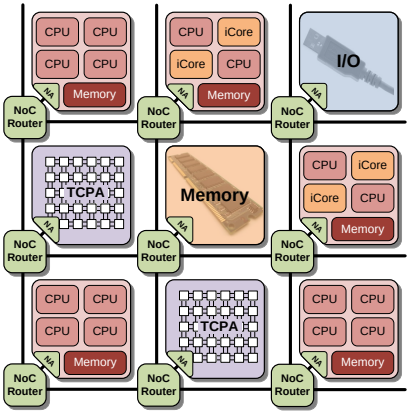


Figure 4.13: Heterogeneous invasive architecture consisting of heterogeneous tiles which are connected through the *i*NoC

B5

Support for predictable real-time stream processing

In the first funding phase, there was no native way to describe a streaming application inside InvadeX10. Now, there are first results towards an actor programming model based on task graphs to enhance the InvadeX10 language. In collaboration with Project A1 and Project C1, a holistic flow from a task graph, specified in InvadeX10, to the mapping in the *i*NoC [Hei+14a] is presented. A programming interface is introduced to express communication requirements at the language level. These requirements are evaluated by an operating-system component, which configures the communication hardware accordingly. The proposed concept enables an intuitive use of NoC features like end-to-end connections and Direct Memory Access (DMA). The speed up for a matrix multiplication exploiting *i*NoC features such as guaranteed service (GS) and prefetching is shown in Figure 4.14.

In [Wei+14a], together with Project A1 and Project A4, an *i*NoC model was incorporated into a hybrid mapping methodology to enable run-time predictability. Hybrid mapping is a combination of design-time analysis of application mappings with run-time management. A key concept here is *composability* which makes it possible to obtain performance guarantees for an individual application in isolation because resource reservations bound the interference effects of other concurrent applications. Guaranteed service connections in the *i*NoC provide exactly the needed resource reservation mechanism to bound the communication latency. The information about the resource reservation of

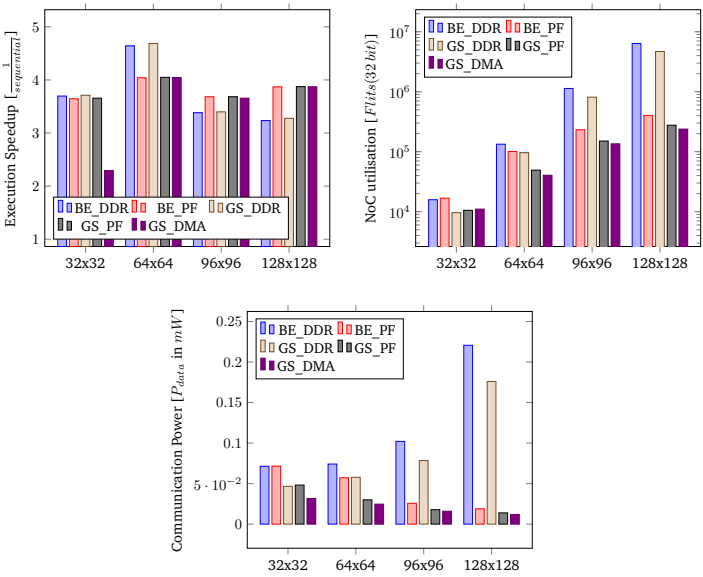
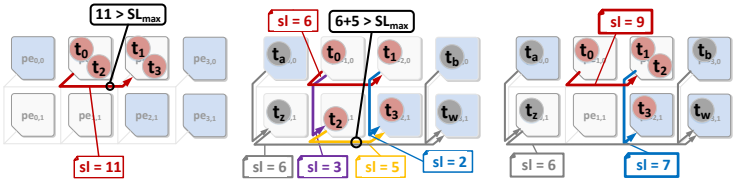


Figure 4.14: Parallel matrix multiplication executed with different settings on a 4 tile prototype of the tiled architecture. Invasive mechanisms (prefetching and guaranteed service) are compared against conventional execution. The plots compare best effort (BE) and guaranteed service (GS), which needs to be explicitly invaded. Three variants of the algorithm are tested: DDR stores the matrix in external DDR3 memory, PF explicitly prefetches data into tile-local memory, and DMA exploits our NoC's direct memory transfer feature. The matrix size varies between 32x32 and 128x128 [Hei+14a].

Pareto-optimal mappings, found during a phase of design-time analysis, are handed over as a *constraint graph* to the run-time system. There, a backtracking algorithm solves a *constraint satisfaction problem* to find a feasible mapping in the system with preoccupied resources. It is also shown that neglecting communication or assuming unlimited network resources leads to mappings which are infeasible and thus not applicable for predictable application execution. In fact, Figure 4.15 exemplifies this with three different mappings of the same application. Only one mapping is feasible whereas the others are infeasible due an overutilisation of network resources (only GS connections with an overall service level (SL) of 10 can be invaded in this example).



(a) Infeasible mapping 1 **(b)** Infeasible mapping 2 **(c)** Feasible mapping 3

Figure 4.15: Several mappings of a task graph (red nodes) with real-time requirements onto a 2x2 *i*NoC architecture. The grey nodes (t_a, t_b, t_z, t_w) denote an already mapped application. (a) and (b) are infeasible due to overutilisation of shared *i*NoC resources (maximal service level (SL) is 10), while (c) represents a feasible mapping [Wei+14a].

Fault-tolerant communication

B5

With shrinking technology sizes, the presence of permanent and transient faults is inevitable. Here, the question is how to detect, localise and how to react to such errors. The *i*NoC currently supports connection-oriented GS communication as well as packet-based Best Effort (BE) communication. In order to ensure GS and BE communication when faults are present, a *second network layer* is implemented for the *i*NoC. A distributed *localisation strategy* is utilised to identify erroneous routers and configuring the second network layer accordingly. The implementation and evaluation of this fault tolerance concept was started in 2014 and will be finished in the mid of 2015.

Inter-tile Task Spawning Support

In NoC-based distributed shared-memory architectures, the synchronisation overhead for spawning a task on a remote tile may lead to high performance penalties. In order to reduce the synchronisation delays during remote task spawning in an invasive architecture, the network adapter architecture is developed to support task spawning between tiles by employing efficient synchronisation mechanisms [Zai+15]. The proposed NA hardware support offloads the software from handling the synchronisation during remote task spawning and hence results in achieving better overall performance. Simulation results highlight that the proposed hardware architecture may improve the performance by up to 42% in comparison to existing state-of-the-art approaches. The FPGA prototype is also used to depict the benefits of the proposed approach for real-world applications.

Integration Work and Demonstration

As in previous years, lot of effort was put into implementation and prototyping of the invasive architecture as shown in Figure 4.13. However, this effort is mandatory to provide a hardware platform including the novel features of the B Projects for the research of the projects of areas C and D. When setting up a hardware architecture with novel components, *debugging* plays a key role [FHMB14]. In order to allow a deep insight into the *i*NoC hardware during system operation, a real-time debugging infrastructure was implemented in the *i*NoC routers [FHB14]. This infrastructure allows a bit-wise capturing and observance of data-transmissions in order to identify and localise hardware and software bugs.

Outlook

In 2015, Project B5 will investigating efficient power gating mechanisms for *i*NoC communication resources under the work package power efficiency. The impact of these mechanisms on real-world application shall be investigated in collaboration with Project D3. Also, Project B5 will further contribute to the demonstrator integration activities of the Project Z2. A first version of a fault-tolerant *i*NoC will be released in 2015 and used in the demonstrator.

Publications

[FHB14] S. Friederich, J. Heisswolf, and J. Becker. “Hardware/software debugging of large scale many-core architectures”. In: *Proceedings of the 27th Symposium on Integrated Circuits and Systems Design (SBCCI)*. IEEE, Sept. 2014, pp. 1–6.

[FHMB14] S. Friederich, J. Heisswolf, D. May, and J. Becker. “Hardware prototyping and software debugging of multi-core architectures”. In: *Proceedings of the Synopsys Users Group Conference (SNUG)*. Munich, Germany, 2014.

[Hei14] J. Heisswolf. “A Scalable and Adaptive Network on Chip for Many-Core Architectures”. Dissertation. Institut für Technik der Informationsverarbeitung, Karlsruhe Institute of Technology, Nov. 2014.

[Hei+14a] J. Heisswolf, A. Zaib, A. Zwinkau, S. Kobbe, A. Weichslgartner, J. Teich, J. Henkel, G. Snelting, A. Herkersdorf, and J. Becker. “CAP: Communication Aware Programming”. In: *Design Automation Conference (DAC), 2014 51th ACM / EDAC / IEEE*. 2014, 105:1–105:6.

[Hei+14b] J. Heisswolf, A. Zaib, A. Weichslgartner, M. Karle, M. Singh, T. Wild, J. Teich, A. Herkersdorf, and J. Becker. “The Invasive Network on Chip - A Multi-Objective Many-Core Communication Infrastructure”. In: *Proceedings of the first International Workshop on Multi-Objective Many-Core Design (MOMAC) in conjunction with International Conference on Architecture of Computing Systems (ARCS)*. Lübeck, Germany: IEEE, Feb. 25, 2014.

[Wei+14a] A. Weichslgartner, D. Gangadharan, S. Wildermann, M. Glaß, and J. Teich. “DAARM: Design-Time Application Analysis and Run-Time Mapping for Predictable Execution in Many-Core Systems”. In: *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2014)*. Oct. 12, 2014, 34:1–34:10. DOI: 10.1145/2656075.2656083.

[Zai+13] A. Zaib, J. Heisswolf, A. Weichslgartner, T. Wild, J. Teich, J. Becker, and A. Herkersdorf. “AUTO-GS: Self-optimization of NoC Traffic Through Hardware Managed Virtual Connections”. In: *Proceedings of the 16th Euromicro Conference on Digital System Design (DSD)*. Santander, Spain: IEEE, Sept. 4–6, 2013, pp. 761–768. DOI: 10.1109/DSD.2013.87.

[Zai+15] A. Zaib, J. Heisswolf, A. Weichslgartner, T. Wild, J. Teich, J. Becker, and A. Herkersdorf. “Network Interface with Task Spawning Support for NoC-based DSM Architectures”. In: *28th GI/ITG International Conference on Architecture of Computing Systems (ARCS)*. To appear. Springer Lecture Notes on Computer Science (LNCS), 2015.

C1: Invasive Run-Time Support System (*i*RTSS)

Wolfgang Schröder-Preikschat, Daniel Lohmann, Jörg Henkel, Lars Bauer

Benjamin Oechslein, Jens Schedel, Christoph Erhardt, Gabor Drescher, Sebastian Kobbe

Project C1 investigates operating-system support for invasive applications. It provides methods, principles and abstractions for the application-aware *extension*, *configuration* and *adaptation* of invasive computing systems. These are technically integrated into the *invasive Run-time Support System* (*i*RTSS), a highly scalable native operating system in close contact and constant touch with a standard Unix-like host operating system. The project works address special-purpose MPSoC-based as well as general-purpose multi/manycore machines.

Outcome of the first funding phase, which ended mid 2014, was (1) OctoPOS, a hardware-centric and lock-free parallel operating system of event-based kernel architecture, (2) an aspect-oriented generator for hardware-tailored cross-tile RPCs and (3) a protocol machine used by our multi-agent system for application-centric bargaining of resource allocations.

In the second funding phase, the focus will be on *partial virtualisation* of selected computing resources—*claim virtualisation* in terms of invasive computing—to provide for multiprogramming and compensate for a temporary "invasion deficit" at the time those resources are requested. An invasion deficit may be functional or non-functional (e.g., in terms of predictability, protection, isolation, or dark silicon) and arises whenever, at the moment of `invade`, the number and characteristics of the resources available deviates to an intolerable degree from the resources requested. As soon as the cause for that virtualisation has ceased to exist, *devirtualisation* shall take place. A central role therein is played by the *strategic control* when to virtualise which (real) resource and reconfigure which entity of *i*RTSS. The goal here is to accomplish *functional transparency* by an interface that allows for the specification of abstract resource demands as *non-functional requirements*, such as the desired degree of predictability, level of timeliness, or amount of performance for a certain computation period. Precomputed Pareto-optimal operat-

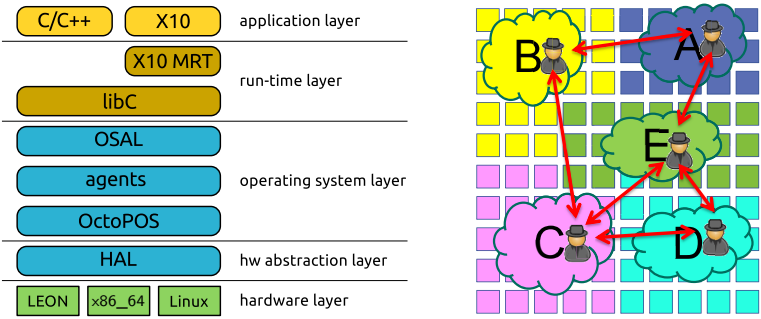


Figure 4.16: *i*RTSS Architecture. Left the overall layered organisation of *i*RTSS (blue), right the structure of the distributed agent system.

ing points as resulting from a process of application characterisation (Project A4) are used as foundation for this translation. However, these models need to be parameterised matching the application running on different system configurations, which might not be possible for the application programmer at design time. Therefore, by employing means of *on-line learning*, *i*RTSS will be able to learn and fine-tune these parameters at run time (e.g., if an initial configuration has been specified or was remembered from previous runs of the application).

C1

Architectural Overview

Figure 4.16 provides a high-level view of the current *i*RTSS architecture. Key elements are OctoPOS, the parallel operating system (POS) that implements the *mechanisms* of *i*RTSS to make all capabilities of the underlying hardware available to higher (software) levels, and the agent system, which provides global *i*RTSS *strategies* for resource management through means of self-adaption to cope with the scalability problem in large multicore systems, logically residing between the operating-system abstraction layer (OSAL) and the OctoPOS kernel.

The Configurable OctoPOS Kernel

One key aspect in the design and development of OctoPOS is to make all the capabilities of the underlying hardware available to higher (software) levels in an “unfiltrated” way by tailoring operating-system mechanisms towards the hardware capabilities—and vice versa.

In this realm, we have further advanced the implementation of the OctoPOS core abstractions [Oec+14] and their support by dedicated

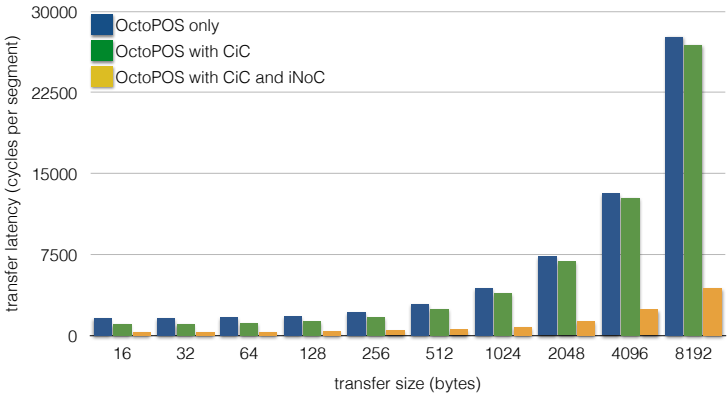


Figure 4.17: OctoPOS *i*-let transfer latencies with respect to payload and hardware support.

hardware (in close collaboration with Project B3 and Project B5) as well as the possible mapping to commodity hardware [Hof14] by "clever" exploitation of their capabilities. On an Infineon AURIX, for instance, we thereby achieve predictable cross-core task activation (i.e., *i*-let infection) latencies as low as 65 clock cycles [MDSPL14]. However, the AURIX features a single-tiled shared-memory architecture (which cannot be expected to scale to hundreds or thousands of cores [Nür+14]) and the respective *i*-lets are basically just events. Hence, *i*-let transfer is much simpler than on the invasive hardware, which exhibits its great benefits as soon as *i*-lets have to cross tile boundaries and carry a payload. Figure 4.17 depicts the respective transfer latency gains of OctoPOS if the hardware features a C*i*C and *i*NoC.

Nevertheless, these benefits also stem from the design of OctoPOS itself. In 2014, we have developed an MPI layer that runs on top of OctoPOS and maps MPI operations to the invasive model to feature better comparison of OctoPOS to other systems. Figure 4.18 shows the results of a comparison to Linux when running the well-known NASA Advanced Supercomputing Parallel Benchmarks on up to 8 Leon cores (not featuring invasive hardware extensions). In all cases, OctoPOS outperforms Linux and generally also scales better than Linux with the number of employed cores. In collaboration with Project D1, we could show that the benefits are even higher when running a resource-aware application [Pau+14a].

We provide OctoPOS for a variety of platforms—with and without dedicated hardware support. The x86_64 bare-metal and Linux "guest

mode" family members (Figure 4.16) not only ease the development and evaluation of the invasive software stack (applications, X10 extensions, compiler). They also make it possible to transfer recent advances in our techniques for energy-aware development and precise energy estimations [Hön+14; WHKSP14] to the *i*RTSS development to support the management of dark silicon in the second funding phase. Also with respect to the second funding phase, we have started to investigate tailorable and problem-specific means for basic isolation [Dan+14].

The Agent System

The resource management strategies of the agent system rely on on-the-fly estimates of different core allocations with respect to functional and non-functional application properties.

In [KBH15], we proposed an adaptive on-the-fly application performance model that allows to learn and predict the application performance for different core allocations. This knowledge will later be used for auto-tuning of claims to match the expressed performance demands (for instance, the execution of the application's main loop 30 times per second). The adaptive on-the-fly application performance model allows reacting to spontaneous workload variations and it considers topological properties of resources. Extensive evaluations using the agent system design space exploration tool developed in the first funding phase show that the average estimation error is reduced from 14.7 percent to 4.5 percent, resulting in high quality of on-the-fly adaptive application mappings.

To evaluate the adaptability of the application performance model, we simulated an application scenario in which the number of applications increases abruptly, shown in Figure 4.20. The resulting application mappings achieved when using our application performance model are almost always better than the mappings from state-of-the-art multi-application multi-step (MAMS) application mapping. The adaptation of our model to the new operating conditions is clearly visible, i.e. the application mapping efficiency (overall achieved speedup per core) increases with each application-re-mapping iteration after the abrupt workload change. The application mapping achieved by using our resource-aware estimation model resulted on average in a 4.3 percent improvement compared to MAMS.

To evaluate the overhead of the application performance model, we measured the execution time of the implementations on an Intel i5-2500 CPU and found that for estimating the speedup of an application running on 40 cores, Downey's topology-agnostic model required 31 ns

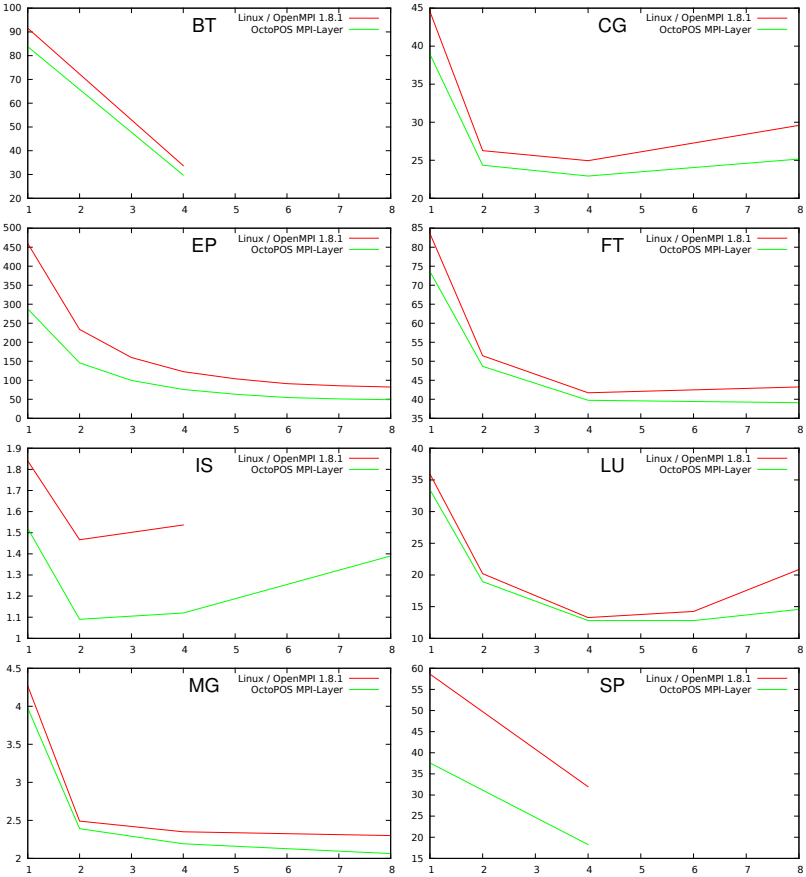


Figure 4.18: The OctoPOS MPI layer in comparison to Linux with OpenMPI 1.8 when running the NAS Parallel Benchmarks on an 8-core SPARC Leon design at 50 MHz. Depicted is the total run time in *s* (y-axis) over the number of employed cores (x-axis) for five kernels and three pseudo applications: *integer sort* (IS), *embarrassingly parallel* (EP), *conjugate gradient* (CG), *multigrid* (MG), *Fourier transformation* (FT) *block tri-diagonal solver* (BT), *scalar penta-diagonal solver* (SP), *lower–upper Gauss–Seidel solver* (LU).

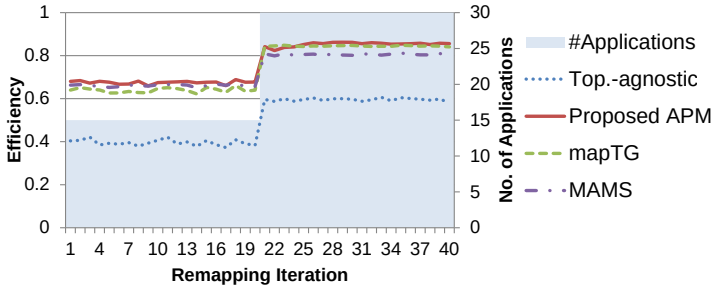


Figure 4.19: Mapping efficiency for an abrupt change in the workload when using different application performance models

(i.e., about 100 cycles) and our resource-aware approach required 1208 ns. The task-mapping heuristic required even 2 ms, a factor 2000 slower per speedup estimation. The execution time does not depend on the number of cores in the system; only the number of cores that is assigned to an application has an influence – additional measurement results for 10 cores and 100 cores are shown in Figure 4.20. In all cases, the overhead evaluations have shown that our application performance model is practicable and feasible.

C1

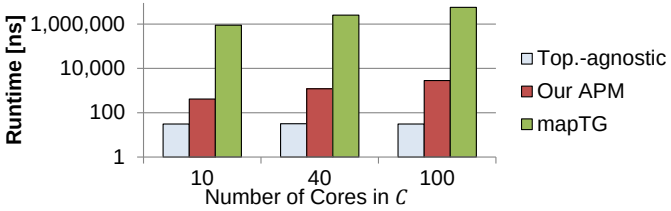


Figure 4.20: Time (in ns) required to estimate the speedup of an application when using different application performance models

Outlook

Based on adaptive on-the-fly application performance mode, the focus of future work for the agent system is on the translation of quality requirements requested by the application (such as the desired degree of predictability, the level of timeliness or the amount of performance for a certain computation period) to actual resource demands and the resulting formation of claims in cooperation with Project A4 as well as

the step-wise integration of the dark silicon management achievements by Project B3.

The OctoPOS kernel will be extended by new mechanisms for claim (de-)virtualisation and reconfiguration at run time, beginning (in cooperation with Project C5) with a focus on adaptable basic isolation. For OctoPOS on standard multicore platforms, we will investigate the software emulation (partial virtualisation) of InvasIC-specific hardware devices (C*i*C, *i*NoC) by means of functionally dedicated cores.

Publications

[Dan+14] D. Danner, R. Müller, W. Schröder-Preikschat, W. Hofer, and D. Lohmann. "Safer Sloth: Efficient, Hardware-Tailored Memory Protection". In: *Proceedings of the 20th IEEE International Symposium on Real-Time and Embedded Technology and Applications (RTAS '14)*. Washington, DC, USA: IEEE Computer Society Press, 2014, pp. 37–47. URL: https://www4.cs.fau.de/~danner/Publications/2014/danner_14_rtas.pdf.

[Hei+14a] J. Heisswolf, A. Zaib, A. Zwinkau, S. Kobbe, A. Weichslgartner, J. Teich, J. Henkel, G. Snelting, A. Herkersdorf, and J. Becker. "CAP: Communication Aware Programming". In: *Design Automation Conference (DAC), 2014 51th ACM / EDAC / IEEE*. 2014, 105:1–105:6.

[Hof14] W. Hofer. "Sloth: The Virtue and Vice of Latency Hiding in Hardware-Centric Operating Systems". PhD thesis. Erlangen: Friedrich-Alexander-Universität Erlangen-Nürnberg, 2014. URL: http://opus4.kobv.de/opus4-fau/files/4875/WanjaHoferDissertation.pdf.

[Hön+14] T. Hönig, H. Janker, O. Mihelic, C. Eibel, R. Kapitza, and W. Schröder-Preikschat. "Proactive Energy-Aware Programming with PEEK". In: *2014 Conference on Timely Results in Operating Systems (TRIOS '14)*. Broomfield, CO: USENIX Association, Oct. 2014. URL: https://www.usenix.org/conference/trios14/technical-sessions/presentation/hoenig.

[KBH15] S. Kobbe, L. Bauer, and J. Henkel. "Adaptive on-the-fly Application Performance Modeling for Many Cores". In: *Proceedings of Design, Automation and Test in Europe Conference (DATE)*. Grenoble, France, Mar. 9–13, 2015.

[MDSPL14] R. Müller, D. Danner, W. Schröder-Preikschat, and D. Lohmann. "MultiSloth: An Efficient Multi-Core RTOS using Hardware-Based Scheduling". In: *Proceedings of the 26th Euromicro Conference on Real-Time Systems (ECRTS '14)*. (Madrid, Spain). Washington, DC, USA: IEEE Computer Society Press, 2014, pp. 289–198. ISBN: 978-1-4799-5798-9. DOI: 10.1109/ECRTS.2014.30.

[Nür+14] S. Nürnberger, G. Drescher, R. Rotta, J. Nolte, and W. Schröder-Preikschat. "Shared Memory in the Many-Core Age". In: *Proceedings of the International Workshop on Runtime and Operating Systems for the Many-core Era (ROME 2014)*. Lecture Notes in Computer Science (LNCS). Porto, Portugal: Springer-Verlag, Aug. 26, 2014.

[Oec+14] B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, and W. Schröder-Preikschat. *OctoPOS: A Hardware-Assisted OS for Many-Cores*. Poster. Amsterdam, NL, Apr. 15, 2014.

[Pau+14a] J. Paul, W. Stechele, M. Kröhnert, T. Asfour, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, and W. Schröder-Preikschat. "Resource-Aware Harris Corner Detection based on Adaptive Pruning". In: *Proceedings of the Conference on Architecture of Computing Systems (ARCS)*. (Luebeck, Germany). LNCS 8350. Springer, Feb. 25–28, 2014, pp. 1–12. ISBN: 978-3-319-04890-1. DOI: 10.1007/978-3-319-04891-8.

[WHKSP14] P. Wägemann, T. Hönig, R. Kapitza, and W. Schröder-Preikschat. *Worst-Case Energy Consumption Analysis for Soft and Hard Energy Systems*. 11th USENIX Symposium on Operating System Design and Implementation (OSDI '14). Poster. Oct. 2014.

C2: Simulative Design Space Exploration

Frank Hannig

Vahid Lari, Sascha Roloff

In the first funding phase, Project C2 investigated novel simulation techniques that enable the validation and variants' exploration of all essential features of invasive computing. It had two major research fields: (a) Timed functional simulation of invasive resource-aware programs and (b) performance evaluation of individual architectures [Han+14; LTHT14] and an integrated simulation methodology [ZRWH14] to co-simulate different types of invasive architectures. In order to handle the complexity and diversity of the considered architectures as well as different invasive programming concepts, resource management and invasion strategies, new methods for the modularisation and orthogonalisation of these exploration concerns have been developed. This enabled to test and evaluate the concepts of invasive computing across all project areas, especially, without the need to have full hardware or software implementations available. Based on this foundation, the main focus of Project C2 in the second funding phase is the automatic *co-exploration* of architectures, invasion strategies and applications across all platform layers.

Timed Functional Simulation

One of the main achievements of Project C2 has been the development of the timed functional simulation environment InvadeSIM [Rol+13], which allows to simulate parallel invasive X10 applications including the invasive run-time system on virtual heterogeneous tiled architectures as they are developed in project area B. It provides (a) a fast simulation approach for simulating hundreds of competing applications on large heterogeneous tiled architectures, (b) capabilities to model such invasive multi-tile architectures, and (c) full support of the parallel X10 language constructs as well as the novel invasive programming constructs such as `invade`, `infect`, etc. as defined in the language extension InvadeX10. This

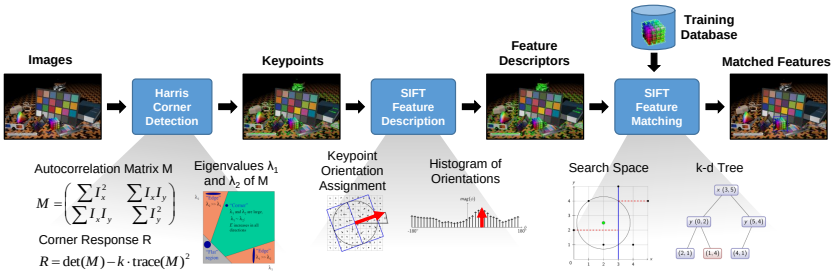


Figure 4.21: Computer vision algorithm chain, which was used in a common demonstration of Project C2 together with Project A1 and Project D1.

required a close cooperation with several projects of the CRC/Transregio (A1, B5, C1, D1, D3).

A big milestone of the simulator development was a common demonstration of the simulation of a complex computer vision application from the robotics application domain Project D1 as shown in Figure 4.21 on both homogeneous and heterogeneous architectures during the DFG review meeting in February 2014. This application detects objects, which have to be put into a training database before, in a stream of images by a chain of different algorithms. These are, among others, a Harris corner detection, a SIFT feature description, and a SIFT feature matching algorithm, which are chained together and can be executed in a pipeline

C2

```
1  /* Get IO tile for Image Source Task and Image Sink Task */
2  val ioClaim = Claim.invade(new Type(PEType.IO) && new PEQuantity(0,1));
3  /* Get RISC tile for Sobel Operator Task */
4  val riscConstraints = new Type(PEType.RISC) && new PEQuantity(0,6) &&
       new ScalabilityHint([1.0, 1.8, 2.6, 3.5] && new PlaceCoherent();
5  val sobelClaim = Claim.invade(riscConstraints);
6  /* Get TCPA or RISC tile for Harris Corner Detection Task */
7  var harrisClaim:Claim = Claim.invade(new Type(PEType.TCPA) && new
       PEQuantity(0,1));
8  if (harrisClaim.size() == 0) {
9    harrisClaim = Claim.invade(riscConstraints);
10   if (harrisClaim.size() == 0) {
11     harrisClaim = sobelClaim;
12   }
13 }
```

Figure 4.22: Extract of the X10 code of an object detection algorithm chain. Depicted is the usage of the invasive computing concepts. For each algorithm shown in Figure 4.21, a proper claim of resources is invaded in a resource-aware manner.

to process a stream of input images. Project D1 provided the algorithmic background of the application and Project C2 the implementation in X10. Furthermore, Project C2 provided first concepts and an initial implementation of an actor model in X10 [RHT14]. This will provide the basis for further investigations in Project A1. This actor model was used to realise a pipelined version of the object detection algorithm chain, where each task is wrapped into one actor and the actors are mapped to proper tiles of the simulated MPSoC architecture. Communication between the actors is simulated by a flit-accurate network-on-chip simulation model [Rol+13]. Invasive computing concepts were used here to claim tiles of resources actor-wise, see Figure 4.22. In the shown X10 code, the compute-intensive Harris corner task is accelerated best by invading a TCPA tile and providing a fallback solution, if no TCPA is available in the target architecture. Results of the simulation for both homogeneous and heterogeneous architectures were demonstrated and visualised by a self-developed graphical user interface. An important outcome of this demonstration was a first public release of the InvadeSIM simulation framework[1].

Towards Simulative Design Space Exploration

C2

Based on the afore described simulation techniques, the main focus of Project C2 in the second funding phase is the automatic *co-exploration* across all platform layers including architectures, invasion strategies and applications as shown in Figure 4.23. In this regard, Project C2 currently investigates the following main research questions:

How should an optimal invasive architecture look like for a given set of applications? In this context, we intend to investigate invasive architecture variants and evaluate them for several application scenarios for multiple objectives (e. g., cost, performance, power). For this purpose, we want to investigate simulative design space exploration techniques by systematically extending our simulation framework InvadeSIM developed in the first funding phase. In addition to available X10 applications, we want to classify and model application sets by so-called *invasive parallel patterns*.

How to model the inherent dynamic behaviour of resource-aware programs as well as of multiple competing applications? In order to model the dynamic behaviour for characteristic parallel patterns, we will introduce the notion of *dynamic invasion graphs* with a clear distinction of control flow stemming from dynamic algorithm behaviour or the

[1] `http://invasic.de/en/tp_c2.php#invadeSIM`

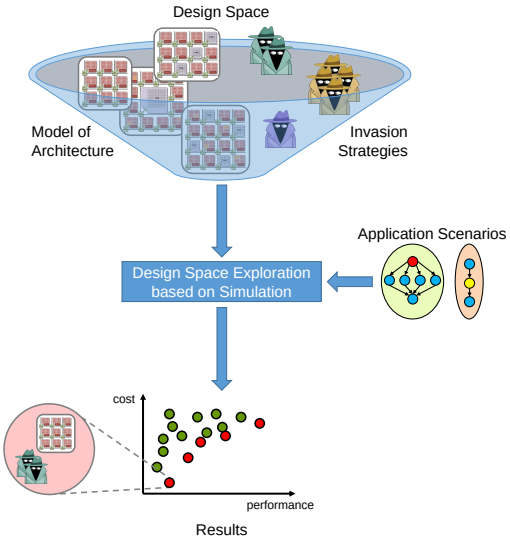


Figure 4.23: Conceptual flow of design space exploration.

invasive run-time system, and the data flow. Here, also the adaptive behaviour within resource-aware programs to changes of allocated resources will be taken into account. Successively, we intend to raise the abstraction level of these graphs by incorporating parameters, scaling functions, probabilities, or probability distributions. We believe that these graph models might provide also an important novel contribution to *multicore benchmarking*.

C2

Organisation of Scientific Events

In order to foster and discuss the increasing importance of resource-aware computing concepts in the research community, we organised an one-and-a-half day workshop on “Resource awareness and adaptivity in multi-core computing (Racing 2014)”, which was co-located with the IEEE European Test Symposium (ETS) in Paderborn, Germany, in May 2014. The Racing workshop programme included several technical sessions with talks as well as a poster and demo session. The peer-reviewed papers of the workshop were published as ePrint proceedings [HT14].

Furthermore, Frank Hannig organised a special session on “Resource-aware and domain-specific computing” at the Asilomar Conference on Signals, Systems, and Computers (ASILOMAR), in Pacific Grove, CA, USA, in November 2014.

Publications

[HT14] F. Hannig and J. Teich, eds. *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. May 29–30, 2014. 62 pp. arXiv:1405.2281 [cs.DC].

[Han+14] F. Hannig, V. Lari, S. Boppu, A. Tanase, and O. Reiche. "Invasive Tightly-Coupled Processor Arrays: A Domain-Specific Architecture/Compiler Co-Design Approach". In: *ACM Transactions on Embedded Computing Systems (TECS)* 13.4s (Apr. 2014), 133:1–133:29. ISSN: 1539-9087. DOI: 10.1145/2584660.

[LTHT14] V. Lari, A. Tanase, F. Hannig, and J. Teich. "Massively Parallel Processor Architectures for Resource-aware Computing". In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 1–7. arXiv:1405.2907 [cs.AR].

[RHT14] S. Roloff, F. Hannig, and J. Teich. "Towards Actor-oriented Programming on PGAS-based Multicore Architectures". In: *Workshop Proceedings of the 27th International Conference on Architecture of Computing Systems (ARCS)*. Lübeck, Germany: VDE Verlag, Feb. 25–28, 2014. ISBN: 978-3-8007-3579-2.

[Rol+13] S. Roloff, A. Weichslgartner, J. Heißwolf, F. Hannig, and J. Teich. "NoC Simulation in Heterogeneous Architectures for PGAS Programming Model". In: *Proceedings of the 16th International Workshop on Software and Compilers for Embedded Systems (M-SCOPES)*. St. Goar, Germany: ACM, June 19–21, 2013, pp. 77–85. ISBN: 978-1-4503-2142-6. DOI: 10.1145/2463596.2463606.

[ZRWH14] A. Zaib, P. Raju, T. Wild, and A. Herkersdorf. "A Layered Modeling and Simulation Approach to investigate Resource-aware Computing in MPSoCs". In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 51–56. arXiv:1405.2917 [cs.OH].

C3: Compilation and Code Generation for Invasive Programs

Gregor Snelting, Jürgen Teich

Matthias Braun, Sebastian Buchwald, Frank Hannig, Manuel Mohr, Tobias Schwarzer, Ericles Sousa, Alexandru Tanase, Michael Witterauf

Project C3 investigates compilation techniques for invasive computing architectures. Its central role is the development of a compiler framework for code generation as well as program transformations and optimisations for a wide range of heterogeneous invasive architectures, including RISC cores, TCPAs (tightly-coupled processor arrays), and *i*-Core reconfigurable processors.

The major achievements of the first funding phase are a compiler for the InvadeX10 language (Project A1) supporting code generation for general-purpose SPARC cores, *i*-Cores, and TCPAs [BHT14] as well as an accompanying invasive X10 run-time library targeting the OctoPOS operating system (Project C1).

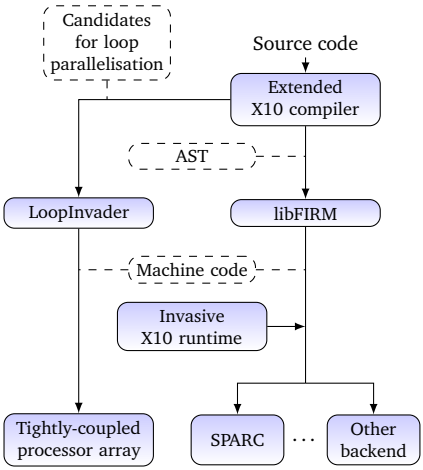


Figure 4.24: Compiler framework for Invasive Computing.

Figure 4.24 shows the structure of the compiler developed in the first funding phase. The compiler is based on an existing X10 compiler, but has been extended with new transformation phases to support tightly-coupled processor arrays (TCPAs) as well as SPARC processors and *i*-Cores through libFIRM. For RISC cores, a new transformation phase builds the intermediate representation FIRM and then generates code using a newly developed SPARC back end. For the compiler front end, a new algorithm to efficiently construct the intermediate representation in static single assignment (SSA) form directly from the source language was devel-

oped [Bra+13]. Furthermore, the extended instruction set of the *i*-Core was explored [Moh+13] in collaboration with Project B1. In particular, Project C3 investigated the use of permutation instructions to accelerate shuffle code that moves values between registers. For TCPAs, the loop compiler *LoopInvader* was developed that detects and extracts nested loop programs from a given X10 input program and transforms these loop nests into single assignment code. For invasive loop programs, *symbolic loop tiling* was proposed as a new static transformation to partition a loop nest using parameterised tile sizes to reflect a (statically) unknown number of processors. A breakthrough in *optimally scheduling loop nests symbolically* was achieved in [TTH13], the result being a static candidate set of optimal loop schedules.

Since the beginning of the second funding phase in July 2014, a major focus of research is the quest for (higher) *predictability* of non-functional aspects of invasive parallel program execution, i. e. *performance*, *fault tolerance*, and *security*. While predictable performance in terms of latency and throughput shall be proven for loop nests compiled to TCPA targets, we will also investigate compiler support to improve the predictability of invasive software and hardware in general. Moreover, the trusted handling of sensitive application data using information flow control will be a focus for RISC targets. Due to a growing vulnerability of complex MPSoC designs to failures, compilation methods exploiting fault tolerance schemes such as dual (DMR) and triple modular (TMR) redundancy for loop programs shall also be exploited systematically for TCPA targets. This poses also challenges to support such schemes on demand by dynamic *i*-Core reconfiguration and for compilation with a *fault tolerance* option.

Additionally, our research efforts are guided by the quest for *productivity* and *automation*. Automatic program *invasification* shall reduce programmer burden by automatically identifying and transforming program parts that benefit from resource invasion. Moreover, by devising compiler optimisations tailored to invasive architectures, we will facilitate the development of efficient invasive programs. Concerning TCPA targets, symbolic loop transformations need to be extended for not only being able to balance loop iterations to an unknown number of available processing elements, but also to trade-off memory and communication requirements. As such, *symbolic multi-level tiling* schemes must balance memory and communication requirements. In the following, we report on major achievements in 2014.

In early 2014, the integration process progressed to a point where invasive applications could be compiled for and run on an invasive hardware platform as proposed by Project Z2. The applications were developed by Project D3 using the invasive computing paradigm (Project A1). The compiler's (Project C3) run-time system uses operating-system interfaces developed by Project C1, and the resulting executables were run on hardware developed by the projects of area B in collaboration with Project Z2. This marks the first time that it became possible to demonstrate an uninterrupted working flow from the abstract language level to execution on the actual hardware. Furthermore, it was demonstrated successfully that the components developed by the individual subprojects do not only work in isolation, but also in conjunction with each other, thereby strongly supporting the project's hardware/software co-design approach.

From a functional perspective, this required work from Project C3 mainly on (1) the garbage collector and (2) the integration of the InvadeX10 language. To support the execution of garbage collected programs, we ported the Boehm-Demers-Weiser conservative garbage collector[2] to OctoPOS. The required interfaces, especially for stopping all *i*-lets on a tile and inspecting *i*-let stacks, were developed in collaboration with Project C1. We followed the design used by X10 for clusters and adapted it to manycore architectures, hence we have one instance of the garbage collector per tile. Currently, only the global memory is managed by the garbage collector; the tile-local memory is managed manually. Furthermore, the InvadeX10 language primitives and constraints had to be supported, to enable the compilation of invasive programs. Therefore, a matching library was developed that translates InvadeX10 calls to calls of the corresponding *i*RTSS interfaces. Especially the handling of application data on a tile during a `reinvade` operation turned out to be a challenge. Possible solutions were proposed [BBMZ14] and we plan to implement and evaluate their implementations in 2015.

Additionally, considerable effort was put into debugging, stabilising and performance tuning of the whole platform.

[2] H.-J. Boehm and M. Weiser. "Garbage collection in an unco-operative environment". In: Software: Practice and Experience 18.9 (1988), pp. 807–820.

Symbolic multi-level (hierarchical) tiling

In [TTH13; TTH14; Han+14], we provided a methodology to map invasive loop programs to a number of processors that is not known at compile time using the *locally sequential, globally parallel* (LSGP) mapping technique. It assigns the iterations within each tile to one processor for sequential (or pipelined execution). The LSGP mapping technique is well-suited for tuning the I/O demand of a partition to the given I/O capacities but requires processor-local data memory proportional to the tile size.

Based on this pioneering work, we subsequently proposed a symbolic parallelisation technique [TWTH14] realising the *locally parallel, globally sequential* (LPGS) scheme for scenarios with tight constraints on the local data memory and unknown number of processors. Here, the loop iterations within a tile are assigned to one processor each for parallel execution, yet the tiles are started to be executed in a sequential order. We proved analytically that it is possible to derive such parameterised LPGS schedules statically and proposed a mixed compile-/runtime approach: First, the compiler determines a unique intra-tile schedule responsible for scheduling the iteration points within one tile in parallel. The compiler subsequently determines all feasible inter-tile schedule candidates based on sequential scanning orders given by stride matrices. At run time, when the size of the available processor array becomes known, a compiler-generated prolog selects the latency-minimal schedule candidate according to an exact, parameterised latency formula.

C3

The resulting schedules achieve constant memory requirements, but may result in I/O requirements that exceed the capacities of the processor array architecture. To now *balance* the required I/O-capacities and local memory size at a fine scale, we plan to investigate symbolic multi-level parallelisation techniques in 2015. In particular, we aim to develop a methodology for *symbolic co-partitioning*. Co-partitioning is a hierarchical combination of both the LSGP and LGPS schemes and is so far only solved for the static case.

Fault-tolerant loop processing

Like all architectures in the age of continuous technology shrinking, TCPAs as investigated in Project B2 may suffer from transitive and permanent faults. To mitigate such scenarios, we proposed a technique called *on-demand fault-tolerant processing of loops* [LTHT14] on many-core architectures. Our approach to achieve fault tolerance uses the principles of invasive computing to claim not only one region of a

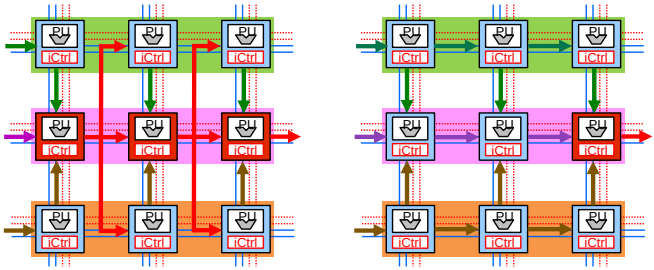


Figure 4.25: Possible voting schemes for TMR: (a) *early voting* (per-PE voting), and (b) *late voting* (voting only at the border of the array).

processor array, but instead one or two contiguous neighbour replica regions to implement redundancy schemes such as DMR (dual modular redundancy) and TMR (triple modular redundancy). Based on user requirements on whether errors, occurring in the computation of loops, should be either detected (DMR) or also corrected (TMR), the compiler shall automatically create and map replica code for a given loop program as well as schedule necessary voting operations.

Currently, based on this preliminary work, we are investigating mathematical methods for two compiler transformations to prepare a given loop for DMR or TMR execution: (a) loop replication to achieve the desired redundancy, and (b) introduction of voting variables and corresponding voting operations to test, resp. correct occurring errors.

C3

Figure 4.25 illustrates our envisioned approach, where an invasive loop application claims three linear arrays (shown in different colours) in order to implement a TMR scheme. Also visualised are two possible voter placement variants: (a) *early voting* which votes on intermediate results at the border of each PE, and (b) *late voting* which votes on results only at the border of the array. We are investigating these variants because they offer different tradeoffs in terms of their error detection latencies, fault coverage, and performance overheads, which makes each variant suitable for different use cases.

Timing predictability and co-design of communication loops

As another result in 2014, we have exploited our achievements on symbolic scheduling of loop programs—parameterised latency and throughput expressions depending on problem and claim size only—to show in [GTHT14] that a claimed processor array is able to execute its loop

schedule provided that all applications mapped on the TCPA will receive guaranteed service on the shared resources of a TCPA tile such as a shared bus. As part of this work, we presented first analytical results for timing analysis of a shared bus access (within the TCPA tile) for data transfers between the tile local memory and the TCPA buffers at the array boundary. These constraints ensure that the latency given by a parametric loop schedule may also be satisfied in view of feeding the input buffers and draining data from the output buffers fast enough to preserve the tight timing predictability of a scheduled application.

Optimisations for invasive architectures

During the first funding phase, we developed a simple and efficient SSA-construction algorithm. One advantage of the algorithm is that it applies *local optimisations* during the IR construction. Local optimisation rules, such as `x + 0` $\rightarrow$ `x` and `x & x` $\rightarrow$ `x`, do not require any global analysis and, thus, can be applied at any time during the compile run. These rules reflect the wisdom of the compiler developers about mathematical identities that hold for the operations of their intermediate representation. Unfortunately, these sets of hand-crafted rules guarantee neither correctness nor completeness.

We solved this problem by implementing a generator for local optimisations, called Optgen [Buc15]. Optgen enumerates all local optimisations up to a given pattern size and verifies them using an SMT solver. Thus, Optgen guarantees completeness and correctness of the generated optimisations rules. Furthermore, we let Optgen generate an optimisation test suite that helps to find missing local optimisations for state-of-the-art compilers. Using this test suite, we tested the latest versions of GCC, ICC and LLVM, and identified more than 50 missing local optimisations that involve only two operations.

Furthermore, we looked into accelerating inter-tile data transfers on the invasive hardware demonstrator platform. During our performance analysis of a X10 multigrid application, we noticed that it spends a significant amount of time moving data between tiles. Hence, it is attractive to optimise these operations in the compiler and run-time system, as the application program can remain unchanged.

In X10, the programmer is able to change between places using the `at` statement. Inside the `at` block, arbitrary data from the source place can be used, which is sent to the destination place before the code is executed. As X10 allows access to complex data structures, the relevant objects must be serialised before they can be sent, and the object graph must be reconstructed by the receiver via deserialisation. Hence, `at`

is potentially very expensive as it entails two traversals of the object graph and requires memory to hold the serialised data twice, once on the sending and once on the receiving side.

Our plan is to accelerate the data transfer by directly accessing the sender's object graph from the receiving tile and cloning it. In general, this does not work because as there is no cache coherence across tile boundaries, the receiver is not guaranteed to read up-to-date values. However, we developed a protocol that issues the necessary cache write-back and invalidation operations to assure correct transfers even in the incoherent case. First tests with a prototype implementation of the cloning approach showed promising results with speedups of up to 33% relative to a pure message passing approach with serialisation. We are currently investigating techniques how to best issue the cache invalidations, i. e. whether to issue them in-place directly after a foreign memory address has been accessed or to issue them "en bloc" after the clone operation is finished.

Publications

[BHT14] S. Boppu, F. Hannig, and J. Teich. "Compact Code Generation for Tightly-Coupled Processor Arrays". In: *Journal of Signal Processing Systems* 77.1–2 (May 31, 2014), pp. 5–29. ISSN: 1939-8018. DOI: 10.1007/s11265-014-0891-2.

[BBMZ14] M. Braun, S. Buchwald, M. Mohr, and A. Zwinkau. *Dynamic X10: Resource-Aware Programming for Higher Efficiency*. Tech. rep. 8. X10 '14. Karlsruhe Institute of Technology, 2014. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000041061.

[Bra+13] M. Braun, S. Buchwald, S. Hack, R. Leißa, C. Mallon, and A. Zwinkau. "Simple and Efficient Construction of Static Single Assignment Form". In: *Compiler Construction*. Ed. by R. Jhala and K. Bosschere. Vol. 7791. LNCS. Springer, 2013, pp. 102–122. DOI: 10.1007/978-3-642-37051-9_6.

[Buc15] S. Buchwald. "Optgen: A Generator for Local Optimizations". In: *Compiler Construction*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2015. forthcoming.

[GTHT14] D. Gangadharan, A. Tanase, F. Hannig, and J. Teich. "Timing Analysis of a Heterogeneous Architecture with Massively Parallel Processor Arrays". In: *DATE Friday Workshop on Performance, Power and Predictability of Many-Core Embedded Systems (3PMCES)*. Dresden, Germany: ECSI, Mar. 28, 2014. URL: http://www.ecsi.org/resource/workshop/2014/3PMCES/DATE/

paper/timing-analysis-heterogeneous-architecture-massively-parallel-processor-arrays.

[GS14] D. Giffhorn and G. Snelting. “A New Algorithm for Low-Deterministic Security”. In: *International Journal on Information Security* (Aug. 2014). DOI: 10.1007/s10207-014-0257-6. URL: http://pp.ipd.kit.edu/uploads/publikationen/giffhorn14lsod.pdf.

[Han+14] F. Hannig, V. Lari, S. Boppu, A. Tanase, and O. Reiche. “Invasive Tightly-Coupled Processor Arrays: A Domain-Specific Architecture/Compiler Co-Design Approach”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 13.4s (Apr. 2014), 133:1–133:29. ISSN: 1539-9087. DOI: 10.1145/2584660.

[LTHT14] V. Lari, A. Tanase, F. Hannig, and J. Teich. “Massively Parallel Processor Architectures for Resource-aware Computing”. In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 1–7. arXiv:1405.2907 [cs.AR].

[Loc14] A. Lochbihler. “Making the Java Memory Model Safe”. In: *ACM Transactions on Programming Languages and Systems* 35.4 (2014), 12:1–12:65. DOI: 10.1145/2518191. URL: http://pp.ipd.kit.edu/uploads/publikationen/lochbihler14toplas.pdf.

[Moh+13] M. Mohr, A. Grudnitsky, T. Modschiedler, L. Bauer, S. Hack, and J. Henkel. “Hardware Acceleration for Programs in SSA Form”. In: *International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*. Montreal, Canada, Oct. 2013. DOI: 10.1109/CASES.2013.6662518.

[Sne15] G. Snelting. “Understanding Probabilistic Software Leaks”. In: *Science of Computer Programming* 97, Part 1.0 (Jan. 2015). Special Issue on New Ideas and Emerging Results in Understanding Software, pp. 122–126. DOI: 10.1016/j.scico.2013.11.008. URL: http://www.sciencedirect.com/science/article/pii/S016764231300292X.

[Sne+14] G. Snelting, D. Giffhorn, J. Graf, C. Hammer, M. Hecker, M. Mohr, and D. Wasserrab. “Checking Probabilistic Noninterference Using JOANA”. In: *IT - Information Technology* (2014). invited article, currently under review. URL: http://pp.ipd.kit.edu/uploads/publikationen/joana14it.pdf.

[TWTH14] A. Tanase, M. Witterauf, J. Teich, and F. Hannig. “Symbolic Inner Loop Parallelisation for Massively Parallel Processor Arrays”. In: *Proceedings of the 12th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*. (Lausanne, Switzerland). Lausanne, Switzerland, Oct. 19–21, 2014, pp. 219–228.

[TTH13] J. Teich, A. Tanase, and F. Hannig. “Symbolic Parallelization of Loop Programs for Massively Parallel Processor Arrays”. In: *Proceedings of the 24th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. Best Paper Award. Washington, DC, USA: IEEE, June 5–7, 2013, pp. 1–9. ISBN: 978-1-4799-0493-8. DOI: 10.1109/ASAP.2013.6567543.

[TTH14] J. Teich, A. Tanase, and F. Hannig. “Symbolic Mapping of Loop Programs onto Processor Arrays”. In: *Journal of Signal Processing Systems* 77.1–2 (July 11, 2014), pp. 31–59. ISSN: 1939-8018. DOI: 10.1007/s11265-014-0905-0.

C5: Security in Invasive Computing Systems

Felix C. Freiling, Wolfgang Schröder-Preikschat, Ingrid Verbauwhede

Ruan de Clercq, Johannes Götzfried, Pieter Maene, Tilo Müller

Project C5 explores security aspects of invasive computing and resource-aware programming. Invasive MPSoC architectures will only be accepted if basic security properties are supported. The final goal is to ensure confidentiality, integrity, and availability in the presence of untrustworthy programs that compete for resources and/or can contain malicious functionality. This requires a comprehensive approach, addressing both hardware and software mechanisms. Project C5 is a new project established within the second funding phase.

Attacker Model

Our security investigation starts with an attacker model in which we focus on software-only attacks. This means that physical attacks are not considered in the current funding phase. In other words, we assume that the attacker cannot physically probe the chip or FPGA, cannot open the device, has no access to the bitstream, and cannot perform active attacks such as fault injection or provoke clock and power glitches.

Our attacker model consists of three hierarchic levels at which an attacker can operate and execute software. Each level includes all levels below. The lowest level is a single core, followed by the tile-level and finally the NoC level. We will first concentrate on the lowest level, i.e., at the core.

A typical processor consists of four generic components: computation, communication, storage and control units. We will use this classification to systematically identify all origins from which an attack can occur, as well as the attack targets. Next, we will evaluate which combinations of attack origins and targets form realistic threats. We consider both active and passive attacks. In case of the former the attacker is considered to be capable of modifying and observing data. However, in case of the latter, the attacker is only capable of observing data.

Some examples of attack targets at each level are given next. First, at the core level, the caches might leak sensitive data and code. Second, because of the run-to-completion semantics, a malicious *i*-let might start an infinite loop. This would make the core unavailable to do any other work. The most obvious attack surface at the tile-level is the TLM. It will be shared by all *i*-lets running on the different cores of the system. Without a protection mechanism, a malicious *i*-let could therefore easily read another's data or code. At the NoC-level, an obvious target is the network communication. One attack target is the traffic passing by the tile's network adapter, which could be snooped. Secondly, a node could read from or write to the DDR or read data from another tile's TLM it does not have access to. Although the NoC has a very high bandwidth, a DoS attack against it might still be possible.

Isolation Concepts

The scientific objective of Project C5 is to embed different degrees of isolation into the invasive computing paradigm through all architectural layers. Basic isolation concepts, like horizontal isolation of the system layer against applications as well as vertical isolation of applications against each other, are developed in close cooperation with Project C1. However, to guarantee the correct execution of intended behaviour of an application not only in the face of competing or malicious programs, but also in view of an untrusted system layer, stronger degrees of isolation are required. Predictability of invasive programs in terms of correct execution must involve a trusted computing base that guarantees the integrity of a program's control flow. Security and predictability can both only be provided if there exists a trust anchor for user applications. This can be assumed to exist in software, but for practical reasons is probably easier to achieve through immutable hardware.

Starting from the state of *no isolation* at the end of the first funding phase, we aim to establish a notion of *basic isolation* which ensures basic read/write protection for applications. An application x enjoys basic isolation if other applications cannot read or write the state of x (using a broad notion of state encompassing data and code). This corresponds to classical OS isolation properties. Basic isolation still allows information to leak from an application through side channels. The concept of ϵ-*isolation* requests the absence of information flow up to a negligible amount ϵ.

Basic isolation does not only encompass the integrity of a program's control flow but also the protection of its intellectual property. A hardware-assisted trusted computing base that shields an application

against malicious manipulation, including manipulation attempts *from the system layer*, can be extended to the ability to protect an application against reverse engineering. We call this variant of basic isolation *blackbox isolation* as it allows the execution of an application inside a virtual blackbox that cannot be manipulated or analysed from its outside environment. This type of isolation is also of interest from an economic point of view, including cloud computing and digital rights management, as shown by emerging technologies such as "Intel SGX" that pursue similar goals.

In cooperation with the Mercator fellow Ingrid Verbauwhede from KU Leuven, we are currently developing a concept for blackbox isolation, based on her previous publication called "Sancus" [Noo+13] that targets embedded single-core systems. Thereafter we are planning to extent our proposal to the invasive computing architecture as blackbox isolation is particularly interesting for resource-aware programming, not only regarding predictability but also regarding future use cases of the invasive paradigm, e.g., for data centre architectures.

Publications

We have recently investigated variants of the isolation concept in different contexts. For example, we studied the problem of memory isolation with respect to data exposure caused by insecure deallocation in common memory management schemes [AFGM14]. We proposed declarative approaches to handle unreasonably long data lifetime at the programming language level, and present several directions on how current platforms can be improved to minimise the lifetime of confidential data. These declarative approaches can be used at the X10 programming language level to prevent information leaks in memory.

We investigated hardware support for memory isolation using secure interrupt handling on a protected module architecture [CPSV14]. Our scheme builds on the idea of using simple memory isolation techniques to ensure leakage free processing of secret information on a microcontroller. Three methods of securely handling interrupts are proposed, each exploring a different trade-off between hardware and software complexity, and interrupt latency.

Code isolation against malicious reverse engineers using a purely software-based approach can be achieved using complex obfuscation techniques. We investigated several such techniques [FPZ14] and constructed complex software obfuscation schemes from sequentially applying simple obfuscation methods. We also discovered several desirable and undesirable properties of such transformations, such as idempo-

tency and monotonicity. A purely software-based approach could be an alternative to using specialised hardware in order to isolate code in an invasive computing system.

We investigated compiler-level protection techniques that isolate return addresses and saved frame pointers on a separate stack that is different from the common user stack [KM14]. By isolating control information from data, we protect sensitive pointers of a program's control flow from being overwritten by buffer overflows. As we make control flow information simply unreachable for overflows, many exploits are stopped at an early stage of progression with only little performance overhead.

Finally, we investigated a tamperproof authentication scheme that mutually authenticates computers and users in order to resist software keylogging attacks during boot [GM14]. As a proof of concept, we implemented trust bootstrapping from a secure token to a PC with trusted platform module. The secure token is a USB drive that verifies the integrity of the PC and indicates the verification status by an LED. This way, users can ensure the integrity of the whole software stack before entering their passwords.

Outlook

For 2015, we are planning to implement the concept of blackbox isolation on single-core embedded systems and then to extend our proposed concept to invasive computing architectures. One of the research challenges is to keep changes to existing hardware minimal, i.e., to establish a provably minimal trusted computing base. The hardware aspects will be a main contribution of the Mercator fellow Ingrid Verbauwhede.

Thereafter, we aim to make different degrees of isolation available to the application level by proposing new constraints for the X10 programming language with which invasive programs formulate their claims. This will be done in cooperation with Project C3. The different degrees of isolation will also have to be integrated into the agent system, what will be done in cooperation with Project C1.

Last but not least, we want evaluate the resistance of blackbox isolation against different types of side channel attacks and quantify the amount ϵ of remaining information flow of a blackbox to its outside environment. This is specifically important for invasive computing since resource-aware programming offers a multitude of possibilities to convey information over unintended communication channels. Indeed, the remaining amount of information flow must be reduced to a minimal ϵ

during further advancements of our isolation concepts, e.g., by requiring processors at the border of an application to remain idle.

Publications

[AFGM14] M. Anikeev, F. Freiling, J. Götzfried, and T. Müller. “Secure garbage collection: Preventing malicious data harvesting from deallocated Java objects inside the Dalvik VM”. In: *Journal of Information Security and Applications*. Amsterdam, 2014. DOI: 10.1016/j.jisa.2014.10.001.

[CPSV14] R. de Clercq, F. Piessens, D. Schellekens, and I. Verbauwhede. “Secure Interrupts on Low-End Microcontrollers”. In: *IEEE 25th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. 2014, pp. 147–152. DOI: 10.1109/ASAP.2014.6868649.

[FPZ14] F. Freiling, M. Protsenko, and Y. Zhuang. “An Empirical Evaluation of Software Obfuscation Techniques applied to Android APKs”. In: *International Workshop on Data Protection in Mobile and Pervasive Computing*. Ed. by J. Lin and T. Müller. Beijing, China, 2014.

[GM14] J. Götzfried and T. Müller. “Mutual Authentication and Trust Bootstrapping towards Secure Disk Encryption”. In: *Transactions on Information and System Security (TISSEC)*. Vol. 17. New York, 2014. DOI: 10.1145/2663348. URL: https://www1.cs.fau.de/mark.

[KM14] C. Kugler and T. Müller. “SCADS: Separated Control- and Data-Stacks (Best Student Paper Award)”. In: *10th International Conference on Security and Privacy in Communication Networks*. Ed. by S. I. ICST (The Institute for Computer Sciences and T. Engineering). Beijing, China, 2014. URL: https://www1.cs.fau.de/filepool/scads/scads-securecomm2014.pdf.

[Noo+13] J. Noorman, P. Agten, W. Daniels, R. Strackx, A. Van Herrewege, C. Huygens, B. Preneel, I. Verbauwhede, and F. Piessens. “Sancus: Low-cost trustworthy extensible networked devices with a zero-software trusted computing base”. In: *Proceedings of the 22nd USENIX Security Symposium*. (Washington, D.C.). USENIX Association, 2013, pp. 479–494.

D1: Invasive Software–Hardware Architectures for Robotics

Tamim Asfour, Walter Stechele

Manfred Kröhnert, Johny Paul

The main research topic of Project D1 is the exploration of benefits and limitations of invasive computing for humanoid robotics. Our goal is to use complex robotic scenarios with concurrent processes and timely varying resource demands to show high-level invasion and negotiation for resources. Invasive computing mechanisms should allow for efficient resource usage and shorter execution times while adhering to predictability of non-functional properties, e.g. power, dependability, security. Therefore, research on techniques of self-organisation are the key to efficient allocation of available resources in situations where multiple applications bargain for the same resources.

Status

In Phase I, we defined a robotics scenario comprising several algorithms which are required for the implementation of visually guided grasping on a humanoid robot. A number of selected applications have been analysed to provide design specification requirements for projects of the A, B, and C areas. Fixed resources but changing load situations were used to do an initial evaluation of these algorithms on an invasive hardware platform. Based upon knowledge gained from these experiments, we established performance metrics which were used later on to adapt parameters of running applications to provide optimal throughput while available computing resources change. The mentioned adaptable parameters range from simple threshold values to more complex algorithm partitioning schemes. Exploiting information provided by a given invasive computing platform and combining it with the previously defined metrics allowed us to demonstrate improved application performance and quality as compared to state-of-the-art techniques.

Object Recognition on MPSoC

The robot ARMAR-III currently uses an object recognition application consisting of three main stages. These include Harris Corner Detection, SIFT (Scale Invariant Feature Transform) feature extraction and SIFT feature matching. During the 1st funding phase of the project, we developed resource-aware models for Harris Corner Detector and SIFT feature matching. These applications can express their resource requirements to the run-time system, based on the image resolution, frame rate, etc. When sufficient resources are not available, the application can adapt to available resources by adapting its own algorithm. Such adaptations helped to improve the quality of the overall results when compared to the non-adaptive versions.

The resource-aware Harris concern detector [Pau+14a] can prune away non-corner like pixels from the scene when inefficient computing resources are not available. The pruning mentioned above is based on a variable threshold value. Hence the number of pixels pruned away can be varied at run time based on the availability of resources. The adaptive pruning technique helped to avoid frame drops and to ensure smooth processing of video frames. The table below shows a comparison between the conventional and resource-aware models. The results were captured by evaluating six different video sequences. Precision and recall represent the accuracy of the detected corners and WCET represents the worst-case execution time. The use of the

	Throughput	WCET	Precision	Recall
Conventional	**81%**	4.31x	0.82	0.81
Resource-aware	100%	1.04x	**0.98**	**0.98**

Table 4.2: Comparison between conventional and resource-aware Harris detectors [Pau+14a]

conventional algorithm leads to a very high worst-case execution time (WCET) and frame drops. The precision and recall values are low for the conventional algorithm as a frame drop leads to zero precision and recall for that particular frame. In brief, the resource-aware Harris detector can operate very well under dynamically changing conditions by adapting the workload, avoiding frame drops and regulating the WCET, leading to high precision and recall rates.

Integrated Robotics Demo

During the DFG review meeting in February 2014 an integrated robotic demo was shown combining algorithms presented in the previous years. All required hardware components and involved high-level tasks are depicted in Figure 4.26.

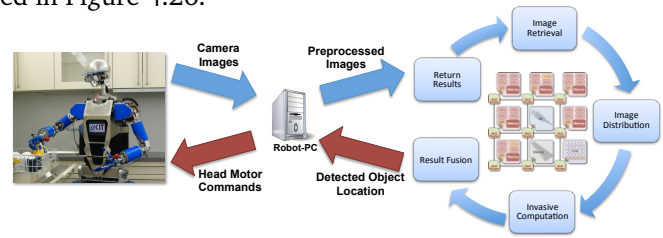


Figure 4.26: Camera images are transferred to an invasive computing demonstration platform via the robot control PC. A new location to look at is sent to the robot after successful object detection.

First, live camera images from the robot head are transferred to the invasive applications where they are processed afterwards. Disparity Map computation is performed at startup together with low resolution HSV Colour Segmentation for both skin and object colour (Figure 4.27 left). A switch to high resolution colour segmentation happens once an object has been detected to provide accurate positions for object tracking (Figure 4.27 right). Disparity Map calculation is stopped while tracking an object since colour segmentation is more crucial in this case and sufficient resources are not available for all algorithms to be running at the same time. After the tracked object vanishes, Disparity Map and low resolution HSV segmentation are resumed.

In order to provide the robot control software sufficiently fast with tracking positions, the invasive algorithms are currently running on the x86 Guestlayer variant of OctoPOS.

D1

Outlook for 2015

Many robotic applications consist of multiple individual application programs in a fixed chain of sequence, e. g., visual object recognition is composed of corner detection, feature extraction, and template matching. During Phase I, we have investigated invasive computing for these individual application programs, based on quality metrics and computing resources for each step. We could show how to respect soft real-time requirements, e. g., frame rate, by adapting application parameters such as threshold for image preprocessing before Harris Corner Detection.

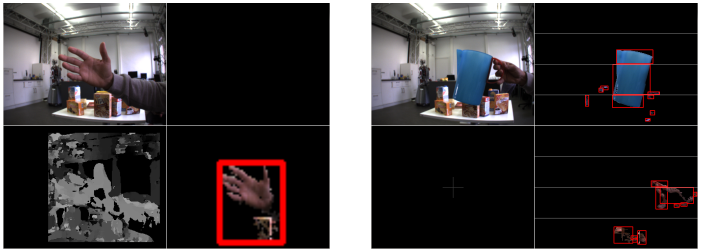


Figure 4.27: Left: Disparity Map and low resolution HSV segmentation calculation. Right: Computation of high resolution HSV segmentation only.

However, from a robotic application perspective, the quality of the object recognition after the complete chain matters more than the quality of the individual steps. Computing resources might be distributed by resource-aware programming mechanisms throughout the individual steps. Starting in Phase II, we plan to extend both quality metrics and soft real-time requirements towards end-to-end coverage over a chain of application programs.

Additionally, we will employ prediction mechanisms to forecast future resource usages of higher level robot applications. Prediction models required for this task need to be selected according to several criteria. One of them being the ability to generate them by learning from experience through self-monitoring concepts as well as to adapt the models at run time. Predicted resource usages will be used in a more complex human-robot interaction scenario with dynamic action selection based on unexpected events to inform the invasive run-time system about requirements of upcoming tasks. Overall, resource allocation and distribution should lead to optimised resource allocation and improved run-time performance.

D1

In 2014, we started working on prediction mechanisms based on the Robot Development Environment ArmarX [Wel+13]. The initial system comprises self-monitoring capabilities of robot programs described as finite state automata, as well as basic prediction and resource models [Krö+14]. Future resource usages were predicted in a Pick and Place scenario containing many fallback states for error recovery.

In 2015, we plan to start with a chain of Harris corner detection, SIFT feature extraction and SIFT feature matching, applying precision-recall graphs, combined with throughput and latency measures, as a quality metric over the complete sequential chain of application programs for visual object recognition. The individual steps (Harris corner detection, SIFT feature extraction and SIFT feature matching) are available from Phase I. Now, we plan to build a control loop over the complete chain in order to model the influence of individual parameter settings on the quality and performance of the object recognition. Parameter settings include general parameters such as image resolution and frame rate, as well as specific parameters such as filter thresholds and KD-tree search depth. Additionally, we continue working on resource usage prediction by refining the presented prediction models and enhancing the current resource profiles with data obtained from profiling executions of real-world robot programs.

We plan to cooperate with Project A1 on applying the actor-oriented programming model and with Project A4 on Pareto analysis over the complete processing chain, in order to identify optimised operating points for the parameter setting within visual object recognition.

Publications

[Krö+14] M. Kröhnert, N. Vahrenkamp, J. Paul, W. Stechele, and T. Asfour. "Resource Prediction for Humanoid Robots". In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 22–28. arXiv:1405.2911 [cs.RO].

[Pau+14a] J. Paul, W. Stechele, M. Kröhnert, T. Asfour, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, and W. Schröder-Preikschat. "Resource-Aware Harris Corner Detection based on Adaptive Pruning". In: *Proceedings of the Conference on Architecture of Computing Systems (ARCS)*. (Luebeck, Germany). LNCS 8350. Springer, Feb. 25–28, 2014, pp. 1–12. ISBN: 978-3-319-04890-1. DOI: 10.1007/978-3-319-04891-8.

[Wel+13] K. Welke, N. Vahrenkamp, M. Wächter, M. Kröhnert, and T. Asfour. "The ArmarX Framework - Supporting high level robot programming through state disclosure". In: *Informatik 2013 Workshop on robot control architectures*. 2013.

D3: Invasion for High-Performance Computing

Hans-Joachim Bungartz, Michael Gerndt, Michael Bader

Isaías Comprés, Andreas Hollmann, Emily Mo-Hellenbrand, Josef Weidendorfer

The overall goal of Project D3 lies in these two areas:

The first one is to provide application level support for the development of invasive computing hardware platforms. In Phase II, we carry out further development on fundamental numerical algorithms, as well as more complex scientific applications to assist research in different areas of invasive hardware and software. These applications are developed in the X10 programming language and modified to support advanced invasion. In these steps, we exchange information and feedback with other projects.

The second area is to investigate and exploit the potentials of invasive computing for state-of-the-art high-performance computing applications [Bad+11; Ger+12a] on standard HPC architectures. Our goal in Phase II is three-fold: firstly, we develop an MPI extension to support the invasive computing concepts on general distributed-memory systems; secondly, we develop high-performance applications for invasion on distributed-memory systems; last but not least, we develop a resource manager, which distributes resources fulfilling the system requirements (such as optimal system throughput, optimal energy efficiency, etc.) as well as the requirements of different types/classes of applications.

1. Demonstrator Platform: X10 Applications

Status

In the past, we developed several invasive numerical core algorithms, such as numerical integration, which computes the integral over a 1-D or 2-D domain; matrix-matrix multiplication, which computes the product of two matrices using space filling curve (SFC)-based or block-wise storage for cache efficiency optimisation [Tra+14b]; LU-decomposition,

which factors a matrix as the product of a lower triangular matrix and an upper triangular matrix with cache optimisation via SFC-based storage; and a multigrid solver, which solves the heat equation using a recursive scheme to correct the solution in each iteration to accelerate convergence [Bun+13].

Since the beginning of Phase II, we have been working on improving and optimising some of the above mentioned algorithms, as well as developing new advanced methods:

- Adding an asynchronous block pre-fetching scheme to the matrix-matrix multiplication algorithm to improve performance efficiency. Multiplicands are partitioned into blocks and stored in the local memory spaces of different processes. This scheme enables processes to pre-fetch data for the next block computation while they are computing a certain block.

- Implementing work stealing functionality for the LU-decomposition algorithm to improve load balancing. The matrix to be decomposed is partitioned into blocks and each process is to work on certain blocks that are stored in its local memory spaces. With this functionality, a process may take over some workload from other processes in case it finishes its own workload faster.

- Implementing sparse grid combination technique functions. The sparse grid combination technique provides multi-level parallelism as well as algorithm-based fault tolerance for solving partial differential equations. These functions will be used for the development of fault-tolerant applications (see below).

Outlook

For the remainder of Phase II, our goals for development, in addition to completing the above mentioned ongoing tasks, include: **(1)** Develop an enhanced multigrid solver for computational fluid dynamics (CFD) simulation, with the long term vision of a Navier-Stokes solver, which can be used for demonstrator developed by Project Z2. **(2)** Contribute to the topic of ”Dark Silicon” (Project B3) and make features accessible to scientific algorithms by providing core-level fault-tolerant applications, such as an iterative solver for a linear system with full local storage of the right-hand-side data, or a simple solver using the sparse grid combination technique[1]. **(3)** Assist the development of TCPA tiles (Project B2) by providing integer only bit-level fault-tolerant applications,

which can be realised with an iterative solver implemented with fixed-point arithmetics[1].

2. Invasion for HPC

Status

In the past, we developed an invasive extension for OpenMP — *i*OMP [Ger+12b] to support the invasive computing concepts for shared-memory based HPC systems. And we implemented an invasive version of a 2-D tsunami simulation based on SFC-based dynamically adaptive grids [SWB13a; SWB13b]. We conducted experiments on a core-distribution scheduler that realises the migration of computational power by distributing cores depending on the requirements specified by one or more parallel program instances. We've successfully demonstrated on a shared-memory system a significant improvement of the overall system throughput using invasive computing [SRNB13; Sch+14; Sch14].

Since the beginning of Phase II, we have been working on the following tasks:

- The resource manager is extended and alternative schedulers are implemented based on machine learning and expert systems in order to run applications without prior benchmarking. These schedulers act on information that is gathered at run time of one or more parallel applications and try to redistribute resources to improve the overall throughput. General performance statistics as well as hardware performance counters, gathered by the Linux perf subsystem, are evaluated for each individual process and are used for the decisions made by the scheduler.

- In order to support resource-aware programming on distributed-memory systems, the implementation of an extension to MPI with invasive operations (*i*MPI) is carried out. The only resources currently abstracted by MPI are the computing processes, which are represented by ranks in communicators and groups. A modification to the way communicators and groups are used by applications is necessary. In addition, efficiency will be a key target since resource adaptations can be relatively expensive operations in distributed systems. Efficient internal representation of key data structures used by communicators and groups need to be devised.

- Development of the invasive 2-D tsunami simulation based on distributed-memory systems using *i*MPI is currently in progress. Similar to the shared-memory based version, the application utilises SFC-based dynamically adaptive grids, which generates different workload at each time step and is optimised for data decomposition. With the support of *i*MPI, it communicates with the resource manager and redistributes data and workload to available computational resources at run time.

Outlook

Our goal in Phase II is to extend the invasive computing concepts to standard distributed-memory based HPC systems. In addition to completing the above mentioned ongoing tasks, for the remainder of Phase II, we also plan for the following: **(1)** Implementation of an invasive 3-D weather/climate simulation, which resembles typical large-scale HPC applications with requirements of dynamic changes of computational resources at run time. **(2)** Development of invasive resource manager for distributed-memory HPC systems. We will continue the resource management research started with the *i*OMP efforts. New MPI specific features for the resource manager, commonly performed by a process manager in MPI, will be developed. **(3)** Development of other classes of applications, which is desirable for increasing the diversity of invasive applications[1].

Publications

[Bad+11] M. Bader, H.-J. Bungartz, M. Gerndt, A. Hollmann, and J. Weidendorfer. "Invasive Programming as a Concept for HPC". In: *Proceedings of the 10th IASTED International Conference on Parallel and Distributed Computing and Networks 2011 (PDCN)*. Feb. 2011.

[Bun+13] H.-J. Bungartz, C. Riesinger, M. Schreiber, G. Snelting, and A. Zwinkau. "Invasive Computing in HPC with X10". In: *X10 Workshop (X10'13)*. X10 '13. New York, NY, USA: ACM, 2013, pp. 12–19. DOI: 10.1145/2481268.2481274.

[1] Will be implemented if time/resources permit.

[Ger+12a] M. Gerndt, F. Hannig, A. Herkersdorf, A. Hollmann, M. Meyer, S. Roloff, J. Weidendorfer, T. Wild, and A. Zaib. “An Integrated Simulation Framework for Invasive Computing”. In: *Proceedings of the Forum on Specification and Design Languages (FDL)*. Vienna, Austria: IEEE, Sept. 18–20, 2012, pp. 209–216. ISBN: 978-1-4673-1240-0.

[Ger+12b] M. Gerndt, A. Hollmann, M. Meyer, M. Schreiber, and J. Weidendorfer. “Invasive computing with iOMP”. In: *Proceedings of the Forum on Specification and Design Languages (FDL)*. Vienna, Austria, Sept. 18–20, 2012, pp. 225–231. ISBN: 978-2-9530504-5-5.

[Sch14] M. Schreiber. “Cluster-Based Parallelization of Simulations on Dynamically Adaptive Grids and Dynamic Resource Management”. Dissertation. Institut für Informatik, Technische Universität München, Jan. 2014. URL: http://www5.in.tum.de/pub/schreiber14dissertation_preprint.pdf.

[SRNB13] M. Schreiber, C. Riesinger, T. Neckel, and H.-J. Bungartz. “Invasive Compute Balancing for Applications with Hybrid Parallelization”. In: *Proceedings of the International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. IEEE, Oct. 23–26, 2013.

[SWB13a] M. Schreiber, T. Weinzierl, and H.-J. Bungartz. “Cluster Optimization and Parallelization of Simulations with Dynamically Adaptive Grids”. In: *Euro-Par 2013*. Aachen, Germany, Aug. 2013.

[SWB13b] M. Schreiber, T. Weinzierl, and H.-J. Bungartz. “SFC-based Communication Metadata Encoding for Adaptive Mesh Refinement”. In: *Proceedings of the International Conference on Parallel Computing (ParCo)*. Ed. by M. Bader. Oct. 2013. In press.

[Sch+14] M. Schreiber, C. Riesinger, T. Neckel, H.-J. Bungartz, and A. Breuer. “Invasive Compute Balancing for Applications with Shared and Hybrid Parallelization”. In: *International Journal of Parallel Programming* (Sept. 2014). DOI: 10.1007/s10766-014-0336-3. URL: http://scholar.google.de/scholar_url?hl=en&q=http://link.springer.com/article/10.1007/s10766-014-0336-3&sa=X&scisig=AAGBfm2kuCkaeu8QOohrdd5TiugbKlCWTQ&oi=scholaralrt.

[Tra+14b] C. Tradowsky, M. Schreiber, M. Vesper, I. Domladovec, M. Braun, H.-J. Bungartz, and J. Becker. “Towards Dynamic Cache and Bandwidth Invasion”. In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Vol. 8405. Lecture Notes in Computer Science. Springer International Publishing, Apr. 2014, pp. 97–107. DOI: 10.1007/978-3-319-05960-0_9.

Z: Central Services

Jürgen Teich

Jürgen Kleinöder, Katja Lohmann, Sandra Mattauch, Ina Derr, Frank Hannig

The central activities and services in InvasIC are coordinated and organised by Project Z. These activities and services are subdivided into two parts:

The first part is administrative support, organisation of meetings (internal project meetings, doctoral researcher retreats) and assistance for visits of guest researchers and for researchers travelling abroad. Technical support and tools for communication and collaboration are provided as well as support and organisation of central publications. Last but not least, financial administration and bookkeeping is one of the central services.

The second part concerns public relations. Contacts with important research sites are established as well as an international Industrial and Scientific Board. Scientific ideas and results are discussed at various workshops and conferences.

For detailed information on the general idea and organisation of InvasIC as well as on the progress made in the different projects, the InvasIC website `http://www.invasic.de` is maintained.

A detailed listing of the scientific meetings and events organised and conducted by Project Z is provided in Part III of this report.

Z2: Validation and Demonstrator

Jürgen Becker, Frank Hannig, Thomas Wild

Markus Blocherer, Srinivas Boppu, Stephanie Friederich, David May, Timo Stripf

In the first funding phase, the major goal of Project Z2 was to provide a common FPGA-based demonstrator environment for validating and demonstrating the principles of invasive computing. As hardware platform, a Synopsys CHIPit Platinum system is used at each site (FAU, KIT, TUM) since the beginning of the collaborative research centre. Thanks to its 6 FPGAs (Virtex5 LX330), the system allows for the prototyping of multi-million gate designs. To show the advantages of invasive computing such as improved quality of service, resource utilisation and speed-up of applications, contributions of the different projects across all project areas were integrated into the common demonstrator platform. In the review meeting held in February this year, the advantages of invasive computing were demonstrated on two demonstrators, which are described in the following.

In cooperation with all projects and the working groups of the CRC, tiled invasive multicore architectures, such as the one shown in Figure 4.28, were developed and prototyped for the demonstration of invasive applications from project area D covering robotics and scientific computing, which make up the common demonstration scenarios. The left part of Figure 4.28 shows an example configuration of an invasive MPSoC: A tiled architecture consisting of RISC-processor–based compute tiles (using LEON3 processor cores or the *i*-Core as developed from Project B1), compute tiles based on TCPAs (Tightly-Coupled Processor Arrays) as developed in Project B2 as well as I/O and memory tiles. The tiles which contain contributions from all projects of project area B are connected via the *i*NoC. The invasive operating system OctoPOS and the run-time support layer *i*RTSS (Project C1) are running on top of this hardware platform. Within Project Z2, different variants of such a heterogeneous hardware architecture, the associated system software and the invasive applications on top were integrated on the CHIPit prototyping system.

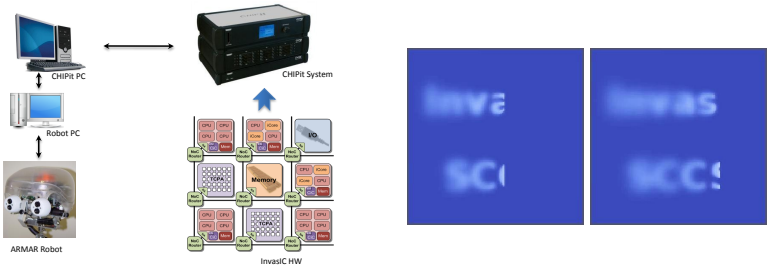


Figure 4.28: The left image shows the robot demo scenario configuration and the right images visualise a simulation of a laser engraving letters on a metal plate using an invasive multigrid algorithm.

Project Z2 contributed in the following three major ways in the realisation of the overall demonstrator:

- Provisioning of all the necessary peripherals like UART, SSRAM, Ethernet, AHB transactors and DDR2 memory for integration into the common demonstrators and establishment of the appropriate tool support for the CHIPit system.
- Support for the integration of the different projects' contributions, i.e., hardware and software components, into the demonstrator platforms.
- Coordination work for the definition of demonstrator scenarios on the prototyping system, including milestone planning.

In the following sections, more detailed information is given on the two demonstrators that were presented during the review meeting in February 2014.

Demonstrator I

During the second half of the first phase of the collaborative research centre, Project Z2 focused on integrating the hardware design developed in project area B into the common demonstrator. The Demonstrator I architecture comprises a 2x2-tile mesh architecture, including one I/O tile. The design has been partitioned over four FPGAs on the CHIPit prototyping platform and each tile is connected via two transactor interfaces with a host PC, see Figure 4.28. The first interface is used for GRMON connection to load and execute the software on the tiles. The second transactor is used to transfer monitoring data for visualisation

of core and network utilisation [FHB14]. The current per-core claim mapping and status of the agent system were transferred over the Ethernet interface [FHMB14] for further visualisation. To show the advantages of invasive computing architectures, a multigrid algorithm with varying resource requirements was chosen. Project D3 provided a laser simulation using such an algorithm. The application output, i.e., the heat distribution on a plate, is shown in Figure 4.28 on the right. Demonstrator I served to showcase a complete tool chain starting from algorithm description in X10, simulation as well as compilation using the InvadeX10 compiler, run-time system *i*RTSS, and finally the execution of the invasive application software on a prototype of an invasive multi-tile hardware architecture.

Demonstrator II

To demonstrate the self-adaptiveness and resource-aware programming concepts of invasive computing, a Harris corner detection application from a robot object tracking application was chosen in Demonstrator II, also composed of a 2×2-tile mesh architecture with two LEON3 tiles, one TCPA tile, and one I/O tile. Project D1 provided this algorithm to explore the benefits of invasive computing for visually guided grasping in humanoid robotics. Using Demonstrator II, it was shown that the Harris corner detection application is able to adapt to the number of LEON cores available on the tiled architecture. Furthermore, it was also shown that the Harris corner detection application is able to switch between the TCPA tile and LEON tiles—showcasing self-adaptiveness and resource awareness of invasive computing. This demonstrator was also successfully presented to public at the DATE 2014 university booth [Sou+14].

Next Steps

During the first funding phase, it turned out that the existing prototyping platform has several limitations in terms of capacity and performance. This limited the ability to demonstrate scalability of our decentralised concepts and architectures. Therefore, for the second funding phase, we applied for a new FPGA-based prototyping platform. The new platform will enable the demonstration of invasive compute principles in a larger (up to 100 processors) and as well faster environment. In order to evaluate which products are suitable for the platform successor, we created a comprehensive list of criteria to be met and compared them with the features of products available on the market. Two products

Figure 4.29: Demonstrator II at the DFG review meeting in Garching, February 2014

were short-listed to be candidates for the new demonstration platform and evaluated in more detail for giving an objective recommendation, on which the acquisition decision could be based. Furthermore, in order to streamline the design and integration activities, a new Git repository was discussed and is on its way to be introduced within the CRC.

In the second funding phase, Project Z2 will continue in providing support for demonstrating the benefits of invasive computing across all involved layers, from algorithmic concepts, software and invasive hardware. Project Z2 will establish the generic infrastructure as required for verification and evaluation of invasive multi-tile architectures by integrating the various HW/SW components from different projects. Special focus will be on supporting the assessment of the major goals of the overall CRC in the second funding phase such as *error resilience* (e. g., by methods for fault injection and emulation of error models), and other qualities of program execution for which *predictability* is of utmost concern such as security and performance. To support investigating the power efficiency of invasive architectures, power models of the individual components will be emulated and aggregated on the demonstrator platform as well.

Publications

[FHB14] S. Friederich, J. Heisswolf, and J. Becker. “Hardware/software debugging of large scale many-core architectures”. In: *Proceedings of the 27th Symposium on Integrated Circuits and Systems Design (SBCCI)*. IEEE, Sept. 2014, pp. 1–6.

[FHMB14] S. Friederich, J. Heisswolf, D. May, and J. Becker. “Hardware prototyping and software debugging of multi-core architectures”. In: *Proceedings of the Synopsys Users Group Conference (SNUG)*. Munich, Germany, 2014.

[Sou+14] É. Sousa, V. Lari, J. Paul, F. Hannig, J. Teich, and W. Stechele. “Resource-Aware Computer Vision Application on Heterogeneous Multi-Tile Architecture”. Hardware and Software Demo at the University Booth at Design, Automation and Test in Europe (DATE), Dresden, Germany. Mar. 24–28, 2014.

5 Working Groups

WG1: Predictability

Coordinators: Michael Gerndt, Michael Glaß

Focal points of interdisciplinary investigation in this working group are all questions around the lead topic of *predictability* of non-functional aspects of computation. The topics of this working group stem from the fact that predictability is an all-encompassing concept that requires consideration across architecture, system software and services, up to the level of applications. Here, WG1 serves as a discussion group for these topics, organises information exchange, and triggers and coordinates collaborations between projects and project areas with respect to predictability. Concrete goals of WG1 are: (1) Identification of the most *relevant topics* concerning predictability within this CRC and the assignment of topic chairs that coordinate individual topics. General topics could be aspects of predictability and composability in the embedded domain but also, e. g., in HPC and their impact on this CRC's project areas and projects. Concrete subtopics include the modelling and language implementation of *requirements*, application and resource (performance) modelling, resource management infrastructures and techniques enforcing predictability through all layers of invasive architectures, optimisation and constraint solving techniques, as well as the question of off-line vs. on-line optimisation and auto-tuning. (2) A *glossary* of all relevant predictability terms to enhance the common understanding of predictability concepts in this CRC. Here, we want to particularly address the notion of predictability in the different communities and application domains present in this CRC. (3) A *predictability landscape* that, based on the identified relevant topics, outlines predictability challenges that (a) can be solved by employing existing analysis techniques, (b) are tackled within this CRC, and (c) may be subject to future research directions. (4) Organisation of *workshops* to

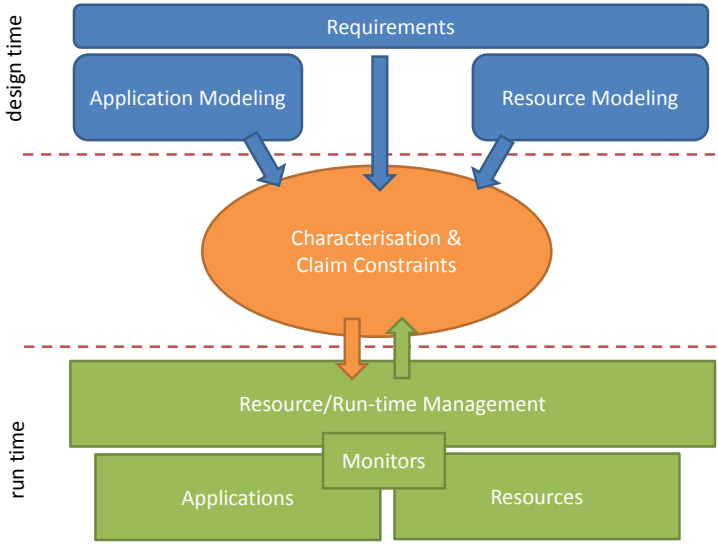


Figure 5.1: Identified topics and the information flow where predictability as a cross-cutting concept has to be enforced within this CRC.

stimulate predictability-related discussions and collaborations outside this CRC with a special focus on bringing together different communities like robotics, OS scheduling, HPC and embedded systems.

As a first step, the WG prepared a questionary for each project of this CRC to get an in-depth view on each project's core predictability work packages, the different views on predictability challenges of each project, as well as further dependencies and cooperation opportunities. In a physical meeting in December 2014, the participants then discussed the major components, concepts, and techniques involved in supporting predictability in and via invasive computing.

Based on the questionary and the intense discussions during the physical meeting, main *topics* have been identified that are—including the flow of information between them—graphically depicted in Figure 5.1: The topics span from *requirements* on non-functional aspects, over *application* and *resource models* (e.g. required for design-time analysis), to concrete measures and techniques to enable predictable *applications* and *resources* at run time. The connecting link between design-time models and requirements on the one hand and run-time aspects on the other is the topic *application characterisation & claim constraints* where applications are analysed and optimised to deliver resource requirements

as claim constraints together with expected (ensured) quality numbers to the *resource/run-time management system*. This way, for example, scalability information is passed from design to run time. The dynamics that arise at run time are then covered by the interplay of *monitors* and the run-time management system. Additional to the identified topics, topic chairs from within the CRC coordinate the projects working in each topic to address and resolve dependencies between projects and establish a seamless information flow and documentation. A possible design flow based on the X10 language that covers the outlined topics as proposed by projects A4, A1, and C3 has been presented in [GBTW14] as part of the joint workshop *Dynamic Co-Optimization of Applications and Resource Management* between this CRC and the BMBF-funded project *FAST*.

WG2: Memory hierarchy

Coordinators: Lars Bauer, Gregor Snelting

Memory organisation is a problem also belonging to multiple domains of expertise including programming model and language (i.e., the X10 memory model) implementation as well as the underlying hardware architecture. Topics of this working group include questions on caches (invadeable caches, flushing, pinning and invalidation, non-cacheable address ranges), memory protection (isolation), DMA via the *i*NoC and interfaces between hardware and software programming.

The extensive kick-off discussions during the Annual Meeting in October 2014 showed that prototyping and integration-specific memory topics should also be covered by this working group, because potential problems that may arise due to individual implementation decisions often cannot be observed by individual projects, but only in a larger integrated system.

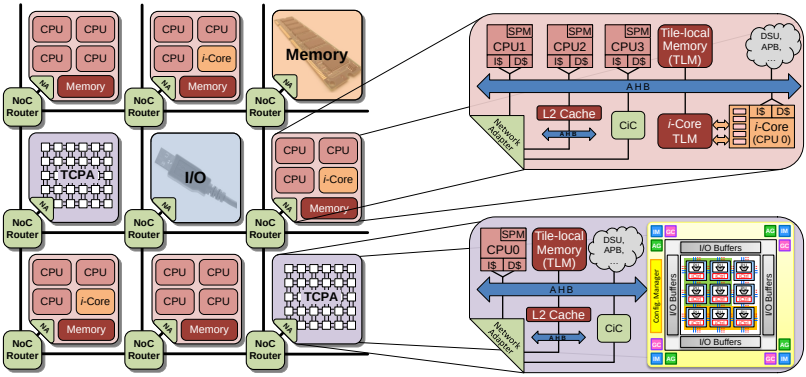


Figure 5.2: Architecture of InvasIC memory hierarchy

WG2

Figure 5.2 was presented at the Annual Meeting in October 2014 and shows the main components of the InvasIC memory hierarchy. Besides

the NoC-based system-level communication and the global memory tile, the focus is on the memory hierarchy within the tiles, i.e. scratchpad memory (SPM), L1 instruction- and data-caches, tile-local memory (TLM), and L2 cache. Based on this memory hierarchy figure, the following memory-related topics were discussed in detail at the annual meeting:

- Several memory-specific prototyping issues along with their solutions or workarounds. Among others, the status of the new L2 cache and DMA copies were discussed, as well as challenges to ensure the global memory order. This showed the importance to discuss the memory hierarchy design in a broader context and to ensure that hardware architects grasp the implications that their implementation decisions may have on the memory model.

- A basic tutorial on general memory model concepts was given. Some more concrete SPARC-V8 and InvasIC-specific examples showed the relevance of the memory model as intermediator between the programmer and the hardware architecture.

- Some limitations/parametrisations of the memory hierarchy are specific to prototyping. Simulation of more realistic memory parameters (e.g., size, speed, usage, etc.) is desirable. The goal is to obtain more realistic measurements and comparisons, i.e., to enable a trade-off between 'demo on prototype' vs. 'realistic measurement that can be scaled'.

- A first discussion about the Project B5 proposal about "dynamic and run-time–adaptive cache coherence" was started. This interesting work package affects the memory coherence and synchronisation between tiles and thus needs to be carefully designed to comply with our memory model.

Altogether, WG2 provides a discussion platform for all topics in the area of memory hierarchy, memory model, and prototyping-specific topics according to memory. WG2 will continue to provide updates at the annual meeting to all members within our collaborative research centre to ensure a homogeneous view.

WG3: Benchmarking and evaluation

Coordinators: Michael Bader, Walter Stechele

WG3 Benchmarking and Evaluation was established in July 2014 with the beginning of Phase II. The major goal of WG3 is to stimulate the exchange between subprojects towards evaluation and benchmarking, including evaluation of invasive cost & benefit, as compared to non-invasive state-of-the-art.

WG3 shall foster technical exchange, identify bottlenecks, bring together complementary expertise between the projects, and establish a repository of standard benchmarks. Specific benchmarks will be derived from the two application projects D1 and D3, with a break-down of test scenarios from D projects to individual subproject level, e.g. SIFT feature calculation from robotic vision might be used for evaluation of *i*-Core performance and power efficiency, visual object recognition might be used for evaluation of *i*NoC bandwidth reservation. Identified benchmark scenarios can also define algorithmic patterns to be studied in Project A4.

Within WG3, researchers discuss criteria for evaluation, including performance, power consumption, quality, dependability, safety, security, hardware overhead, and resource usage efficiency. During the annual meeting in October 2014 in Aalen, two topics have been discussed:

(1) Related work on Application Heartbeats[1], on Application Auto-Tuning[2], and on Tessellation OS[3].

[1] H. Hoffmann, J. Eastep, M. D. Santambrogio, J. E. Miller, and A. Agarwal. "Application heartbeats: a generic interface for specifying program performance and goals in autonomous computing environments." In: *Proceedings of the 7th international conference on Autonomic computing (ICAC '10)*. ACM, New York, NY, USA, 2010, pp. 79-88.

[2] E. Paone, D. Gadioli, G. Palermo, V. Zaccaria, and C. Silvano. "Evaluating orthogonality between application auto-tuning and run-time resource management for adaptive OpenCL applications." In: *Proceedings of IEEE 25th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. June 2014, pp. 161-168.

[3] J. A. Colmenares, G. Eads, S. Hofmeyr, S. Bird, M. Moreto, D. Chou, B. Gluzman,

With respect to predictability, as a major difference between invasive computing concepts and related work publications, the concept of invasive resource reservation has been identified.

(2) Matching criteria between the work plan of individual subprojects from the Phase II proposal and WG3 goals.

E. Roman, D. B. Bartolini, N. Mor, K. Asanovic, J. D. Kubiatowicz. "Tessellation: Refactoring the OS around explicit resource containers with continuous adaptation." In: *Proceedings of Design Automation Conference (DAC)*. June 2013, pp. 1-10.

WG4: Power efficiency and dark silicon

Coordinators: Frank Hannig, Jörg Henkel

The focus of this working group is to align research problems in the direction of the "Dark Silicon" challenge.

The term "Dark Silicon" has been coined with respect to the increasing power density problem: since the classical Dennard scaling will be no longer applicable in upcoming silicon technology nodes, the power density i.e., the amount of electrical power that is dissipated per chip area, increases drastically. In the past, the power density could be kept at tolerable levels since the increased amount of transistors per chip was (power-density-wise) compensated by lowering V_{dd}. This, however, is not possible any longer. "Dark Silicon" denotes the problem of future multicore/manycore systems where a considerate amount of computing and/or communication resources needs to stay "dark" i.e., unused in order not to exceed was is called the "thermal design power": this, in short, is the maximum amount of power that a chip can be operated at without suffering short or long term damage. As a matter of fact, "Dark Silicon" is becoming a severe problem for all future manycore systems where performance, predictability and efficiency matter. Especially in invasive computing, the dark silicon problem will significantly matter because it is targeted towards high efficiency, e. g., how to make best use of on-chip computing and communication resources at the lowest cost possible (e. g., amount of consumed energy, cost of chip packaging for cooling, etc.). In other words: if the dark silicon problem would not be addressed, invasive computing would lose its advantages compared to competing manycore systems. Hence, addressing the dark silicon problem will enable the invasive computing paradigm to come ahead for the upcoming generations of technology nodes.

Towards this end, this working group aims at facilitating information exchange by building power and dark silicon models for an InvasIC-wide power and dark silicon estimation. Furthermore, it targets collecting results for power/dark silicon estimation and agent knowledge as well as

investigate common possibilities for cross-level energy efficiency optimisations. Through integrated dark silicon efforts across different projects, it would not only ensure that the (worst-case) thermal constraints of the on-chip computation and communication resources are not violated, but would also aid a lot in the predictability of an invasive computing system as a shutdown of cores can be avoided, thereby satisfying the constraints of also applications with predictable execution requirements.

Because of its key importance, this working group enables information exchange across different projects to not only build a common understanding and nomenclature for power efficiency and dark silicon problem, but also facilitates integration of research efforts of various projects including B3, B2, B4, C1, etc.

As a starting point, in the annual meeting of InvasIC, the leads of WG4 presented a set of refined goals and targets of the WG aligned to the roadmap of planned InvasIC-wide dark silicon research activities. A summary of key goals is listed below:

1. Aligning research problems to the dark silicon, power/energy efficiency, and temperature challenges.
2. InvasIC-wide dark silicon modelling and estimation that will also require full system power modelling and estimation.
3. Interfacing and facilitate information exchange, for instance,
   a) How to pass the dark silicon information/constraints to the agent-layer?
   b) How to interface between the monitoring (B4) and iDoC (B3)?
   c) Knowledge exchange on power models (e.g., abstraction level, technologies, estimation tools) as independent activities of individual projects.
4. **The overarching goal is**: Infrastructure Development through
   a) Building and integrating power models, dark silicon models.
   b) Full system simulation with dark silicon of multi-tile invasive architectures.
   c) Integration of different dark silicon works, e.g., B2, B3, B4, B5, C1, etc.
   d) Prototyping dark silicon effects through emulation and demonstration and how application projects can incorporate Dark Silicon and energy efficiency effects.

Initial important joint collaborative research activities were identified. A couple of examples under discussion along with the early research efforts are listed below.

1. The early research work of B3 explored the impact of Dark Silicon Patterning on the performance and thermal efficiency. The Dark Silicon Patterning determines the temporal and spatial shutdown of on-chip resources with the goal of minimising peak temperature without violating TDP. Selection of dark cores impacts the chip power density and temperature profile and consequently the power budget utilisation and the amount of total dark silicon. Dark Silicon Patterning alleviates the power density issues, thus leading to a more effective use of TDP budget. During the meeting, the next collaborative steps towards the InvasIC-wide dark silicon patterning were discussed, as summarised below.
    a) **Collaborative Activities between B3 and B2**: Integrating other fabrics like TCPA and *i*-Core and exploring the dark silicon patterning impact. Exploring the temperature and power density variations for TCPA, RISC-like cores, and *i*-Core. Requirements: ASIC synthesis output, layout / floorplan, power estimates or power model, performance traces, same technology, etc. Integrated system simulations. Derive Thermal Safe Power (TSP) for heterogeneous fabrics.
    b) **Collaborative Activities between B3 and C1**: Coordinated Resource and Dark Silicon Management: Interfacing with C1 to forward the dark silicon constraints to the agent-layer? Studying the impact of dark silicon management on the efficiency *i*RTSS. Analysing potential conflicts between DaSiM and *i*RTSS.
    c) **Collaborative Activities between B3 and B5**: Integrated *i*NoC and Tiles Patterning for Dark Silicon. Interesting problems to answer are: Darkening the routers or dark tiles or not? Multi-layer vs. Multiple V-f levels for *i*NoC? *i*NoC and Multiple voltage islands? Single Layer: Re-routing in case of dark routers within the active layer.

Dark Silicon WG Activities: The WG team has also performed several activities for dissemination at a wider and international level through workshop and special session organisations and keynote talks. A summary of these activities is given below.

1. *Workshop on "A Roadmap for EDA Research in the Dark Silicon Era"*

collocated at IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2014: This workshop was intended to provide a common platform for EDA experts to discuss their vision and perspectives on the dark silicon problem, and to define a research roadmap for the next decade. This workshop brought together researchers and experts from industry and academia to dwell on whether fundamentally new solutions are required in the context of dark silicon, or conversely, whether existing solutions can be retro-fitted to address these problems. In either scenario, a lively, but informative technical debate was performed. More details about the workshop can be found at http://ces.itec.kit.edu/EDA4DS/

2. *Special Session "Dark Silicon as a Challenge for Hardware/Software Co-Design" at International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), 2014 at ESWeek*: The goal of this special session was to expose dark silicon challenges for hardware-software co-design along with an overview of some of the early research efforts that are attempting to shape the design and run-time management of future generation heterogeneous dark silicon processors. In a big picture, this special session aimed at making a point that whether "dark silicon introduces fundamentally new challenges for the hardware-software co-design" or "the dark silicon is merely an additional constraint for the specification, design, analysis, and implementation phases of hardware-software co-design of embedded systems".

3. *Keynote Talks: Two keynote talks were given by Prof. Dr. Jörg Henkel.*
   a) J. Henkel, "Dependability of On-Chip Systems in the Dark Silicon Era", The 32nd IEEE International Conference on Computer Design (ICCD) 2014, October 19-22, Seoul, Korea.
   b) J. Henkel, "The Dark Silicon Problem in Multi-Core Systems – Invasive Computing as a Solution", Thematic Session at HiPEAC Computer Systems Week 2014, May 13, Barcelona, Spain.

III

Events and Activities

Summary

The central activities and services in InvasIC are coordinated and conducted by Project Z.

Figure 5.3: From left to right: Ina Derr (Financial Services), Dr.-Ing. Jürgen Kleinöder (Managing Director), Dr. Katja Lohmann (Deputy Managing Director), Prof. Jürgen Teich (Coordinator and PI) and Dr. Sandra Mattauch (Public Relations)

In the following sections, we summarise major events such as the DFG Review meeting (Section 6) and further activities in 2014. These include Internal Meetings (Section 7), Trainings and Tutorials (Section 8) as well as further InvasIC Activities (Section 9). Last but not least, we present the current composition of the Industrial and Scientific Board and provide a short summary of the Board's meeting in October 2014 in Section 10.

Figure 5.4: At the Annual Meeting in Aalen, October 2014

6 DFG Review

In February 2014, our CRC/Transregio 89 was successfully reviewed for its second funding phase by the German Research Foundation (DFG) and 15 peer experts. Already during 2013, nearly all members were involved in preparing the funding proposal and the review. In the beginning of 2014, two rehearsals took place in Garching where researchers from all projects met to prepare for the DFG review: talks were given, posters presented and the demonstrators were set up and rehearsed.

The review started in the morning of February 11 with a plenary meeting of the review panel. In the following session, the PIs presented their individual project results in short talks and the doctoral researchers provided further information during a poster session. The doctoral researchers also introduced two demonstrators to the reviewers: One comprehensive platform was developed to demonstrate all major aspects of invasive computing at language, application, OS-level and also highlighted all architectural design features of the developed multicore architecture. The second demonstrator served to show resource awareness of invasive applications for robot vision on a heterogeneous tiled architecture. The first day ended with a closed meeting of the review panel.

At the plenary discussion in the morning of the second day, representatives from the applicant universities outlined the structural integration and role of the Collaborative Research Centre for the respective university and how each university intends to support the CRC/Transregio 89. During the second internal meeting, the review panel decided to recommend our proposal to the Grants Committee on Collaborative Research Centres.

In May 2014, the Grants Committee finally decided to fund our CRC/-Transregio 89 for another term of 4 years!

Figure 6.1: At the DFG review in Garching, February 2014

7 Internal Meetings

Collaboration between the researchers of the three sites Karlsruhe, München and Erlangen is essential for the success of the CRC/Transregio 89 – InvasIC. In 2014, researchers met at the following opportunities:

Event	Date	
Rehearsal I	Jan. 10, 2014, Garching	In Garching, researchers from all projects met to prepare the DFG review in february.
Rehearsal II	Jan. 28, 2014, Garching	In Garching, researchers from all projects met to prepare the DFG review in february.
Review	Feb. 11/12, 2014, Garching	The CRC/Transregio 89 presented results of the first funding phase in plenary talks, posters and demonstrators to the reviewers.
Doctoral Researcher Retreat	Sep. 15–17, 2014, Ellwangen	The 6th InvasIC DRR took place in Ellwangen to discuss further challenges of the second funding phase.
Annual Meeting 2014	Oct. 23/24, 2014, Aalen	New projects and working groups presented their ideas with focus on the second funding phase. On the meeting's second day, members of the "Industrial and Scientific Board" attended to evaluate the ideas and progress of the presented projects.

Figure 7.1: At the DFG review in Garching, February 2014

8 Trainings and Tutorials

Workshops and trainings were organised under the coordination of Project Z, to give InvasIC members the opportunity to strengthen their soft skills, train their key qualifications, and improve their knowledge on invasive computing related topics. Moreover, members of InvasIC took the opportunity to participate at a workshop on "Intracultural Communication" (organised by Graduiertenkolleg "Heterogene Bildsysteme").

Event	Date	
Workshop Voice, Standing and Communication	Nov. 18, 2014 Munich	The seminar of Astrid Herrmann was organised for young researchers to train their voice and performance for presentations.
MPSoC Winter School Design, Programming and Applications of Multi-Processor Systems on Chip	Nov. 26, 2014 Tunis,Tunisia	Prof. Dr.-Ing. J. Teich (FAU) gave an invited lecture about "Coarse-Grained Reconfigurable Architectures - Design and Programming".
Workshop Poster Presentation	Nov. 27/28, 2014 Erlangen	In this two-day workshop, Paul Gahman taught basics of poster design and presentation.

Figure 8.1: Voice, Standing and Communication Workshop, Munich, November 2014

9 InvasIC Activities

To promote the ideas and results of InvasIC and discuss them with leading experts from industry and academia, guest speakers were invited to the "InvasIC Seminar". Moreover, PIs of InvasIC gave talks and seminars at important research sites and conferences ("Invited Talks and Seminars") or organised workshops ("Workshops" and "Conferences") on the topics of Invasive Computing.
The "InvasIC Seminar" is a series of talks given at one of the three sites. Videos of the respective talks are provided at our website `http://www.invasic.de`.

Figure 9.1: Prof. Peter Marwedel together with Prof. Jürgen Teich during an InvasIC Seminar

Figure 9.2: Prof. Albert Cohen giving a talk at the InvasIC Seminar

InvasIC Seminar

Time and Place	Title	Speaker
Munich, Feb. 18, 2014	GungHo!ing Weather and Climate	Dr. I. Kavčič (University of Exeter)
Erlangen, March 21, 2014	An Application-Specific Processor for Real-Time Medical Monitoring	Prof. P. Brisk (University of California)
Erlangen, March 21, 2014	Execution Stream Fingerprinting for Low-cost Safety-critical System Design	Prof. B. Meyer (McGill University)
Erlangen, March 21, 2014	Heterogeneous Multi-cores in the Dark Silicon Era	Prof. T. Mitra (National University of Singapore)
Munich, May 8, 2014	An Asymptotic Parallel-in-Time Method for Highly Oscillatory PDEs	Prof. B. Wingate (University of Exeter)
Erlangen, Jun. 13, 2014	Crown Scheduling: Energy-Efficient On-Chip Pipelining for Moldable Parallel Streaming Tasks	Prof.C. Kessler (Linköping University)
Erlangen, Jun. 27, 2014	Efficient Computing in Cyber-Physical Systems	Prof. P. Marwedel (TU Dortmund)
Erlangen, Jul. 4, 2014	Application-driven Embedded System Design	Prof. A. Cohen (École polytechnique)
Erlangen, Nov. 21, 2014	Making data flow, dynamically	Prof. T. Basten (Eindhoven University of Technology)

Figure 9.3: Guests in the InvasIC Seminar. From left to right: Prof. Philip Brisk, Dr. Frank Hannig, Prof. Jürgen Teich, Prof. Tulika Mitra, Prof. Brett Meyer

Invited Talks and Seminars

Date and Place	Title	Speaker
Nuremberg, Germany, Feb. 25, 2014 Special Session 9: Multicore processors for embedded systems: Are we ready?, Embedded World Conference 2014	Talk: Multicore processors for embedded systems: Are we ready?	Prof. J. Teich (FAU)
Barcelona, Spain, May 13, 2014 HiPEAC Computer Systems Week 2014	Keynote: The Dark Silicon Problem in Multi-Core Systems - Invasive Computing as a Solution	Prof. J. Henkel (KIT)
Barcelona, Spain, May 13, 2014 Special Session: Dynamic co-optimization of applications and resource management, HiPEAC Computer Systems Week 2014	Talk: Assisting Run-time Optimization of Many-Core Systems by Design-time Characterization	Prof. M. Glaß (FAU)
Barcelona, Spain, May 13, 2014 Special Session: Dynamic co-optimization of applications and resource management, HiPEAC Computer Systems Week 2014	Talk: Resource Aware Programming with Invasive MPI	I. Compres Urena (TUM)

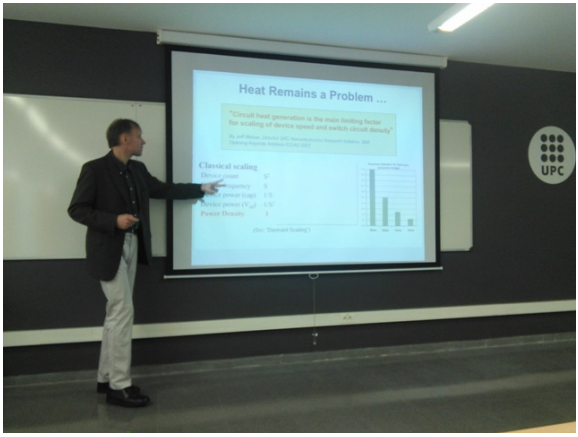


Figure 9.4: Prof. Jörg Henkel giving a talk at HiPEAC Computer Systems Week 2014, Barcelona

Date and Place	Title	Speaker
Bologna, Italy, May 23, 2014 Seminar Series "Trends in Electronics"	Talk: Foundations and Benefits of Invasive Computing	Prof. J. Teich (FAU)
New York, USA, Jun. 23, 2014 D.E. Shaw Research	Talk: Automatic Tailoring of System Software: Rethinking the Application–Hardware Bridge	Dr.-Ing. D. Lohmann (FAU)
Montreal, Mc Gill University, Canada, Jul. 29, 2014	Seminar: Foundations and Benefits of Invasive Computing	Prof. J. Teich (FAU)
Frankfurt, Germany, Sep. 25, 2014 Deutsche Forschungsgesellschaft für Automatisierung und Mikroelektronik (DFAM)	Talk: System-Level Design Automation of Embedded Systems	Prof. J. Teich (FAU)
Madrid, Spain, Oct. 8, 2014 Conference on Design and Architectures for Signal and Image Processing (DASIP)	Keynote: Invasive Computing - Principles and Benefits	Prof. J. Teich (FAU)
Seoul, Korea, Oct. 21, 2014 32nd IEEE International Conference on Computer Design (ICCD)	Keynote: Dependability of On-Chip Systems in the Dark Silicon Era	Prof. J. Henkel (KIT)

Figure 9.5: Prof. Michael Glaß giving a talk at HiPEAC Computer Systems Week 2014, Barcelona

Organised Workshops

Date and Place	Title	Organiser
Luebeck, Germany, Feb. 25-28, 2014 International Conference on Architecture of Computing Systems (ARCS 2014)	First International Workshop on Multi-Objective Many-Core Design (MOMAC)	Dr.-Ing. S. Wildermann (FAU) Prof. M. Glaß (FAU)
Dresden, Germany, Mar. 28, 2014 (DATE 2014)	Workshop on Performance, Power and Predictability of Many-Core Embedded Systems (3PMCES)	Prof. W. Stechele (TUM)
Barcelona, Spain, May 13, 2014 HiPEAC Computer Systems Week 2014	Workshop on Dynamic co-optimization of applications and resource management	Prof. M. Gerndt (TUM) Dr. J. Weidendorfer (TUM)
Paderborn, Germany, May 29/30, 2014 IEEE European Test Symposium (ETS)	Workshop on Resource awareness and adaptivity in multi-core computing (Racing 2014)	Prof. J. Teich (FAU) Dr.-Ing. F. Hannig (FAU)
New Delhi, India, Oct. 13, 2014 Special Session at CODES+ISSS, ESWEEK 2014	Special Session on Dark Silicon as a Challenge for Hardware-Software Co-Design	Dr.-Ing. M. Shafique (KIT) Prof. S. Garg (University of Waterloo, Canada)
San Jose, USA, Nov. 6, 2014 International Conference on Computer-Aided Design (ICCAD)	Workshop on A Roadmap for EDA Research in the Dark Silicon Era	Dr.-Ing. M. Shafique (KIT)
Pacific Grove, USA, Nov. 3, 2014 Asilomar Conference on Signals, Systems and Computers	Special Session on Resource-aware and Domain-specific Computing	Dr.-Ing. F. Hannig (FAU)

10 Industrial and Scientific Board

For the promotion of our ideas to the industrial community and for the discussion with peer colleagues world-wide, we established the InvasIC Industrial and Scientific Board. Members of the board in its current constitution are 8 experts from 7 institutions industry and university:

IBM

Dr. Peter Roth (IBM Böblingen)

Dr. Patricia Sagmeister (IBM Rüschlikon)

Intel

Hans-Christian Hoppe (Intel Director of ExaCluster Lab Jülich, Intel Director of Visual Computing Institute Saarbrücken)

Siemens

Urs Gleim (Head of Research Group Parallel Systems Germany, Siemens Corporate Technology)

University of Edinburgh

Prof. Dr. Michael O'Boyle
(Director Institute for Computing Systems Architecture)

Georg-Simon-Ohm Hochschule Nürnberg

Prof. Dr. Christoph von Praun
(Faculty Member and Associate Department Chair)

IAV - Automotive Engineering

Elmar Maas (IAV, Gifhorn)

XILINX

Michaela Blott (Xilinx, Dublin)

The members of the InvasIC Industrial and Scientific Board are periodically informed about progress and news of the CRC/Transregio InvasIC. In October 2014, the members of the Board met during the Annual Meeting in Aalen. In talks and demonstrations given by the different projects as well as during a poster session, the Board's members could get an idea about the current state of research in InvasIC. In a concluding plenary session, the opinions and suggestions of the Board's members were collected and discussed.

Figure 10.1: Members of the Industrial and Scientific Board during the plenary session at the Annual Meeting in Aalen, October 2014. From left to right: Elmar Maas (IAV), Peter-Hans Roth (IBM), Michaela Blott (Xilinx), Klaus-Dieter Schubert (IBM), Urs Gleim (Siemens), Hans-Christian Hoppe (Intel)

11 Publications

[AFGM14] M. Anikeev, F. Freiling, J. Götzfried, and T. Müller. “Secure garbage collection: Preventing malicious data harvesting from deallocated Java objects inside the Dalvik VM”. In: *Journal of Information Security and Applications*. Amsterdam, 2014. DOI: 10.1016/j.jisa.2014.10.001.

[BBHR14] M. Bader, A. Breuer, W. Hölzl, and S. Rettenberger. “Vectorization of an Augmented Riemann Solver for the Shallow Water Equations”. In: *Proceedings of the 2014 International Conference on High Performance Computing and Simulation (HPCS 2014)*. Ed. by W. W. Smari and V. Zeljkovic. IEEE, 2014, pp. 193–201.

[Bad+11] M. Bader, H.-J. Bungartz, M. Gerndt, A. Hollmann, and J. Weidendorfer. “Invasive Programming as a Concept for HPC”. In: *Proceedings of the 10th IASTED International Conference on Parallel and Distributed Computing and Networks 2011 (PDCN)*. Feb. 2011.

[BHT14] S. Boppu, F. Hannig, and J. Teich. “Compact Code Generation for Tightly-Coupled Processor Arrays”. In: *Journal of Signal Processing Systems* 77.1–2 (May 31, 2014), pp. 5–29. ISSN: 1939-8018. DOI: 10.1007/s11265-014-0891-2.

[BBMZ14] M. Braun, S. Buchwald, M. Mohr, and A. Zwinkau. *Dynamic X10: Resource-Aware Programming for Higher Efficiency*. Tech. rep. 8. X10 ’14. Karlsruhe Institute of Technology, 2014. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000041061.

[Bra+13] M. Braun, S. Buchwald, S. Hack, R. Leißa, C. Mallon, and A. Zwinkau. “Simple and Efficient Construction of Static Single Assignment Form”. In: *Compiler Construction*. Ed. by R. Jhala and K. Bosschere. Vol. 7791. LNCS. Springer, 2013, pp. 102–122. DOI: 10.1007/978-3-642-37051-9_6.

[Buc15] S. Buchwald. “Optgen: A Generator for Local Optimizations”. In: *Compiler Construction*. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2015. forthcoming.

[Bun+13] H.-J. Bungartz, C. Riesinger, M. Schreiber, G. Snelting, and A. Zwinkau. “Invasive Computing in HPC with X10”. In: *X10 Workshop (X10’13)*. X10 ’13. New York, NY, USA: ACM, 2013, pp. 12–19. DOI: 10.1145/2481268.2481274.

[CPSV14] R. de Clercq, F. Piessens, D. Schellekens, and I. Verbauwhede. “Secure Interrupts on Low-End Microcontrollers”. In: *IEEE 25th International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. 2014, pp. 147–152. DOI: 10.1109/ASAP.2014.6868649.

[Dan+14] D. Danner, R. Müller, W. Schröder-Preikschat, W. Hofer, and D. Lohmann. “Safer Sloth: Efficient, Hardware-Tailored Memory Protection”. In: *Proceedings of the 20th IEEE International Symposium on Real-Time and Embedded Technology and Applications (RTAS ’14)*. Washington, DC, USA: IEEE Computer Society Press, 2014, pp. 37–47. URL: https://www4.cs.fau.de/~danner/Publications/2014/danner_14_rtas.pdf.

[FSS13] P. Flick, P. Sanders, and J. Speck. “Malleable Sorting”. In: *Parallel Distributed Processing (IPDPS), 2013 IEEE 27th International Symposium on*. ISSN 1530-2075. 2013, pp. 418–426. DOI: 10.1109/IPDPS.2013.90.

[FPZ14] F. Freiling, M. Protsenko, and Y. Zhuang. “An Empirical Evaluation of Software Obfuscation Techniques applied to Android APKs”. In: *International Workshop on Data Protection in Mobile and Pervasive Computing*. Ed. by J. Lin and T. Müller. Beijing, China, 2014.

[FHB14] S. Friederich, J. Heisswolf, and J. Becker. “Hardware/software debugging of large scale many-core architectures”. In: *Proceedings of the 27th Symposium on Integrated Circuits and Systems Design (SBCCI)*. IEEE, Sept. 2014, pp. 1–6.

[FHMB14] S. Friederich, J. Heisswolf, D. May, and J. Becker. “Hardware prototyping and software debugging of multi-core architectures”. In: *Proceedings of the Synopsys Users Group Conference (SNUG)*. Munich, Germany, 2014.

[GTHT14] D. Gangadharan, A. Tanase, F. Hannig, and J. Teich. “Timing Analysis of a Heterogeneous Architecture with Massively Parallel Processor Arrays”. In: *DATE Friday Workshop on Performance, Power and Predictability of Many-Core Embedded Systems (3PMCES)*. Dresden, Germany: ECSI, Mar. 28, 2014. URL: http://www.ecsi.org/resource/workshop/2014/3PMCES/DATE/paper/timing-analysis-heterogeneous-architecture-massively-parallel-processor-arrays.

[Gan+14] D. Gangadharan, É. Sousa, V. Lari, F. Hannig, and J. Teich. “Application-driven Reconfiguration of Shared Resources for Timing Predictability of MPSoC Platforms”. In: *Proceedings of Asilomar Conference on Signals, Systems, and Computers (ASILOMAR)*. (Pacific Grove, CA, USA). IEEE, Nov. 2–5, 2014. Forthcoming.

[Ger+12a] M. Gerndt, F. Hannig, A. Herkersdorf, A. Hollmann, M. Meyer, S. Roloff, J. Weidendorfer, T. Wild, and A. Zaib. “An Integrated Simulation Framework for Invasive Computing”. In: *Proceedings of the Forum on Specification and Design Languages (FDL)*. Vienna, Austria: IEEE, Sept. 18–20, 2012, pp. 209–216. ISBN: 978-1-4673-1240-0.

[Ger+12b] M. Gerndt, A. Hollmann, M. Meyer, M. Schreiber, and J. Weidendorfer. “Invasive computing with iOMP”. In: *Proceedings of the Forum on Specification and Design Languages (FDL)*. Vienna, Austria, Sept. 18–20, 2012, pp. 225–231. ISBN: 978-2-9530504-5-5.

[GS14] D. Giffhorn and G. Snelting. “A New Algorithm for Low-Deterministic Security”. In: *International Journal on Information Security* (Aug. 2014). DOI: 10.1007/s10207-014-0257-6. URL: http://pp.ipd.kit.edu/uploads/publikationen/giffhorn14lsod.pdf.

[GBTW14] M. Glaß, M. Bader, J. Teich, and S. Wildermann. “Assisting Run-time Optimization of Many-Core Systems by Design-time Characterization”. HiPEAC Spring Computing Systems Week, Barcelona, Invited Talk. May 13, 2014.

[Glo+14] E. Glocker, S. Boppu, Q. Chen, U. Schlichtmann, J. Teich, and D. Schmitt-Landsiedel. “Temperature modeling and emulation of an ASIC temperature monitor system for Tightly-Coupled Processor Arrays (TCPAs)”. In: *Advances*

in Radio Science 12 (2014), pp. 103–109. DOI: 10.5194/ars-12-103-2014. URL: http://www.adv-radio-sci.net/12/103/2014/.

[Glo+14a] E. Glocker, Q. Chen, A. Zaidi, U. Schlichtmann, and D. Schmitt-Landsiedel. "Emulated ASIC Power and Temperature Monitor System for FPGA Prototyping of an Invasive MPSoC Computing Architecture". In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 14–15. arXiv:1405.2909 [cs.AR].

[Glo+14b] E. Glocker, Q. Chen, A. Zaidi, U. Schlichtmann, and D. Schmitt-Landsiedel. "Emulierung eines ASIC-Leistungsverbrauchs- und Temperaturmonitorsystems für FPGA-Prototyping eines ressourcengewahren Computersystems". In: *16. Workshop Analogschaltungen, Wien, Österreich*. 2014.

[GM14] J. Götzfried and T. Müller. "Mutual Authentication and Trust Bootstrapping towards Secure Disk Encryption". In: *Transactions on Information and System Security (TISSEC)*. Vol. 17. New York, 2014. DOI: 10.1145/2663348. URL: https://www1.cs.fau.de/mark.

[GRGT14] S. Graf, F. Reimann, M. Glaß, and J. Teich. "Towards Scalable Symbolic Routing for Multi-Objective Networked Embedded System Design and Optimization". In: *Proceedings of the International Conference on Hardware/-Software Codesign and System Synthesis (CODES+ISSS 2014)*. Oct. 12, 2014, 2:1–2:10. DOI: 10.1145/2656075.2656102.

[GBH14a] A. Grudnitsky, L. Bauer, and J. Henkel. "COREFAB: Concurrent Reconfigurable Fabric Utilization in Heterogeneous Multi-Core Systems". In: *International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656106.2656119.

[GBH14b] A. Grudnitsky, L. Bauer, and J. Henkel. "MORP: Makespan Optimization for Processors with an Embedded Reconfigurable Fabric". In: *Proceedings of the 22nd ACM/SIGDA International Symposium on Field Programmable Gate Arrays*

(FPGA). Monterey, CA, USA, Feb. 26–28, 2014, pp. 127–136. DOI: 10.1145/2554688.2554782.

[HBH14] M. Haaß, L. Bauer, and J. Henkel. "Automatic Custom Instruction Identification in Memory Streaming Algorithms". In: *International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656106.2656114.

[HT14] F. Hannig and J. Teich, eds. *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. May 29–30, 2014. 62 pp. arXiv:1405.2281 [cs.DC].

[Han+14] F. Hannig, V. Lari, S. Boppu, A. Tanase, and O. Reiche. "Invasive Tightly-Coupled Processor Arrays: A Domain-Specific Architecture/Compiler Co-Design Approach". In: *ACM Transactions on Embedded Computing Systems (TECS)* 13.4s (Apr. 2014), 133:1–133:29. ISSN: 1539-9087. DOI: 10.1145/2584660.

[Hei14] J. Heisswolf. "A Scalable and Adaptive Network on Chip for Many-Core Architectures". Dissertation. Institut für Technik der Informationsverarbeitung, Karlsruhe Institute of Technology, Nov. 2014.

[Hei+14a] J. Heisswolf, A. Zaib, A. Zwinkau, S. Kobbe, A. Weichslgartner, J. Teich, J. Henkel, G. Snelting, A. Herkersdorf, and J. Becker. "CAP: Communication Aware Programming". In: *Design Automation Conference (DAC), 2014 51th ACM / EDAC / IEEE*. 2014, 105:1–105:6.

[Hei+14b] J. Heisswolf, A. Zaib, A. Weichslgartner, M. Karle, M. Singh, T. Wild, J. Teich, A. Herkersdorf, and J. Becker. "The Invasive Network on Chip - A Multi-Objective Many-Core Communication Infrastructure". In: *Proceedings of the first International Workshop on Multi-Objective Many-Core Design (MOMAC) in conjunction with International Conference on Architecture of Computing Systems (ARCS)*. Lübeck, Germany: IEEE, Feb. 25, 2014.

[Hen14a] J. Henkel. "Adaptive Embedded Computing with i-Core". Keynote Talk, 6th Workshop on Adaptive and Reconfigurable Embedded Systems, CPSWeek (APRES). Berlin, Germany, Apr. 14, 2014.

[Hen14b] J. Henkel. “Dependability of On-Chip Systems in the Dark Silicon Era”. Keynote Talk, 32nd IEEE International Conference on Computer Design (ICCD). Seoul, Korea, Oct. 19–22, 2014.

[Hen14c] J. Henkel. “The Dark Silicon Problem in Multi-Core Systems – Invasive Computing as a Solution”. Keynote Talk, Thematic Session at HiPEAC Computer Systems Week. Barcelona, Spain, May 13, 2014.

[HBGZ14] J. Henkel, L. Bauer, A. Grudnitsky, and H. Zhang. “Adaptive Embedded Computing with i-Core”. In: *ACM SIGBED Review – Special Issue on the 6th Workshop on Adaptive and Reconfigurable Embedded Systems*. Vol. 11. Extended Abstract for Keynote Talk. Oct. 2014, pp. 20–21. DOI: 10.1145/2692385.2692387.

[Hof14] W. Hofer. “Sloth: The Virtue and Vice of Latency Hiding in Hardware-Centric Operating Systems”. PhD thesis. Erlangen: Friedrich-Alexander-Universität Erlangen-Nürnberg, 2014. URL: http://opus4.kobv.de/opus4-fau/files/4875/WanjaHoferDissertation.pdf.

[Hön+14] T. Hönig, H. Janker, O. Mihelic, C. Eibel, R. Kapitza, and W. Schröder-Preikschat. “Proactive Energy-Aware Programming with PEEK”. In: *2014 Conference on Timely Results in Operating Systems (TRIOS ’14)*. Broomfield, CO: USENIX Association, Oct. 2014. URL: https://www.usenix.org/conference/trios14/technical-sessions/presentation/hoenig.

[KSH14] M. U. K. Khan, M. Shafique, and J. Henkel. “Software Architecture of High Efficiency Video Coding for Many-core Systems with Power-efficient Workload Balancing”. In: *Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Dresden, Germany, Mar. 24–28, 2014. DOI: 10.7873/DATE.2014.232.

[Kle+14] V. B. Kleeberger, M. Barke, C. Werner, D. Schmitt-Landsiedel, and U. Schlichtmann. “A Compact Model for NBTI Degradation and Recovery under Use-Profile Variations and its Application to Aging Analysis of Digital Integrated Circuits”. In: *Microelectronics Reliability* 54.6-7 (2014), pp. 1083–1089. DOI: 10.1016/j.microrel.2013.12.002. URL: http://www.sciencedirect.com/science/article/pii/S0026271413004368.

[KBH15] S. Kobbe, L. Bauer, and J. Henkel. “Adaptive on-the-fly Application Performance Modeling for Many Cores”. In: *Proceedings of Design, Automation and Test in Europe Conference (DATE)*. Grenoble, France, Mar. 9–13, 2015.

[Krö+14] M. Kröhnert, N. Vahrenkamp, J. Paul, W. Stechele, and T. Asfour. “Resource Prediction for Humanoid Robots”. In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 22–28. arXiv:1405.2911 [cs.RO].

[KM14] C. Kugler and T. Müller. “SCADS: Separated Control- and Data-Stacks (Best Student Paper Award)”. In: *10th International Conference on Security and Privacy in Communication Networks*. Ed. by S. I. ICST (The Institute for Computer Sciences and T. Engineering). Beijing, China, 2014. URL: https://www1.cs.fau.de/filepool/scads/scads-securecomm2014.pdf.

[LTHT14] V. Lari, A. Tanase, F. Hannig, and J. Teich. “Massively Parallel Processor Architectures for Resource-aware Computing”. In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 1–7. arXiv:1405.2907 [cs.AR].

[Lar+13] V. Lari, S. Muddasani, S. Boppu, F. Hannig, M. Schmid, and J. Teich. “Hierarchical Power Management for Adaptive Tightly-Coupled Processor Arrays”. In: *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 18.1 (Jan. 2013), 2:1–2:25. DOI: 10.1145/2390191.2390193.

[Loc14] A. Lochbihler. “Making the Java Memory Model Safe”. In: *ACM Transactions on Programming Languages and Systems* 35.4 (2014), 12:1–12:65. DOI: 10.1145/2518191. URL: http://pp.ipd.kit.edu/uploads/publikationen/lochbihler14toplas.pdf.

[LBS14] D. Lorenz, M. Barke, and U. Schlichtmann. “Monitoring of aging in integrated circuits by identifying possible critical paths”. In: *Journal of Microelectronics Reliability* 54 (2014), pp. 1075 –1082. DOI: http://dx.doi.org/10.1016/j.microrel.2014.01.013. URL: http://www.

sciencedirect.com/science/article/pii/S0026271414000298.

[MDSPL14] R. Müller, D. Danner, W. Schröder-Preikschat, and D. Lohmann. “MultiSloth: An Efficient Multi-Core RTOS using Hardware-Based Scheduling”. In: *Proceedings of the 26th Euromicro Conference on Real-Time Systems (ECRTS ’14)*. (Madrid, Spain). Washington, DC, USA: IEEE Computer Society Press, 2014, pp. 289–198. ISBN: 978-1-4799-5798-9. DOI: 10.1109/ECRTS.2014.30.

[Moh+13] M. Mohr, A. Grudnitsky, T. Modschiedler, L. Bauer, S. Hack, and J. Henkel. “Hardware Acceleration for Programs in SSA Form”. In: *International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES)*. Montreal, Canada, Oct. 2013. DOI: 10.1109/CASES.2013.6662518.

[Mun+14] W. Munawar, H. Khdr, S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel. “Peak Power Management for Scheduling Real-time Tasks on Heterogeneous Many-Core Systems”. In: *20th IEEE International Conference on Parallel and Distributed Systems (ICPADS)*. Hsinchu, Taiwan, Dec. 16–19, 2014.

[Noo+13] J. Noorman, P. Agten, W. Daniels, R. Strackx, A. Van Herrewege, C. Huygens, B. Preneel, I. Verbauwhede, and F. Piessens. “Sancus: Low-cost trustworthy extensible networked devices with a zero-software trusted computing base”. In: *Proceedings of the 22nd USENIX Security Symposium*. (Washington, D.C.). USENIX Association, 2013, pp. 479–494.

[Nür+14] S. Nürnberger, G. Drescher, R. Rotta, J. Nolte, and W. Schröder-Preikschat. “Shared Memory in the Many-Core Age”. In: *Proceedings of the International Workshop on Runtime and Operating Systems for the Many-core Era (ROME 2014)*. Lecture Notes in Computer Science (LNCS). Porto, Portugal: Springer-Verlag, Aug. 26, 2014.

[Oec+14] B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, and W. Schröder-Preikschat. *OctoPOS: A Hardware-Assisted OS for Many-Cores*. Poster. Amsterdam, NL, Apr. 15, 2014.

[Pag+14] S. Pagani, H. Khdr, W. Munawar, J.-J. Chen, M. Shafique, M. Li, and J. Henkel. “TSP: Thermal Safe Power - Efficient power budgeting for Many-Core Systems in Dark Silicon”. In: *IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. Best Paper Award. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656075.2656103.

[Pau+14a] J. Paul, W. Stechele, M. Kröhnert, T. Asfour, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, and W. Schröder-Preikschat. “Resource-Aware Harris Corner Detection based on Adaptive Pruning”. In: *Proceedings of the Conference on Architecture of Computing Systems (ARCS)*. (Luebeck, Germany). LNCS 8350. Springer, Feb. 25–28, 2014, pp. 1–12. ISBN: 978-3-319-04890-1. DOI: 10.1007/978-3-319-04891-8.

[Pau+14b] J. Paul, W. Stechele, É. Sousa, V. Lari, F. Hannig, J. Teich, M. Kröhnert, and T. Asfour. “Self Adaptive Harris Corner Detection on Heterogeneous Many-core Processor”. In: *Proceedings of the Conference on Design & Architectures for Signal & Image Processing (DASIP)*. Oct. 2014, pp. 1–8.

[RHT14] S. Roloff, F. Hannig, and J. Teich. “Towards Actor-oriented Programming on PGAS-based Multicore Architectures”. In: *Workshop Proceedings of the 27th International Conference on Architecture of Computing Systems (ARCS)*. Lübeck, Germany: VDE Verlag, Feb. 25–28, 2014. ISBN: 978-3-8007-3579-2.

[Rol+13] S. Roloff, A. Weichslgartner, J. Heißwolf, F. Hannig, and J. Teich. “NoC Simulation in Heterogeneous Architectures for PGAS Programming Model”. In: *Proceedings of the 16th International Workshop on Software and Compilers for Embedded Systems (M-SCOPES)*. St. Goar, Germany: ACM, June 19–21, 2013, pp. 77–85. ISBN: 978-1-4503-2142-6. DOI: 10.1145/2463596.2463606.

[Sam+14a] F. Sampaio, M. Shafique, B. Zatt, S. Bampi, and J. Henkel. “dSVM: Energy-efficient distributed Scratchpad Video Memory Architecture for the next-generation High Efficiency Video Coding”. In: *Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Dresden, Germany, Mar. 24–28, 2014. DOI: 10.7873/DATE.2014.033.

[Sam+14b] F. Sampaio, M. Shafique, B. Zatt, S. Bampi, and J. Henkel. “Content-Driven Memory Pressure Balancing and Video Memory Power Management for Parallel High Efficiency Video Coding”. In: *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. La Jolla, CA, USA, Aug. 11–13, 2014. DOI: 10.1145/2627369.2627615.

[Sam+14c] F. Sampaio, M. Shafique, B. Zatt, S. Bampi, and J. Henkel. “Energy-Efficient Architecture for Advanced Video Memory”. In: *IEEE/ACM 33rd International Conference on Computer-Aided Design (ICCAD)*. San Jose, CA, USA, Nov. 3–6, 2014.

[Sch14] M. Schreiber. “Cluster-Based Parallelization of Simulations on Dynamically Adaptive Grids and Dynamic Resource Management”. Dissertation. Institut für Informatik, Technische Universität München, Jan. 2014. URL: http://www5.in.tum.de/pub/schreiber14dissertation_preprint.pdf.

[SRNB13] M. Schreiber, C. Riesinger, T. Neckel, and H.-J. Bungartz. “Invasive Compute Balancing for Applications with Hybrid Parallelization”. In: *Proceedings of the International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD)*. IEEE, Oct. 23–26, 2013.

[SWB13a] M. Schreiber, T. Weinzierl, and H.-J. Bungartz. “Cluster Optimization and Parallelization of Simulations with Dynamically Adaptive Grids”. In: *Euro-Par 2013*. Aachen, Germany, Aug. 2013.

[SWB13b] M. Schreiber, T. Weinzierl, and H.-J. Bungartz. “SFC-based Communication Metadata Encoding for Adaptive Mesh Refinement”. In: *Proceedings of the International Conference on Parallel Computing (ParCo)*. Ed. by M. Bader. Oct. 2013. In press.

[Sch+14] M. Schreiber, C. Riesinger, T. Neckel, H.-J. Bungartz, and A. Breuer. “Invasive Compute Balancing for Applications with Shared and Hybrid Parallelization”. In: *International Journal of Parallel Programming* (Sept. 2014). DOI: 10.1007/s10766-014-0336-3. URL: http://scholar.google.de/scholar_url?hl=en&q=http://link.springer.com/article/10.1007/s10766-014-0336-3&

sa=X&scisig=AAGBfm2kuCkaeu8QOohrdd5TiugbKlCWTQ&oi=scholaralrt.

[SBH14] M. Shafique, L. Bauer, and J. Henkel. “Adaptive Energy Management for Dynamically Reconfigurable Processors”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 33.1 (Jan. 2014), pp. 50–63. DOI: 10.1109/TCAD.2013.2282265.

[SGHM14] M. Shafique, S. Garg, J. Henkel, and D. Marculescu. “The EDA Challenges in the Dark Silicon Era: Temperature, Reliability, and Variability Perspectives”. In: *Proceedings of the The 51st Annual Design Automation Conference on Design Automation Conference (DAC)*. San Francisco, CA, USA, June 1–5, 2014. DOI: 10.1145/2593069.2593229.

[SH14] M. Shafique and J. Henkel. “Low Power Design of the Next-Generation High Efficiency Video Coding”. In: *Proceedings of the 19th Asia and South Pacific Design Automation Conference (ASP-DAC)*. Singapore, Jan. 20–23, 2014, pp. 274–281. DOI: 10.1109/ASPDAC.2014.6742902.

[SKH14] M. Shafique, M. U. K. Khan, and J. Henkel. “Power Efficient and Workload Balanced Tiling for Parallelized High Efficiency Video Coding”. In: *21st IEEE International Conference on Image Processing (ICIP)*. Paris, France, Oct. 27–30, 2014.

[Sha+14] M. Shafique, S. Garg, T. Mitra, S. Parameswaran, and J. Henkel. “Dark Silicon As a Challenge for Hardware/-Software Co-design: Invited Special Session Paper”. In: *IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. New Delhi, India, Oct. 12–17, 2014. DOI: 10.1145/2656075.2661645.

[Sne15] G. Snelting. “Understanding Probabilistic Software Leaks”. In: *Science of Computer Programming* 97, Part 1.0 (Jan. 2015). Special Issue on New Ideas and Emerging Results in Understanding Software, pp. 122–126. DOI: 10.1016/j.scico.2013.11.008. URL: http://www.sciencedirect.com/science/article/pii/S016764231300292X.

[Sne+14] G. Snelting, D. Giffhorn, J. Graf, C. Hammer, M. Hecker, M. Mohr, and D. Wasserrab. “Checking Probabilistic Noninterference Using JOANA”. In: *IT - Information Technology* (2014). invited article, currently under review. URL: http://pp.ipd.kit.edu/uploads/publikationen/joana14it.pdf.

[SGHT14] É. Sousa, D. Gangadharan, F. Hannig, and J. Teich. “Runtime Reconfigurable Bus Arbitration for Concurrent Applications on Heterogeneous MPSoC Architectures”. In: *Proceedings of the EUROMICRO Digital System Design Conference (DSD)*. Verona, Italy: IEEE, Aug. 27–29, 2014, pp. 74–81. DOI: 10.1109/DSD.2014.105.

[Sou+14] É. Sousa, V. Lari, J. Paul, F. Hannig, J. Teich, and W. Stechele. “Resource-Aware Computer Vision Application on Heterogeneous Multi-Tile Architecture”. Hardware and Software Demo at the University Booth at Design, Automation and Test in Europe (DATE), Dresden, Germany. Mar. 24–28, 2014.

[TWTH14] A. Tanase, M. Witterauf, J. Teich, and F. Hannig. “Symbolic Inner Loop Parallelisation for Massively Parallel Processor Arrays”. In: *Proceedings of the 12th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*. (Lausanne, Switzerland). Lausanne, Switzerland, Oct. 19–21, 2014, pp. 219–228.

[Tei14a] J. Teich. “Foundations and Benefits of Invasive Computing”. University of Bologna, Italy, Invited Talk in the Seminar Series Trends in Electronics. May 23, 2014.

[Tei14b] J. Teich. “Foundations and Benefits of Invasive Computing”. Seminar, Mc Gill University, Montreal. July 29, 2014.

[Tei14c] J. Teich. “Introduction to Invasive Computing”. Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014), Paderborn, Germany, Tutorial Talk. May 29, 2014.

[Tei14d] J. Teich. “Invasive Computing – Concepts and Benefits”. Keynote, DASIP 2014, Conference on Design and Architectures for Signal and Image Processing, Madrid, Spain. Oct. 8, 2014.

[Tei14e] J. Teich. “Reconfigurable Computing for MPSoC”. Invited Lecture, Winter School Design and Applications of Multi Processor System on Chip, Tunis, Tunesia. Nov. 26, 2014.

[Tei14f] J. Teich. “System-Level Design Automation of Embedded Systems”. Invited Talk, Tagung Deutsche Forschungsgesellschaft für Automatisierung und Mikroelektronik e.V. (DFAM). Sept. 25, 2014.

[TTH13] J. Teich, A. Tanase, and F. Hannig. “Symbolic Parallelization of Loop Programs for Massively Parallel Processor Arrays”. In: *Proceedings of the 24th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. Best Paper Award. Washington, DC, USA: IEEE, June 5–7, 2013, pp. 1–9. ISBN: 978-1-4799-0493-8. DOI: 10.1109/ASAP.2013.6567543.

[TTH14] J. Teich, A. Tanase, and F. Hannig. “Symbolic Mapping of Loop Programs onto Processor Arrays”. In: *Journal of Signal Processing Systems* 77.1–2 (July 11, 2014), pp. 31–59. ISSN: 1939-8018. DOI: 10.1007/s11265-014-0905-0.

[Tra+14a] C. Tradowsky, T. Gädeke, T. Bruckschlögl, W. Stork, K.-D. Müller-Glaser, and J. Becker. “SmartLoCore: A Concept for an Adaptive Power-Aware Localization Processor”. In: *Parallel, Distributed and Network-Based Processing (PDP), 2014 22nd Euromicro International Conference on*. Feb. 2014, pp. 478–481. DOI: 10.1109/PDP.2014.118.

[THMB15] C. Tradowsky, T. Harbaum, L. Masing, and J. Becker. “A Novel ADL-based Approach to Design Adaptive Application-Specific Processors”. In: *Best of IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. 2015. forthcoming.

[Tra+14b] C. Tradowsky, M. Schreiber, M. Vesper, I. Domladovec, M. Braun, H.-J. Bungartz, and J. Becker. “Towards Dynamic Cache and Bandwidth Invasion”. In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Vol. 8405. Lecture Notes in Computer Science. Springer International Publishing, Apr. 2014, pp. 97–107. DOI: 10.1007/978-3-319-05960-0_9.

[WHKSP14] P. Wägemann, T. Hönig, R. Kapitza, and W. Schröder-Preikschat. *Worst-Case Energy Consumption Analysis for Soft and Hard Energy Systems*. 11th USENIX Symposium on Operating System Design and Implementation (OSDI '14). Poster. Oct. 2014.

[Wei+14a] A. Weichslgartner, D. Gangadharan, S. Wildermann, M. Glaß, and J. Teich. "DAARM: Design-Time Application Analysis and Run-Time Mapping for Predictable Execution in Many-Core Systems". In: *Proceedings of the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS 2014)*. Oct. 12, 2014, 34:1–34:10. DOI: 10.1145/2656075.2656083.

[WBUW14] T. Weinzierl, M. Bader, K. Unterweger, and R. Wittmann. "Block Fusion on Dynamically Adaptive Spacetree Grids for Shallow Water Waves". In: *Parallel Processing Letters* 24.3 (2014), p. 1441006.

[Wei+14b] T. Weinzierl, R. Wittmann, K. Unterweger, M. Bader, A. Breuer, and S. Rettenberger. "Hardware-aware block size tailoring on adaptive spacetree grids for shallow water waves". In: *HiStencils 2014 – 1st International Workshop on High-Performance Stencil Computations*. 2014.

[Wel+13] K. Welke, N. Vahrenkamp, M. Wächter, M. Kröhnert, and T. Asfour. "The ArmarX Framework - Supporting high level robot programming through state disclosure". In: *Informatik 2013 Workshop on robot control architectures*. 2013.

[WGT14] S. Wildermann, M. Glaß, and J. Teich. "Multi-Objective Distributed Run-time Resource Management for Many-Cores". In: *Proceedings of Design, Automation and Test in Europe (DATE 2014)*. Dresden, Germany, Mar. 2014, pp. 1–6. DOI: 10.7873/DATE.2014.234.

[ZRWH14] A. Zaib, P. Raju, T. Wild, and A. Herkersdorf. "A Layered Modeling and Simulation Approach to investigate Resource-aware Computing in MPSoCs". In: *Proceedings of the First Workshop on Resource Awareness and Adaptivity in Multi-Core Computing (Racing 2014)*. (Paderborn, Germany). May 29–30, 2014, pp. 51–56. arXiv:1405.2917 [cs.OH].

[Zai+13] A. Zaib, J. Heisswolf, A. Weichslgartner, T. Wild, J. Teich, J. Becker, and A. Herkersdorf. “AUTO-GS: Self-optimization of NoC Traffic Through Hardware Managed Virtual Connections”. In: *Proceedings of the 16th Euromicro Conference on Digital System Design (DSD)*. Santander, Spain: IEEE, Sept. 4–6, 2013, pp. 761–768. DOI: 10.1109/DSD.2013.87.

[Zai+15] A. Zaib, J. Heisswolf, A. Weichslgartner, T. Wild, J. Teich, J. Becker, and A. Herkersdorf. “Network Interface with Task Spawning Support for NoC-based DSM Architectures”. In: *28th GI/ITG International Conference on Architecture of Computing Systems (ARCS)*. To appear. Springer Lecture Notes on Computer Science (LNCS), 2015.