Transregional Collaborative Research Centre 89

Invasive Computing

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

Annual Report 2015

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
Martensstraße 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 2015.

The CRC/Transregio 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.

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).

A highlight in 2015 was a Doctoral Seminar on Predictability held at the Sarntal Academy in September, organised by Prof. Michael Glaß and Prof. Michael Gerndt. About 20 doctoral researchers and 5 guest lecturers presented and discussed the state-of-the-art and ongoing research on predictability in the individual projects.

We would like to thank all members of the SFB/Transregio InvasIC and all our partners from industry and academia for the fruitful collaborations and inspiring discussions in the last year! We do hope that you will enjoy reading about the progress achieved in 2015, as well as about our research planned for the following years.

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.

Current 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 current funding phase, we 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 may not only be of type performance (e.g. execution time, throughput, etc.),

but do 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 current 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
- PD 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 CRC/Transregio is organised in five 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. rer. nat. 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, PD 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 Project 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. We describe our results obtained in these research areas during 2015 in the following.

Invasive Programming Patterns

The Malleable Pattern

We investigate mechanisms that enable dynamic resource adaptation in a safe way. The concept of malleable applications is proven to be efficient[1]. The common definition of "malleable"[2] already applies to all invasive applications. We research a more narrow class of applications, called "async-malleable", which are able to react to *external* adaptations of the set of assigned resources at *any* time (e. g. triggered by another application retreating). Such applications are not restricted to pick specific points where adaptation is possible. We designed an interface [BMZ15],

[1] P. Flick, P. Sanders, and J. Speck, "Malleable Sorting", IPDPS 2013

[2] D. G. Feitelson and L. Rudolph, "Towards Convergence in Job Schedulers for Parallel Supercomputers", IPPS 1996

```
1 val ilet = (id:IncarnationID)=>{
2   for (job in queue) {
3     if (queue.checkTermination(id)) break;
4     job.do(); } }
5 val resizeHandler = (add:List[PE], remove:List[PE])=>{
6   for (pe in add) queue.addWorker(pe,ilet);
7   queue.adapt();
8   for (pe in remove) queue.signalTermination(pe); }
9 val constraints = new PEQuantity(4,10)
10  && new Malleable(resizeHandler)
11  && new ScalabilityHint(speedupCurve);
12 val claim = Claim.invade(constraints);
13 queue.adaptTo(claim);
14 claim.infect(ilet);
15 claim.retreat();
```

Figure 4.1: Example of a asynchronously-malleable invasive application with a master-slave structure and a global queue of jobs. First, `ilet` states the actual computation to perform. Second, `resizeHandler` defines how to handle resource changes. Then, `constraints` describes a malleable claim of 4 to 10 PEs and a speedup curve defined elsewhere. The `invade` call returns a resource claim and `queue` gets a chance to internally adapt itself to the claimed resources (e. g. for work stealing). Then `infect` starts an *i*-let on each PE of the claim, which execute jobs from the global queue. Finally, we `retreat` the claim to free the resources.

where a malleable application must provide a handler function, which the resource manager calls whenever it changes the claim. The handler must have access to resources which are about to be removed as well as the freshly gained resources. An example of the programmers interface is shown by Figure 4.1. Such applications promise to be much more adaptive and thus efficient. An evaluation is ongoing.

The Job Queue Pattern

Invasive computing requires the programmer to think in a resource-aware way and this is the price to pay. Nevertheless, we want to lower this price. Hence, another objective in the current funding phase is to provide more convenient tools for the programmer. We developed an invasive distributed job queue framework [Böw15], which can be applied to sorting or numerical integration. Applications using the framework are implicitly async-malleable.

Predictable Invasive Applications

Project A1 contributed a lot to defining the program and organising the Sarntal 2015 course “Benchmarking for Multi-Criteria-Predictable Multi-Core Computing”. One result of this course was the definition of new language elements and compiler annotations for specifying future actor-based applications within the parallel processing language of choice X10 like in the following example:

```
1
2     //This closure describes the behaviour of an actor
3     //with only connection to other actors via Channels.
4     val actorA = (a:Actor) => {
5       // closure implementing A
6     };
7
8     val actorB = (a:Actor) => {
9       // closure implementing B
10    }
11
12    // Create actor objects
13    val aA = new Actor(actorA);
14    val aB = new Actor(actorB);
15
16    // Create an Actor Graph (with global requirements).
17    val ag = @REQUIRE(Latency(10, "ms", "hard")) new ActorGraph();
18    tg.addActor(aA);
19    tg.addActor(aB);
20    tg.addChannel(aB, aA);
21
22    // keyword for triggering the characterization of the actor graph.
23    Magic.run(ag);
```

The source code above illustrates the creation of two actors and their communication by an actor graph. In this example, the application has the *requirement* that its execution latency should not exceed 10 ms, specified via the new `@REQUIRE` annotation. Line 22 specifies that the generated actor graph should be characterised by the design space exploration (DSE) of Project A4. The result of the DSE is a set of optimised *operating points* with different resource needs, where each point is verified by DSE to fulfil the requirements as specified in the source code. This characterisation is then transformed back into X10 source code. For example, line 22 above will be replaced by following code snippet during compilation and after DSE:

```
1    //OperatingPoints from characterization
2    val operatingPoints = Characterization.getOperatingPoints(ag);
3
4    //Tell run-time system to allocate resource acc. to one operating
         point.
5    val binding = Claim.invade(operatingPoints);
6
7    //Bind and execute actors on allocated resources.
8    binding.infect();
```

The set of operating points can be generated by the DAARM flow [Wei+14], developed together with Project A4 and illustrated in Figure 4.2.

Analysis of Predictable Invasive Computing

Manycore systems will need to support highly dynamic workloads, i. e. varying application mixes and different modes of operation controlled by power and temperature management. The challenge here is, on the one hand, that the temporal (and further non-functional execution) characteristics of a mapped program or application can be influenced by other applications in such a dynamic environment because of resource sharing, so that e. g. deadlines might be violated if no precaution is taken. In [RWHT15], we have shown, in close cooperation with Project C2, that by means of exclusive resource reservations, it is possible to spatially isolate applications from each other and avoid any unwanted interferences. On the other hand, pure design-time application mapping techniques are not able to support such dynamic run-time scenarios and online variations of workload. Therefore, our idea of the DAARM flow (see in Figure 4.2) was to generate and evaluate multiple of such resource reservation constellations at design time, called *operating points*. This information is then handed over to the run-time system (as shown in the code snippet above) to find any suitable and available resource constellation (claim) according to one of these operating points.

However, already in [Wei+14] we have shown that the assembling of such claims to be feasible has to fulfil multiple *binding and routing constraints* for guaranteeing that the application's execution properties, which were analysed at design time, will actually be satisfied at run time despite resource sharing. As a consequence, a constraint satisfaction problem (CSP) has to be solved during run-time application mapping for being able to give these guarantees. Particularly, when the goal is to find not only a feasible allocation, but to find a mapping of all pending applications such that the energy consumption is minimised,

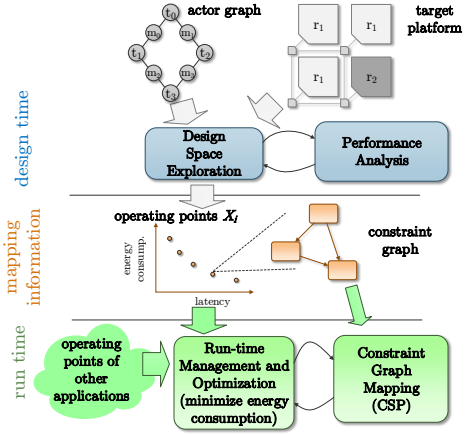
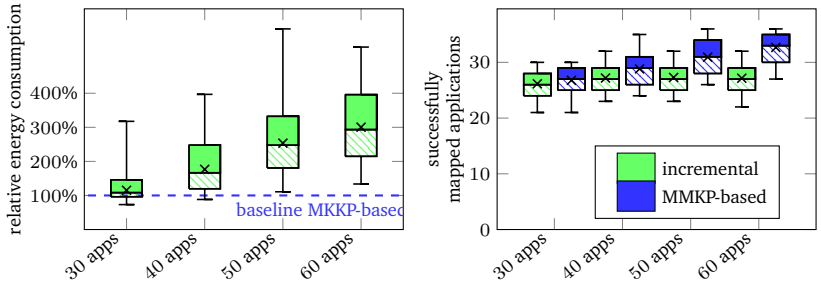


Figure 4.2: Design flow for hybrid application mapping (cf.[WWT15]). DSE generates a set of design points, each providing a constraint graph to the run-time system that describe constellations of resources that need to be assigned exclusively so to provide guaranteed execution qualities.

the problem becomes *hard*. Then, it may be necessary to not only solve a single CSP per application, but potentially multiple CSPs to test multiple mappings of all of these applications and finding the one that optimises the energy consumption. This is shown in the lower part of Figure 4.2: The set of operating points of each active application is handed to the run-time optimiser. This searches for a combination of operating points that (a) minimises the overall energy consumption and (b) can be feasibly mapped by solving the CSP.

In [WWT15], we have analysed this problem in depth. Particularly, we have provided a novel heuristic that improves the optimisation technique applied in the initial DAARM flow significantly. Some results are shown in Fig. 4.3. Here, we have tested to place different application mixes of varying sizes (30, 40, 50 and 60 applications), all assembled from task graph specifications of the E3S benchmark, to an invasive 10×10 NoC architecture. This test case constitutes the corner case of managing a heavily over-utilised system. We randomly generated and tested 500 different test cases per application mix size. Each mix was mapped by applying the incremental mapping strategy from DAARM and the mapping strategy proposed in [WWT15] (called MMKP-based mapping). For each test case, we measured (a) the amount of applications from the mix which were successfully mapped and (b) the overall energy



(a) Relative overall energy consumption E_{rel}. **(b)** Number of successfully mapped applications.

Figure 4.3: Box plots summarising the results of mapping application mixes of different sizes onto an invasive 10×10 NoC architecture.

consumption. Let $E_{inc}(j)$ be the energy consumption obtained for incremental mapping and $E_{MMKP}(j)$ the energy consumption obtained for the proposed approach in the j-th test case. Then the *relative energy consumption* obtained for incremental mapping put into relation to the result of the proposed mapping algorithm is

$$E_{rel}(j) = \frac{E_{inc}(j)}{E_{MMKP}(j)} \cdot 100\%. \tag{4.1}$$

Fig. 4.3 summarising 500 test cases per application mix size. The box plots indicate minimum, maximum, 10th, 50th, and 90th percentile of the results, and the mark (x) represents the average result. It shows that the novel technique significantly outperforms the former approach in both objectives, i. e. reduced energy consumption, at the same time increased number of successfully mapped applications.

Demonstrator Activities and Integration Work

Project A1 kept supporting the integration of the invasive platform. In collaboration with members from Projects C1, C2, Z2, and others we contributed to a problem report that highlighted important unresolved issues mostly within the agent system. While some smaller issues (support the TileSharing constraint) were promptly fixed, others issues remain and stall our work. For example, support for the Malleable constraint in the agent system is yet to be completed. This prohibits authoritative evaluation and publication in top venues.

We also integrated the InvadeSIM simulator, as continuously developed and extended in Project C2, and the actual compiler and operating system into the same regression tester. So, any divergence between the two systems becomes now clearly visible and automatically updated with every revision.

Scientific Activities

Project A1 also contributed a lot in spreading the ideas of invasive computing in invited talks and keynotes [Tei15a; Tei15b; Tei15c; Wil15a; Wil15b] as well as the aforementioned Sarntal academy.

While practical work on the memory model for invasive X10 has not started yet, a literature survey turned up related work like the JDMM[3].

Next Steps in Language Design

We also want to support actor-based applications due to their more predictable performance. This requires an interface to turn operating points as provided by the characterisation into actual claims via an alternative invasion mechanism. We need to find a representation which is efficient for the agent system in its optimisation task and also convenient for the programmer and the source-to-source translation tool, since timing-predictable applications are usually developed with additional tool support.

Publications

[Böw15] N. C. Böwing. “Invasives Verteiltes Job Queue Framework”. Master thesis. IPD Snelting, Dec. 2015.

[BMZ15] S. Buchwald, M. Mohr, and A. Zwinkau. “Malleable Invasive Applications”. In: *Proceedings of the 8th Working Conference on Programming Languages (ATPS)*. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, Mar. 18, 2015.

[RWHT15] S. Roloff, S. Wildermann, F. Hannig, and J. Teich. “Invasive Computing for Predictable Stream Processing: A Simulation-based Case Study”. In: *Proceedings of the 13th IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia)*. (Amsterdam, The Netherlands). IEEE, Oct. 8–9, 2015, 2 pp. ISBN: 978-1-4673-8164-2. DOI: 10.1109/ESTIMedia.2015.7351761.

[3] F. S. Zakkak and P. Pratikakis. “JDMM: A Java Memory Model for Non-cache-coherent Memory Architectures”, ISMM ’14.

[Tei15a] J. Teich. “Adaptive Isolation for Predictable MPSoC Stream Processing”. Keynote, SCOPES 2015, 18th International Workshop on Software and Compilers for Embedded Systems, Schloss Rheinfels, St. Goar, Germany. June 2, 2015.

[Tei15b] J. Teich. “Adaptive Isolation for Predictable MPSoC Stream Processing”. In: *Proceedings of the 18th International Workshop on Software and Compilers for Embedded Systems (SCOPES 2015)*. June 2015, p. 2. DOI: 10.1145/2764967.2771821.

[Tei15c] J. Teich. “Invasive Computing”. Invited Talk, SE 2015, Software Engineering and Management, Special Session Software Engineering in der DFG, Dresden, Germany. Mar. 19, 2015.

[Wei+14] 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. 15, 2014, pp. 1–10. DOI: 10.1145/2656075.2656083.

[Wil15a] S. Wildermann. “Time-predictable multi-core programming using Invasive Computing”. Invited Talk at ESSEI TecDay: Multicore – The challenge in avionics. Oct. 13, 2015.

[Wil15b] S. Wildermann. “Wieviele Prozessoren passen in eine Hosentasche?” Invited Talk at Öffentliche Vortragsreihe Faszination Technik. Apr. 2015.

[WWT15] S. Wildermann, A. Weichslgartner, and J. Teich. “Design Methodology and Run-time Management for Predictable Many-Core Systems”. In: *Proceedings of the 6th IEEE Workshop on Self-Organizing Real-Time Systems (SORT)*. Apr. 13, 2015, pp. 1–8.

[Wit+15] M. Witterauf, A. Tanase, J. Teich, V. Lari, A. Zwinkau, and G. Snelting. “Adaptive Fault Tolerance through Invasive Computing”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231155.

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

Michael Glaß, Michael Bader

Tobias Schwarzer, Alexander Pöppl

Project A4 investigates existing, develops novel, and analyses invasive algorithmic patterns w. r. t. diverse 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.

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 define their quality requirements from which a precise characterisation of achievable qualities must be derived, 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 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 thereby contributes to the entire programming path 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 4.4, Project A4 focuses on the interplay of algorithm development and invasive scheduling and mapping decisions that shall be guided by application characteristics that are automatically determined at design time. This characterisation delivers which quality numbers may be achieved by a certain claim composition and mapping (specified by *claim constraints*), and is indispensable for any program execution at predictable quality.

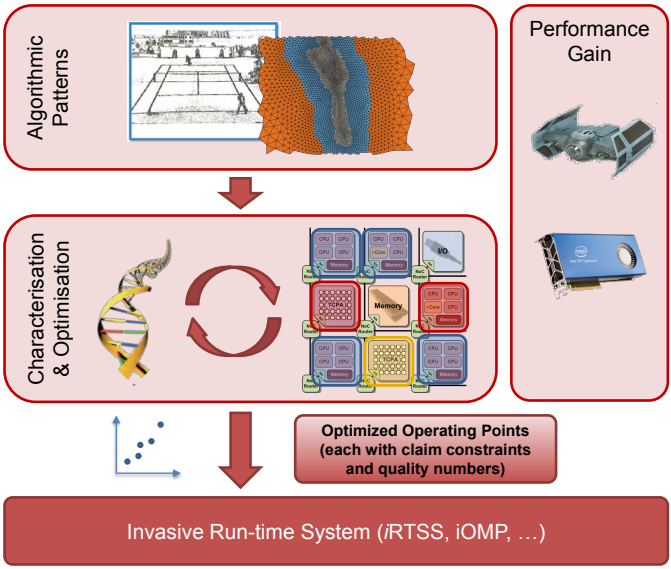


Figure 4.4: Project A4 in a nutshell: The co-design of invasive algorithmic patterns and a design-time characterisation as novel contribution to invasive computing shall result in performance gains and enable predictability by providing statically analysed operating points (tuples of claim constraints and quality numbers) to the invasive run-time system for dynamic resource allocation.

Stencil Computations

As a first step, we ported the SWE (Shallow Water Equations) software package, based on[4], to X10. This implementation was tuned to exhibit performance characteristics that are comparable to native codes on the same hardware, a prerequisite to obtaining usable results in subsequent experiments. Finally, we integrated a prototype of the X10 actor library (see Project A1 and [RHT14]) in order to enable an exploration of the design space that consists of the possible mappings of the simulation space onto the target hardware.

The main challenge when porting the SWE software package was to retain the original performance characteristics while still embracing the APGAS[5] nature of the X10 programming language. In contrast to C and C++, where the choice of where an object should be allocated is largely left to the programmer and temporarily allocated objects and arrays on the stack are possible, most objects in X10, even temporary array variables, are allocated on the heap[6]. While it is possible to force the compiler to generate code for a stack allocation of objects, any further allocations that may happen in the library code are out of the programmer's control. One important consequence of this is the absence of stack-allocated arrays of floating-point values. Another problem is auto-vectorisation: For C- or Fortran-based code, compilers are able to generate vectorised versions as long as the code adheres to a certain structure. This is an issue in X10, as the standard X10 compiler—our target compiler in the HPC arena—generates C++ code that does not exhibit the structure required for the C compiler to generate vectorised versions of the serial functions.

The first problem is solvable by pre-allocating all necessary data structures prior to executing the main loop of the system. In the solver—the part of the code that is called most exhaustively—we replaced the statically sized arrays with separate variables for each floating-point value.

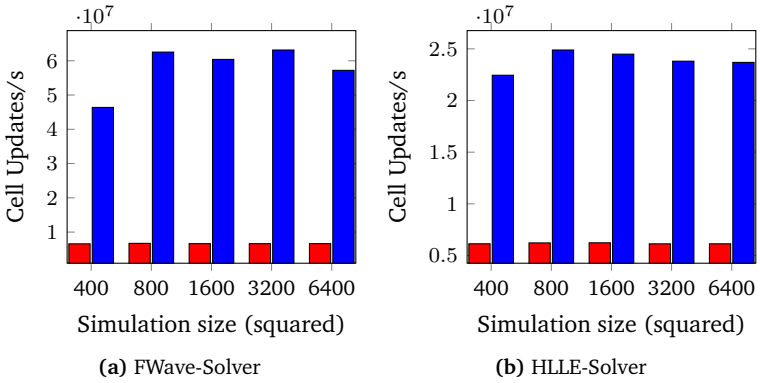
The second problem, vectorisation, has been addressed by introducing hand-written[7] native C++-classes into the X10 code. We investigated two possible options: The first one uses an Augmented Riemann solver

[4] M. Bader and A. Breuer. "Teaching Parallel Programming Models on a Shallow-Water Code". In: *ISPDC 2012 - 11th International Symposium on Parallel and Distributed Computing*. IEEE Computer Society, Nov. 2012, pp. 301–308.

[5] Asynchronous Partitioned Address Space

[6] A notable exception being `structs`, which enable the creation of immutable value types that are allocated on the stack and passed by value.

[7] Usually, C++ code is generated from X10 source code as an intermediate step.



(a) FWave-Solver **(b)** HLLE-Solver

Figure 4.5: Comparison of performance of vectorised vs. unvectorised versions of the available solvers. The red bars show the scalar version of the solvers, the blue bars show the vectorised version.

based on vector intrinsics [BBHR14]. Integrating this has proven fairly simple, one just has to wrap the call to the vectorised solver, as there are no facilities for calling by reference on primitive values in arrays in X10. The second option is an auto-vectorisable version of the net update calculation loop. Again, we used the existing flux solvers developed at the Chair of Scientific Computing at TUM. Both available solvers, the FWave and the HLLE-Solver can auto-vectorise, and, using these, we were able to achieve significant speedups (see Figure 4.5) over the scalar X10 version. If the application is to be compiled with the `x10i` compiler or executed on the proFPGA prototyping platform (Project Z2), one may disable the compilation of the segments calling C++-code.

In [RHT14], Roloff et al. describe a first prototype of an actor library for PGAS-based languages. We integrated this library into our software, obtaining an actor-based version of SWE-X10. The structure of the actor graph is derived from the layout and communication pattern of the original software. There, the overall simulation is split into multiple patches that execute in their own thread and communicate with their respective neighbours on their four borders (top, bottom, left, and right). This may be translated intuitively to one actor per patch and to two channels between every two neighbouring actors. An example for such an actor graph with nine actors is depicted in Figure 4.6.

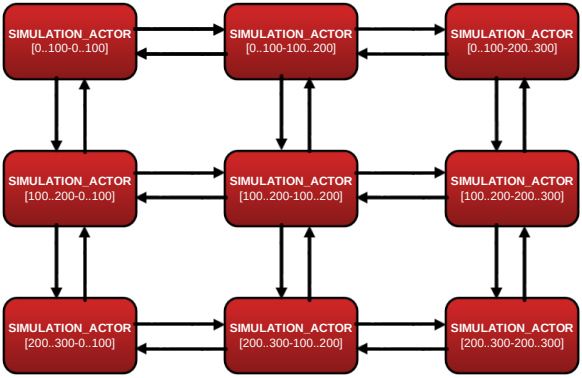


Figure 4.6: Actor graph for a SWE-X10 simulation run with a 300×300 grid, subdivided into nine 100×100 sized patches, each being controlled by an actor. They are connected to each of their neighbours by two channels, an outgoing channel to deliver updates to the neighbour and an incoming channel to receive the necessary updates for the next time step.

Symmetry-Eliminating Design Space Exploration

Existing DSE (Design Space Exploration) approaches used in hybrid application mapping techniques do not consider any *architectural symmetries*. The significance of this is outlined via an example in Fig. 4.7(b): Four different mapping options on a concrete tiled invasive MPSoC architecture are shown with all of them varying in the assignment of three tasks to two different tiles of different resource types (indicated by the colour). All four mappings are obviously equivalent: They all feature two allocated PEs of resource types r_type_0 and r_type_1 and the same hop distance, direction, and allocation of bandwidth for communication between the tasks $t_0 \rightarrow t_2$ and $t_1 \rightarrow t_2$ as indicated by the abstract representation in Figure 4.7(c). Extrapolating this scenario to larger multicore/manycore systems, it becomes clear that using state-of-the-art DSE techniques developed for mapping task graph based application models to concrete architectures results in massively redundant search spaces. This not only results in many unnecessary and costly evaluations during DSE. It is also well-known that redundant search spaces may deteriorate the optimisation process and distract it from finding high-quality solutions.

We started to work on WP A4.3 and WP A4.4 by defining a novel *symmetry-free search space* that eliminates the outlined architectural

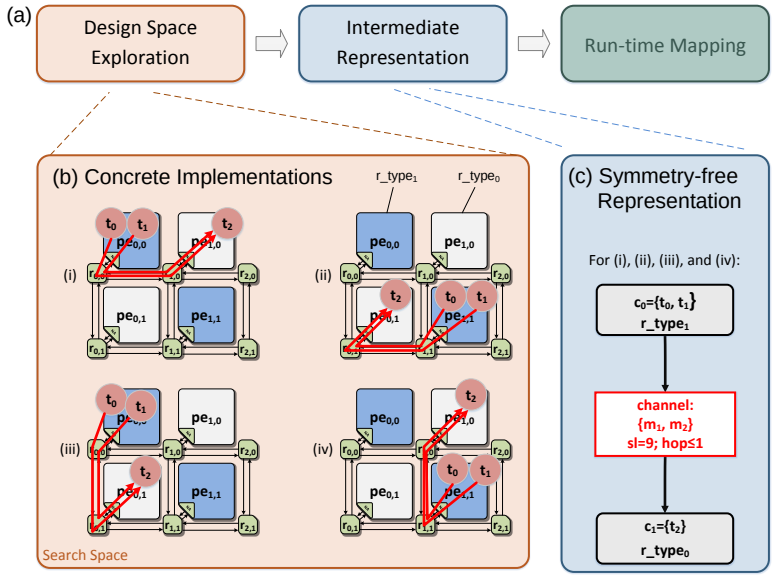


Figure 4.7: (a) General hybrid application mapping flow shown on top. (b) A design-time DSE that explores mapping options for task graphs onto a concrete architecture. (c) Each is transformed to an intermediate representation called *constraint graph* [Wei+14]. The latter basically encodes rules for the run-time mapper how to feasibly embed the application tasks and inter-task communications. Architectural symmetries are shown by four different concrete mapping options for three tasks t_0, t_1, and t_2 on a heterogeneous invasive NoC architecture containing four single PE tiles $pe_{0,0}$, $pe_{1,0}$, $pe_{0,1}$, and $pe_{1,1}$ with $pe_{0,0}$ and $pe_{1,1}$ being of a different type than the others. As depicted, the resulting constraint graph is identical in all four cases.

symmetries completely. This is achieved by reformulating the design-time mapping problem: We do not encode the mapping of application tasks to concrete resources, but we employ an abstract representation as depicted in Figure 4.7(c) which was initially developed to serve as a data structure to encode the design-time mapping for the run-time system. In this representation, each solution covers a complete equivalence class of concrete mappings with the same quality numbers instead of concrete mappings. In the novel symmetry-free search space, we directly explore this structure, instead of deriving it from a concrete mapping as proposed in [Wei+14]. Thus, some structures may not have a feasible mapping on the concrete architecture. Our approach therefore integrates formal feasibility checks that ensure that each explored abstract representation has at least one feasible mapping on the con-

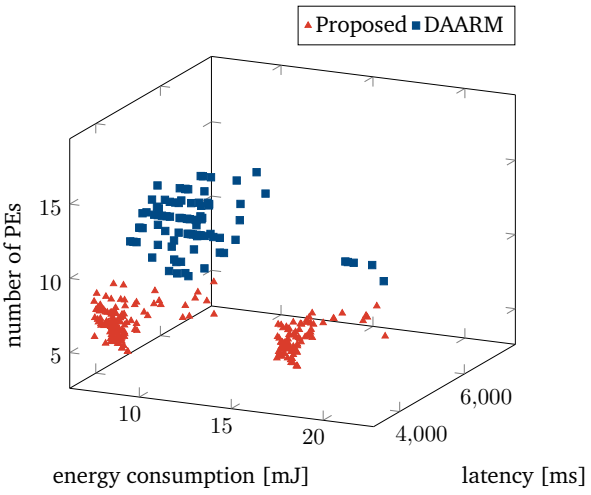


Figure 4.8: The best set of operating points obtained for an automotive application in terms of energy consumption, latency, and required number of PEs evaluated for a heterogeneous 8×8 NoC using a symmetry-free encoding (proposed) in comparison with a previous approach called DAARM [Wei+14].

crete architecture. This way, our approach solves the same exploration problem as state-of-the-art approaches, but with a dramatically reduced search space. First results as depicted in Figure 4.8 show that the novel approach significantly outperforms an existing method [Wei+14] w. r. t. the quality of the delivered implementations.

Outlook

Our first version of the actor library will be improved upon. Together with Project A1, Project C2 and Project A4, we will extend it with features such as type-safe channels and the ability to allow for multiple communication channels between two actors. Moreover, a novel concept for actors to allow for dynamic reconfiguration shall be developed.

For the SWE-X10 package, we aim to implement an adaptive mesh-refinement scheme with a real local time stepping as well as a dynamic rearrangement of the structure and distribution of the simulation as well as its governing actor graph. Once the functionality is available, the optimal distribution of the—now heterogeneous—actor graph needs to be determined. This shall be accomplished by performing our design-time characterisation.

Furthermore, we intent to advance our symmetry-free DSE by employing Satisfiability Modulo Theories (SMT) techniques that allow to efficiently prove whether there exists a feasible mapping (embedding) of a set of applications on the given platform with respect to both, functional and non-functional constraints (requirements).

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, Aug. 2014, pp. 193–201.

[PH16] A. Pöppl and A. Herz. "A cache-aware performance prediction framework for GPGPU computations". In: *Euro-Par 2015: Parallel Processing Workshops*. Ed. by S. Hunold, J. L. Träff, and F. Versaci. Lecture Notes in Computer Science. accepted. Berlin, Heidelberg: Springer-Verlag, 2016.

[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.

[Wei+14] 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. 15, 2014, pp. 1–10. DOI: 10.1145/2656075.2656083.

B1: Adaptive Application-Specific Invasive Microarchitecture

Jörg Henkel, Jürgen Becker, Lars Bauer

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

Project B1 investigates mechanisms that provide run-time adaptivity at the microarchitectural level (μArch) 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). The i-Core is an integral part of the InvasIC hardware. Therefore, we integrated an updated version of our i-Core prototype into the joint demonstration platform during the first months of second funding phase. In the following, we briefly describe our current activities and results for Intra-Tile Cache Re-allocation, Auto-SI as well as Heterogeneous Fabric.

Intra-Tile Cache Re-allocation

Nowadays, cache memory is still statically allocated at design time. This leads to a distribution of memory resources that is optimised for an average use case. We propose a dynamic cache architecture that enables parameterisation and resource allocation of cache memory resources between cores during run time. This hardware representation makes it possible to deeply integrate the adaptive cache into an existing processor microarchitecture. The reallocation is done with little overhead such that each algorithm class can be more efficiently executed on the manycore platform [Tra+16].

For the first test, the software application is run 5 times on previously defined hardware scenarios. Each time the application runs, a different cache configuration is applied by the software. The configuration parameter to be changed is the amount of exclusive cache blocks available as cache memory for the processor. A cache block is the smallest instance of the cache architecture consisting of multiple lines with multiple words each i. e. a set. By changing the amount of cache blocks that the cache can utilise, not only the cache's size will change depending on the size of each cache block, but also the cache associativity will change. For

each scenario, we vary the amount of exclusive cache blocks from 4 to 8 in steps of 1. We measure the cache miss rate during the execution of the application as a performance metric.

Figure 4.9 upper graph shows the behaviour of the cache misses over the total cache size. We can notice a non-linear behaviour and a performance gain that can be achieved when running the benchmark with a different cache configuration. For instance, a cache size of 3.5 kB built using 7 cache blocks of 512 bytes per cache block would lead in this case to the lowest miss rate, hence the best cache performance.

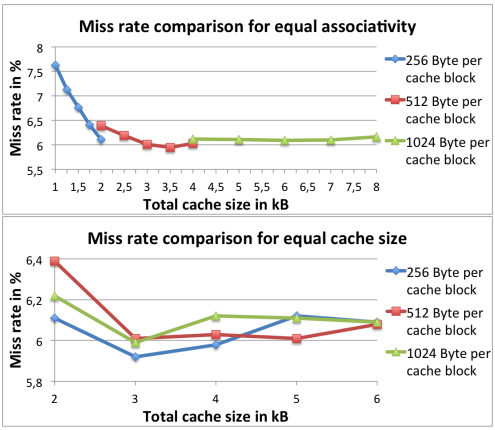


Figure 4.9: Miss rate for the Coremark benchmark with 2000 iterations on three scenarios of the adaptive cache architecture with same associativity (upper graph) and with same cache size (lower graph).

In this test, we always selected the same amount of cache blocks for each hardware scenario, which lead to different total cache sizes for each case. This is the case because each scenario has cache blocks of different sizes.

For the second test, the same three hardware scenarios previously presented are being used. However, this time the total cache size will be varied from 2 to 6 kB in steps of 1 kB by adjusting the amount of cache blocks respectively for each hardware scenario. This has the advantage that the effect of the cache block size and amount of cache blocks can be directly seen for all scenarios for each cache size. The hardware designs have been expanded to contain more available cache blocks as needed, however the amount of cache blocks to be used in each case is still configured at run time by the software and varies depending on the hardware scenario. For example, to build a 5 kB cache in all three scenarios we would need 20 cache blocks of 256 bytes, 10 cache blocks of 512 bytes but only 5 cache blocks of 1024 bytes.

Figure 4.9 lower graph shows the different miss rates measured for each cache configuration on each hardware scenario. We see the miss rates staying in the range of 5 % to 8 % with variations in the miss rate in a range of about 2 %. The data shows that for this particular application

a minimum miss rate can be achieved at a total cache size of 3.5 kB with 7 cache blocks of 512 bytes per cache block. The same analysis for other applications would provide different results, thus showing the importance of an adaptive cache structure that can match the individual application needs and that can be controlled by the software.

Our cache architecture exploits fine grain run-time adaptation, which enables performance gains while keeping the hardware implementation overhead to a minimum. As the evaluation has shown, the architecture provides multiple advantages and performance gains with an expandable potential for multicore architectures.

Auto-SI

Modern computer systems demand for more performance. One approach is to speed up loops. However, if no Special Instructions for the *i*-Core are provided during compile-time (offline SI), then no acceleration is possible during run-time. Thus, we propose to automatically detect loop kernels during execution and then translate & migrate them onto the *i*-Core fabric. Several steps are necessary to accelerate a loop transparently, dynamically, and automatically: i) monitor instructions, ii) prepare configuration, iii) configure hardware accelerator, iv) use accelerator on *i*-Core fabric.

The Auto-SI is detected and configured during the first run of the inner loop in software. Auto-SI is then available from the 2nd run of the outer loop. This uses the same approach as with offline SI. We then dynamically insert the instruction to start Auto-SI.

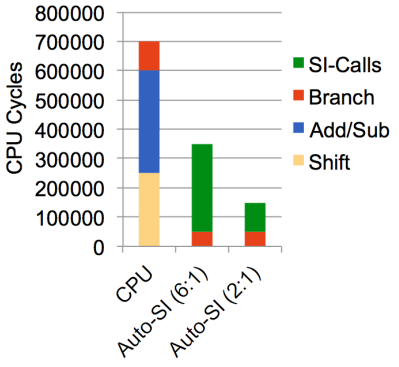


Figure 4.10: Auto-SI speedup with realistic 6:1 and Leon3 2:1 CPU/FPGA clock frequency ratio compared to the CPU baseline.

We integrated Auto-SI into the experimental *i*-Core setup on the FPGA prototyping board. Figure 4.10 shows the evaluation for an inner loop of the basic maths square root algorithm. 50,000 iterations are run on the CPU in 700,000 cycles (left). First, a realistic scenario, in which the CPU frequency is $6\times$ of the FPGA frequency is assumed (centre). Second, the invasive LEON3 scenario is used, in which the CPU runs at twice the frequency of the FPGA fabric (right). In both scenarios, we achieve a significant speedup of $2\times$ and $4.67\times$, respectively.

Heterogeneous Fabric

During the first year of our second funding phase, we introduced COREFAB (COncurrent Reconfigurable FABric utilisation, cf. last year's report) for efficient utilisation of the *i*-Core by providing an intra-tile multicore invasion of the *i*-Core reconfigurable fabric using Remote Special Instructions. This year, we started investigating a heterogeneous fabric structure for the *i*-Core for better area and power efficiency. In the following, we detail some case study results carried out using the accelerators we designed for Project D1 and Project D3 as well as our internal benchmark applications to motivate the need for the heterogeneous structure in the *i*-Core fabric.

At the end of the first funding phase, we designed specific SIs for SIFT feature matching and floating point matrix multiplication (fmm) to demonstrate the performance benefits for floating point algorithms on the *i*-Core fabric. This year, we presented in [Bau+15; Pau+15] the design and architecture of a *flexible multi-mode accelerator for floating point operations* which supports the SIFT and fmm SIs and integrated it into the InvasIC hardware. Figure 4.11 shows its block level architecture. The interface to the accelerator is compliant to the fabric interconnect of the *i*-Core and comprises two 32 bit inputs, one 32 bit output, 6 bit control inputs and the clock signal. The FMAV accelerator performs single precision floating point operations and supports 12 operation modes. This allows flexible combinations of multiplication, addition and subtraction with optional storing of the result in a register.

The resource utilisation of the FMAV accelerator is 1,051 LUTs (132 CLBs), which effectively requires reconfigurable containers of 140 CLBs as they need to be resized in multiples of 20. The current design of the *i*-Core has fixed size homogeneous reconfigurable containers of size 20×5 CLBs. With 20×7 CLBs, the FMAV accelerator requires a significantly bigger reconfigurable container compared to the existing accelerators for integer operations. Resizing all containers to 20×7 CLBs would lead to high internal fragmentation and therefore inefficient fabric use. A possible solution to overcome this is to reduce the CLB

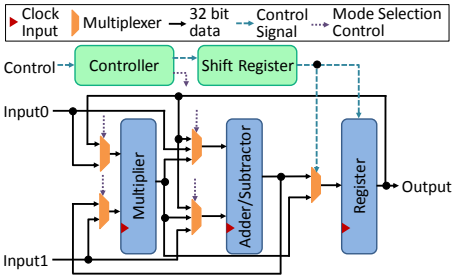


Figure 4.11: FMAV Architecture

demands of FMAV by utilising DSP blocks for a more area-efficient implementation. An evaluation of the FMAV implementation on the *i*-Core using DSP blocks shows a significantly reduced use of LUTs of 562 (71 CLBs) while using 5 DSP blocks compared to 1,051 LUTs when using LUTs only. In a homogeneous fabric, this requires every container to provide DSP resources even though only a single accelerator makes use of them, leading to inefficient resource usage again. Therefore, we want to investigate a reconfigurable fabric that is heterogeneous in container size and in availability of specialised resources (e. g. DSP blocks).

To investigate container sizes, a case study with homogeneous container sizes was carried out using our H.264 video encoder application which can utilise up to 14 different accelerators on the *i*-Core reconfigurable fabric. The reconfigurable fabric area utilised by some of the accelerators was 25% less than available. This leads to wastage of fabric area and an unnecessarily high leakage power. Therefore, we introduce a reconfigurable fabric with heterogeneous size containers which allows us to fit an additional container into the same area. A preliminary evaluation of this design was carried out using the *i*-Core simulator. The results (Figure 4.12) showed a 1.2× performance improvement compared to the homogeneous design of reconfigurable containers.

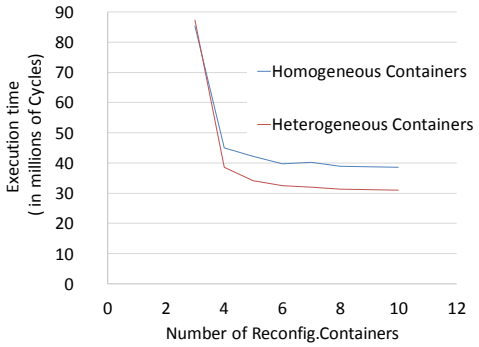


Figure 4.12: *i*-Core simulation results with homogeneous and heterogeneous size reconfigurable containers

In the future, we plan to redesign the current homogeneous container structure in the *i*-Core to a heterogeneous one. The new heterogeneous container structure will have different size containers, use dedicated DSP and memory blocks within some containers and provide containers with different number of I/Os. This allows more efficient area utilisation of the fabric with improvements in performance and leakage power. The key algorithms of the run-time system need to be redesigned to manage and explore the heterogeneous fabric.

Next Steps in Plan

To make the performance benefits of the *i*-Core amenable to predictable execution on the InvasIC hardware (one of the main topics of Phase II), models are required to obtain timing guarantees for the run-time–reconfigurable instruction set as well as execution of SIs. We are working on models which enable a precise timing analysis of tasks utilising run-time reconfiguration. Eventually, the information obtained by analysing our models will be integrated in state-of-the-art timing analysis tools and enable run-time guarantees for tasks running on the *i*-Core.

B1

Additionally, we are continuously extending and improving our *i*-Core prototype and will make new features available on the joint InvasIC hardware prototype. At the beginning of the current funding phase, we integrated an updated version of the *i*-Core. The next steps entail a migration to the new InvasIC demonstrator platform and a new version of the Gaisler IP Library[8] which our *i*-Core prototype is based on.

Publications

[Bau+15] L. Bauer, A. Grudnitsky, M. Damschen, S. R. Kerekare, and J. Henkel. “Floating Point Acceleration for Stream Processing Applications in Dynamically Reconfigurable Processors”. In: *IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia’15)*. Invited Paper for the Special Session “Dynamics and Predictability in Stream Processing – A Contradiction?” Amsterdam, The Netherlands, Oct. 2015, 2 pp. DOI: 10.1109/ESTIMedia.2015.7351762.

[Fig+15] P. Figuli, C. Tradowsky, J. Martinez, H. Sidiropoulos, K. Siozios, H. Stenschke, D. Soudris, and J. Becker. “A Novel Concept for Adaptive Signal Processing on Reconfigurable Hardware”. In: *Applied Reconfigurable Computing*. Vol. 9040. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 311–320.

[Gru15] A. Grudnitsky. “A Reconfigurable Processor for Heterogeneous Multi-Core Architectures”. Dissertation. Chair for Embedded Systems (CES), Department of Computer Science, Karlsruhe Institute of Technology (KIT), Germany, Dec. 21, 2015.

[8] http://www.gaisler.com

[Pau+15] J. Paul, W. Stechele, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, W. Schröder-Preikschat, M. Kröhnert, T. Asfour, Éricles R. Sousa, V. Lari, F. Hannig, J. Teich, A. Grudnitsky, L. Bauer, and J. Henkel. "Resource Awareness on Heterogeneous MPSoCs for Image Processing". In: *Journal of Systems Architecture* 61.10 (Nov. 6, 2015), pp. 668–680. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.09.002.

[THMB16] 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)*. 2016. Forthcoming.

[Tra+16] C. Tradowsky, E. Cordero, C. Orsinger, M. Vesper, and J. Becker. "A Dynamic Cache Architecture for Efficient Memory Resource Allocation in Many-Core Systems". In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Springer International Publishing, Mar. 2016. Forthcoming.

B2: Invasive Tightly-coupled Processor Arrays

Jürgen Teich

Marcel Brand, 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 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. 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. These research achievements have resulted in a successful PhD defence of Vahid Lari in November 2015 [Lar15].

Predictable multicore computing is also the major focus of research for Project B2 in the 2nd funding phase. Whereas it has been shown 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 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* drive our current research.

The high integration density of future multicore systems will inevitably lead also 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 dual (DMR) and triple modular redundancy (TMR) to the level of loop programs and their parallel processing on multicores has not been investigated yet. Here, we are investigating approaches in which based on application *requirements* on reliable execution, an invasive loop program 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 [Wit+15]. Without creating any inefficiency for error detection circuits in our hardware, the regular structure of TCPAs does ideally offer an application to claim (a) a non-redundant, (b) a dual-replicated (see Figure 4.13(a)), or even (c) a triple-replicated array instance for computing the parallel program in lock-step mode (see Figure 4.13(b)–(d)). In [Lar+15a], we demonstrated this idea and how to also adaptively provide (a) signal replication of input and output signal streams, and (b) voting (hard-wired vs. dedicated functional units). Here, Project C3 provides the necessary compiler support for loop program transformation for fault-tolerant loop computation (see their report). Figure 4.13 illustrates the essential ideas of our proposed 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 [Tan+15] (Project C3). Based on a compile-time quantitative analysis of the execution time and reliability gains in terms of probability of failure in dependence of observed soft error rate during operation, we showed in [Lar+15b] how a suitable redundancy scheme might be selected by the run-time system (Project C1) automatically to enforce a desired safety-level. Moreover, we evaluated and compared the implementations of the voting instructions in both hardware and software for their hardware cost and timing overheads in [Lar+15a; Lar+15b]. An example of such

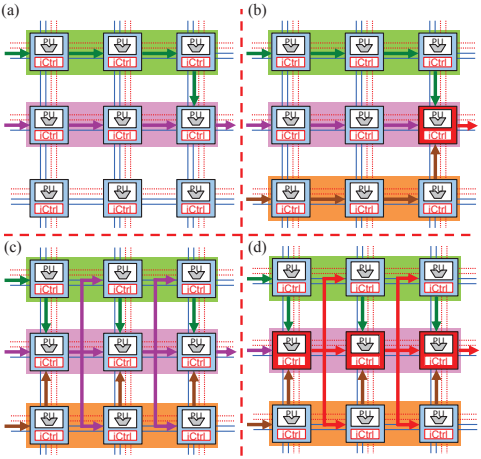
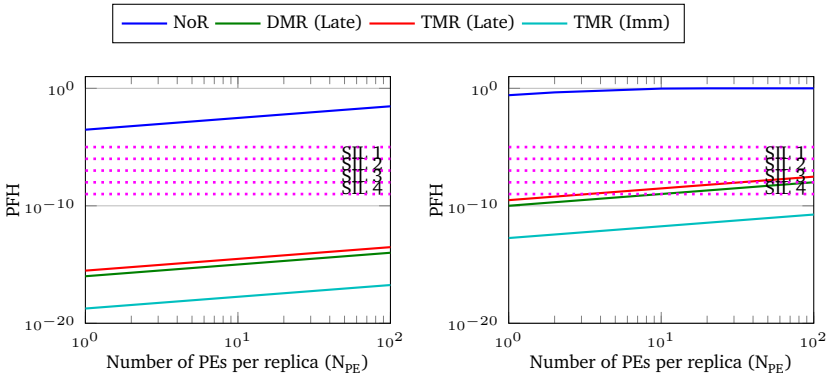


Figure 4.13: Redundancy is allocated on-demand by claiming identical subarrays to realise (a) (DMR), and (b)–(d) TMR. Comparison and voting are performed at the array boundary in (a) and (b), respectively. In (c) and (d), voting is performed in software, respectively in hardware, inside the processor array for earlier detection/correction of potential errors.

quantitative analysis is plotted in Figure 4.14 in terms of the probability of failure per hour (PFH) for a matrix-matrix multiplication kernel of size $[200 \times 400][400 \times 300]$. The figures show PFH values depending on the size of PEs in each replica and two different soft error rates at the level of each PE, i. e. 2,000 failures in time (FIT)[9] or 2,000,000 FIT. Also annotated in the figures are the margins of the four safety integrity levels, namely, *SIL 1, SIL 2, SIL 3, SIL 4* defined by the IEC 61508 standard. The PFH values for the non-redundant variant (see Figure 4.14) would not meet any SIL requirement and redundancy schemes have to be applied if needed by the application. In Figure 4.14, if less than ten PEs are claimed, the DMR/TMR (Late) version may satisfy requirements beyond the defined SILs. But for mappings on $N_{PE} > 10$ PEs, only the TMR replication with immediate voting remains to satisfy the requirements of all four SILs. For other levels, the replication scheme to satisfy a certain SIL needs to be adjusted properly in dependence on N_{PE}.

As the next step, we envision an extension of the provided analysis 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

[9]Number of failures in one billion (10^9) device-hours of operation.



(a) SER 2,000 FIT **(b)** SER 2,000,000 FIT

Figure 4.14: Probability of failure per hour (PFH) for a matrix-matrix multiplication loop nest application of size $[200 \times 400][400 \times 300]$ based on the number of PEs (N_{PE}) for different soft error rates, i. e. 2,000 FIT and 2,000,000 FIT for the cases of no replication (blue), DMR with late comparison (DMR (Late), green), TMR with late voting (TMR (Late), red), and finally TMR with immediate voting (TMR (Imm), cyan).

to investigate proper techniques for fault recovery for parallel loop executions on TCPAs.

Ultra-Low Power/Dark Silicon

TCPAs have proven to not only offer high performance but high energy efficiency in running compute-intensive loop kernels as well compared to general purpose processors. In order to exploit this capability within heterogeneous multi-processor system-on-chip (MPSoC) architectures, we have investigated techniques for:

Power density-aware resource management for heterogeneous tiled multicores: In cooperation with Project B3, we are working towards maximising the overall system performance under critical temperature constraints for heterogeneous tiled multicores, where all cores or accelerators (i. e. TCPAs) inside a tile share the same voltage and frequency levels. We have profiled the power consumption of a TCPA considering several computationally intensive applications from the domain of digital signal processing. The TCPA power profiles were derived based on post-synthesis results from the Cadence Encounter RTL Compiler using a NanGate 45 nm low power process technology node. These profiles may be provided to the dark silicon management

(Project B3) for adapting the power density constraint according to available thermal headroom and application requirements.

Run-time adaptation of application execution under thermal and power constraints: In [Sou+15], we presented a solution for run-time adaptation of application execution under thermal and power constraints in TCPAs. There, we consider a scenario where applications running on mobile platforms require a certain performance level or quality (e. g. high-resolution image processing) that need to be satisfied while adhering to a certain power budget and temperature threshold. As a solution to these conflicting goals, we considered the principles of invasive computing to exploit run-time adaptations of resource usage without violating any thermal and/or power constraint for a TCPA target. In cooperation with Project B4, we developed a mathematical model for estimating the power consumption based on the post-synthesis implementation of a TCPA in different CMOS technologies while the temperature variation was emulated. We also presented our hardware/-software solutions to load new configurations into the accelerator array. The results show that the average power consumption of 25 processing elements is much less than the idle power consumption of a single embedded processor running at the same frequency and in the same technology. Moreover, the reconfiguration overhead is so low that no frame needs to be dropped during run-time adaptation.

Still regarding the run-time adaptation, our most recent research investigates how to accelerate computationally intensive applications on TCPAs by means of increasing data reuse. We presented a reconfigurable buffer architecture in [SHT15] that can be configured at run time to select between different schemes for memory access, i. e. addressable RAMs or pixel buffers. We have successfully showcased the benefits of our approach by prototyping a heterogeneous MPSoC architecture containing a RISC processor and a TCPA. The architecture is prototyped in FPGA technology (Project Z2). For a class of image processing algorithms, we have demonstrated that our proposed buffer structures for system integration allow to increase the memory bandwidth utilisation. The buffers also allow for a considerable performance improvement in comparison to state-of-the-art solutions for image processing.

Outlook

Our current work and next activities include the development of required hardware and software interfaces to signal non-maskable errors, especially in case of DMR execution, to the application level by developing an exception mechanism. We are developing 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. Our next research also includes the investigation and development of error recovery techniques, e. g. synchronous halting an application execution on a TCPA, or re-execution of loop computations or full programs on a TCPA without considerable timing overheads. Moreover, our architecture research will be guided by the analysis and realisation of orthogonal instruction processing functional units within the VLIW PEs of a TCPA.

Finally, the next activities also focus on investigating novel techniques to provide bandwidth guarantees as well as predictable QoS for communication between accelerators and general purpose processors. To achieve these goals, a run-time management system for dynamic memory allocation needs to be designed.

Publications

[Lar15] V. Lari. “Invasive Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, Nov. 18, 2015.

[Lar+15a] V. Lari, A. Tanase, J. Teich, M. Witterauf, F. Khosravi, F. Hannig, and B. Meyer. “A Co-Design Approach for Fault-Tolerant Loop Execution on Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231157.

[Lar+15b] V. Lari, J. Teich, A. Tanase, M. Witterauf, F. Khosravi, and B. Meyer. “Techniques for On-Demand Structural Redundancy for Massively Parallel Processor Arrays”. In: *Journal of Systems Architecture (JSA)* 61.10 (Nov. 2015), pp. 615–627. ISSN: 1383-7621. DOI: http://dx.doi.org/10.1016/j.sysarc.2015.10.004.

[Pau+15] J. Paul, W. Stechele, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, W. Schröder-Preikschat, M. Kröhnert, T. Asfour, Éricles R. Sousa, V. Lari, F. Hannig, J. Teich, A. Grudnitsky, L. Bauer, and J. Henkel. “Resource Awareness on Heterogeneous MPSoCs for Image Processing”. In: *Journal of Systems Architecture* 61.10 (Nov. 6, 2015), pp. 668–680. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.09.002.

[SHT15] É. Sousa, F. Hannig, and J. Teich. “Reconfigurable Buffer Structures for Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the International Embedded Systems Symposium (IESS)*. Foz do Iguaçu, Brazil: LNCS, Nov. 2015, 10 pp.

[Sou+15] Éricles R. Sousa, F. Hannig, J. Teich, Q. Chen, and U. Schlichtmann. “Runtime Adaptation of Application Execution under Thermal and Power Constraints in Massively Parallel Processor Arrays”. In: *Proceedings of the 18th International Workshop on Software and Compilers for Embedded Systems (SCOPES)*. (St. Goar, Germany). ACM, June 1–3, 2015, pp. 121–124. ISBN: 978-1-4503-3593-5. DOI: 10.1145/2764967.2771933.

[Tan+15] A. Tanase, M. Witterauf, J. Teich, F. Hannig, and V. Lari. “On-Demand Fault-Tolerant Loop Processing on Massively Parallel Processor Arrays”. In: *Proceedings of the 26th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. (Toronto, Canada). IEEE, July 27–29, 2015, pp. 194–201. ISBN: 978-1-4799-1925-3. DOI: 10.1109/ASAP.2015.7245734.

[Tei15a] J. Teich. “Adaptive Isolation for Predictable MPSoC Stream Processing”. Keynote, SCOPES 2015, 18th International Workshop on Software and Compilers for Embedded Systems, Schloss Rheinfels, St. Goar, Germany. June 2, 2015.

[Tei15c] J. Teich. “Invasive Computing”. Invited Talk, SE 2015, Software Engineering and Management, Special Session Software Engineering in der DFG, Dresden, Germany. Mar. 19, 2015.

[Wit+15] M. Witterauf, A. Tanase, J. Teich, V. Lari, A. Zwinkau, and G. Snelting. “Adaptive Fault Tolerance through Invasive Computing”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231155.

B3: Power-Efficient Invasive Loosely-Coupled MPSoCs

Andreas Herkersdorf, Jörg Henkel

Santiago Pagani, Anuj Pathania, Heba Khdr, 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: Modelling 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 for Performance Optimisation

In [HKPS15], we present new trends in dark silicon reflecting, among others, the deployment of FinFETs in recent technology nodes and the impact of voltage/frequency scaling, which lead to *new less-conservative dark silicon predictions*. The focus is on dark silicon from a thermal perspective: we show that it is not simply the chip's total power budget, e. g. the Thermal Design Power (TDP), that leads to the dark silicon problem, but instead it is the power density and related thermal effects.

In [KPSH15], we proposed a dark silicon-aware resource management technique that distributes the chip's resources (cores and power) among different applications under a thermal constraint. In particular, in order

to maximise the overall system performance, we determine the number of active cores and the voltage and frequency level of each application, while considering the Thread Level Parallelism (TLP) and the Instruction Level Parallelism (ILP) of the applications.

In [SGGH15], we proposed a variability-aware dark silicon management technique that exploits the so-called *dark silicon patterning*, along with the core-to-core leakage power variations, to optimise the system performance under peak temperature constraints. The dark silicon patterns denote various spatio-temporal decisions for the cores' power state. This directly influences the resulting chip thermal profile due to improved heat dissipation, and thereby enables powering-on more cores to facilitate high-TLP applications, and/or boosting certain cores to facilitate high-ILP applications. An improved thermal profile also reduces the number of Dynamic Thermal Management (DTM) events.

In [KSH15], we propose a hierarchical scheme for distributing the resources and TDP budget among concurrently executing applications with multi-threaded workloads under throughput constraints. The application-level TDP budget is partitioned among its threads depending upon their workloads, which can then be fine-tuned at run time considering workload variations.

In [Pag+15], we present an efficient and lightweight run-time boosting technique based on transient temperature estimation. This technique guarantees meeting run-time performance requirements surges, by executing the boosted cores at the required frequencies for the entire boosting intervals, while throttling down the non-boosted cores. In order to minimise the performance losses for the non-boosted cores, the throttling down levels are chosen such that the maximum frequencies among all cores reaches the critical temperature precisely when the boosting is expected to expire.

In [PCSH15a], we developed a method for analytically computing transient temperature peaks. Our method works for any compact thermal model consisting in a system of first-order differential equations, like the RC thermal networks used in HotSpot. Instead of using regular numerical methods, we find an analytical solution for the differential equations using matrix exponentials and linear algebra. This results in a mathematical expression which can easily be analysed and differentiated to compute the maximum transient temperatures. Furthermore, we can also efficiently compute any future transient temperatures without accuracy losses or a need to compute all step-wise partial temperatures.

Some of these (and other) research efforts are also summarised in [PSCH15; PCSH15b; SH15].

Dark Silicon Management for Aging Optimisation

In [Gna+15], we developed a technique that harnesses the available dark silicon and manufacturing process induced core-to-core variability in frequency and leakage power to decelerate and/or balance the chip-wide temperature-dependent aging effects. It accounts for the thermal profile and core usage when taking temperature-dependent aging optimisation decisions. Core usage is important for early-stage aging, while temperature effects are relatively more dominating in aggravating the aging rates during the later years. We also consider the leakage power variations to optimise the health map of a chip (i. e. aged state of different cores) when executing many applications concurrently.

Dark Silicon Management for Energy Optimisation

In [PPSH15], we present a power-performance characterisation of multi-threaded mobile games by executing them on a real-world mobile platform with an asymmetric multicore. This analysis is leveraged to propose a QoS-aware Governor, integrated into the platform's operating system, running a lightweight online heuristic that holistically accounts for thread-to-core mapping and Dynamic Power Management (DPM).

In [PCH15], we present a simple and practical solution both for energy minimisation and peak power reduction for performance constrained tasks on tiled multicore systems. Namely, all cores in a tile run at the same voltage, but the frequency of each core is individually chosen such that the tasks in each core can meet their deadlines without running at unnecessarily high frequencies. We provide comprehensive analysis of this technique compared to the optimal solutions, deriving its worst-case behaviour both for energy minimisation and peak power reduction.

In [HKS15], we present a cross-layer approach for analysing and designing energy-efficient advanced multimedia systems with next-generation High-Efficiency Video Coding (HEVC) standard. Our approach leverages both algorithmic and architectural layers of system design abstractions in order to achieve a high power/energy efficiency. We present an analysis and design of an HEVC-based multimedia system while leveraging algorithmic-architectural collaborative optimisations and video content properties to achieve high energy efficiency.

Online Data Analysis to Support Dark Silicon Management

With invasive diagnosis on chip (iDoC), we extend the InvasIC run-time system with hardware components to observe the software execution

on-chip. The goal is to identify potential run-time inefficiencies that can be exploited for power saving while still meeting performance requirements. Unprocessed data on the hardware level, such as CPU register contents, NoC flits, etc., however, is rarely useful and requires post-processing to yield useful information for higher abstraction levels. Towards this goal, we create a flexible library of data analysis components, catering different trade-offs of computational density and flexibility. As part of this library, we developed a freely programmable general-purpose data analysis unit, which is able to generate metrics out of the observed system state. This transformation utilises knowledge about the system, which is formalised and programmed onto the data analysis unit. For example, if the observation of the program counter in a processor core is combined with knowledge about the program structure, the metric "frames per second" in a video streaming application can be obtained. In [Wag+15], we describe the concept of knowledge-based diagnosis and explain how knowledge from different fields can be combined.

In current work, we use the iDoC analysis infrastructure to measure application-level quality-of-service metrics, such as frames per second for a video application (such as InvasIC robotic demonstration). This quality metric can then be used to improve power management decisions.

In the coming year, we plan to further extend the iDoC library of analysis components. We also plan to investigate self-learning analysis components with the goal to automatically identify critical states in the software execution.

Organisation of Scientific Events and Public Dissemination

Project B3 organised two special sessions on dark silicon: The first special session on "Dark Silicon: No Way Out?" at the ACM/EDAC/IEEE 52nd Design Automation Conference (DAC), 2015. More information is available at http://www2.dac.com/events/eventdetails.aspx?id=182-55. The second special session on "Dark Silicon - From Computation to Communication" at the International Symposium on Networks-on-Chip September (NOCS), 2015 [Hen+15].

A workshop on "Towards Efficient Computing in the Dark Silicon Era" was organised at the IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2015. More information about the workshop can be found at https://wp.nyu.edu/iccad_darksil_workshop.

Prof. Henkel and Dr. Shafique gave one keynote each about dark silicon [Hen15; Sha15]. Prof. Herkersdorf gave a talk about the challenges of run-time observation in MPSoCs [Her15].

Publications

[Gna+15] D. Gnad, M. Shafique, F. Kriebel, S. Rehman, D. Sun, and J. Henkel. “Hayat: Harnessing Dark Silicon and Variability for Aging Deceleration and Balancing”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). HiPEAC Paper Award. ACM, June 7–11, 2015, 180:1–180:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2744849.

[HKPS15] J. Henkel, H. Khdr, S. Pagani, and M. Shafique. “New Trends in Dark Silicon”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). HiPEAC Paper Award. ACM, June 7–11, 2015, 119:1–119:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2747938.

[Hen+15] J. Henkel, H. Bukhari, S. Garg, M. U. K. Khan, H. Khdr, F. Kriebel, U. Ogras, S. Parameswaran, and M. Shafique. “Dark Silicon - From Computation to Communication”. In: *International Symposium on Networks-on-Chip (NOCS)*. Invited Special Session Paper. Sept. 2015.

[Hen15] J. Henkel. “Dark Silicon and Dependability”. Keynote Talk, International Symposium on Computer Architecture & Digital Systems (CADS). Oct. 8, 2015.

[HKS15] J. Henkel, M. U. K. Khan, and M. Shafique. “Energy-Efficient Multimedia Systems for High Efficiency Video Coding”. In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. (accepted as a reviewed Special Session paper). Lisbon, Portugal, May 24–27, 2015.

[Her15] A. Herkersdorf. “What happens on an MPSoC stays on an MPSoC – unfortunately!” Keynote Talk at MPSoC Forum. Ventura, CA, USA, July 13, 2015.

[KSH15] M. U. K. Khan, M. Shafique, and J. Henkel. “Hierarchical Power Budgeting for Dark Silicon Chips”. In: *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. July 2015, pp. 213–218. DOI: 10.1109/ISLPED.2015.7273516.

[KPSH15] H. Khdr, S. Pagani, M. Shafique, and J. Henkel. “Thermal Constrained Resource Management for Mixed ILP-TLP Workloads in Dark Silicon Chips”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). HiPEAC Paper Award. ACM, June 7–11, 2015, 179:1–179:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2744916.

[PCH15] S. Pagani, J.-J. Chen, and J. Henkel. “Energy and Peak Power Efficiency Analysis for the Single Voltage Approximation (SVA) Scheme”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 34.9 (Sept. 2015), pp. 1415–1428. DOI: 10.1109/TCAD.2015.2406862.

[PCSH15a] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. “MatEx: Efficient Transient and Peak Temperature Computation for Compact Thermal Models”. In: *Proceedings of the 18th Design, Automation and Test in Europe (DATE)*. Grenoble, France, Mar. 9–13, 2015, pp. 1515–1520.

[PCSH15b] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. “Thermal-Aware Power Budgeting for Dark Silicon Chips”. In: *Proceedings of the 2nd Workshop on Low-Power Dependable Computing (LPDC) at the International Green and Sustainable Computing Conference (IGSC)*. Las Vegas, Nevada, USA, Dec. 14–16, 2015.

[PSCH15] S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel. “Thermal-Aware Power Budgeting for Dark Silicon Chips (Invited talk)”. In: *Workshop on System-to-Silicon Performance Modeling and Analysis at the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. San Francisco, CA, USA, June 8–12, 2015.

[Pag+15] S. Pagani, M. Shafique, H. Khdr, J.-J. Chen, and J. Henkel. “seBoost: Selective Boosting for Heterogeneous Manycores”. In: *Proceedings of the 10th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. Amsterdam, The Netherlands, Oct. 4–9, 2015, pp. 104–113.

[PPSH15] A. Pathania, S. Pagani, M. Shafique, and J. Henkel. “Power Management for Mobile Games on Asymmetric Multi-Cores”. In: *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. July 2015, pp. 243–248. DOI: 10.1109/ISLPED.2015.7273521.

[SH15] M. Shafique and J. Henkel. “Mitigating Power Density and Temperature Problems in the Nano-Era”. In: *IEEE/ACM 34th International Conference on Computer-Aided Design (ICCAD)*. Special Session Paper. San Jose, CA, USA, Nov. 2, 2015.

[Sha15] M. Shafique. “Run-Time Resource and Reliability Management in Dark Silicon Many-Core Chips”. Keynote Talk, International Workshop on Multi-Objective Many-Core Design (MOMAC). Mar. 2015.

[SGGH15] M. Shafique, D. Gnad, S. Garg, and J. Henkel. “Variability-Aware Dark Silicon Management in On-Chip Many-Core Systems”. In: *18th IEEE/ACM Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Grenoble, France, Mar. 9–13, 2015.

[Wag+15] P. Wagner, L. Li, T. Wild, A. Mayer, and A. Herkersdorf. “Knowledge-Based On-Chip Diagnosis for Multi-Core Systems-on-Chip”. en. In: *edaWorkshop 15*. Dresden, Germany, May 2015, pp. 39–45. ISBN: 978-3-86386-914-4.

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

The hardware monitoring system of Project B4 provides the invasive computing system with monitoring data that gives valuable information about the current hardware health. Those data can be used to support a flexible resource allocation that adapts to (temporal) changes within the hardware (e. g. changes in supply voltage or temperature or increased aging) as well as manufacturing variations. The monitoring data can also be used to foresee potential problems before they occur: Hardware components that approach a hardware failure (catastrophic failures as well as parametric failures such as not meeting frequency requirements or exceeding power limitations) can be predicted based on the current and the past monitoring data. That gives an invasive computing system a chance to take corrective measures to prevent such failures or in case of a catastrophic failure at least reduce its impact. To offer an optimal trade-off between accurate monitoring data and used resources to obtain the data, Project B4 continuously works on the optimisation of the overall monitoring system (like types, quantities, duty cycles, needed accuracy, power consumption to alleviate problems of dark silicon). The monitoring system should not pose a potential security risk. So we clarified the issue, that side channels, lowering the security of the invasive computing system, could not be established due to the storage and transfer of monitoring data.

We are currently working on further improvements to our eTPMon (emulated Temperature and Power Monitor) framework [Glo+15]. With eTPMon, we presented an approach to emulate the behaviour of an ASIC power and temperature monitoring system for the FPGA prototyping of MPSoC computing architectures. It is implemented in VHDL and integrated into the FPGA demonstrator. eTPMon delivers power and temperature values on a per-core granularity that are processed by the C*i*C monitor aggregator. The power consumption is modelled

assuming the instructions executed in an ASIC implementation of an invasive computing system. The temperature is calculated from the power consumption based on a thermal RC model. Further improvements of eTPMon comprise the inclusion of cache memory and FPU instructions as well as power modes, and voltage, frequency and temperature dependencies in the model for the power monitor. The improvements also include transient temperature changes in the model for the temperature monitor. We used eTPMon to study the thermal behaviour of LEON3-based as well as TCPA (Tightly-Coupled Processor Array)-based architectures, since different architectures have different usage scenarios with different thermal profiles. That means that the optimisation of temperature and power monitors for both architectures leads to different solutions in terms of data accuracy and duty cycle as well as the placement and the required quantity of the monitors.

In cooperation with Project B2, we presented in [Sou+15] a resource-aware computing paradigm, using run-time power and temperature data received from hardware monitors, to exploit run-time adaptation without violating any thermal and/or power constraints in TCPA. There, we considered a scenario where applications running on mobile platforms require a certain performance level or quality (e. g. high-resolution image processing) that needs to be satisfied while adhering to a certain power budget and temperature threshold. For estimating the power consumption at run time, a mathematical model based on post-synthesis implementations of a TCPA was developed for different CMOS technologies. The temperature variation was emulated at run time.

Post-silicon clock tuning elements are widely used in high-performance designs to mitigate the effects of process variations and aging. Located on clock paths to flip-flops, these tuning elements can be configured through the scan chain so that clock skews to these flip-flops can be adjusted after manufacturing. Owing to the delay compensation across consecutive register stages enabled by the clock tuning elements, higher yield and enhanced robustness can be achieved. These benefits are, nonetheless, attained by increasing die area due to the inserted clock tuning elements. For balancing performance improvement and area cost, an efficient timing analysis algorithm is needed to evaluate the performance of such a circuit. So far, this evaluation is only possible by Monte Carlo simulation which is very time-consuming. In [LS15], we proposed an alternative method using graph transformation, which computes a parametric minimum clock period and is more than 100 times faster than Monte Carlo simulation while maintaining a good accuracy. This

method also identifies the gates that are critical to circuit performance, so that a fast analysis/optimisation flow becomes possible. This method can be used to achieve performance optimisation of invasive computing systems in post-silicon steps.

The aging of integrated circuits cannot be neglected in advanced processor technologies. With the varying workload between components of the invasive computing system as well as within single components, some components are more prone to aging then others.

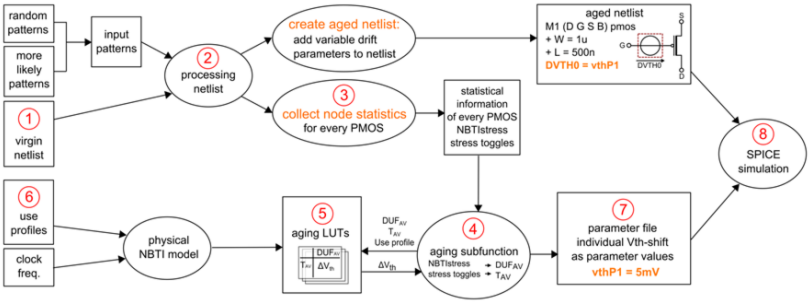


Figure 4.15: Data flow diagram of the reliability assessment tool extrapolating aging for circuit lifetime.

In [Ary+14], we developed and tested a reliability assessment tool, that can simulate the NBTI degradation (including its recovery phase) during the design phase, in order to perform an evaluation of the timing behaviour of the circuit over its lifetime (Fig. 4.15). With the tool, it is possible to detect aging paths that experience a different amount of stress and thus age at different speeds. So it provides a useful means when evaluating the reliability within the lifetime of a circuit to prevent early circuit failures. By using aging monitors within the invasive computing system, it is possible to monitor the aging of the components. This offers the possibility to analyse the aging behaviour and also develop measures against aging effects. As for power and temperature monitors, the approach to integrate aging monitors into the FPGA demonstrator platform will be to emulate the behaviour of an ASIC aging monitor (eAMon) and to integrate it into our eTPMon framework (leading to eTPAMon). A real aging monitor naturally “remembers” the aging status, while an emulated monitor loses such information when powered off. So the aging status of eAMon needs to be stored and loaded at every new start of the demonstrator. But by storing and loading the aging status, it is possible to store and

load every point in the system's lifetime easily. Also timing pre-errors could be manually injected. All in all, this offers high flexibility for demonstrations on the FPGA demonstrator platform. Currently, we are working on the implementation of the eAMon. A possible solution is to first identify the possible critical paths (PCPs) during the lifetime with our aging-aware timing analyses (TA) [LBS14] and our reliability assessment tool [Ary+14]. And afterwards to replace the sink flip-flops (FFs) there with our in-situ delay monitors[10]. The in-situ delay monitoring concept is based on the pre-error rate, that is used as an indicator for the remaining timing slack. For the FPGA demonstrator this concept can be further simplified into a look-up-table (LUT)-based approach as done for the power and temperature monitor, usable during run-time. Since the aging of integrated circuits evolves much slower than changes in, e. g. temperature and power, a periodical online test of the PCPs (at idle time) is sufficient to detect any aging that might endanger correct computations. The eAMon delivers the pre-error rate for each PCP that could be processed to a per-core, per-region or per-tile value and that could be further abstracted to an aging/degradation value. Determination of the needed granularity for higher levels and the component that does further abstractions (monitors, iDoC, software) will be subject of our future work.

Publications

[Ary+14] N. P. Aryan, A. Listl, L. Heiss, C. Yilmaz, G. Georgakos, and D. Schmitt-Landsiedel. "From an analytic NBTI device model to reliability assessment of complex digital circuits." In: *International On-Line Testing Symposium (IOLTS)*. 2014, pp. 19–24.

[Glo+15] E. Glocker, Q. Chen, A. Zaidi, U. Schlichtmann, and D. Schmitt-Landsiedel. "Emulation of an ASIC power and temperature monitor system for FPGA prototyping". In: *Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2015 10th International Symposium on*. June 2015, pp. 1–8. DOI: 10.1109/ReCoSoC.2015.7238083.

[LS15] B. Li and U. Schlichtmann. "Statistical Timing Analysis and Criticality Computation for Circuits With Post-Silicon Clock Tuning Elements". In: *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 34.11 (Nov. 2015), pp. 1784–1797. ISSN: 0278-0070. DOI: 10.1109/TCAD.2015.2432143.

[10] M. Wirnshofer. *Variation-Aware Adaptive Voltage Scaling for Digital CMOS Circuits*. Vol. 41. Springer Series in Advanced Microelectronics, 2013.

[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: 10.1016/j.microrel.2014.01.013.

[Sou+15] Éricles R. Sousa, F. Hannig, J. Teich, Q. Chen, and U. Schlichtmann. “Runtime Adaptation of Application Execution under Thermal and Power Constraints in Massively Parallel Processor Arrays”. In: *Proceedings of the 18th International Workshop on Software and Compilers for Embedded Systems (SCOPES)*. (St. Goar, Germany). ACM, June 1–3, 2015, pp. 121–124. ISBN: 978-1-4503-3593-5. DOI: 10.1145/2764967.2771933.

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

Jürgen Becker, Andreas Herkersdorf, Jürgen Teich

Stephanie Friederich, Jan Heißwolf, Leonard Masing, Preethi Parayil, Ravi Kumar Pujari, Andreas Weichslgartner, Thomas Wild, 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.

The current research focus of Project B5 lies on the predictability of non-functional properties such *timing*, *security*, *fault tolerance*, and *energy consumption* of invasive NoCs. Our other research tackles the NoC topology (including 3D NoCs) and cache coherence as well as novel synchronisation mechanisms.

Support for Predictable Real-Time Stream Processing

Already in 2014, together with Project A1 and Project A4, a hybrid application mapping methodology to enable run-time predictability was introduced [Wei+14]. Hybrid application mapping combines the advantages of compute intensive design-time analysis and dynamic run-time management. In an *i*NoC, the guaranteed service connections enable us to bound the interference of other applications and hence enable us to give upper bounds for the end-to-end latency of communicating applications or tasks thereof. With a new heuristic presented in [WWT15], we could also increase the number of successfully mapped applications and decrease the energy consumption of a number of mapped applications (for details see the report of Project A1).

Secure Communication for Security-Critical Applications

The sharing of resources in general and communication resources in particular may open side channels and disclose application character-

istics to an attacker. Especially in the case of packet-switched NoCs, e. g. our *i*NoC architecture, this may become a severe issue when executing any security-critical application. For example, in Figure 4.16, the communications between two tasks B_1 and B_2 of a security-critical application overlap with the communication of a malicious application (tasks A_1 and A_2). By sending constantly dummy data over the channel and measuring the transmission time, the malicious application can draw conclusions on the amount of data sent by the security-critical application. To prevent such attacks, together with Project C5, we developed first concepts to close this side channel by usage of spatial isolation in the context of hybrid application mapping.

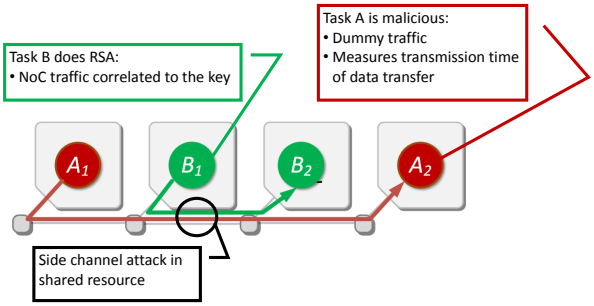


Figure 4.16: Possible side channel attack in an *i*NoC.

Network Topologies

For decreasing the average distance between network nodes, the NoC architecture has been extended by a third dimension. In addition to the 2D mesh architecture, there are now implementations of 3D mesh and torus networks available. Higher dimensional networks offer shorter longest path in the network and hence resulting in a lower average latency for inter tile communication. The 3D *i*NoC architecture will be partitioned across multiple layers of 3D integrated circuits.
To manage the restricted number of inter layer connections in three dimensional integrated circuits, we introduced a novel router implementation with heterogeneous bandwidth ports.

Fault-Tolerant Communication

Manufacturing defects and aging effects are expected to cause permanent faults in future highly integrated technology nodes which are

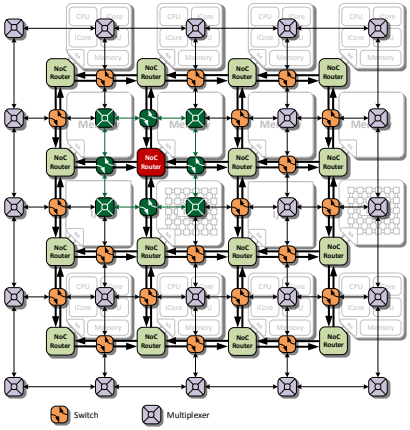


Figure 4.17: Example of a 1x1 ring bus configuration in a 4x4 NoC architecture with a *second layer network* [Hei+15].

B5

targeted by NoC-based architectures. To deal with permanent faults, we introduced a fault-tolerant NoC-based architecture based on a light weight *second layer network* which takes over the duties of defective routers. A distributed *localisation strategy* is utilised to identify erroneous routers and for configuring the second network layer accordingly [Hei+15]. A configuration for bypassing a single defective router is shown in Figure 4.17. In case of no defects, the *second layer network* can be used for power saving. The implementation of the fault tolerance concept based on the second layer network is finished and a first version of a fault-tolerant *i*NoC has been released.

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 MPSoC architecture, the network adapter architecture has been 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 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.

Support of Region-based Cache Coherence

In the invasive MPSoC architecture as developed in the first funding phase, cache coherence is provided only within one compute tile for the associated *tile local memory (TLM)*. This limits the maximum size of an X10 place, which is characterised by common shared memory, to only a single tile. In order to support applications that can profit from places with more cores cache coherency between several tiles (inter-tile coherency) is required. For this and to limit the overhead associated with global coherence, Project B5 proposed to provide means that allow a dynamic creation and dissolving of coherency regions depending on the requirements of the applications as shown in Figure 4.18. In discussions with application programmers, it turned out that applications like DNA sequencing, sparse matrix processing, or tree traversals could profit from such inter-tile coherence. Further it emerged that such inter-tile coherence support was preferred for the TLMs of the tiles. Therefore, this will make up the initial focus of our work. Extending the support for global memory was considered as an option of lower priority.

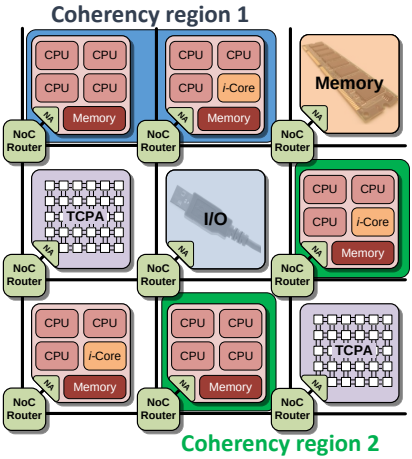


Figure 4.18: Invasive architecture with two coherency regions.

In respect to the realisation of inter-tile coherence, an initially conceived approach with snoop extensions among neighbouring tiles turned out to have too many restrictions and will be therefore not followed. In-

stead, a directory based concept with distributed directories associated to the TLMs will be further investigated. For this, we will extend the coherency mechanisms based on distributed directories as described in [DWH14] and [Par+15] to enable dynamic inter-tile coherence.

As a first step, we established a SystemC-based cache simulation model of a tiled invasive MPSoC architecture as shown in Figure 4.18 based on the model used in [Par+15]. The model is configurable in terms of number of processors per tile and reflects the memory hierarchy of the invasive architecture. During simulation, the processor model replays traces that were generated via Gem5 for appropriate workloads. In every compute tile of the architecture model a *coherency region manager (CRM)* module is introduced that provides the required operations for inter-tile coherence on TLMs. This means monitoring cache line sharing among all tiles in a coherency region and taking action in terms of invalidation and enforced write back when required. The necessary functionalities have been elaborated for all relevant memory access variants as part of investigations of a first CRM design study.

Work is ongoing to study the performance of inter-tile coherence with the mentioned simulation model. Topics of investigation are the scalability of the concept in terms of number of tiles and their location making up a coherence region.

Integration Work and Demonstration

As in previous years, a lot of effort was put into implementation and prototyping of invasive MPSoC architectures. 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.

Outlook

One of the main tasks in 2016 for Project B5 will be the investigation of efficient power gating mechanisms for *i*NoC communication resources. The impact of these mechanisms on real-world applications shall be investigated in collaboration with Project D3. Also, Project B5 will further contribute to the demonstrator integration activities of Project Z2.

Publications

[DWH14] P. Damodaran, S. Wallentowitz, and A. Herkersdorf. “Distributed cooperative shared last-level caching in tiled multiprocessor system on chip”. In: *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*. Mar. 2014, pp. 1–4. DOI: 10.7873/DATE.2014.096.

[Hei+15] J. Heisswolf, A. Weichslgartner, A. Zaib, S. Friederich, L. Masing, C. Stein, M. Duden, R. Klöpfer, T. Wild, A. Herkersdorf, J. Teich, and J. Becker. “Fault-tolerant Communication in Invasive Networks on Chip”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8.

[Par+15] P. Parayil, A. Zaib, T. Wild, W. Stefan, and A. Herkersdorf. “Sharer Status-based Caching in tiled Multiprocessor Systems-on-Chip”. In: *HPC 2015 - 23rd High Performance Computing Symposia*. SCS, The Society for Modeling & Simulation, Apr. 2015, pp. 67–74.

B5

[Sch15] M. Schmid. “Rapid Prototyping for Hardware Accelerators in the Medical Imaging Domain”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 24, 2015.

[Wei+14] 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. 15, 2014, pp. 1–10. DOI: 10.1145/2656075.2656083.

[Wei+15] A. Weichslgartner, J. Heisswolf, A. Zaib, T. Wild, A. Herkersdorf, J. Becker, and J. Teich. “Position Paper: Towards Hardware-Assisted Decentralized Mapping of Applications for Heterogeneous NoC Architectures”. In: *Proceedings of the second International Workshop on Multi-Objective Many-Core Design (MOMAC) in conjunction with International Conference on Architecture of Computing Systems (ARCS)*. Porto, Portugal: IEEE, Mar. 2015, pp. 1–4.

[WWT15] S. Wildermann, A. Weichslgartner, and J. Teich. “Design Methodology and Run-time Management for Predictable Many-Core Systems”. In: *Proceedings of the 6th IEEE Workshop on Self-Organizing Real-Time Systems (SORT)*. Apr. 13, 2015, pp. 1–8.

[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)*. Vol. 9017. Lecture Notes in Computer Science (LNCS). Springer, 2015, pp. 186–198. ISBN: 978-3-319-16085-6. DOI: 10.1007/978-3-319-16086-3_15.

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, Anuj Pathania, Volker Wenzel

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.

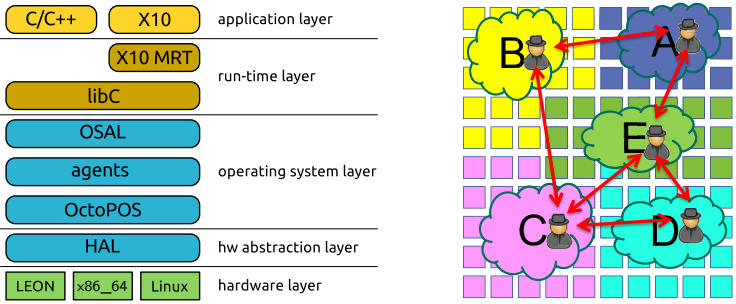


Figure 4.19: *i*RTSS Architecture. Left the overall layered organisation of *i*RTSS (blue), right the structure of the distributed agent system.

Architectural Overview

Figure 4.19 provides a high-level view of the current *i*RTSS architecture. Key elements are OctoPOS,[11] the parallel operating system (POS) that

[11] The prefix "Octo" stems from the denotation of a nature which is highly parallel in its actions as well as adaptable to its particular environment: the octopus, being able to act in parallel by means of its tentacles, adapt itself through colour change, and, due to its highly developed nervous system, attune to dynamic environmental conditions and impact.

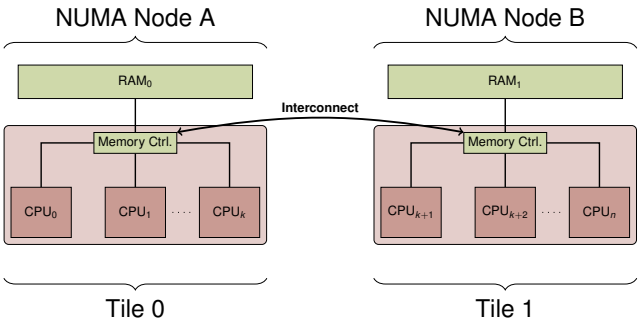


Figure 4.20: OctoPOS x86_64 for invasive computing. InvasIC tiles are mapped to NUMA domains.

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.

C1

The Configurable OctoPOS Kernel

We provide OctoPOS for a variety of platforms—with and without support for dedicated hardware (Figure 4.19). The 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.

In 2015, we could, in cooperation with many other projects, further demonstrate the benefits of this approach for invasive applications [Pau+15] as well as the code generation of the InvadeX10 compiler developed by Project C3 [Moh+15].

Native x86_64 Support: One important milestone we have reached in 2015 was the finalising and provision of the native x86_64 member of the OctoPOS family. The main purpose of the x86_64 version is comparability: In cooperation with Project C3, which provides a new compiler back end for OctoPOS x86_64, application projects can now use this platform to compare the invasive approach with existing (e. g. Linux-based) approaches on common, state-of-the art hardware. Invasive hardware concepts are mapped as far as possible to the standard hardware and emulated by software otherwise: Tiles are mapped to

NUMA domains (Figure 4.20 depicts this for a 2-socket system). The *i*NoC and C*i*C functionalities are currently emulated by synchronous syscalls, but will eventually be offloaded to an exclusively reserved *system core*. We expect this core to be highly underutilised in practice, but consider this as fruitful approach to investigate and quantify the possible benefits of dedicated hardware support for latency hiding, as provisioned by the original invasive platform.

In this realm, Project C1 also finally ordered the requested 4-socket Xeon E7v3 (Haswell) system, which had been prolonged because of the original Haswell's TSX bugs. The C1 system primarily serves OctoPOS development; a second identical system is provisioned by Project Z2, providing invasive applications with the invasive interface on up to 96 logical cores.

The possibility to run OctoPOS and invasive workloads on standard hardware also facilitate utilising further tools and techniques for energy-aware development and predictable energy estimations [Hön+15; HJSP15; Wäg+15] to support the management of dark silicon in the second funding phase.

C1

LEON Platform Improvements: Besides the x86_64 port, the OctoPOS team also spent significant resources in 2015 on stability improvements for the invasive hardware platform. This includes the provisioning of test cases, the development of an automatic testing infrastructure, as well as general debugging support.

Dynamic Memory Protection: We have also designed and implemented mode-/MMU-based *dynamic* basic isolation support for *i*RTSS. Hardware-based isolation with respect to protection levels and memory access shall be a run-time configurable feature in *i*RTSS in order to support dynamic (de-)virtualisation: There is little need to use the MMU and provide the corresponding operating-system functions if, for instance, the machine is used in single-user mode or all application processes stem from programs written in a type-safe language, for which certain properties can be guaranteed ahead of time (e. g. X10). Neither of those processes benefits from that feature in functional respect, but all of them are handicapped in non-functional respect, as that feature does not come for free: it creates unconcealable software-induced overhead in the course of memory management, system call and return, interrupt handling, process dispatching and inter-process communication. We quantified this overhead, which is indeed significant. Table 4.1 depicts some results for the x86_64 platform: While the sole provisioning of memory-protection support increases the run-time of the `invade` syscall by only seven percent (in comparison to a variant that does not sup-

	protection: n/a	protection: off	protection: on
invade	1.00×	1.07×	22.2×
allocate page	1.00×	4.73×	41.8×
free page	1.00×	4.07×	41.9×

Table 4.1: Relative run-time costs for syscalls induced by memory protection on OctoPOS x86_64.

port protection at all), actually activating it at run time induces an overhead of several thousand percent! For memory-management operations, the overhead is even larger: Necessary bookkeeping slows down the allocation and release of memory pages by a constant difference. With protection switched on, expensive TLB shootdowns have to be performed with increasing costs depending on the number of cores.

This underlines the *i*RTSS goal to charge only applications with theses costs that actually need it, which requires memory protection to be a dynamic feature that at run time can be dynamically enabled/disabled on claim granularity.

Wait-Free Synchronisation Support: A most important design goal of OctoPOS is *internal scalability* and *predictability* by prohibiting any kind of blocking lock-based synchronisation inside the kernel. Waiting in OctoPOS happens only for purposes of *logical synchronisation*, but never for mutual exclusion. With *guarded sections* [DSP15], we have provided a new abstraction for wait-free mutual exclusion that can also be easily applied in user-mode applications. The basic principle is depicted in Figure 4.21 (left side): Upon entering an already-taken guarded section, a *requester* thread never blocks, but instead enqueues the operation to be performed inside the guarded section and passes by. The operation itself is guaranteed to be executed eventually by the current owner or some other thread (*sequencer*). If the requester, ultimately, needs the respective result (point of *logical* synchronisation), it can wait for it by means of a *future* object.

C1

Figure 4.21 shows the scalability of this approach (in comparison to MCS locks and read spinlocks) on up to 64 cores on x86_64. The measurements relate to concurrently entering and leaving an empty critical section (maximal contention). While guarded sections perform worse than locks if one *always* has to immediately wait for the result (GS-wait), they scale much better if logical synchronisation on the result is not required (GS-nowait). Locks, in contrast, unconditionally force the entering threads to wait, hampering predictability and throughput.

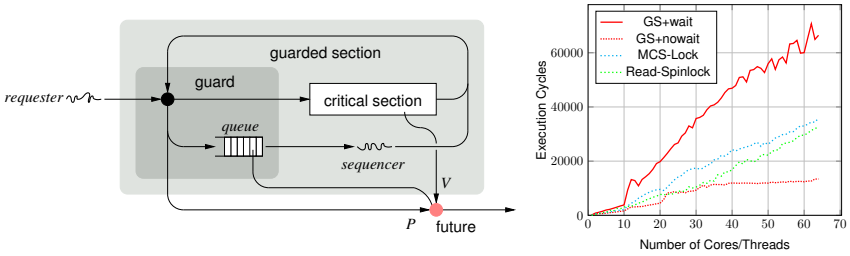


Figure 4.21: Guarded sections as a new means for wait-free mutual exclusion.

The Agent System

The task of the *i*RTSS agent system is to provide a decentralised, scalable resource management for invasive applications. Per application, agents negotiate among each other for computational resources by comparing speedup curves to optimise the performance of their applications.

Integration with *DaSiM*: A concept for the combination of Dark Silicon Management (*DaSiM*) and the *i*RTSS agent system has been developed together with Project B3. During normal agent resource negotiation, *DaSiM* will be queried to provide the agent system a set of 'dark silicon safe' alternatives for resource allocation. The agent system will choose from this set with respect to optimisation of performance. Additionally, *DaSiM* will make use of so-called "dark agents", which acquire resources from application agents to power-gate them in order to prevent thermal violations.

Implementation Refinement: Issues with the current implementation of the agent system were discovered and circulated within the project in the so called "problem report". Subsequently, considerable efforts have been invested into refinement and reduction of imperfections of the agent system's implementation.

Malleability Support: Applications can be either static, mouldable or malleable with respect to their level of parallelism. The number of resources allocated to a malleable application can be changed at any time during it's execution, allowing them to exploit the full potential of *Invasive* architecture through Invade and Retreat system calls. An effort was made to add support for malleable applications to the *i*RTSS agent system. The actual implementation of malleable functionality involved modifying and extending multiple classes within the agent subsystem of the *i*RTSS. To maintain maximum flexibility, malleability is implemented with per-claim granularity.

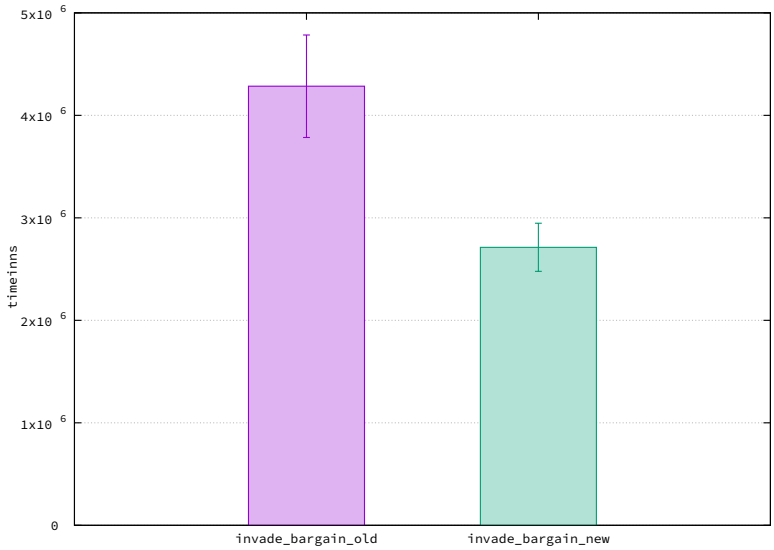


Figure 4.22: Implementation run-time comparison of the function invade_bargain.

C1

Performance Improvements: The bargaining part of the invade algorithm was mostly rewritten to fix performance problems detected in the implementation. The changes lead to a average speedup of 1.58, with the percentual standard deviation dropping from 11.67% to 8.65%. The performance improvement are visualised in Figure 4.22.

Agent System Visualisation: There is an ongoing effort to develop a visualisation tool to illustrate the inner workings of the agent system on the final demonstrator. The visualisation tool is based on *InvadeView*, a Qt tool developed within Project C2. The visualisation tool is divided into two parts, one running on the *Invasive Demonstrator* and the other running on a *Linux* machine connected to the demonstrator. System state data will be generated by the agent system, which will then be processed by an on-demonstrator utility and sent over to the *Linux* machine. An on-Linux utility then receives the data and sends it to an extended *InvadeView* Qt application for display.

Outlook

We will continue the work on the x86_64 and LEON variants of the OctoPOS kernel. For x86_64, we investigate a Linux–OctoPOS side-by-

side operation: Linux will run on a single core and a small portion of the memory, but serve with access to device drivers and file systems we do not intend to implement for OctoPOS. This should make it significantly easier to port and compare I/O-intensive applications. We will also evaluate and quantify the benefits of InvasIC-specific hardware devices (C*i*C, *i*NoC) by software emulation (partial virtualisation) on dedicated cores.

Future plans also include the extension of the agent system's supported constraints. An application provides constraints to the agent system, for instance, about the minimum, maximum amount of required cores and the required type (RISC, *i*-Core, TCPA). In the future, this shall be extended to support more constraints that have been already defined for the invadeX10 framework. We intend to apply methods of *Distributed Constraint Solving*. This will be done in close collaboration with Project A1, where experiences with constraint satisfaction techniques exist already.

Publications

[DSP15] G. Drescher and W. Schröder-Preikschat. "Guarded Sections: Structuring Aid for Wait-Free Synchronisation". In: *Proceedings of the 18th IEEE Symposium on Real-Time Computing (ISORC 2015)*. Washington, DC, USA: IEEE Computer Society Press, 2015, pp. 280–283. ISBN: 978-1-4799-8781-8.

[HJSP15] T. Hönig, H. Janker, and W. Schröder-Preikschat. "The FigarOS Operating System Kernel for Fine-Grained System-Level Energy Analysis". In: *DAC 2015 Workshop on System-to-Silicon Performance Modeling and Analysis: Power, Temperature and Reliability*. San Francisco, CA, USA, June 7, 2015.

[Hön+15] T. Hönig, C. Eibel, B. Herzog, H. Janker, P. Wägemann, and W. Schröder-Preikschat. "Playing Hare and Tortoise: The FigarOS Kernel for Fine-Grained System-Level Energy Optimizations". In: *2015 Brazilian Symposium on Computing Systems Engineering (SBESC '15)*. Fóz do Iguaçú, Brazil: IEEE, Nov. 2015.

[IHSH16] S. Iqtedar, O. Hasan, M. Shafique, and J. Henkel. "Formal Probabilistic Analysis of Distributed Resource Management Schemes in On-Chip Systems". In: *IEEE/ACM 19th Design, Automation and Test in Europe Conference (DATE'16)*. (Dresden, Germany). (accepted). Mar. 2016.

[Moh+15] M. Mohr, S. Buchwald, A. Zwinkau, C. Erhardt, B. Oechslein, J. Schedel, and D. Lohmann. “Cutting Out the Middleman: OS-Level Support for X10 Activities”. In: *Proceedings of the fifth ACM SIGPLAN X10 Workshop*. X10 ’15. Portland, OR, USA: ACM, June 14, 2015, pp. 13–18. DOI: 10.1145/2771774.2771775.

[Pat+16] A. Pathania, V. Venkataramani, M. Shafique, T. Mitra, and J. Henkel. “Distributed Fair Scheduling for Many-Cores”. In: *Design Automation and Test in Europe (DATE)*. 2016.

[Pau+15] J. Paul, W. Stechele, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, W. Schröder-Preikschat, M. Kröhnert, T. Asfour, Éricles R. Sousa, V. Lari, F. Hannig, J. Teich, A. Grudnitsky, L. Bauer, and J. Henkel. “Resource Awareness on Heterogeneous MPSoCs for Image Processing”. In: *Journal of Systems Architecture* 61.10 (Nov. 6, 2015), pp. 668–680. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.09.002.

[SSKH16] A. K. Singh, M. Shafique, A. Kumar, and J. Henkel. “Resource and Throughput Aware Execution Trace Analysis for Efficient Run-time Mapping on MPSoCs”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 35.1 (Jan. 2016), pp. 72–85. ISSN: 0278-0070. DOI: 10.1109/TCAD.2015.2446938.

[Wäg+15] P. Wägemann, T. Distler, T. Hönig, H. Janker, R. Kapitza, and W. Schröder-Preikschat. “Worst-Case Energy Consumption Analyis for Energy-Constrained Embedded Systems”. In: *Proceedings of the 27th EUROMICRO Conference on Real-Time Systems (ECRTS 2015)*. Lund, Sweden, July 9, 2015, pp. 105–114.

C2: Simulative Design Space Exploration

Frank Hannig

Sascha Roloff, Vahid Lari

Project C2 investigates novel simulation techniques that enable the validation and variants' exploration of architecture options as well as invasion strategies. The timed functional simulation, evaluation, and co-exploration of both invasive resource-aware programs and invasive heterogeneous tiled architectures are the major objectives of Project C2. Likewise, the architectures to be investigated are highly complex and diverse as well as the workloads to be simulated. For example, a simulation scenario could be made up of: (a) an architecture under investigation that is composed of dozens to hundreds heterogeneous processor cores organised in compute tiles and interconnected via a network-on-chip (i. e. the *i*NoC), and (b) multiple competing applications that consist again of highly parallel programs. In order to handle this complexity, novel parallel simulation methods and environments have been developed. Parallel simulation techniques enabled us 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. In addition to this foundation, Project C2 advanced its research in two major directions within this year. The first research direction has focused on simulation-based analysis of non-functional properties such as timing and fault tolerance. The second research focus has laid the foundation for the projected design space exploration. Here, first concepts and implementations for the generation of architecture variants as well as pattern-based parallel workloads have been developed.

Parallel Simulation

A major milestone of Project C2's research was the parallelisation of the InvadeSIM simulator [RSHT15]. Here, modern multicore processors are exploited in order to speedup the simulation of complex architectures

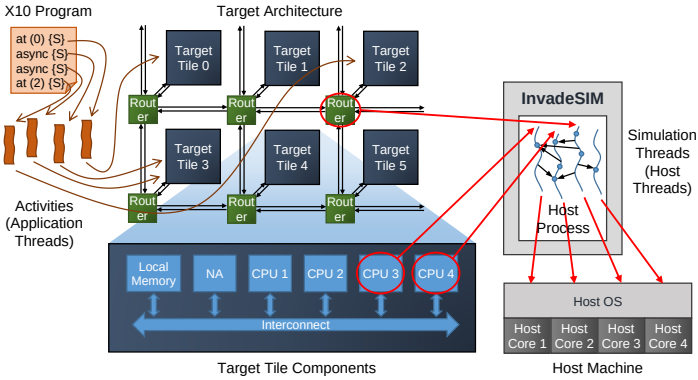


Figure 4.23: Overview of the heterogeneous MPSoC simulator InvadeSIM. A parallel X10 application executes several activities on a tiled architecture. Each activity is executed by one target processor, which in turn is simulated by one simulation thread on the host machine. For parallel simulation of the simulation threads, multicore host processors are exploited to substantially increase simulation performance.

and long running benchmarks. The parallel simulation was achieved by extending the direct-execution simulation approach of InvadeSIM[12] by different parallelisation strategies. In [RSHT15], we proposed four novel parallel discrete-event simulation techniques, which map thread-level parallelism within X10 applications to core-level parallelism on the target architecture and back to thread-level parallelism on the host machine. In order to achieve this, the correct synchronisation and activation of the host threads was required. Experiments with parallel real-world X10 applications have been used to compare the different parallelisation techniques against each other and demonstrated that 10 times faster simulations than a sequential simulation can be achieved on a 12-core Intel Xeon processor. The InvadeSIM simulation kernel simulates the functional execution of spawned activities on the individual processors including their temporal behaviour, which in general depends on the type of processor assigned to each activity. This is achieved by assigning each hardware resource of the target architecture a distinct simulation thread on the host machine as indicated in Figure 4.23 by red arrows from the target architecture to the InvadeSIM kernel. Using parallel discrete-event simulation, the simulation threads are then assigned

C2

[12] S. Roloff, F. Hannig, and J. Teich. "Approximate Time Functional Simulation of Resource-Aware Programming Concepts for Heterogeneous MPSoCs". In: *Proceedings of the 17th Asia and South Pacific Design Automation Conference (ASP-DAC)*. Sydney, Australia, Jan. 30–Feb. 2, 2012, pp. 187–192. ISBN: 978-1-4673-0770-3. DOI: 10.1109/ASPDAC.2012.6164943.

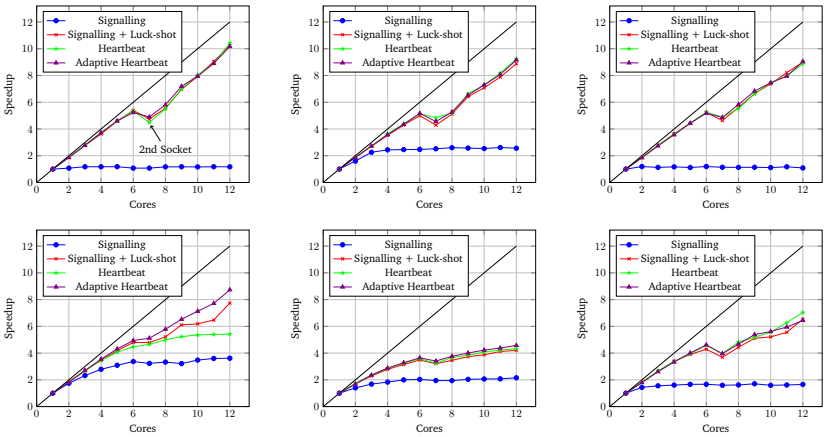


Figure 4.24: Evaluation of selected X10 benchmarks (from left to right: MontyPi, KMeans, Random Access, Graph Analysis, FFT, Streaming) [RSHT15]. The applications are mapped/simulated in a distributed-parallel manner on a 4x4-tile architecture, each tile has four cores. The achievable speedups on a simulation host with up to 12 cores are shown for our four novel parallel simulation techniques.

for parallel execution to the cores of the host machine as indicated in Figure 4.23 by red arrows from the InvadeSIM kernel to the host machine. The first parallelisation technique introduced in [RSHT15] is called *signalling*, which is based on performance counter signals that are raised if a certain simulation thread passes the simulation time of a waiting simulation thread. In this case, the simulation thread is stopped and checks, whether the waiting thread can be waked up. If yes, it is activated and both threads run concurrently. Nevertheless, this technique might drastically reduce the number of simultaneously running simulation threads. This disadvantage can be compensated by an improvement called *luck-shot* in which snapshots of the current simulated time of all running simulation threads are determined and used for comparison at a synchronisation point. The other investigated parallelisation technique is called *heartbeat* and is based on a dedicated thread, which periodically checks, whether simulation threads can be activated. Here, the choice of the period has been shown to be essential for the resulting speedup and a statically determined period might be not the best choice. Thus, we finally proposed an *adaptive heartbeat* technique, which dynamically changes its period according to the dynamic behaviour of the application to maximise the speedup gained through parallel simulation. The different parallelisation techniques are compared in terms of speedup in Figure 4.24.

Simulation-based Timing Predictability and Fault Tolerance Analysis

Predictability is one of the key features of invasive computing in our current funding phase. Here, simulation-based analysis techniques can help to early quantify that the concepts of invasive computing increase predictability. In this regard, Project C2 investigated timing predictability of complex streaming applications from the domain of computer vision on invasive MPSoCs using InvadeSIM [RWHT15]. In particular, an object detection algorithm chain was separated into several communicating actors using actor-based programming in X10 together with Project A1 and Project A4. Here, each actor can be mapped onto a different tile of the simulated architecture, which allows to analyse various mappings. Depending on the claimed resources and the interferences between the algorithm chain and other applications on the MPSoC, different latency and throughput values as well as different variations of these values (jitter) were analysed. Different execution scenarios like non-invasive versus invasive execution of the algorithm chain competing with two other applications as well as different resource mappings of a single-application execution were considered. It could be shown that using the concepts of invasive computing (isolating applications through exclusive reservation of resources), variations in latency and throughput as measure of predictability could be substantially reduced and given user requirements can be certainly satisfied. A demo with small videos of these results can be found at: `http://invasic.cs.fau.de/en/demos.php`

In collaboration with Project B2 and Project C3, we also extended the simulation engine for tightly-coupled processor arrays (TCPAs) so that to empirically evaluate the proposed mechanisms for supporting different redundancy levels, such as dual (DMR) and triple modular redundancy (TMR) when executing loop programs on TCPAs with either late or immediate error handling capabilities [Lar+15a; Tan+15; Lar+15b; Lar15]. Thanks to the developed simulation-based fault injection environment, we could evaluate the reliability gains when using each of the mentioned redundancy mechanisms compared to non-redundant executions.

Automatic Generation of Architecture Variants and Pattern-based Parallel Workloads

Based on the aforesaid achievements in suitable simulation techniques, the other major focus of Project C2 in the current funding phase is an automatic *co-exploration* across all platform layers including architectures, invasion strategies and applications as shown in Figure 4.25 as

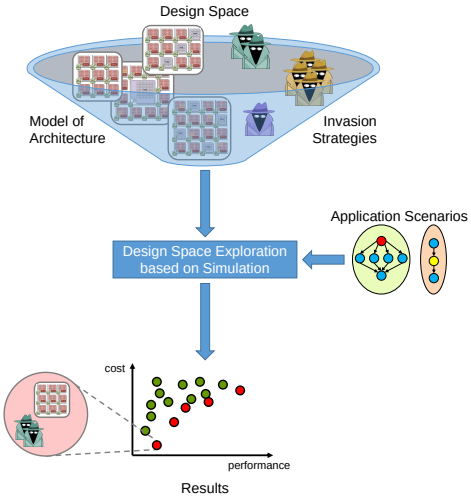


Figure 4.25: Conceptual flow of architecture and strategy design space exploration.

well as the automatic generation of pattern-based workload scenarios. In this regard, we have stated to define a basic infrastructure for architecture exploration and pattern-based generation of generic simulation workloads.

C2

Architecture exploration is used to determine a set of optimal architecture configurations for a given mix of applications with respect to multiple objectives, including execution latency, temperature distribution, and power consumption. For this purpose, Project C2 investigated how existing optimisation frameworks (e. g. Opt4J[13]) can be utilised to generate different architecture variants as well as to couple it with InvadeSIM as evaluation engine for the aforementioned objectives. Here, important problems to be solved are the efficient coding of an architecture variant within the optimisation framework, the definition of appropriate operations to manipulate the architecture coding as well as techniques to reduce the search space by, e. g. mirroring smaller architecture variants to bigger ones. Furthermore, a generic XML-based architecture description format was defined to interface the different tools.

Another important aspect of architecture exploration is the employed

[13] M. Lukasiewycz, M. Glaß, F. Reimann, and J. Teich. “Opt4J: A Modular Framework for Meta-heuristic Optimization”. In: *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO)*. (Dublin, Ireland). ACM, 2011, pp. 1723–1730. ISBN: 978-1-4503-0557-0. DOI: 10.1145/2001576.2001808.

workload model, which is crucial for the overall performance of the optimisation process. One option is to use existing X10 applications, which are functionally and temporally simulated by InvadeSIM. Another option that Project C2 investigates is the use of task graphs based on parallel programming patterns. Here, we developed a first method to systematically generate task graphs, which represent the concurrent execution behaviour of common parallel programming patterns, such as *divide and conquer*, *MapReduce*, or *embarrassingly parallel*. This tool allows to select and parametrise existing programming patterns as well as to define new patterns. Nodes of the generated task graphs can be annotated different properties, such as execution time, temperature, or energy consumption. Here, generic scaling functions manipulate the evolution of the different properties, e. g. the workload is divided by the number of nodes into which another node is split.

Organisation of Scientific Events and Public Dissemination

Frank Hannig organised a special session on "Dynamics and Predictability in Stream Processing—A Contradiction?" at the 13th IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia), Amsterdam, The Netherlands, in October 2015. The special session was composed of four presentations whereof three originated from the CRC.

Also in October 2015, researchers and students of Project C2 vividly demonstrated the principles of invasive computing at the *Long Night of Sciences* ("Lange Nacht der Wissenschaften", see also photographies in Chapter 8 "Trainings and Tutorials") in Erlangen. For this purpose, a pan-tilt-zoom camera in form of a Martian was tracking a tennis ball, which could be freely moved by the visitors in the entire room. This object tracking application was simulated and visualised in real-time using our simulator InvadeSIM. Thereby, the visitors of various age and educational background got fundamental insights into tomorrow's multicore processor architectures and the concepts of invasive computing.

Lastly, CRC's principal investigators Hannig and Herkersdorf edited a special issue on "Testing, Prototyping, and Debugging of Multi-Core Architectures" in Elsevier's Journal of Systems Architecture [HH15]. This special issues is partly a follow-up of the Racing Workshop[14] that we organised last year.

[14] 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.

Publications

[HH15] F. Hannig and A. Herkersdorf. “Introduction to the Special Issue on Testing, Prototyping, and Debugging of Multi-Core Architectures”. In: *Journal of Systems Architecture* 61.10 (Nov. 7, 2015), p. 600. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.11.003.

[Lar15] V. Lari. “Invasive Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, Nov. 18, 2015.

[Lar+15a] V. Lari, A. Tanase, J. Teich, M. Witterauf, F. Khosravi, F. Hannig, and B. Meyer. “A Co-Design Approach for Fault-Tolerant Loop Execution on Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231157.

[Lar+15b] V. Lari, J. Teich, A. Tanase, M. Witterauf, F. Khosravi, and B. Meyer. “Techniques for On-Demand Structural Redundancy for Massively Parallel Processor Arrays”. In: *Journal of Systems Architecture (JSA)* 61.10 (Nov. 2015), pp. 615–627. ISSN: 1383-7621. DOI: http://dx.doi.org/10.1016/j.sysarc.2015.10.004.

[RSHT15] S. Roloff, D. Schafhauser, F. Hannig, and J. Teich. “Execution-driven Parallel Simulation of PGAS Applications on Heterogeneous Tiled Architectures”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). ACM, June 7–11, 2015, 44:1–44:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2744840.

[RWHT15] S. Roloff, S. Wildermann, F. Hannig, and J. Teich. “Invasive Computing for Predictable Stream Processing: A Simulation-based Case Study”. In: *Proceedings of the 13th IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia)*. (Amsterdam, The Netherlands). IEEE, Oct. 8–9, 2015, 2 pp. ISBN: 978-1-4673-8164-2. DOI: 10.1109/ESTIMedia.2015.7351761.

[Tan+15] A. Tanase, M. Witterauf, J. Teich, F. Hannig, and V. Lari. “On-Demand Fault-Tolerant Loop Processing on Massively Parallel Processor Arrays”. In: *Proceedings of the 26th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. (Toronto, Canada). IEEE, July 27–29, 2015, pp. 194–201. ISBN: 978-1-4799-1925-3. DOI: 10.1109/ASAP.2015.7245734.

C3: Compilation and Code Generation for Invasive Programs

Gregor Snelting, Jürgen Teich

Srinivas Boppu, 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, tightly-coupled processor arrays (TCPAs), and *i*-Core reconfigurable processors.

Figure 4.26 shows the structure of the developed compiler. The compiler is based on an existing X10 compiler, but has been extended with new transformation phases to support TCPAs as well as general purpose cores, such as SPARC/x86 processors as well as *i*-Cores, through libFIRM.

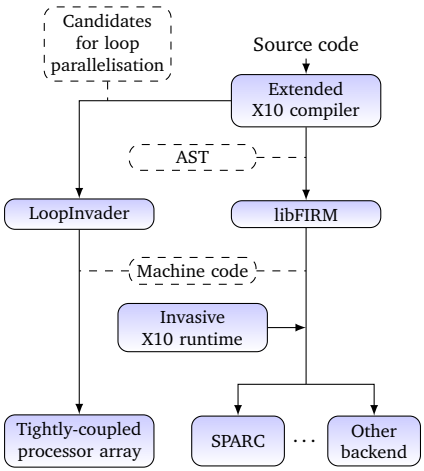


Figure 4.26: Compiler framework for invasive computing.

For TCPAs, we first investigated and developed a mathematical framework to symbolically tile invasive loop programs on multiple levels (hierarchically), which allows the compiler to generate code although the number of PEs to run on is unknown at compile time, while still balancing I/O bandwidth and memory requirements. Second, to counter the growing susceptibility of complex MPSoC designs to faults, we developed novel compilation methods that automatically transform a given loop program to utilise redundancy schemes such as dual (DMR) and triple (TMR) modular redundancy.

For general purpose cores, we stabilised the existing SPARC-based demonstrator platform and additionally extended the X10 compiler to generate code for x86-based standard hardware, which improves comparability of the invasive paradigm to other state of the art approaches. We reached and will maintain full feature parity between both versions. Furthermore, we investigated optimisations for invasive architectures on multiple levels: in the run-time system, at the interface from language run-time to the operating system, in the compiler intermediate representation, as well as during code generation. The research achievements on code generation for TCPAs have resulted in a successful PhD defence of Srinivas Boppu in December 2015 [Bop15].

The obtained results for the TCPA-related research topics will be explained next, followed by the results for general purpose cores.

Symbolic Multi-Level (Hierarchical) Tiling

Previously, we have described our breakthrough result that it is possible to determine a set of latency-minimal schedules for nested loop programs symbolically using either the *locally sequential, globally parallel* (LSGP) or the *locally parallel, globally sequential* (LPGS) scheme of tiling for parallelisation. Having *symbolic* schedules allows the number of PEs to execute on to remain unknown until run time, where a prologue selects the overall optimal schedule among all candidates according to the then known number of processors. This compile/run-time hybrid approach significantly reduces the run-time overhead compared to dynamic or just-in-time compilation.

C3

However, both LSGP and LPGS entail (different) disadvantages: Using LSGP, the necessary local memory within the PEs is proportional to the input size which prohibits the approach to be applicable in scenarios with limited data memory; using LPGS, the necessary I/O bandwidth is larger and might exceed existing I/O capacities. Consequently, both approaches might not have feasible mappings, for example, on a particular TCPA architecture. To alleviate such drawbacks, we were able to propose a symbolic multi-level parallelisation approach [TWTH15] that may balance necessary I/O-bandwidth and memory size to fit a given TCPA architecture.

First, the original iteration space of a given loop program is hierarchically tiled on h levels using h parametric tiling matrices $P^\ell = \mathrm{diag}(p_1^\ell, \ldots, p_n^\ell)$. This is demonstrated in Figure 4.27, where an iteration space of a 2-D loop nest of size $N \times T$ depicted in Figure 4.27(a) is hierarchically and symbolically tiled on $h = 2$ levels with $P^1 = \mathrm{diag}(2, 3)$

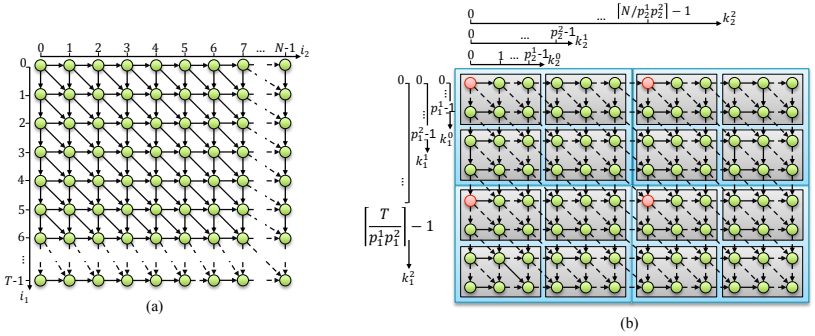


Figure 4.27: (a) Iteration space of a 2-D loop nest of size $T \times N$. Each node represents an iteration of the loop program. Each directed edge represents a data dependence between two iterations. (b) Iteration space hierarchically and symbolically tiled on two levels.

(size of the grey tiles) and $P^2 = \mathrm{diag}(2, 2)$ (size of the blue tiles), resulting in Figure 4.27(b).

Next, each iteration must be assigned a start time, i. e. be scheduled. Our approach uses a scheme that assigns start times to iterations within tiles that are scheduled sequentially on level 1—which correspond to the actual PEs—, and parallel start times to all tile origins on levels 2 to $h - 1$, and finally sequential start times again to the tile origins on level h. We analytically proved that it is possible to derive symbolic schedule vectors that describe the above parallelisation. As proposed in earlier findings, a prologue selects the latency-minimal schedule vector at run time that satisfies given resource constraints (such as I/O-bandwidth and memory size).

In 2016, we intend to complement our findings by software pipelining for symbolic scheduling to increase throughput.

C3

Fault Tolerance in Invasive Computing and Loop Processing

Since the feature sizes of silicon devices have continued to shrink, it is imperative to counter the increasing proneness to errors of modern, complex systems by applying appropriate fault tolerance measures. In [Wit+15; Tan+15; Lar+15a], we therefore proposed new techniques that leverage the advantages of invasive computing to implement fault tolerance on MPSoCs adaptively.

In particular, in [Wit+15] we demonstrated how invasive computing lets applications flexibly and dynamically adapt the level of fault tolerance according to external (e. g. radiation, workload) and internal (e. g.

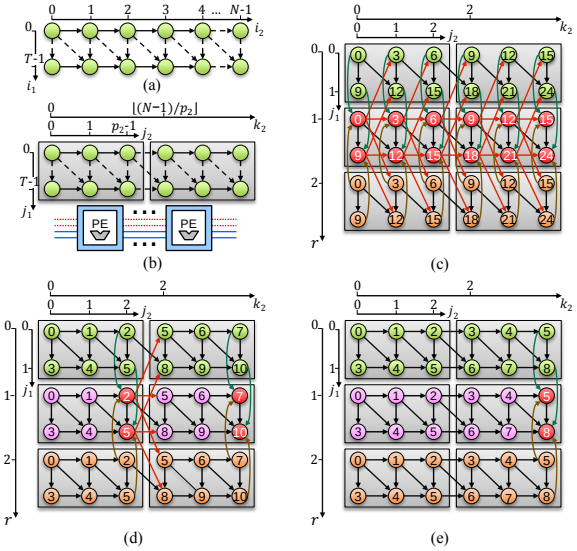


Figure 4.28: (a) Iteration space and data dependencies of an FIR filter. (b) Tiled iteration space. (c)–(e) Loop replication in the case of TMR for $T = 2$, $N = 6$, $p_2 = 3$ with (c) *immediate voting* (every iteration), (d) *early voting* (once per tile) and (e) *late voting* (once at the border). The start times of each iteration are also shown. The coloured edges represent the extra dependencies introduced by voting: red edges show the propagation of the voting results, green and brown edges show the propagation of results from the first and third replica to the second replica where voting on them takes place.

C3

security concerns) factors. Invasive computing furthermore supports the application programmer in all aspects of fault tolerance, but especially error and fault handling: it is, for example, possible to react to a fault by re-execution on the same resources, requesting a higher level of fault tolerance, or by executing on different resources.

In [Tan+15], we presented a compile-time transformation that introduces modular redundancy into a loop program to protect it against soft errors. Our approach uses the abundant number of processing elements (PEs) within a TCPA to claim not only one region of a processor array, but instead two (for *double modular redundancy* (DMR)) or three (for *triple modular redundancy* (TMR)) contiguous neighbouring regions of PEs. The transformation itself consists of two sub-transformations: The first one creates the desired level of redundancy by *loop replication*, and the second sub-transformation, *voting insertion*, appropriately inserts voting operations into the replicated loop program.

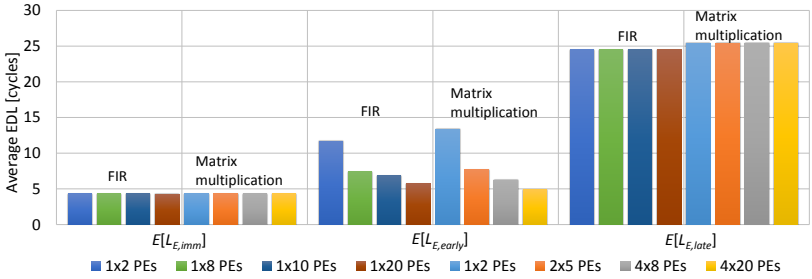


Figure 4.29: Average error detection latencies measured by functional analysis for the three proposed variants for voting insertion into the program applied to an FIR filter ($T = 1000$ samples and $N = 60$ taps) and a matrix-matrix multiplication application ($[10 \times 40][40 \times 20]$). The iteration space is kept constant while the coloured bars represent the different numbers of PEs that are used for execution.

For *voting insertion*, we proposed three different variants to detect, respectively correct errors: for each variable to be protected, (a) in every loop iteration (*immediate voting*), (b) once per PE (*early voting*), and (c) at the border of the allocated processor array region (*late voting*). Each of the variants exhibits a different tradeoff in terms of *latency* (time to finish computation) and *error detection latency* (time to detect a fault). These latency tradeoffs are easily seen in Figure 4.28 where the start time of each iteration is shown. As expected, immediate voting introduces the most amount of latency overhead while late voting introduces the least.

C3

Figure 4.29 shows a comparison of the average error detection latencies (EDL) for immediate voting ($E[L_{E,imm}]$), early voting ($E[L_{E,early}]$), and late voting ($E[L_{E,late}]$). The EDL grows larger from immediate to early and from early to late voting. It is noteworthy that for early voting, the EDL is inversely proportional to the number of PEs (when the iteration space size is left unchanged) because the tile size decreases.

Finally, in close collaboration with Project B2, we have combined software and hardware fault tolerant techniques in a co-design approach [Lar+15a] in which an individual safety level may be defined for each loop program as a requirement for execution. A desired safety level may be directly translated into an appropriate level of redundancy and voting insertion scheme according to a quantitative reliability analysis that we developed based on applications' timing characteristics (derived from our compilation flow) and observed soft error rates (SER) on a system [Lar+15b]. The core idea is to adapt the degree of fault protection for invasive loop programs to the required level of reliability at run

time; depending on application needs and observed SER on the array, the proposed redundancy schemes are dynamically and appropriately chosen and used at invasion time.

Optimisations for Invasive Architectures

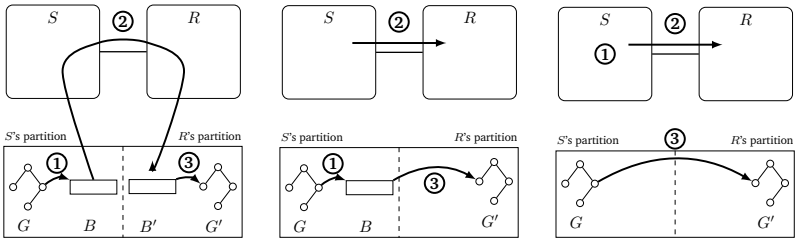
During the first funding phase, we developed a simple and efficient SSA-construction algorithm. Since then, we formally verified the correctness of our algorithm[15] using the theorem prover Isabelle/HOL. We also proved that the algorithm always constructs pruned SSA form, and minimal SSA form in case of reducible control flow graphs.

One advantage of our SSA construction 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 compilation. 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.

We further continued our work on permutation instructions from the first funding phase. In the first phase, we focused on showing that the concept itself is feasible: we implemented a hardware prototype, proposed an efficient heuristic for generating code, and conducted an extensive evaluation to determine the possible speedup for real-world programs. Recently, we worked on the theoretical foundations of the problems we solved in the first funding phase. The code generation approach we proposed in previous work consists of two steps: the conversion step to transform a given register transfer graph into a permutation of registers and a set of copy operations, and the decomposition step to implement the permutation with as few instructions as possible. We now

[15] S. A. Ullrich. “Verified Construction of SSA Form”. Bachelor thesis. IPD Snelting, Oct. 2013.



(a) Message passing. **(b)** Message passing via shared memory. **(c)** Cloning.

Figure 4.30: Schematic comparison of approaches to transfer an object graph G from sending tile S to receiving tile R. Temporary buffers are denoted by B, B'; and G' is the resulting copy of G.

proved [BMR15] that the proposed greedy decomposition algorithm is indeed optimal. Furthermore, we developed an efficient dynamic programming based algorithm for solving the complete problem and proved it to be optimal. We plan to implement this new algorithm and compare it to the heuristic.

In collaboration with Project C1 [Moh+15], we described how to avoid having a user-level scheduler as part of the X10 run-time and investigated the performance characteristics of our system. OctoPOS offers the abstraction of an *i*-let, a short snippet of code that is assumed to run to completion most of the time. Additionally, as OctoPOS is built around exclusive resource allocation, it does not support preemption, making it possible to efficiently create and schedule a large number of *i*-lets. This approach can be viewed as putting a user-level-like scheduler into the kernel. Hence, there is no need for a user-level scheduler in the X10 run-time system. Thus, we directly map each activity (`async` block) to exactly one *i*-let and leave all further scheduling decisions to OctoPOS. Additionally, we described how to implement inter-tile operations (`at`) using *i*-lets. We showed that the resulting system offers the common operations with a very low overhead: 1469 cycles to start an activity on the local tile, and 1981 cycles to start an activity on a remote tile. We plan to conduct the same performance analysis on the x86_64 version of OctoPOS.

Furthermore, we continued our work on accelerating inter-tile data transfers on non-cache-coherent tiled architectures. We extended existing techniques for transferring data via shared memory to non-contiguous data structures and called the resulting approach *cloning*. As

each at statement in an X10 program makes it necessary to transfer such a non-contiguous data structure (called object graph in this context), this is a worthwhile optimisation target. Figure 4.30 shows a schematic comparison of data transfer approaches. We implemented multiple strategies for the necessary cache operations (write back and invalidate) for cloning and performed a detailed evaluation of their performance on the Synopsys CHIPit Platinum. We found that for large complex object graphs, our technique offers a speedup of up to $2.3\times$ over the state of the art, and $3.4\times$ over the reference implementation.

Information Flow Control For Invasive Applications

G. Snelting's group develops JOANA, a tool for software security analysis (information flow control, IFC). JOANA is one of the few IFC tools worldwide that handles full Java with unlimited threads, and provides high precision (few false alarms). It is planned to use JOANA also for invasive X10 programs. An X10 front end is under development and will be integrated in 2016. Recently, a new *(i)RLSOD* algorithm was developed for JOANA, which discovers all probabilistic leaks, while still allowing secure parallel execution of "public" (so-called "low") statements [GS15]. (i)RLSOD requires a probabilistic scheduler, as well sequential consistency; the latter must be provided by the invasive memory model (see Project A1).

C3

Demonstrator Activities and Integration Work

In 2015, Project C3 continued the work on stabilising and performance tuning of the common hardware platform. In collaboration with members from Project C1, Project Z2, Project B1, Project B3, and Project B5, we created a problem report that highlighted some unresolved issues. As part of a concentrated effort, numerous problems could be solved or at least described more precisely, leading to new ideas for possible causes and greatly accelerating debugging efforts.

Additionally, we added the x86_64 bare metal version of OctoPOS as a new target for the X10 compiler, alongside the existing SPARC and Linux guest versions. OctoPOS provides the same interface on all supported platforms, thus the addition mainly required code generation support. Hence, we developed a new backend for libFIRM capable of generating 64-bit x86 code. As FIRM's optimisation passes are mainly architecture-independent, the new backend immediately profits from improvements made in the first funding phase. We will improve architecture-dependent parts, such as code selection, in 2016.

We reached a milestone in October 2015, when the invasive X10 multigrid application, which we also used during the project review in early 2014, ran successfully on a 48 core server machine provided by Project C1. This constitutes feature parity between the new x86_64 target and the existing SPARC and Linux targets. The x86_64 version of OctoPOS enables us to investigate the principles of invasive computing on standard hardware and compare it to other approaches on the same hardware platform. Additionally, we plan to show the benefit of having hardware support for invasion and resource guarantees by running the same invasive X10 programs on the invasive multi-tile MPSoC architecture (project area B) when ported on the new ProDesign FPGA prototyping platform (Project Z2) and on standard hardware. While the execution times will not be directly comparable, we hope to show an advantage of supporting invasive constructs on all levels, including the hardware, for other metrics, such as system efficiency. Hence, we will support both the SPARC and the x86_64 target in the X10 compiler.

Publications

[Bop15] S. Boppu. “Code Generation for Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, Dec. 18, 2015.

[Buc15] S. Buchwald. “Optgen: A Generator for Local Optimizations”. In: *Proceedings of the International Conference on Compiler Construction (CC)*. Ed. by B. Franke. Vol. 9031. Lecture Notes in Computer Science. Springer Berlin Heidelberg, Apr. 18, 2015, pp. 171–189. DOI: 0.1007/978-3-662-46663-6_9.

[BMR15] S. Buchwald, M. Mohr, and I. Rutter. “Optimal Shuffle Code with Permutation Instructions”. In: *Algorithms and Data Structures*. Ed. by F. Dehne, J.-R. Sack, and U. Stege. Vol. 9214. Lecture Notes in Computer Science. Springer International Publishing, Aug. 5, 2015, pp. 528–541. DOI: 10.1007/978-3-319-21840-3_44.

[GS15] D. Giffhorn and G. Snelting. “A new algorithm for low-deterministic security”. In: *International Journal of Information Security* 14.3 (2015), pp. 263–287. DOI: 10.1007/s10207-014-0257-6.

[Lar+15a] V. Lari, A. Tanase, J. Teich, M. Witterauf, F. Khosravi, F. Hannig, and B. Meyer. “A Co-Design Approach for Fault-Tolerant Loop Execution on Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware*

and Systems (AHS). Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231157.

[Lar+15b] V. Lari, J. Teich, A. Tanase, M. Witterauf, F. Khosravi, and B. Meyer. "Techniques for On-Demand Structural Redundancy for Massively Parallel Processor Arrays". In: *Journal of Systems Architecture (JSA)* 61.10 (Nov. 2015), pp. 615–627. ISSN: 1383-7621. DOI: http://dx.doi.org/10.1016/j.sysarc.2015.10.004.

[Moh+15] M. Mohr, S. Buchwald, A. Zwinkau, C. Erhardt, B. Oechslein, J. Schedel, and D. Lohmann. "Cutting Out the Middleman: OS-Level Support for X10 Activities". In: *Proceedings of the fifth ACM SIGPLAN X10 Workshop*. X10 '15. Portland, OR, USA: ACM, June 14, 2015, pp. 13–18. DOI: 10.1145/2771774.2771775.

[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.

[TWTH15] A. Tanase, M. Witterauf, J. Teich, and F. Hannig. "Symbolic Loop Parallelization for Balancing I/O and Memory Accesses on Processor Arrays". In: *Proceedings of the 13th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*. (Austin, TX, USA). IEEE, Sept. 21–23, 2015, pp. 188–197. ISBN: 978-1-5090-0237-5. DOI: 10.1109/MEMCOD.2015.7340486.

[Tan+15] A. Tanase, M. Witterauf, J. Teich, F. Hannig, and V. Lari. "On-Demand Fault-Tolerant Loop Processing on Massively Parallel Processor Arrays". In: *Proceedings of the 26th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. (Toronto, Canada). IEEE, July 27–29, 2015, pp. 194–201. ISBN: 978-1-4799-1925-3. DOI: 10.1109/ASAP.2015.7245734.

[Wit+15] M. Witterauf, A. Tanase, J. Teich, V. Lari, A. Zwinkau, and G. Snelting. "Adaptive Fault Tolerance through Invasive Computing". In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231155.

C5: Security in Invasive Computing Systems

Felix C. Freiling, Wolfgang Schröder-Preikschat,
Ingrid Verbauwhede (Mercator Fellow)

Ruan de Clercq, Gabor Drescher, Johannes Götzfried, Pieter Maene,
Tilo Müller, Alexander Würstlein

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. In 2015, we began to devise the attacker model and study security properties (especially isolation properties) as constraints for invasive computing (work package 1). We subsequently were able to make progress on achieving different levels of isolation at different architectural layers. The main focus was the hardware (i. e. isolating applications running on the same core, work package 4) and the systems software layer (isolating memory abstractions in collaboration with Project C1, work package 3).

Attacker Model

For the second funding phase, our attacker model considers software-only attacks, meaning that physical attacks are not considered. 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. These are shown in Fig. 4.31. Note that each level includes all levels below. The X10-attacker corresponds to an attacker who can run programs written in X10. These

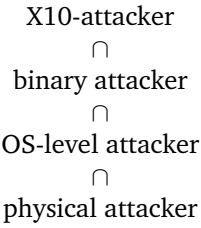


Figure 4.31: Hierarchy of attacker models.

programs can be statically checked through a trusted X10 compiler and so malicious behaviour is strongly inhibited, e. g. by the type system of X10. In contrast, the binary attacker may execute arbitrary (binary) code that runs with privileges associated with normal applications (usually user level privileges). With the OS-level attacker, we allow an attacker to take over control of the operating system. This is the strongest attacker we assume at the moment. As mentioned above, physical attacks are the most powerful ones and are currently not considered.

Isolation Concepts

The scientific objective of Project C5 is to embed different degrees of isolation into the invasive computing paradigm through all architectural layers, by means of dynamic software and hardware mechanisms.

C5

Basic Isolation Starting from the state of *no isolation* at the end of the first funding phase, Project C5 aims to establish a notion of *basic isolation* which ensures basic read/write protection for applications. 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 by employing dynamic memory protection.

Strong Isolation 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 or in the absence of a system layer, stronger degrees of isolation are required. Predictability of invasive programs in terms of correct execution must involve a *trusted computing base* (TCB) that guarantees the confidentiality and

integrity of a program's code and data. Confidentiality and integrity can both only be provided if there exists a trust anchor for applications. This can be assumed to exist in software, but to resist attackers who have gained system level privileges, a zero-software TCB must be developed on top of an immutable hardware trust anchor.

ϵ-Isolation TCB-based strong isolation concepts, however, may still allow information to leak from an application through side channels. Only the concept of *ϵ-isolation* additionally requests the absence of information flow up to a negligible amount ϵ.

> We define system S to satisfy ϵ-isolation for attacker A and environment E iff there exists evidence of attacks on S by A in E that lead to unauthorised information leakage of at most ϵ bits per second (bps).

Intuitively, the definition can be rephrased as follows: *S satisfies ϵ-isolation iff the best published attack on S achieves ϵ bps.* This definition takes the "state of the art" protection against side channels into account, it is therefore relative to the knowledge of existing attacks on S. Therefore, a system that satisfies ϵ-isolation for $\epsilon = 5$ bps today might not satisfy this property anymore if a new attack on S becomes known that achieves an information leakage of 50 bps. If this is the best possible attack today, then the system satisfies ϵ-isolation with $\epsilon = 50$ bps. Theoretically, if nobody has attacked the system yet, it satisfies ϵ-isolation for $\epsilon = 0$ bps. This shows that ϵ-isolation is only meaningful if many people have tried very hard to attack S and have published their results. Our definition therefore corresponds to security definitions of cryptographic primitives that also depend on how well-researched a primitive is.

Results

We have recently investigated variants of the strong isolation concept in different contexts. First, we studied the problem of memory isolation with respect to data exposure caused by insecure deallocation in common memory management schemes [AFGM15]. 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. In 2016, we plan to investigate how these declarative

approaches can be used at the X10 programming language level to prevent information leaks in invasive programming.

Second, 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 [KM15]. 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 unreachable for overflows, many exploits are stopped at an early stage of progression with only little performance overhead.

Third, we presented a bytecode interpreter that isolates code and data of programs against memory attacks by executing them without using RAM for any sensitive content [SGM15]. The memory contents of a program executed usually rests unprotected in RAM. An attacker can exploit this fact to obtain information about both, programs running on the target system, and the data they are operating on. As a countermeasure, we provide a Turing complete execution environment running on x86 commodity hardware, which allows program execution to treat RAM as untrusted. To this end, we use a bytecode interpreter executing programs without directly using RAM for code and data. This interpreter stores its state in CPU registers and uses RAM only to store encrypted data.

Fourth, we developed a system for offline software protection within low-cost embedded devices called *Soteria* [Göt+15]. Protecting the intellectual property of software that is distributed to third-party devices which are not under full control of the software author is difficult to achieve today. With Soteria, we are able to protect software by considering a small part of the hardware as trusted. At its heart, Soteria is a program-counter based memory access control extension for the TI MSP430 microprocessor. Based on our open implementation as an openMSP430 extension, and our FPGA-based evaluation, we have shown that Soteria has a minimal performance, size and cost overhead while effectively protecting the confidentiality and integrity of an application's code against all kinds of software attacks.

C5

Fifth, we analysed one of the fundamental building blocks that will allow us to integrate encryption-based isolation into the SPARC LEON3 core in the future, namely single-cycle implementations of block ciphers [MV15]. This work compares unrolled combinatorial hardware implementations of six lightweight block ciphers, along with an AES implementation as a baseline. The majority of such ciphers were designed for area-constrained environments where speed is often not crucial, but

single-cycle, low latency block ciphers with limited area requirements will be needed for the invasive computing architecture.

Outlook

For 2016, we plan to implement operating system support for isolating applications such that they can transparently work on encrypted data. In every enabled application, data will only be stored in cleartext for the moment it is processed, and otherwise stays encrypted in RAM. In particular, the required encryption keys will not reside in RAM, but will be stored in CPU registers.

However, isolating applications from each other by means of encryption is costly in terms of performance and can only serve as a prototype implementation. Hence, we will also present a hardware-based isolation mechanism that relies on transparent memory encryption. In contrast to existing solutions, this mechanism will scale to an unlimited number of applications, will be able to handle heterogeneous memory and will have a zero-software TCB. Securely shared memory between multiple applications will be supported to serve as a secure communication channel. This solution is planned to be implemented in the SPARC LEON3 processor and to be evaluated on FPGA.

Finally, we are planning to identify the requirements of a trusted computing base for heterogeneous multicore systems with a top-down approach in mind. While our previous solutions have been focused on the building blocks necessary for zero-software TCBs in single cores, such as the SPARC LEON3, we want to leverage those building blocks to support trusted computing at the tile level and the NoC level of invasive multi-tile computing architectures in the future. This will lead to the development of new security mechanisms for such architectures.

Publications

[AFGM15] 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: Elsevier, 2015, pp. 81–86. DOI: 10.1016/j.jisa.2014.10.001.

[Göt+15] J. Götzfried, T. Müller, R. de Clercq, P. Maene, F. Freiling, and I. Verbauwhede. “Soteria: Offline Software Protection within Low-cost Embedded Devices”. In: *Proceedings of the 31th Annual Computer Security Applications Conference (ACSAC’15)*. Los Angeles, California, USA: ACM, 2015, pp. 241–250.

[KM15] C. Kugler and T. Müller. “Separated Control and Data Stacks to Mitigate Buffer Overflow Exploits”. In: *Endorsed Transactions on Security and Safety*. European Alliance for Innovation (EAI), Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (ICST), 2015, pp. 1–36.

[MV15] P. Maene and I. Verbauwhede. “Single-Cycle Implementations of Block Ciphers”. In: *Lightweight Cryptography for Security and Privacy*. Lecture Notes in Computer Science. Springer-Verlag, 2015.

[SGM15] M. Seitzer, M. Gruhn, and T. Müller. “A Bytecode Interpreter for Secure Program Execution in Untrusted Main Memory”. In: *20th European Symposium on Research in Computer Security (ESORICS'15)*. Vienna, Austria: SBA Research, 2015, pp. 376–395.

D1: Invasive Software–Hardware Architectures for Robotics

Tamim Asfour, Walter Stechele

Manfred Kröhnert, Johny Paul, Dirk Gabriel

The main research topic of Project D1 is the exploration of benefits and limitations of invasive computing in humanoid robotics. Our goal is to demonstrate invasion mechanisms and negotiation for resources in the context of complex robotic scenarios with concurrent processes and timely varying resource demands. 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 key to efficient allocation of available resources in situations where multiple applications bargain for the same resources.

In 2015, we focused on profiling real-world robotics applications in order to extract resource usage models from the gathered data, which should serve as basis for the development of speculative resource management techniques. In particular, we investigated typical application chains in the context of motion planning and vision-based execution.

Profiling of Robotic Applications

Initial work on prediction of robot actions influenced by environmental parameters was presented in [Krö+14]. However, the prediction of future resource utilisation was based on an artificial resource models trained with synthetic data. Thus, one goal was to generate resource models based on real sensorimotor data acquired during the actual execution of robot tasks. To achieve this, the robot framework ArmarX ([Vah+15]) was enhanced with monitor capabilities for extracting CPU and memory usage of running ArmarX applications. Figure 4.32 shows an example of such sensorimotor robot data. It is acquired online during robot operation and used to build the resource usage models. As CPU and memory usage depend on the complexity of the current scene and the current state of each robot application, the models are therefore

context-dependent and implicitly represent the dynamics of the current scenario.

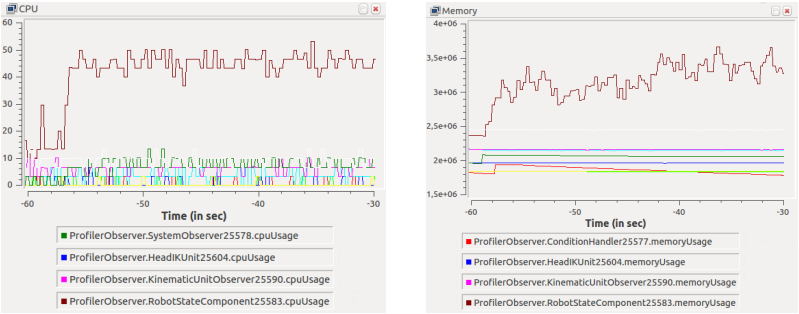


Figure 4.32: Screenshots of CPU (left) and memory (right) usage of a running ArmarX robotics application. A change in execution leads to higher CPU and memory usage of the RobotStateComponent which provides the internal representation of the robot to the applications responsible for solving the forward and inverse kinematics problem.

Resource-Aware Motion Planning

Robots employ motion planning techniques for generating collision-free movements in a given environment. A wide variety of motion planning algorithms exist in the literature, including several parallel implementations. However, a fixed amount of processing resources is usually assigned to these algorithms which can lead to resource underutilisation in the case of easy solvable motion planning problems. To this end, we investigated how resource-aware concepts can be used, and developed a resource-aware motion planner within the ArmarX framework [KGVA16], which adapts the amount of allocated resources based on the difficulty of the underlying planning problem. We evaluated the new planner in tasks similar to the one shown in Figure 4.33, where the humanoid robot ARMAR-4 has to remove a bottle from a crate. The difficulty of the problem arises from the narrow free passages of the crate, which leaves little free space for collision free movements.

The major design goal of the resource-aware, distributed, and parallel motion planner was to provide fast planning while keeping resource utilisation high. To achieve this, the planner starts with a minimum of computational resources and acquires more resources based on the estimated planning problem difficulty. This estimation is performed by different strategies allocating new resources over time (ΔT) or ad-

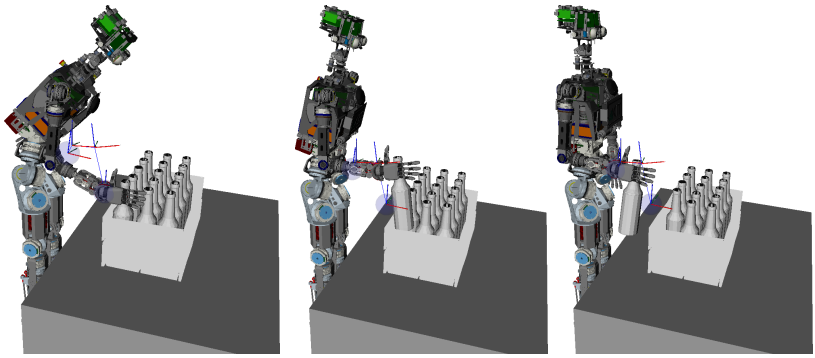


Figure 4.33: ARMAR-4 removing a bottle from a crate. The motion was generated by the resource-aware motion planner.

ditionally taking failed planning attempts into account (NNΔT). The evaluation of different resource acquisition strategies in comparison to static resource allocation is shown in the graphs in Figure 4.34 (lower numbers are better). Additionally, an example of the *Serial Walls* planning problem used for evaluation is shown on the left of Figure 4.34. Here, the green box must be moved from left to right through the holes in the walls and the planning problem difficulty increases with the number of walls present.

The blue and green bars in the middle of Figure 4.34 show, that the total time to solve the problem is slightly higher for all resource allocation strategies as compared to static resource allocation with more than one planning thread. However, the efficiency or resource utilisation is better when compared to static allocation (right graph in Figure 4.34). These results clearly show that adding concepts from invasive computing to existing software architectures can be very beneficial.

D1

Application Chains

In addition to motion planning, the recognition of objects in the environment is a major task for robotic systems. Therefore, ARMAR-III uses an object recognition application consisting of three stages. These stages are Harris Corner Detection, SIFT (Scalable Invariant Feature Transform) feature extraction and SIFT feature matching. A resource-aware model allows Harris Corner Detection and SIFT feature matching to express their resource requirements to the run-time system and split their workload based on the granted resources. Since it can occur that

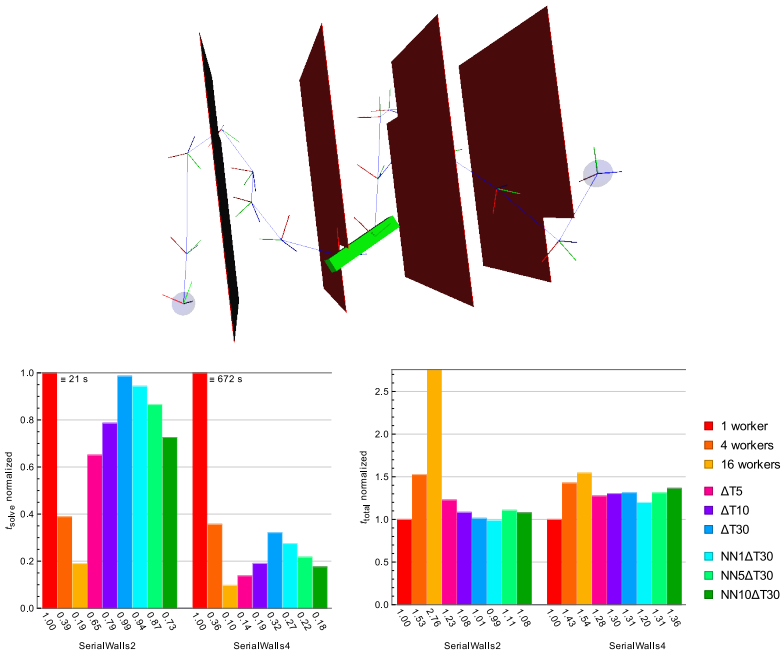


Figure 4.34: Evaluation scenario SerialWalls4, where the green box must be moved through the holes in the brown walls from left to right (upper figure). The normalised execution times t_{solve} and efficiency values t_{total} of the planning algorithm is shown is shown in the left and right graph, respectively.

not enough resources are available to complete calculation within given time limits, the Harris Corner Detection has been modified as presented in [Pau+14]. Controlled by a threshold value, the application stage can prune away non-corner like pixels and hence reduce the calculation effort and execution time at the expense of quality.

Reducing the quality of a single stage influences the results and resource demands of the following stages and therefore of the overall application. Consequently, this influence must be considered while parametrising one stage. In order to take this into account, we have planned to build up an control loop for the application chain to determine all stage parameters.

To demonstrate the correct behaviour of the control loop, the application has to be integrated on a platform capable of executing it with a reasonable frame rate. Furthermore, multiple applications must be

executed on the platform to cause situations where different applications bargain for the available resources and need to adapt. Therefore, the object recognition and two further applications for detecting single-coloured objects and calculating a depth map are currently ported to a newly available x64-machine, which supports a natively running OctoPOS.

In order to realise the behavioural model used by the control loop, certain information must be collected without influencing the application's behaviour. Thus, in 2015, a module supporting Intel Hardware-Performance-Counters has been integrated in OctoPOS in cooperation with Project C1. It provides detailed information about timing and hardware usage to the application and an external monitor connected to the machine via Ethernet.

Outlook

In the next year, we plan to extend the presented work by generating specific resource models based on execution phases of robot applications. To determine concrete numbers of the application chains, they will be integrated into the design space exploration tool developed by Project A1 as hardware-in-the-loop. Afterwards, the models will be evaluated and combined with the existing prediction mechanism in order to provide resource predictions for robot applications based on realistic data sets. Furthermore, we plan to consider and evaluate enhanced prediction models and compare them to the quality of the existing approach.

Publications

[KGVA16] M. Kröhnert, R. Grimm, N. Vahrenkamp, and T. Asfour. "Resource-Aware Motion Planning". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2016, submitted.

[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+14] 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+15] J. Paul, W. Stechele, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, W. Schröder-Preikschat, M. Kröhnert, T. Asfour, Éricles R. Sousa, V. Lari, F. Hannig, J. Teich, A. Grudnitsky, L. Bauer, and J. Henkel. "Resource Awareness on Heterogeneous MPSoCs for Image Processing". In: *Journal of Systems Architecture* 61.10 (Nov. 6, 2015), pp. 668–680. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.09.002.

[Vah+15] N. Vahrenkamp, M. Wächter, M. Kröhnert, K. Welke, and T. Asfour. "The Robot Software Framework ArmarX". In: *Information Technology* 57.2 (2015), pp. 99–111.

D3: Invasion for High-Performance Computing

Hans-Joachim Bungartz, Michael Gerndt, Michael Bader

Isaías Comprés, Andreas Hollmann, Ao 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[16,17] on standard HPC architectures [LGG16]. Our goal in Phase II is three-fold: firstly, to develop an MPI extension to support the invasive computing concepts on general distributed-memory systems; secondly, the development of high-performance applications for invasion on distributed-memory systems; last but not least, to develop a resource manager, which distributes resources fulfilling system requirements (such as optimal system throughput, optimal energy efficiency, etc.) as well as requirements of different types/classes of applications.

[16] M. Bader et al. "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. DOI: 10.2316/P.2011.719-070.

[17] M. Gerndt et al. "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.

Development & Progress

1. Demonstrator Platform: X10 Applications

In the area of MPSoCs, we continue our efforts on the development of different applications in X10 that could be run on the invasive multi-tile hardware demonstrator platform. These applications should be of practical value and be able to showcase the benefits of invasive computing. Our goal for the current funding phase is to demonstrate on the InvasIC hardware platform design features such as reliability, predictability, flexibility, security, among others. This is our main motivation for choice of application development:

- A computational fluid simulation on arbitrary 2-D geometries is chosen for development. This is another classical grid-based simulation application, besides the heat simulation developed in Phase I. This application has an increased complexity and greater resource requirements compared to the previously considered heat simulation. The plain version has been implemented in X10. An extension with multigrid solver and invasive features is under development.

- A prototype of a malleable simulation framework utilising the sparse grid combination technique is under investigation. To fully demonstrate the flexibility of invasive computing, malleable applications are highly desirable. Experiments are being carried out on simulations with finite difference methods (FDM), since FDM is proven to be suitable for computations on sparse grids.

- Two fundamental linear algebraic algorithms with different cache optimisation approaches were developed. This was done in collaborative research with Project B1 on *adaptive caches*. Experiments were carried out with different hardware cache configurations. We strive for finding the optimal software/hardware co-design for cache efficiency to achieve high performance on the InvasIC hardware. Two papers are evolving from this project, one of which primarily focused on the algorithmic part and the other on the hardware part.

2. Invasion for HPC

A complete update to the software stack of a computing system is required to enable invasive computing. In the case of HPC systems, this means that the Resource Manager (RM), the Distributed Runtime

Environment (DRTE) and the applications need to be updated. The Resource Manager is the component that allows users to submit work in batch files. The work is then organised in queues and a scheduler tries to make the best use of available resources while performing the required work. The DRTE is in our case MPI centric; it consists of the MPI library and supporting infrastructure for process startup, IO redirection, etc. In the case of invasive computing, the DRTE must also allow for resource adaptations. Finally, applications need to be modified to make use of the resource-aware features supported by the invasive software stack. In the following sections, we describe what has been achieved so far, as we approach our first full integrated prototype for distributed memory HPC systems.

Resource Management

A three tier design has been established and is partially implemented, as shown in Figure 4.35. The implementation is based on SLURM, the currently most widely used open-source resource manager. The core design of the centralised controller and distributed network of daemons is preserved. The controller and daemons have been extended to support resource adaptations. As part of the InvasIC Resource Manager (iRM) design, a new intermediate component has been integrated: the Invasive Runtime Scheduler (iRS). The iRS sits between the network of daemons and the centralised controller. This new arrangement results in a three-tier software architecture with a clear separation of concerns:

- Controller: Handles incoming user jobs, manages queues and performs batch scheduling. Handles partitions dedicated to static jobs directly. Forwards adaptive jobs to the Invasive Runtime Scheduler through a negotiation protocol. This component will contain a resource-aware batch scheduling algorithm that is now in active development.

- Invasive Runtime Scheduler (iRS): Accepts adaptive jobs from the Controller through a negotiation protocol. Performs accepted jobs in parallel. Maximises application and system wide metrics by controlling running applications' mappings. Performs changes to the mappings as a continuous process, dictated by its run-time scheduling algorithm that is in active development.

- Node Daemons: Performs instructions dictated by the iRS. These include: forking individual MPI processes, setting up local MPI communicator data, redirection of `STDOUT`, `STDERR` and `STDIN` to

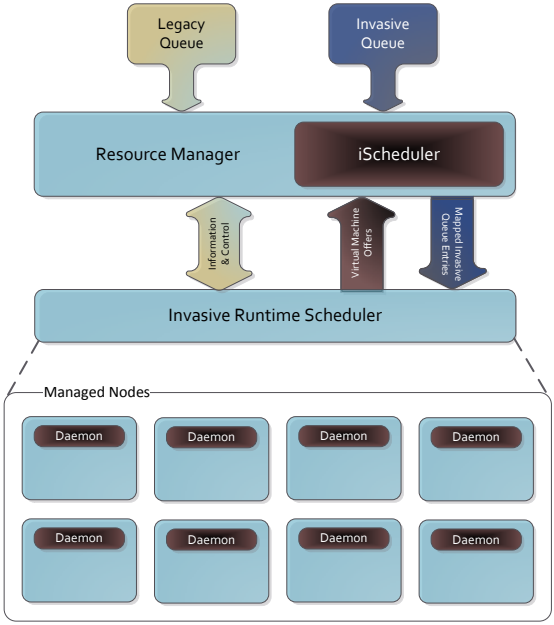


Figure 4.35: Invasive Resource Manager three-tier architecture.

the client node (where `mpiexec` resides). It also tracks its local child processes and reports errors and other events. It is also responsible of tracking performance and energy metrics of the node.

D3

Invasive MPI

Our current Distributed Runtime Environment (DRTE) is MPI-centric. Our extended MPI library is the component that gets linked into the binaries of our invasive HPC applications. The DRTE is composed of the MPI library, node daemons and the Invasive Runtime Scheduler (iRS). The MPI library provides all the new API calls that allow for the development of invasive HPC applications using MPI. The API extension has been finalised and consists of 4 new operations:

- `MPI_Init_adapt`: Initialises the library in adaptive mode.
- `MPI_Probe_adapt`: Probes for resource adaptations.

- `MPI_Comm_adapt_begin`: In case of an adaptation, begins the repartitioning window.
- `MPI_Comm_adapt_commit`: Signals the completion of the repartitioning window.

The library is currently being actively developed. The main design goals are to hide the latencies involved during adaptations and to have nearly free probe operations. The creation of new processes in new allocations can be several seconds and it is therefore not desirable to block running processes during these phases. The probe operation should be performed very frequently to minimise the time new resources remain idle; therefore, the latency of this operations must be kept as low as possible.

Invasive HPC Applications

The following applications are being developed alongside the Invasive Resource Manager (iRM) and the Invasive MPI (iMPI) library:

- **Statistical Inverse Problem – Locating Obstacles in Fluid Channel:**

  This is our pilot application for the Invasive HPC framework. This application takes as an input a set of fluid velocity measurements from a few sensors placed across a channel, and produces a probability density function (PDF) of the locations of multiple obstacles inside the channel. This PDF can then help us to find out point coordinates or regions at which the obstacles are most likely located.

  Solving this problem involves running many forward simulations with different plausible guesses of obstacle locations. In order to reduce computational costs and run time, we employ an adaptive surrogate model instead of the full simulation model for the forward simulations. This adaptive surrogate model is actively refined and updated during the computation of the PDF, allowing for increasing accuracy. Such a dynamic behaviour requires changes in computational resources and fits perfectly in the Invasive HPC framework. Better yet, this algorithm is embarrassingly parallel due to its nature, meaning that the bulk computations by parallel processes are independent to one another.

  The implementation of this application and its integration with iMPI are completed. However, run-time measures and performance analysis are yet to be done.

- **2D Tsunami Simulation:**

  As compared to the malleable pilot application, the tsunami simulation is quite rigid in terms of data distribution. This simulation is based on adaptive triangular grids. Therefore, it also requires a change of resources during run time and thus fits well to the Invasive HPC framework. However, every time step it advances, exchange of data between neighbouring processes is required, i. e. parallel processes have dependencies upon one another. Moreover, every time the application adapts to a new set of resources, grid cells have to be re-distributed among processes, which generates a lot of communication. Therefore, the integration of grid-based simulation applications with the Invasive HPC framework is much more complicated than that of malleable applications.

  The normal parallelised version of the 2D tsunami simulation with MPI already exists. However, the integration with iMPI requires a structural re-design of the application. This is because in the middle of execution, newly joined processes need to reach the same state of the existing processes before they can start computing, i. e. they have to go through all the code from the very beginning in order to obtain the same computation variables and function call stacks. To accelerate this process as much as possible, a re-design of the application with an explicit resource adaption routine in the main compute loop is necessary.

  Re-structuring of the application and integration with iMPI is currently in progress.

Outlook

In addition to completing the above mentioned ongoing tasks, our next steps include:

- **X10 Development:** implement the malleable simulation framework, and extend it to integrate more simulations with different discretisation schemes, i. e. finite difference, finite element and finite volume methods.

- **X10 Development:** contribute to the topic of “Dark Silicon” (Project B3) and make features accessible to scientific algorithms. Provide 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.

- **HPC Development:** improve and extend the resource manager with *intelligence*, i. e. the resource manager is pre-trained with certain features such as application type, application priority, energy level, among others, such that it can make better decisions in shorter time.
- **HPC Development:** implement an invasive 3-D weather/climate simulation prototype, which resembles typical large-scale HPC applications with requirements of dynamic changes of computational resources at run time.
- **HPC Development:** develop other classes of applications. Currently, a malleable grid-based simulation framework is intended and under investigation.

Publications

[BHRB15] A. Breuer, A. Heinecke, L. Rannabauer, and M. Bader. "High-Order ADER-DG Minimizes Energy- and Time-to-Solution of SeisSol". In: *High Performance Computing*. Ed. by J. M. Kunkel and T. Ludwig. Vol. 9137. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 340–357. ISBN: 978-3-319-20118-4. DOI: 10.1007/978-3-319-20119-1_25.

[LGG16] W. Liu, M. Gerndt, and B. Gong. "Model-based MPI-IO tuning with Periscope tuning framework". In: *Concurrency and Computation: Practice and Experience* 28.1 (2016), pp. 3–20. ISSN: 1532-0634. DOI: 10.1002/cpe.3603.

[PZZB15] B. Peherstorfer, S. Zimmer, C. Zenger, and H.-J. Bungartz. "A Multigrid Method for Adaptive Sparse Grids". In: *SIAM Journal on Scientific Computing* 37.5 (2015), S51–S70. DOI: 10.1137/140974985.

[Unt+15] K. Unterweger, R. Wittmann, P. Neumann, T. Weinzierl, and H.-J. Bungartz. "Integration of FULLSWOF2D and PeanoClaw: Adaptivity and Local Time-Stepping for Complex Overland Flows". In: *Recent Trends in Computational Engineering - CE2014*. Ed. by M. Mehl, M. Bischoff, and M. Schäfer. Vol. 105. Lecture Notes in Computational Science and Engineering. Springer International Publishing, 2015, pp. 181–195. ISBN: 978-3-319-22996-6. DOI: 10.1007/978-3-319-22997-3_11.

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

Marcel Brand, Stephanie Friederich, Leonard Masing, David May

The major goal of Project Z2 is to provide a common demonstrator environment for validating and demonstrating the principles of invasive computing. To accommodate an invasive MPSoC architecture with a multitude of processor cores, this demonstrator environment is realised on a prototyping platform containing several FPGAs. Currently, Project Z2 employs a Synopsys CHIPit Platinum platform featuring six Virtex-5 LX330 FPGAs. From 2015 on, a ProDesign proFPGA containing four Virtex-7 2000T is available to scale the prototyped architecture further.

To show the advantages of invasive computing contributions of the different projects across all project areas, the developed concepts at the hardware as well as compile and application level are integrated into the common demonstrator platform. Whereas in the first funding phase, the focus was mainly on improved quality of service, resource utilisation, and speed-up of individual applications, the major goal of the second phase is showing the benefits of invasive computing with regard to increased predictability of execution times, energy efficiency, and security.

In cooperation with all projects and the working groups of the CRC, our tiled invasive multicore architectures are continuously expanded and prototyped for demonstration of invasive applications from project area D.

An example configuration of an invasive MPSoC architecture as shown in Figure 4.36 consists of the following components: Several computing tiles, each with a set of of RISC processors (using LEON3 processor cores or the *i*-Core as developed in 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 hardware projects of project area B are connected via the *i*NoC. The operating system OctoPOS and the run-time support layer *i*RTSS (Proj-

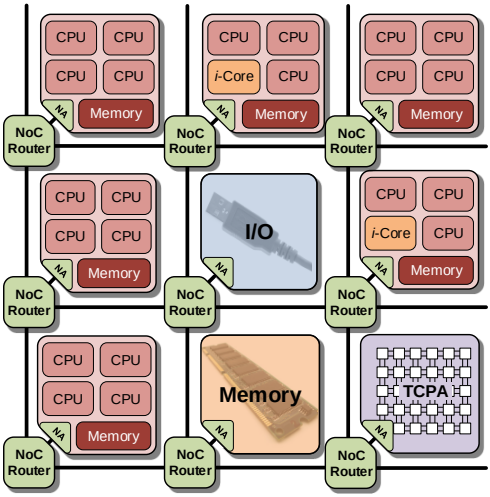


Figure 4.36: A 3x3 InvasIC design consisting of seven compute tiles, as well as one memory and one I/O tile.

ect 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. The same architecture is currently in the process of being ported to the new proFPGA prototyping platform which is intended to run scaled up designs compared to the CHIPit.

New Demonstrator Platform

One of the main tasks of Project Z2 in 2015 was the careful evaluation of potential new demonstrator systems and—after the acquisition of the ProDesign proFPGA[18] prototyping system—its upbringing. It will replace the CHIPit system as the main demonstrator platform at each affiliated site (FAU, KIT, TUM).

The proFPGA system consists of a housing (see Figure 4.37(a)), similar to that of a desktop PC, which has the space to harbour the acquired components: a ProDesign quad Motherboard, a fan and the power-supply unit. The motherboard contains four Virtex-7 2000T FPGAs (see Figure 4.37(b)) and additionally provides 32 connectors for extension

[18] http://www.prodesign-europe.com/profpga

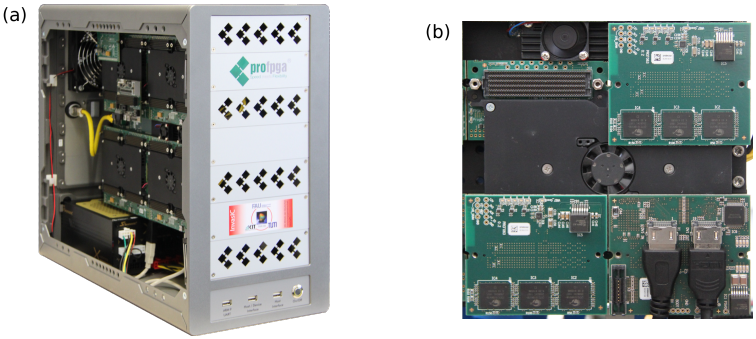


Figure 4.37: (a) On the left side, the proFPGA is shown. The open case shows the ProDesign quad Motherboard equipped with 4 Virtex-7 2000T FPGAs.
(b) On the right, there is a detailed view of one Virtex-7 2000T sitting on the motherboard. Also visible are one empty connector and three extension boards. Two SSRAM extension boards are shown with the 3 memory banks being visible on each. On the bottom right the DVI extension board is shown.

boards, eight per FPGA. Combined with the hardware, ProDesign provides the software for accessing and configuring the system and the FPGAs. Access to the system is possible via Ethernet, USB, and PCIe. The provided software can be used for loading bitstreams to and setting up all required voltages and clock networks in the system. Furthermore, a so-called mmi64 interface can be used to create custom designs, such as transactors or debugging interfaces, that use the interface to the motherboard and connect directly to the FPGAs. Each system is complemented with various extension boards containing a DDR3 DRAM memory, a DVI transmitter and receiver, an Ethernet interface and eight SSRAM extension boards, specifically designed and assembled for the CRC. An invasive tiled design uses tile local memory (TLM) in every tile, which using the SSRAM extension boards can now be implemented on external chips, because the available on-chip memories(BRAMs) are limited and are mainly used for L1 and L2 caches. The overall proFPGA system is displayed in Figure 4.37 on the left side whereas on the right side a detailed view of one Virtex-7 2000T is shown. It is equipped with a DVI and two SSRAM extension boards. An SSRAM extension board consists of three banks with each 8MB static memory that can be accessed separately.
Before every project can start working with the proFPGA, the InvasIC architecture has to be ported to it. For this, Project Z2 has been in charge of the following preparatory work packages in 2015:

- The careful testing of the correct functionality and programmability of the new prototyping systems as well as the memory and DVI extension boards by developing and running of predesigned and customised tests.

- The provision and testing of the new toolflow for the proFPGA system based on Xilinx Vivado[19]. The previously used Xilinx ISE Design Suite does not support the synthesis and implementation of the novel Virtex-7 2000T which are used in the proFPGA platform.

- Redesign and implementation of the controllers for DRAM, SSRAM, DVI, and Ethernet to be compliant to the InvasIC architecture and the proFPGA system. The DRAM and SSRAM controller are functional and are currently in the state of implementation into a combined design. The integration of the DVI and Ethernet controller is currently under development. The DVI controller will support reading in DVI input from, e. g. a PC, and writing DVI output to an output device like an LCD monitor. The implementation that will be provided by Project Z2 will support single or double buffering schemes depending on frame size and frame rate. For buffering, one of the additional SSRAM banks that are not used in the multi-tile design will be utilised.

- Integrate the transactor hardware design, a custom component provided by ProDesign for the project, into the design of a multi-tile invasive MPSoC hardware architecture and provide appropriate driver support on the debug host. The transactor is connected to the AMBA bus as a master module and provides an interface between a host and the hardware. In case of the proFPGA system, a host can connect to it by using the mmi64 interface. The transactor then enables a host system, e. g. to configure other modules or read and write from and to memory. This is useful for running programs on connected LEON3 processor cores and for debugging purposes.

- Port templates of multi-tile invasive reference architecture configurations to the proFPGA. The increased size of the Virtex-7 2000T shall offer to accommodate a factor 4 larger design than that

[19] http://www.xilinx.com/products/design-tools/vivado.html

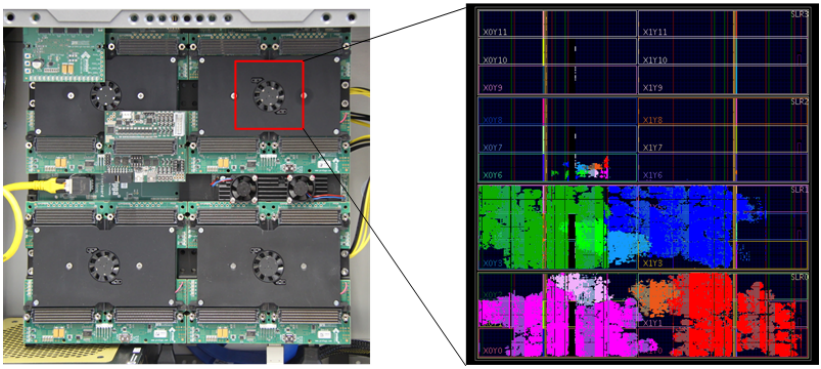


Figure 4.38: The proFPGA motherboard containing four Virtex-7 2000T is shown to the left. A mapping of a 2x2 tile design onto one of the FPGAs using the new toolflow based on Vivado is illustrated on the right.

supported currently by the CHIPit system. The port of the architecture depends on the finalisation of the mentioned controllers, which—at the time of writing—is still work in progress.

Project Support

In addition to the above work packages related to the migration to the new prototyping system, Project Z2 continued its contribution to the realisation of the overall CRC demonstrator platform. In the last year, a significant effort was also spent on the elimination of final bugs that were still remaining in the overall demonstrator. In joint effort with all scientific projects, Project Z2 helped to trace down and identify the reasons of several temporary malfunctions either in modules provided by individual projects or in the interplay with generic components of the LEON3-based multicore architecture.

Further work items covered by Project Z2 were:

- Update to a new GRLIB version version, which is necessary with the transition to the new demonstrator platform, since Virtex7 FPGAs are not supported by the old GRLIB version which has been used so far. Project Z2 initiated the update to Version 1.4.3 and managed the required adaptations to the custom designs of the involved projects in project area B.

- The repository for the Project Z2 was revisited and ported to a GIT repository. This shall result in a better version management

structure combining the individual projects and a more flexible work flow by utilising GIT's superior branching functionality. Furthermore, GIT tends to be much faster for large projects. Its decentralised approach enables the developers to commit changes and switch between branches even without having access to the projects server.

Outlook

Project Z2 will continue providing support for demonstrating the benefits of invasive computing across all involved layers, from algorithmic concepts, software, and invasive hardware. Towards this end, Project Z2 will establish the generic infrastructure as required for verification and evaluation of invasive multi-tile architectures by supporting the integration of the various HW/SW components from different projects.

A major focus for 2016 is supporting all projects of the CRC to successfully port their designs to the new demonstration platform. This is of great importance to support in a timely manner the evaluation of the benefits of invasive computing, the achievable predictability of security, reliability, and timing of parallel applications by isolation of resources. After finalising the designs for the new components and the port to the new Gaisler library, a fully integrated design shall become operational soon. The integration of all components into a design which is at the same level as the CHIPit demonstrator is envisioned to be available already in spring 2016.

5 Working Groups

WG1: Predictability

Coordinators: Michael Gerndt, Michael Glaß

Focal points of interdisciplinary investigation in this working group are all questions around the current lead topic of the CRC: The *predictability* of non-functional aspects of parallel computation. The term **-predictability* was defined and used to express a current deficiency of today's multicore systems and parallel applications to provide a bounded (guaranteed) quality of the their execution—not only w. r. t. execution time, but also to security, reliability and/or power consumption. 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. Starting already in 2014, the following initial goals for this WG could be achieved in 2015:

- Identification of the most *relevant topics* concerning **-predictability* within this CRC and the assignment of topic chairs that coordinate individual topics.
- Creation of a *glossary* of all relevant predictability terms to enhance the common understanding of predictability concepts in this CRC.
- Providing 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.

As a constant goal, this WG is active in the organisation of *workshops* to 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.

In a physical meeting in January 2015 at FAU, implications of the aspect of security and reliability on the WG's topics have been discussed in great detail. A major outcome of the discussion is that not only *requirements* specified by the designer, but also *assumptions* on the application, the system, and the environment—for example the attacker model or the radiation rate—have to be considered. In follow-up physical meeting in May 2015 at TUM, the initially determined topics have been reconsidered in detail and recognised ambivalences were resolved. In this context, the need for a formal definition of the topics and their interplay has been raised and set as a goal of this WG. Also, concrete questions and examples on each individual topic with involved projects and their models and tools have been developed and documented to guide future collaborations.

The most important working meeting of WG1 took place as part of the doctoral-researcher course "Benchmarking for Multi-Criteria-Predictable Multi-Core Computing" at the Sarntal Academy in South Tyrol, Italy from September 20 to October 02, 2015. Located at the picturesque Jägerhof in Durnholz, the course brought together seven PIs and ten doctoral researchers from the CRC as well as two additional doctoral researchers and a Master's student to discuss and work on the topic of predictability. As a tradition of the Sarntal Academy, the course combined working sessions with group activities, of which hiking in the beautiful mountains of the Sarntal served as a perfect activity to further enhance the team spirit of all participants. The working sessions were organised in two main parts—talks and lectures as well as discussion and working groups—with each week featuring both parts. The organisers (Prof. Michael Gerndt and Prof. Michael Glaß), as well as the guest lecturers (Prof. Jürgen Teich, Prof. Wolfgang Schröder-Preikschat, Prof. Michael Bader, PD Dr. Daniel Lohmann, and Dr. Stefan Wildermann) presented a wide range from fundamentals, to state-of-the-art, and ongoing research to enhance and deepen the predictability awareness across all projects. The doctoral researchers presented predictability aspects of their ongoing research and outlined open research questions as well as possible collaborations w. r. t. predictability. The complete list of talks that were given during the Sarntal Academy is given in the following table.

Title	Speaker
Predictability The Compiler's Point of View	Sebastian Buchwald
Simulating Heterogeneous MPSoCs and Predictability Analysis using Simulation	Sascha Roloff
Predictability in Operating Systems	W. Schröder-Preikschat
Predictability of Energy Demand: Energy-Optimization Strategies at Operating-System	Timo Hönig
Predictability by Hardware-Centric Operating-System Design	Daniel Lohmann
Predicting Worst-Case Energy Consumptions for Energy-Neutral Systems	Peter Wägemann
Multi-Objective System Design and Optimization	Tobias Schwarzer
Predictability in Resource Management	Anuj Pathania
A statistical Inverse Problem with adaptive surragate model implemented with invasive MPI	Emily Mo-Hellenbrand
Asynchronous Adaptive Operations and Resource Management	Isaías A. Comprés Ureña
Dependability Fundamentals	Michael Glaß
Runtime Exploitation of Application Dynamism for Energy-efficient eXascale computing	Michael Gerndt
Implementation of an actor-based Shallow Water Equations Simulation in X10	Alexander Pöppl
How bees allocate resources	Andreas Zwinkau
HW/SW Mechanisms for Adaptive Resource Isolation for *-Predictability	Jürgen Teich
Optimization Techniques for Managing Many-Core Systems	Stefan Wildermann
High Level Modelling and Simulation for Regional Cache Coherence in Multicore Systems	Akshay Srivatsa
Adaptive Application-Specific Invasive Microarchitecture	Marvin Damschen
ProFPGA: Analysis of the new demonstrator platform	Marcel Brand

Based on these talks, working groups were formed that discussed topics such as (i) the modelling of requirements and assumptions w. r. t. quality numbers, (ii) a formal underpinning for the general design flow, (iii) challenges for predictability in the context of invasive applications, system software and architectures, as well as (iv) concepts for

benchmarks and demonstrators. Besides enhancing the awareness of predictability aspects and challenges across all projects and abstraction layers, the course was also very productive in achieving and documenting concrete results and outlining future work. Also, concepts for demonstrations of the benefits of invasive computing to provide predictable execution qualities through resource isolation, interference minimisation, novel run-time–management techniques, and reconfigurable hardware have been elaborated as follows:

- **Interplay of Actors, Characterisation, and Compilation in X10.** A a possible actor implementation in X10 that will serve as a template for future actor-based applications, their characterisation, and the following source-to-source compilation was outlined.

- **Formalisation of Topics and their Interplay.** Based on the WG's topics, a formal definition of the most important terms and their interplay tailored to invasive computing was proposed and will be provided as an internal white paper. For example, what is a *claim constraint*, how are claim constraints generated, and how are they represented/encoded so they are available for the run-time–management system.

- **Formalisation of Predictability w. r. t. Invasive Computing** A novel formal definition of predictability in the context of invasive computing was proposed and discussed.

- **Quality Numbers and Prerequisites for their Predictability.** Quality numbers relevant for the CRC were investigated and the general prerequisites in order to achieve predictability for them were extracted and provided as an internal white paper.

- **Garbage Collection.** Garbage collectors in general and in particular in X10 and their impact on predictability were discussed and analysed. The discussion, implications, and possible solutions have been documented in an internal white paper.

- **Demonstrators.** First ideas on how to showcase predictability in combination with the most important aspects of invasive computing were outlined and next steps were documented.

The course was completed with an X10 programming tutorial that introduced all participants to a basic actor-oriented programming in X10, using the SWE application developed in Project A4.

Forthcoming Activities

In the following, we present a list of forthcoming activities related to the topic of WG1:

1. *Keynote RAPIDO, the 8th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools, Prague, Czech Republic, 18 January, 2016*. Prof. Teich will give a keynote titled *The Role of Restriction and Isolation for Increasing the Predictability of MPSoC Stream Processing* including definitions of *-predictability as well as the capability of invasive computing to decrease the degree of variability and thus increase the timing predictability in terms of observed latency and frame rates for streaming applications through resource isolation.

2. *Keynote IMPACT, the 6th International Workshop on Polyhedral Compilation Techniques, Prague, Czech Republic, 19 January, 2016*. This keynote titled *Symbolic Loop Parallelization for Adaptive Multi-Core Systems – Recent Advances and Benefits* reports on the ability of tightly coupled processor arrays (TCPAs), see Project B2, to create strictly predictable execution times for loop nests in dependence of claim sizes determined at runtime.

3. *Special Issue on Invasive Computing in it – Information Technology*. Prof. Teich is a guest editor of a special issue on the topic of invasive computing with a focus on *-predictability. This special issue will provide first results of our second funding phase on tackling the problems of *-predictability of multi-objective execution qualities of parallel invasive programs and including their 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), but also include aspects of security and power. Through analysis of application requirements from different domains including stream processing and malleable task applications, not only efficiency but also predictable execution qualities may be demonstrated nicely for applications stemming from robotics, imaging, as well as HPC. The special issue will be enriched by an overview article on a recent international initiative called “Elastic Computing” that you will see exploits quite similar ideas on resource allocation on demand,

yet for cloud computing, in order to dynamically trade off multiple qualities of services that may be observed in reaction.

4. *Dagstuhl-Seminar 16441 "Adaptive Isolation for Predictability and Security", October 30–November 4, 2016*. This seminar has been jointly proposed by PI Teich and Prof. Verbauwhede together with Prof. Tulika Mitra, NUS, and Prof. Lothar Thiele, ETH Zurich. The cross-cutting topics of the proposed seminar are methods for temporal and spatial isolation. These methods shall be discussed for their capabilities to enforce non-functional properties on timing predictability as well as security without sacrificing any efficiency or resource utilisation. To be able to provide isolation instantaneously, e. g. even just for segments of a program under execution, adaptivity is essential at all hardware and software layers. Support for adaptivity will be the second focal aspect of the seminar. Here, virtualisation and new adaptive resource reservation protocols will be discussed and analysed for their capabilities to provide application/job-wise predictable program execution qualities on demand and for their costs and overheads. In case the latter may be kept low, there is a chance that adaptive isolation, the title of our seminar, may be a door opener of MPSoC technology for many new application areas of embedded systems.

WG2: Memory Hierarchy

Coordinators: Lars Bauer, Gregor Snelting

In the first funding phase of InvasIC, cache coherence was 'only' provided within a tile. In the current second funding phase, Project B5 proposed the work package "Dynamic and run-time-adaptive cache coherence", which essentially proposes to dynamically extend the size of a cache-coherence region beyond the boundaries of a single tile. For instance, it could be extended over two or three (possibly neighboured) tiles. As it was foreseeable that such an extension could affect several aspects of the memory hierarchy (e. g. memory consistency) and other projects as well, the memory hierarchy working group WG2 and Project B5 organised a meeting to discuss these topics within InvasIC.

All relevant projects participated in the meeting at March 16, 2015 in Munich. This WG2 annual report summaries the results and open questions that came up during the WG2 Meeting and that were refined thereafter.

Supported Memory for Coherence Extension

Inter-tile cache coherence would preferably be provided for Tile local memory (TLM), but not necessarily for global shared memory. The extension to global shared memory is optional.

Addressing Problems Related to Coherency Extension (Local vs. Global TLM Address)

The TLM in a tile can be accessed by its local address (only valid within the tile) and its global address (valid from all other tiles). The local address always accesses the TLM of the tile that issued the access, whereas the global address can be used to access the TLM of other tiles. However, an *i*-let running on tile A cannot access the TLM of tile A by using its global address. It has to use the local address, because the TLM of tile A can hardware-wise only be mapped to one address region (here: the local address), but not to two separate address regions (local

and global address).[1]

This is problematic for the planed coherence extension. For instance, when tiles A and B are merged into one cache coherent region, then it would no longer be possible to distinguish the TLMs of tiles A and B by using the local address. For instance, if an *i*-let that executes on tile A writes some data to the TLM of tile A (by using its local address) and then wants to pass a pointer to that data to another *i*-let, but that other *i*-let happens to execute on tile B (same cache coherent region), then the *i*-let on tile B would mistakenly access the local TLM of tile B.

Therefore, when sharing the TLMs of two or more tiles in a coherency region, it has to be ensured that the applications only use the global TLM addresses (with corresponding hardware modifications to support that) even when accessing the local TLM. However, because OctoPOS and the entire software stack require the usage of local TLM addresses[2], it was suggested to use 'local to global' address translation (basically the MMU) between the core and the cache.

This means: OctoPOS and the software stack could still use local addresses, as they would be automatically translated to the global address of the local TLM. For the same reason, applications could still uses local TLM addresses if they do not envision to benefit from extended cache coherence regions. Only applications that explicitly want to support extended cache coherence tiles would need to use global TLM addresses to explicitly make clear which of the multiple TLMs inside the extended coherence region should be accessed.

To realise this, multiple actions need to be taken:

- The hardware TLM needs to be changed to only react on its global addresses.
- Each Leon3 needs to incorporate an MMU that needs to be initialised to map all requests to a local TLM address to the global addresses of the tile's TLM.
- For applications that want to make use of a cache coherence region across multiple tiles, something like an OS-level application loader

[1] Even when using a workaround (e. g. adding a second slave for the other address region to the bus that accesses the same memory array under the hood), the data caches would not understand that accesses to the local and global address actually go to the same physical memory, which would lead to cache coherence issues.

[2] Currently, all program/data is stored in TLM. The software demands that the memory layout is identical on all tiles that shall execute *i*-lets. This enables to realise a remote-infect by passing a function pointer (pointing to a local TLM address) to the target tile. If different addresses would be used per tile (e. g. due to different memory layouts or due to using global TLM addresses), then these would need to be translated first.

would be required that changes the addresses of all statically known accesses to a TLM within the cache coherence region to the global TLM address of any TLM within the particular cache coherence region that the application invaded.

Memory Consistency Model

Total Store order (TSO) consistency should be followed. If a separate medium is used for forwarding coherence traffic (e. g. a configurable bridge that connects two cache coherent tiles), it should be ensured that the writes from the same processor may not arrive out of order at the destination, independent whether they are transmitted via the *i*NoC or the separate medium. In addition, the cache invalidation requests should be handled to minimise the blocking of bus and still keeping the consistency.

Writing the updated values in L2 Cache back to the home tile

If L2 of a tile caches the TLM of a remote tile, it needs to be ensured that –if this value gets updated– it has to be written back to the home tile's TLM who is actually the owner of that data.

Atomic memory operations

Atomic memory operations like SWAP, LDSTUB, and CASA must behave as they would as defined in the memory consistency model (and as they do on a single tile). In addition, the fairness of lock access by remote tiles as compared to local tile should be ensured.

Distribution of *i*-lets in tiles within coherent regions

It needs to be investigated how the *i*-lets would be distributed by Octo-POS in the tiles which make one coherent region.

Exchange of coherency messages

Originally, two alternatives were envisaged for exchanging coherency messages among the tiles making up a coherency region: a) via bridges between the AHB buses of adjacent tiles that can be activated on demand, or b) via the *i*NoC. It has turned out that case a) would not only limit coherency regions to be built with adjacent tiles. Also the size of such regions would be limited to two tiles as otherwise the originator of accesses to shared memory could not be identified any longer if more than one bridge has to be traversed. Therefore, only communication via *i*NoC will be further investigated.

Potential Target Applications

Applications like DNA sequencing, sparse matrix processing, or tree traversals could be the best use cases and are expected to benefit from inter-tile cache coherence. Distributed MPI-based applications like multi-grid are not considered as good use cases for inter-tile coherence because these application work on their own data sets and require message passing-based communication instead of shared memory communication.

Evaluation of inter-tile cache coherence overhead

The penalty and cost of inter-tile cache coherence need to be evaluated by Project B5 with the help of appropriate modelling/analysis. It should be considered that the coherence messages are exchanged only when necessary and without blocking the tile bus.

Required extensions in the language, agent system and OctoPOS

Mechanisms have to be defined in which way an application can request a coherency region of two or more tiles, how the individual tiles are identified that should make up the requested regions and how the HW support for the coherency extension is activated by OctoPOS. This all requires the definition of appropriate interfaces between the involved software and hardware entities.

Memory Model on Language Level

Also related to the memory considerations regarding inter-tile cache coherence is the memory model of the programming language X10, which the compiler must map to the hardware memory model. Starting point could be a verified Java memory model[3], but weaker models[4] might be necessary depending on the guarantees the hardware can provide.

[3] Andreas Lochbihler, "Making the Java Memory Model Safe", ACM ToPLaS 2014.

[4] F. S. Zakkak and P. Pratikakis. "JDMM: A Java Memory Model for Non-cache-coherent Memory Architectures", ISMM 2014.

WG3: Benchmarking and Evaluation

Coordinators: Michael Bader, Walter Stechele

In February 2015, WG3 and Project Z2 have jointly initiated a survey on invasive applications for the demonstrator platforms, i. e. the hardware prototyping platform (invasive MPSoC) and the x86-based standard multicore platform. The goal of the survey was two-fold: WG3 started a discussion on how to demonstrate the benefit of invasion w. r. t. quality numbers and compared to the state of the art; Project Z2 collected hardware (and other) requirements to run these applications and demonstrate the benefits of invasive computing, including predictability of multiple applications on large manycore platforms. A special focus is on benefits and limitations of resource reservation (cores, memory, NoC bandwidth). Predictability includes execution time of mixed-criticality soft and hard real-time applications, security and fault tolerance, dark silicon and energy-efficient computing.

In July 2015, WG3 and Project Z2 have jointly organised a demonstrator workshop in Munich. The goal was to update planning for potential demonstration scenarios on both platforms, FPGA-based invasive MPSoC implementations and x86-based demonstration. Overall, contributions have indicated first ideas for common demonstrations between projects, including HPC and dark silicon as well as using the X10 actor library for demonstrations of applications running on *i*-Core and TCPA. A more detailed planning and requirements on the demonstrator platform is subject of upcoming discussions.

At the annual meting in October 2015, the focus of WG3 was on the performance of the FPGA-based invasive MPSoC implementations. The topic has been initiated by Manuel Mohr, addressing the low CPU clock frequency as compared to the memory clock frequency on the FPGA platform. Michael Bader presented the Roofline Model (Williams et al. 2008) to evaluate the ratio between memory and computing. Further evaluation of invasive demonstration scenarios with respect to the Roofline Model will be discussed.

For the DATE conference in March 2016, Walter Stechele is a co-organiser of the First Workshop on Resource Awareness and Application Autotuning in Adaptive and Heterogeneous Computing, together with Cristina Silvano from Politecnico di Milano and Stephan Wong from TU Delft, with contributions from high performance computing arranged by Michael Bader. The goal of the workshop is to bring together researchers from the area of resource awareness and application autotuning, to discuss their various approaches, their commonalities and differences, to foster collaboration between them and to share their most recent research achievements with the international research community. The workshop program will include invited talks from top-level specialists coming from both industry and academia. There is an open Call for Posters in order to give young researchers and PhD students a chance to introduce their research and to get in personal contact with each other. Researchers from invasive computing are cordially invited to contribute to the workshop. Further information can be found on `http://res4ant.deib.polimi.it`.

WG4: Power Efficiency and Dark Silicon

Coordinators: Jörg Henkel, Frank Hannig

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 is not only ensured that the (worst-case) thermal constraints of the on-chip computation and communication resources are not violated, but this also aids 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.

A summary of the key goals and targets of the WG, aligned to the roadmap of planned InvasIC-wide dark silicon research activities, 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 (Project B4) and iDoC (Project 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. Projects 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 have already been identified. A couple of examples under discussion along with the early research efforts are listed below.

1. The early research work of Project B3 explored the impact of Dark Silicon Management for Performance Optimisation, Dark Silicon Management for Aging Optimisation, and Dark Silicon Management for Energy Optimisation (for more details and a list of publications, we refer to Project B3).
2. **Open-Source Software**: Two tools implementing some work of B3, specifically, the Thermal Safe Power (TSP) power budgeting technique and the MatEx transient/peak temperature computation framework, are available at http://ces.itec.kit.edu/download.
3. Some collaborative steps towards the InvasIC-wide dark silicon patterning are summarised below.
   a) **Collaborative Activities between Project B3 and Project 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 Project B3 and Project C1**: Coordinated Resource and Dark Silicon Management: Interfacing with Project 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 Project B3 and Project 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. *Special Session on "Dark Silicon: No Way Out?" at ACM/EDAC/IEEE 52nd Design Automation Conference (DAC), 2015*: The goal of this special session is to expose Dark Silicon challenges for design automation, architecture, and system-level design communities 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 particular, this special session aims at clarifying whether "Dark Silicon introduces fundamentally new challenges for the community" or "the Dark Silicon is merely an additional constraint for the system designer". Our broader goal is to spur greater awareness and discussion of these challenges in the design automation community, and to position the Dark Silicon problem as one that this community can have a large impact on solving. More details about the special session can be found at http://www2.dac.com/events/eventdetails.aspx?id=182-55.

2. *Special Session on "Dark Silicon - From Computation to Communication" at International Symposium on Networks-on-Chip September (NOCS), 2015*: This special session aims at exposing dark silicon challenges to the NOCS community, particularly, covering both the computation and communication perspectives [Hen+15].
3. *Workshop on "Towards Efficient Computing in the Dark Silicon Era" at IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2015*: This workshop is intended to be a forum to synthesise emerging perspectives and research directions on the dark silicon problem from across industry and academia. More details can be found at https://wp.nyu.edu/iccad_darksil_workshop.
4. *Special Issue on "Computing in the Dark Silicon Era" at IEEE Design & Test:* Accepted special issue proposal planned for mid 2016. The objective of this special issue is to provide a forum for academic and industry researchers to publish new research results that solve the outstanding challenges posed by the dark silicon problem.
5. *Tutorial on "Designing and Architecting Advanced Embedded Systems" at the National University of Sciences and Technology (NUST)*: by M. Shafique, S. Rehman, December 2014, Islamabad, Pakistan.
6. *Ph.D. Forums:*
   a) M. U. K. Khan, M. Shafique, and J. Henkel, "Hardware-Software Co-Design for Next Generation Dark Silicon Multimedia Systems", at the ACM/EDAC/IEEE 52nd Design Automation Conference (DAC), June 2015.
   b) M. U. K. Khan, M. Shafique, and J. Henkel, "Hardware-Software Co-Design for Next Generation Dark Silicon Multimedia Systems", at the IEEE/ACM 16th Design Automation and Test in Europe Conference (DATE), **Ph.D. Forum Best Poster Award**, March 2015.

7. *Keynote Talks: Prof. Dr. Henkel and Dr. Shafique gave one keynote each about dark silicon.*
   a) J. Henkel, “Dark Silicon and Dependability”, at the International Symposium on Computer Architecture & Digital Systems (CADS), October 8, 2015, Tehran, Iran.
   b) M. Shafique, “Run-Time Resource and Reliability Management in Dark Silicon Many-Core Chips”, at the International Workshop on Multi-Objective Many-Core Design (MOMAC) in conjunction with International Conference on Architecture of Computing Systems (ARCS), March 24, 2015, Porto, Portugal.
8. Additional invited talks and special session papers [HKS15; PSCH15; PCSH15b; SH15].

Publications

[Hen+15] J. Henkel, H. Bukhari, S. Garg, M. U. K. Khan, H. Khdr, F. Kriebel, U. Ogras, S. Parameswaran, and M. Shafique. “Dark Silicon - From Computation to Communication”. In: *International Symposium on Networks-on-Chip (NOCS)*. Invited Special Session Paper. Sept. 2015.

[HKS15] J. Henkel, M. U. K. Khan, and M. Shafique. “Energy-Efficient Multimedia Systems for High Efficiency Video Coding”. In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. (accepted as a reviewed Special Session paper). Lisbon, Portugal, May 24–27, 2015.

[PCSH15b] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. “Thermal-Aware Power Budgeting for Dark Silicon Chips”. In: *Proceedings of the 2nd Workshop on Low-Power Dependable Computing (LPDC) at the International Green and Sustainable Computing Conference (IGSC)*. Las Vegas, Nevada, USA, Dec. 14–16, 2015.

[PSCH15] S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel. “Thermal-Aware Power Budgeting for Dark Silicon Chips (Invited talk)”. In: *Workshop on System-to-Silicon Performance Modeling and Analysis at the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. San Francisco, CA, USA, June 8–12, 2015.

[SH15] M. Shafique and J. Henkel. “Mitigating Power Density and Temperature Problems in the Nano-Era”. In: *IEEE/ACM 34th International Conference on Computer-Aided Design (ICCAD)*. Special Session Paper. San Jose, CA, USA, Nov. 2, 2015.

III

Events and Activities

Summary

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

Figure 5.1: 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 Sarntal Academy (Section 6) and many further activities in 2015. 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 in Section 10.

Figure 5.2: At the Annual Meeting in Nuremberg, October 2015

6 Sarntal Academy

From September 20 to October 02, 2015 a Doctoral Seminar on the topic of Predictability through principles of invasive computing took place at the Sarntal Academy. Prof. Michael Gerndt and Prof. Michael Glaß organised this doctoral seminar for the working group on predictability of the CRC/Transregio 89 titled “Benchmarking for Multi-Criteria-Predictable Multi-Core Computing” at the Sarntal Academy in South Tyrol, Italy. The organisers, doctoral researchers, as well as the guest lecturers Prof. Jürgen Teich, Prof. Wolfgang Schröder-Preikschat, Prof. Michael Bader, PD Dr. Daniel Lohmann, and Dr. Stefan Wildermann presented state-of-the-art and ongoing research to enhance and deepen the predictability-awareness across all projects. Topics included (i) the modelling of quality requirements for sections of parallel program code and assumptions wrt. quality numbers such as timing, reliability, security, or power, (ii) a formal underpinning for the general design flow, (iii) challenges for predictability in the context of invasive applications, system software, and architectures. Finally, (iv) concepts for benchmarks and demonstrators have been discussed and results been documented in concrete code examples and white papers. Following the tradition of the Sarntal Academy, hiking as well as other group activities enabled to compensate for the hard but successful work.

Figure 6.1: InvasIC Doctoral Seminar, Sarntal Academy, 2015

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 2015, researchers met at the following opportunities:

Event	Date	
WG Predictability	Jan. 21, 2015, Erlangen	In Erlangen, researchers from the working group Predictability met to debate security requirements.
Semi-annual Meeting 2015	Feb. 12/13, 2015, Adelsried	At the semi-annual meeting, the status quo of projects and the working groups were summarised.
WG Memory Hierarchy	March 16, 2015, Munich	Members from the working group Memory Hierarchy met in Munich to discuss the technical details and their implications on memory hierarchy, memory model, OS and applications
Doctoral Researcher Retreat	Sept. 23–25, 2015, Aalen	The 7th InvasIC DRR took place in Aalen to discuss the challenges of the second funding phase.
WG Predictability	May 22, 2015, Munich	In Munich, researchers from the working group Predictability met again.
WG Demonstrator	July 16, 2015, Munich	Members from the working group Demonstrator met in Munich to discuss ongoing work. They started with the application projects, continued with OctoPOS, agents, compiler, then with hardware projects, TCPA, *i*-Core, *i*NoC, and finally the Z2 demonstrator platform.
Sarntal Academy	Sept. 20–Oct. 02, 2015 Sarntal	Doctoral researchers and guest lecturers met in Sarntal to investigate issues of predictability through invasive computing
Annual Meeting 2015	Oct. 15/16, 2015 Nuremberg	Projects and working groups presented their progress with focus on their next working packages.

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.

Event	Date	
Tutorial Software Engineering in der DFG at Multikonferenz Software Engineering and Management 2015	March 18, 2015 Dresden, Germany	Prof. Dr.-Ing. J. Teich (FAU) gave a tutorial on “Invasives Rechnen”.
Tutorial Lange Nacht der Wissenschaften	Oct. 24, 2015 Erlangen, Germany	Researchers of Project C2 vividly demonstrated the principles of invasive computing at the Long Night of Sciences.
Workshop TEAM - Toll, ein anderer macht’s!	Nov. 26/27, 2015 Erlangen, Germany	The seminar of Barbara Berndt was organised for young researchers to learn the basics of team building and team leading.

Figure 8.1: Lange Nacht der Wissenschaften, October 2015 in Erlangen

TEAM - Toll, ein anderer macht's!

During a two-day workshop in November 2015, doctoral researchers learned about essential team work and team building. The soft-skill seminar was given by Barbara Berndt, a professional didactics trainer and management coach. Step by step, the participants learned the theoretical basics of how teams are formed and which aspects are important for a team to succeed. It was explored how someone's type of personality can affect the role they play in a group, along with techniques how to lead a team effectively and how to resolve internal conflicts.

The theory was put into practice through various exercises in which the doctoral researchers interacted with each other to fulfil given tasks as a team. After each practice session, the participants reflected on the insights gained and were given helpful personal feedback by the coach. The workshop was packed full of useful tips and information, and was met with very positive resonance.

Figure 8.2: Workshop on team building, Erlangen, November 2015

9 InvasIC Activities

To promote the ideas and results of InvasIC and discuss them with leading experts from industry and academia, international 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`.

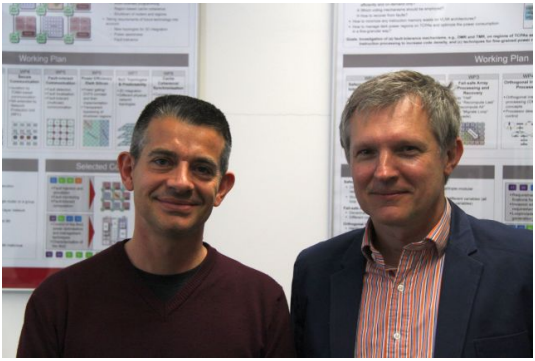


Figure 9.1: Prof. Luca Benini together with Prof. Jürgen Teich at the InvasIC Seminar

Figure 9.2: Prof. Katharina Morik giving a talk at the InvasIC Seminar

InvasIC Seminar

Time and Place	Title	Speaker
Erlangen, January 30, 2015	Reliable Real-Time Communication in Cyber-Physical Systems: Towards Cooperative Vehicular Networks	Prof. Dr. F. Dressler (University of Paderborn)
Erlangen, February 27, 2015	Accelerating Data Processing Using FPGAs	Prof. Dr. J. Teubner (TU Dortmund)
Erlangen, March 26, 2015	Predictability for Uni- and Multi-Core Real-Time/Cyber-Physical Systems	Prof. F. Mueller (North Carolina State University)
Erlangen, March 27, 2015	Building, Programming, and Validating Low-power Heterogeneous Multi-core Image Processors	Dr. M. Lindwer (Intel Corporation, Eindhoven)
Erlangen, March 27, 2015	Multi- and Many-Core Architectures - A Trip over a Bumpy Road	Prof. Dr.-Ing. J. Nolte (BTU Cottbus–Senftenberg)
Erlangen, April 10, 2015	Toward Energy-neutral Computational Sensing - Challenges and Opportunities	Prof. Dr. L. Benini (ETH Zurich)
Munich, April 28, 2015	Pattern-based Parallelization of Sequential Software	Korbinian Molitorisz (Karlsruhe Institute of Technology)
Erlangen, June 5, 2015	Fault-Tolerant Task Pools	Prof. Dr. C. Fohry (Universität Kassel)

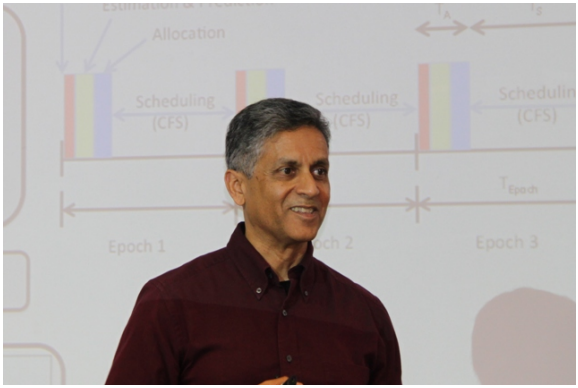


Figure 9.3: Chancellor's Professor Nikil Dutt giving a talk at the InvasIC Seminar

Time and Place	Title	Speaker
Erlangen, June 12, 2015	Constructing Time-Predictable MPSoCs – Never Give up Control	Prof. Dr. P. Puschner (TU Wien)
Erlangen, July 17, 2015	Big Data — Small Devices	Prof. Dr. K. Morik (TU Dortmund)
Erlangen, July 24, 2015	Proof-Carrying Services	Prof. Dr. M. Platzner (Universität Paderborn)
Erlangen, August 14, 2015	Dataflow based Design and Implementation for Multicore Digital Signal Processors	Prof. Dr. S. S. Bhattacharyya (University of Maryland)
Erlangen, October 28, 2015	Towards Sentient Chips: Self-Awareness through On-Chip Sensemaking	Chancellor's Professor N. Dutt (University of California, Irvine)
Erlangen, November 20, 2015	Real-Time Operating Systems for Multicore Platforms: Scheduling, Locking, and Open Problems	Dr. B. B. Brandenburg (Max Planck Institute for Software Systems)
Erlangen, November 27, 2015	Energy-Efficient Algorithms	Prof. Dr. S. Albers (TU München)
Munich, November 30, 2015	Data-driven Online Adaptive Model Reduction For Outer Loop Applications	Dr. B. Peherstor (Massachusetts Institute of Technology)
Munich, December 10, 2015	Integrated Circuit Test Structures: Lessons Learned	Dr. Sani Nassif (Radyalis)

Figure 9.4: Prof. Susanne Albers giving a talk at the InvasIC Seminar

Invited Talks and Seminars

Date and Place	Title	Speaker
Porto, Portugal, March 24, 2015 Second International Workshop on Multi-Objective Many-Core Design (MOMAC)	Keynote: Run-Time Resource and Reliability Management in Dark Silicon Many-Core Chips	Dr.-Ing. M. Shafique (KIT)
Auckland, New Zealand, April 10, 2015 Auckland University	Talk: Invasive Computing: A Systems-Programming Perspective	Prof. W. Schröder-Preikschat (FAU)
St. Goar, Germany, June 2, 2015 18th International Workshop on Software and Compilers for Embedded Systems, Scopes 2015	Keynote: Adaptive Isolation for Predictable MPSoC Stream Processing	Prof. J. Teich (FAU)
San Francisco, USA, June 7, 2015 Workshop on System-to-Silicon Performance Modeling and Analysis, DAC 2015	Talk: Thermal-Aware Power Budgeting for Dark Silicon Chips	Dr.-Ing. M. Shafique (KIT)
San Francisco, USA, June 7, 2015 Workshop on System-to-Silicon Performance Modeling and Analysis, DAC 2015	Talk: Enabling Dependable MPSoC Task Migration with On-Chip Interconnect Virtualization	Prof. A. Herkersdorf (TUM)
San Francisco, USA, June 7, 2015 Workshop on System-to-Silicon Performance Modeling and Analysis, DAC 2015	Talk: The FIGAROS Operating System Kernel for Fine-Grained System-Level Energy Analysis	T. Hönig (FAU)
San Francisco, USA, June 8–12, 2015 ACM/EDAC/IEEE 52nd Design Automation Conference (DAC)	Talk: New Trends in Dark Silicon	Prof. J. Henkel (KIT)
Berkeley, USA, June 18, 2015 International Computer Science Institute (ICSI)	Talk: Plan Ahead: Making Energy-Aware Computing Systems	T. Hönig (FAU)

Date and Place	Title	Speaker
Berkeley, USA, June 18, 2015 University of California	Talk: When Less is More: Invasive Energy Optimizations of System Software	T. Hönig (FAU)
Ventura, USA, July 13, 2015 15th International Forum on MPSoC for Software-defined Hardware	Keynote: What happens on an MPSoC stays on an MPSoC	Prof. A. Herkersdorf (TUM)
Cracow, Poland, September 24, 2015 Conference on Design and Architectures for Signal and Image Processing, DASIP 2015	Keynote: Resource Awareness on Heterogeneous MPSoCs for Image Processing	Prof. W. Stechele (TUM)
Hanover, Germany, September 9, 2015 Leibnitz-University Hanover	Talk: Two-Dimensional Tailoring of RTOS Kernels: Rethinking the Hardware–Application Bridge	Dr.-Ing. D. Lohmann (FAU)
Hanover, Germany, September 9, 2015 Leibnitz-University Hanover	Talk: Adaptable System Software	Dr.-Ing. D. Lohmann (FAU)
Gilching, Germany, October 13, 2015 ESSEI TecDay 2015	Talk: Time-predictable Multi-core Programming using Invasive Computing	Dr.-Ing. S. Wildermann (FAU)
Dortmund, Germany, October 2, 2015 Technical University Dortmund	Talk: Predictability Issues in Operating Systems	Prof. W. Schröder-Preikschat (FAU)
Tehran, Iran, October 8, 2015 International Symposium on Computer Architecture and Digital Systems (CADS)	Keynote: Dark Silicon and Dependability	Prof. J. Henkel (KIT)
Amsterdam, The Netherlands, October 4–9, 2015 13th IEEE Symposium on Embedded Systems For Real-time Multimedia (ESTIMedia 2015)	Talk: Invasive Computing for Predictable Stream Processing: A Simulation-based Case Study	Prof. J. Teich (FAU)
Dresden, Germany, November 7, 2015 Technical University Dresden	Talk: Invasive Computing: A Systems-Programming Perspective	Prof. W. Schröder-Preikschat (FAU)

Organised Workshops

Date and Place	Title	Organiser
Adelsried, Germany, Feb. 1, 2015	InvasIC meets AVACS	Dr. D. Gangadharan (FAU), Prof. J. Teich (FAU)
Grenoble, France, March 13, 2015 Design, Automation and Test in Europe (DATE 2015)	Friday Workshop on Heterogeneous Architectures and Design Methods for Embedded Image Systems (HIS 2015)	Dr.-Ing. F. Hannig (FAU), D. Fey (FAU) A. Lokhmotov (ARM, UK)
Porto, Portugal, March 24, 2015 International Conference on Architecture of Computing Systems (ARCS 2015)	Second International Workshop on Multi-Objective Many-Core Design (MOMAC)	Dr.-Ing. S. Wildermann (FAU) Prof. M. Glaß (FAU)
Amsterdam, The Netherlands, October 8/9, 2015 13th IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia 2015)	Special Session on Dynamics and Predictability in Stream Processing – A Contradiction?	Dr.-Ing. F. Hannig (FAU)

InvasIC meets AVACS

The Transregional Collaborative Research Center 14 AVACS is based on a joint venture and partnership between the Universities of Freiburg, Oldenburg and Saarbrücken and the Max Planck Institut für Informatik in Saarbrücken and was established in 2003. They investigate techniques for the automatic verification and analysis of complex systems. AVACS concentrates on the verification of safety-critical systems controlling and interacting with physical and technical processes.
In early 2015 we organised a cooperation workshop “InvasIC meets AVACS” in Adelsried. For one day, researchers from both CRC/Transregios met to discuss the application of formal techniques developed in AVACS for the analysis of invasive computing applications and systems. They focused on predictability and dependability.

Figure 9.5: Workshop InvasIC meets AVACS in Adelsried, February 2015

DATE 2015 Friday Workshop HIS

A Workshop on “Heterogeneous Architectures and Design Methods for Embedded Image Systems” (HIS 2015) organised by Frank Hannig and Dietmar Fey was incorporated into the framework of the DATE Friday Workshops. The workshop was co-organised with the Conference on Design, Automation and Test in Europe (DATE) which took place from March 9th to March 13th, 2015, in Grenoble, France.

Awards

Date and Place	Title	
San Francisco, USA, June 7–11, 2015 Design Automation Conference (DAC 2015)	ACM SIGDA Outstanding New Faculty Award	Dr.-Ing. Muhammad Shafique
Erlangen, July 30, 2015	Siemens Masterpreis 2015	Michael Schadhauser
Amsterdam, The Netherlands, October 4–9, 2015 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)	Best Paper Award: R2Cache: Reliability-Aware Reconfigurable Last-Level Cache Architecture for Multi-Cores	F. Kriebel, A. Subramaniyan, S. Rehman, S. J. Ahandagbe, M. Shafique, J. Henkel

Figure 9.6: Award winner Dr.-Ing. M. Shafique

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 in 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)

11 Publications

[AFGM15] 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: Elsevier, 2015, pp. 81–86. DOI: 10.1016/j.jisa.2014.10.001.

[Ary+14] N. P. Aryan, A. Listl, L. Heiss, C. Yilmaz, G. Georgakos, and D. Schmitt-Landsiedel. “From an analytic NBTI device model to reliability assessment of complex digital circuits.” In: *International On-Line Testing Symposium (IOLTS)*. 2014, pp. 19–24.

[BB12] M. Bader and A. Breuer. “Teaching Parallel Programming Models on a Shallow-Water Code”. In: *ISPDC 2012 - 11th International Symposium on Parallel and Distributed Computing*. IEEE Computer Society, Nov. 2012, pp. 301–308.

[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, Aug. 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. DOI: 10.2316/P.2011.719-070.

[Bau+15] L. Bauer, A. Grudnitsky, M. Damschen, S. R. Kerekare, and J. Henkel. “Floating Point Acceleration for Stream Processing Applications in Dynamically Reconfigurable Processors”. In: *IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia’15)*. Invited Paper for the Special Session “Dynamics and Predictability in Stream Processing – A Contradiction?” Amsterdam, The Netherlands, Oct. 2015, 2 pp. DOI: 10.1109/ESTIMedia.2015.7351762.

[Bop15] S. Boppu. “Code Generation for Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, Dec. 18, 2015.

[Böw15] N. C. Böwing. “Invasives Verteiltes Job Queue Framework”. Master thesis. IPD Snelting, Dec. 2015.

[BHRB15] A. Breuer, A. Heinecke, L. Rannabauer, and M. Bader. “High-Order ADER-DG Minimizes Energy- and Time-to-Solution of SeisSol”. In: *High Performance Computing*. Ed. by J. M. Kunkel and T. Ludwig. Vol. 9137. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 340–357. ISBN: 978-3-319-20118-4. DOI: 10.1007/978-3-319-20119-1_25.

[Buc15] S. Buchwald. “Optgen: A Generator for Local Optimizations”. In: *Proceedings of the International Conference on Compiler Construction (CC)*. Ed. by B. Franke. Vol. 9031. Lecture Notes in Computer Science. Springer Berlin Heidelberg, Apr. 18, 2015, pp. 171–189. DOI: 0.1007/978-3-662-46663-6_9.

[BMR15] S. Buchwald, M. Mohr, and I. Rutter. “Optimal Shuffle Code with Permutation Instructions”. In: *Algorithms and Data Structures*. Ed. by F. Dehne, J.-R. Sack, and U. Stege. Vol. 9214. Lecture Notes in Computer Science. Springer International Publishing, Aug. 5, 2015, pp. 528–541. DOI: 10.1007/978-3-319-21840-3_44.

[BMZ15] S. Buchwald, M. Mohr, and A. Zwinkau. “Malleable Invasive Applications”. In: *Proceedings of the 8th Working Conference on Programming Languages (ATPS)*. Lecture Notes in Computer Science (LNCS). Springer Berlin Heidelberg, Mar. 18, 2015.

[DWH14] P. Damodaran, S. Wallentowitz, and A. Herkersdorf. “Distributed cooperative shared last-level caching in tiled multiprocessor system on chip”. In: *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*. Mar. 2014, pp. 1–4. DOI: 10.7873/DATE.2014.096.

[DSP15] G. Drescher and W. Schröder-Preikschat. “Guarded Sections: Structuring Aid for Wait-Free Synchronisation”. In: *Proceedings of the 18th IEEE Symposium on Real-Time Computing (ISORC 2015)*. Washington, DC, USA: IEEE Computer Society Press, 2015, pp. 280–283. ISBN: 978-1-4799-8781-8.

[Fig+15] P. Figuli, C. Tradowsky, J. Martinez, H. Sidiropoulos, K. Siozios, H. Stenschke, D. Soudris, and J. Becker. “A Novel Concept for Adaptive Signal Processing on Reconfigurable Hardware”. In: *Applied Reconfigurable Computing*. Vol. 9040. Lecture Notes in Computer Science. Springer International Publishing, 2015, pp. 311–320.

[Ger+12] 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.

[GS15] D. Giffhorn and G. Snelting. “A new algorithm for low-deterministic security”. In: *International Journal of Information Security* 14.3 (2015), pp. 263–287. DOI: 10.1007/s10207-014-0257-6.

[Glo+15] E. Glocker, Q. Chen, A. Zaidi, U. Schlichtmann, and D. Schmitt-Landsiedel. “Emulation of an ASIC power and temperature monitor system for FPGA prototyping”. In: *Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2015 10th International Symposium on*. June 2015, pp. 1–8. DOI: 10.1109/ReCoSoC.2015.7238083.

[Gna+15] D. Gnad, M. Shafique, F. Kriebel, S. Rehman, D. Sun, and J. Henkel. “Hayat: Harnessing Dark Silicon and Variability for Aging Deceleration and Balancing”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). HiPEAC Paper Award. ACM, June 7–11, 2015, 180:1–180:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2744849.

[Göt+15] J. Götzfried, T. Müller, R. de Clercq, P. Maene, F. Freiling, and I. Verbauwhede. “Soteria: Offline Software Protection within Low-cost Embedded Devices”. In: *Proceedings of the 31th Annual Computer Security Applications Conference (ACSAC’15)*. Los Angeles, California, USA: ACM, 2015, pp. 241–250.

[Gru15] A. Grudnitsky. “A Reconfigurable Processor for Heterogeneous Multi-Core Architectures”. Dissertation. Chair for Embedded Systems (CES), Department of Computer Science, Karlsruhe Institute of Technology (KIT), Germany, Dec. 21, 2015.

[HH15] F. Hannig and A. Herkersdorf. “Introduction to the Special Issue on Testing, Prototyping, and Debugging of Multi-Core Architectures”. In: *Journal of Systems Architecture* 61.10 (Nov. 7, 2015), p. 600. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.11.003.

[Hei+15] J. Heisswolf, A. Weichslgartner, A. Zaib, S. Friederich, L. Masing, C. Stein, M. Duden, R. Klöpfer, T. Wild, A. Herkersdorf, J. Teich, and J. Becker. “Fault-tolerant Communication in Invasive Networks on Chip”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8.

[HKPS15] J. Henkel, H. Khdr, S. Pagani, and M. Shafique. “New Trends in Dark Silicon”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). HiPEAC Paper Award. ACM, June 7–11, 2015, 119:1–119:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2747938.

[Hen+15] J. Henkel, H. Bukhari, S. Garg, M. U. K. Khan, H. Khdr, F. Kriebel, U. Ogras, S. Parameswaran, and M. Shafique. “Dark Silicon - From Computation to Communication”. In: *International Symposium on Networks-on-Chip (NOCS)*. Invited Special Session Paper. Sept. 2015.

[Hen15] J. Henkel. “Dark Silicon and Dependability”. Keynote Talk, International Symposium on Computer Architecture & Digital Systems (CADS). Oct. 8, 2015.

[HKS15] J. Henkel, M. U. K. Khan, and M. Shafique. “Energy-Efficient Multimedia Systems for High Efficiency Video Coding”. In: *IEEE International Symposium on Circuits and Systems (ISCAS)*. (accepted as a reviewed Special Session paper). Lisbon, Portugal, May 24–27, 2015.

[Her15] A. Herkersdorf. “What happens on an MPSoC stays on an MPSoC – unfortunately!” Keynote Talk at MPSoC Forum. Ventura, CA, USA, July 13, 2015.

[HJSP15] T. Hönig, H. Janker, and W. Schröder-Preikschat. “The FigarOS Operating System Kernel for Fine-Grained System-Level Energy Analysis”. In: *DAC 2015 Workshop on System-to-Silicon Performance Modeling and Analysis: Power, Temperature and Reliability*. San Francisco, CA, USA, June 7, 2015.

[Hön+15] T. Hönig, C. Eibel, B. Herzog, H. Janker, P. Wägemann, and W. Schröder-Preikschat. “Playing Hare and Tortoise: The FigarOS Kernel for Fine-Grained System-Level Energy Optimizations”. In: *2015 Brazilian Symposium on Computing Systems Engineering (SBESC ’15)*. Fóz do Iguaçú, Brazil: IEEE, Nov. 2015.

[IHSH16] S. Iqtedar, O. Hasan, M. Shafique, and J. Henkel. “Formal Probabilistic Analysis of Distributed Resource Management Schemes in On-Chip Systems”. In: *IEEE/ACM 19th Design, Automation and Test in Europe Conference (DATE’16)*. (Dresden, Germany). (accepted). Mar. 2016.

[KSH15] M. U. K. Khan, M. Shafique, and J. Henkel. “Hierarchical Power Budgeting for Dark Silicon Chips”. In: *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. July 2015, pp. 213–218. DOI: 10.1109/ISLPED.2015.7273516.

[KPSH15] H. Khdr, S. Pagani, M. Shafique, and J. Henkel. “Thermal Constrained Resource Management for Mixed ILP-TLP Workloads in Dark Silicon Chips”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). HiPEAC Paper Award. ACM, June 7–11, 2015, 179:1–179:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2744916.

[KGVA16] M. Kröhnert, R. Grimm, N. Vahrenkamp, and T. Asfour. “Resource-Aware Motion Planning”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2016, submitted.

[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].

[KM15] C. Kugler and T. Müller. “Separated Control and Data Stacks to Mitigate Buffer Overflow Exploits”. In: *Endorsed Transactions on Security and Safety*. European Alliance for Innovation (EAI), Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (ICST), 2015, pp. 1–36.

[Lar15] V. Lari. “Invasive Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, Nov. 18, 2015.

[Lar+15a] V. Lari, A. Tanase, J. Teich, M. Witterauf, F. Khosravi, F. Hannig, and B. Meyer. “A Co-Design Approach for Fault-Tolerant Loop Execution on Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231157.

[Lar+15b] V. Lari, J. Teich, A. Tanase, M. Witterauf, F. Khosravi, and B. Meyer. “Techniques for On-Demand Structural Redundancy for Massively Parallel Processor Arrays”. In: *Journal of Systems Architecture (JSA)* 61.10 (Nov. 2015), pp. 615–627. ISSN: 1383-7621. DOI: http://dx.doi.org/10.1016/j.sysarc.2015.10.004.

[LS15] B. Li and U. Schlichtmann. “Statistical Timing Analysis and Criticality Computation for Circuits With Post-Silicon Clock Tuning Elements”. In: *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on* 34.11 (Nov. 2015), pp. 1784–1797. ISSN: 0278-0070. DOI: 10.1109/TCAD.2015.2432143.

[LGG16] W. Liu, M. Gerndt, and B. Gong. “Model-based MPI-IO tuning with Periscope tuning framework”. In: *Concurrency and Computation: Practice and Experience* 28.1 (2016), pp. 3–20. ISSN: 1532-0634. DOI: 10.1002/cpe.3603.

[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: 10.1016/j.microrel.2014.01.013.

[LGRT11] M. Lukasiewycz, M. Glaß, F. Reimann, and J. Teich. “Opt4J: A Modular Framework for Meta-heuristic Optimization”. In: *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO)*. (Dublin, Ireland). ACM, 2011, pp. 1723–1730. ISBN: 978-1-4503-0557-0. DOI: 10.1145/2001576.2001808.

[MV15] P. Maene and I. Verbauwhede. “Single-Cycle Implementations of Block Ciphers”. In: *Lightweight Cryptography for Security and Privacy*. Lecture Notes in Computer Science. Springer-Verlag, 2015.

[Moh+15] M. Mohr, S. Buchwald, A. Zwinkau, C. Erhardt, B. Oechslein, J. Schedel, and D. Lohmann. “Cutting Out the Middleman: OS-Level Support for X10 Activities”. In: *Proceedings of the fifth ACM SIGPLAN X10 Workshop*. X10 ’15. Portland, OR, USA: ACM, June 14, 2015, pp. 13–18. DOI: 10.1145/2771774.2771775.

[PCH15] S. Pagani, J.-J. Chen, and J. Henkel. “Energy and Peak Power Efficiency Analysis for the Single Voltage Approximation (SVA) Scheme”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 34.9 (Sept. 2015), pp. 1415–1428. DOI: 10.1109/TCAD.2015.2406862.

[PCSH15a] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. “MatEx: Efficient Transient and Peak Temperature Computation for Compact Thermal Models”. In: *Proceedings of the 18th Design, Automation and Test in Europe (DATE)*. Grenoble, France, Mar. 9–13, 2015, pp. 1515–1520.

[PCSH15b] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. “Thermal-Aware Power Budgeting for Dark Silicon Chips”. In: *Proceedings of the 2nd Workshop on Low-Power Dependable Computing (LPDC) at the International Green and Sustainable Computing Conference (IGSC)*. Las Vegas, Nevada, USA, Dec. 14–16, 2015.

[PSCH15] S. Pagani, M. Shafique, J.-J. Chen, and J. Henkel. “Thermal-Aware Power Budgeting for Dark Silicon Chips (Invited talk)”. In: *Workshop on System-to-Silicon Performance Modeling and Analysis at the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. San Francisco, CA, USA, June 8–12, 2015.

[Pag+15] S. Pagani, M. Shafique, H. Khdr, J.-J. Chen, and J. Henkel. “seBoost: Selective Boosting for Heterogeneous Manycores”. In: *Proceedings of the 10th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*. Amsterdam, The Netherlands, Oct. 4–9, 2015, pp. 104–113.

[Par+15] P. Parayil, A. Zaib, T. Wild, W. Stefan, and A. Herkersdorf. “Sharer Status-based Caching in tiled Multiprocessor Systems-on-Chip”. In: *HPC 2015 - 23rd High Performance Computing Symposia*. SCS, The Society for Modeling & Simulation, Apr. 2015, pp. 67–74.

[PPSH15] A. Pathania, S. Pagani, M. Shafique, and J. Henkel. “Power Management for Mobile Games on Asymmetric Multi-Cores”. In: *ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED)*. July 2015, pp. 243–248. DOI: 10.1109/ISLPED.2015.7273521.

[Pat+16] A. Pathania, V. Venkataramani, M. Shafique, T. Mitra, and J. Henkel. “Distributed Fair Scheduling for Many-Cores”. In: *Design Automation and Test in Europe (DATE)*. 2016.

[Pau+14] 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+15] J. Paul, W. Stechele, B. Oechslein, C. Erhardt, J. Schedel, D. Lohmann, W. Schröder-Preikschat, M. Kröhnert, T. Asfour, Éricles R. Sousa, V. Lari, F. Hannig, J. Teich, A. Grudnitsky, L. Bauer, and J. Henkel. “Resource Awareness on Heterogeneous MPSoCs for Image Processing”. In: *Journal of Systems Architecture* 61.10 (Nov. 6, 2015), pp. 668–680. ISSN: 1383-7621. DOI: 10.1016/j.sysarc.2015.09.002.

[PZZB15] B. Peherstorfer, S. Zimmer, C. Zenger, and H.-J. Bungartz. “A Multigrid Method for Adaptive Sparse Grids”. In: *SIAM Journal on Scientific Computing* 37.5 (2015), S51–S70. DOI: 10.1137/140974985.

[PH16] A. Pöppl and A. Herz. “A cache-aware performance prediction framework for GPGPU computations”. In: *Euro-Par 2015: Parallel Processing Workshops*. Ed. by S. Hunold, J. L. Träff, and F. Versaci. Lecture Notes in Computer Science. accepted. Berlin, Heidelberg: Springer-Verlag, 2016.

[RHT12] S. Roloff, F. Hannig, and J. Teich. “Approximate Time Functional Simulation of Resource-Aware Programming Concepts for Heterogeneous MPSoCs”. In: *Proceedings of the 17th Asia and South Pacific Design Automation Conference (ASP-DAC)*. Sydney, Australia, Jan. 30–Feb. 2, 2012, pp. 187–192. ISBN: 978-1-4673-0770-3. DOI: 10.1109/ASPDAC.2012.6164943.

[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.

[RSHT15] S. Roloff, D. Schafhauser, F. Hannig, and J. Teich. “Execution-driven Parallel Simulation of PGAS Applications on Heterogeneous Tiled Architectures”. In: *Proceedings of the 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. (San Francisco, CA, USA). ACM, June 7–11, 2015, 44:1–44:6. ISBN: 978-1-4503-3520-1. DOI: 10.1145/2744769.2744840.

[RWHT15] S. Roloff, S. Wildermann, F. Hannig, and J. Teich. “Invasive Computing for Predictable Stream Processing: A Simulation-based Case Study”. In: *Proceedings of the 13th IEEE Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia)*. (Amsterdam, The Netherlands). IEEE, Oct. 8–9, 2015, 2 pp. ISBN: 978-1-4673-8164-2. DOI: 10.1109/ESTIMedia.2015.7351761.

[Sch15] M. Schmid. “Rapid Prototyping for Hardware Accelerators in the Medical Imaging Domain”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 24, 2015.

[SGM15] M. Seitzer, M. Gruhn, and T. Müller. “A Bytecode Interpreter for Secure Program Execution in Untrusted Main Memory”. In: *20th European Symposium on Research in Computer Security (ESORICS’15)*. Vienna, Austria: SBA Research, 2015, pp. 376–395.

[SH15] M. Shafique and J. Henkel. “Mitigating Power Density and Temperature Problems in the Nano-Era”. In: *IEEE/ACM 34th International Conference on Computer-Aided Design (ICCAD)*. Special Session Paper. San Jose, CA, USA, Nov. 2, 2015.

[Sha15] M. Shafique. “Run-Time Resource and Reliability Management in Dark Silicon Many-Core Chips”. Keynote Talk, International Workshop on Multi-Objective Many-Core Design (MOMAC). Mar. 2015.

[SGGH15] M. Shafique, D. Gnad, S. Garg, and J. Henkel. “Variability-Aware Dark Silicon Management in On-Chip Many-Core Systems”. In: *18th IEEE/ACM Design, Automation and Test in Europe Conference and Exhibition (DATE)*. Grenoble, France, Mar. 9–13, 2015.

[SSKH16] A. K. Singh, M. Shafique, A. Kumar, and J. Henkel. “Resource and Throughput Aware Execution Trace Analysis for Efficient Run-time Mapping on MPSoCs”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* 35.1 (Jan.

2016), pp. 72–85. ISSN: 0278-0070. DOI: 10.1109/TCAD.2015.2446938.

[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.

[SHT15] É. Sousa, F. Hannig, and J. Teich. “Reconfigurable Buffer Structures for Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the International Embedded Systems Symposium (IESS)*. Foz do Iguaçu, Brazil: LNCS, Nov. 2015, 10 pp.

[Sou+15] Éricles R. Sousa, F. Hannig, J. Teich, Q. Chen, and U. Schlichtmann. “Runtime Adaptation of Application Execution under Thermal and Power Constraints in Massively Parallel Processor Arrays”. In: *Proceedings of the 18th International Workshop on Software and Compilers for Embedded Systems (SCOPES)*. (St. Goar, Germany). ACM, June 1–3, 2015, pp. 121–124. ISBN: 978-1-4503-3593-5. DOI: 10.1145/2764967.2771933.

[TWTH15] A. Tanase, M. Witterauf, J. Teich, and F. Hannig. “Symbolic Loop Parallelization for Balancing I/O and Memory Accesses on Processor Arrays”. In: *Proceedings of the 13th ACM-IEEE International Conference on Formal Methods and Models for System Design (MEMOCODE)*. (Austin, TX, USA). IEEE, Sept. 21–23, 2015, pp. 188–197. ISBN: 978-1-5090-0237-5. DOI: 10.1109/MEMCOD.2015.7340486.

[Tan+15] A. Tanase, M. Witterauf, J. Teich, F. Hannig, and V. Lari. “On-Demand Fault-Tolerant Loop Processing on Massively Parallel Processor Arrays”. In: *Proceedings of the 26th IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)*. (Toronto, Canada). IEEE, July 27–29, 2015, pp. 194–201. ISBN: 978-1-4799-1925-3. DOI: 10.1109/ASAP.2015.7245734.

[Tei15a] J. Teich. “Adaptive Isolation for Predictable MPSoC Stream Processing”. Keynote, SCOPES 2015, 18th International Workshop on Software and Compilers for Embedded Systems, Schloss Rheinfels, St. Goar, Germany. June 2, 2015.

[Tei15b] J. Teich. “Adaptive Isolation for Predictable MPSoC Stream Processing”. In: *Proceedings of the 18th International Workshop on Software and Compilers for Embedded Systems (SCOPES 2015)*. June 2015, p. 2. DOI: 10.1145/2764967.2771821.

[Tei15c] J. Teich. “Invasive Computing”. Invited Talk, SE 2015, Software Engineering and Management, Special Session Software Engineering in der DFG, Dresden, Germany. Mar. 19, 2015.

[THMB16] 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)*. 2016. Forthcoming.

[Tra+16] C. Tradowsky, E. Cordero, C. Orsinger, M. Vesper, and J. Becker. “A Dynamic Cache Architecture for Efficient Memory Resource Allocation in Many-Core Systems”. In: *Reconfigurable Computing: Architectures, Tools, and Applications*. Springer International Publishing, Mar. 2016. Forthcoming.

[Ull13] S. A. Ullrich. “Verified Construction of SSA Form”. Bachelor thesis. IPD Snelting, Oct. 2013.

[Unt+15] K. Unterweger, R. Wittmann, P. Neumann, T. Weinzierl, and H.-J. Bungartz. “Integration of FULLSWOF2D and PeanoClaw: Adaptivity and Local Time-Stepping for Complex Overland Flows”. In: *Recent Trends in Computational Engineering - CE2014*. Ed. by M. Mehl, M. Bischoff, and M. Schäfer. Vol. 105. Lecture Notes in Computational Science and Engineering. Springer International Publishing, 2015, pp. 181–195. ISBN: 978-3-319-22996-6. DOI: 10.1007/978-3-319-22997-3_11.

[Vah+15] N. Vahrenkamp, M. Wächter, M. Kröhnert, K. Welke, and T. Asfour. “The Robot Software Framework ArmarX”. In: *Information Technology* 57.2 (2015), pp. 99–111.

[Wäg+15] P. Wägemann, T. Distler, T. Hönig, H. Janker, R. Kapitza, and W. Schröder-Preikschat. “Worst-Case Energy Consumption Analyis for Energy-Constrained Embedded Systems”. In: *Proceedings of the 27th EUROMICRO Conference on Real-Time Systems (ECRTS 2015)*. Lund, Sweden, July 9, 2015, pp. 105–114.

[Wag+15] P. Wagner, L. Li, T. Wild, A. Mayer, and A. Herkersdorf. “Knowledge-Based On-Chip Diagnosis for Multi-Core Systems-on-Chip”. en. In: *edaWorkshop 15*. Dresden, Germany, May 2015, pp. 39–45. ISBN: 978-3-86386-914-4.

[Wei+14] 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. 15, 2014, pp. 1–10. DOI: 10.1145/2656075.2656083.

[Wei+15] A. Weichslgartner, J. Heisswolf, A. Zaib, T. Wild, A. Herkersdorf, J. Becker, and J. Teich. “Position Paper: Towards Hardware-Assisted Decentralized Mapping of Applications for Heterogeneous NoC Architectures”. In: *Proceedings of the second International Workshop on Multi-Objective Many-Core Design (MOMAC)*

in conjunction with International Conference on Architecture of Computing Systems (ARCS). Porto, Portugal: IEEE, Mar. 2015, pp. 1–4.

[Wil15a] S. Wildermann. “Time-predictable multi-core programming using Invasive Computing”. Invited Talk at ESSEI TecDay: Multi-core – The challenge in avionics. Oct. 13, 2015.

[Wil15b] S. Wildermann. “Wieviele Prozessoren passen in eine Hosentasche?” Invited Talk at Öffentliche Vortragsreihe Faszination Technik. Apr. 2015.

[WWT15] S. Wildermann, A. Weichslgartner, and J. Teich. “Design Methodology and Run-time Management for Predictable Many-Core Systems”. In: *Proceedings of the 6th IEEE Workshop on Self-Organizing Real-Time Systems (SORT)*. Apr. 13, 2015, pp. 1–8.

[Wir13] M. Wirnshofer. *Variation-Aware Adaptive Voltage Scaling for Digital CMOS Circuits*. Vol. 41. Springer Series in Advanced Microelectronics, 2013.

[Wit+15] M. Witterauf, A. Tanase, J. Teich, V. Lari, A. Zwinkau, and G. Snelting. “Adaptive Fault Tolerance through Invasive Computing”. In: *Proceedings of the 2015 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)*. Montreal, Canada: IEEE, June 15–18, 2015, pp. 1–8. DOI: 10.1109/AHS.2015.7231155.

[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)*. Vol. 9017. Lecture Notes in Computer Science (LNCS). Springer, 2015, pp. 186–198. ISBN: 978-3-319-16085-6. DOI: 10.1007/978-3-319-16086-3_15.