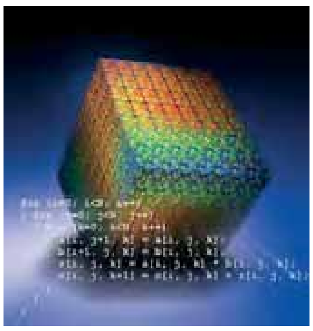
Sonderforschungsbereich/Transregio 89
Transregional Collaborative Research Center 89

Invasive Computing Annual Report 2018



TRR 89

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

Transregional Collaborative Research Centre 89

Invasive Computing

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

Annual Report 2018

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

Managing Director
Dr.-Ing. Jürgen Kleinöder
Lehrstuhl für Informatik 4
Friedrich-Alexander-Universität Erlangen-Nürnberg
Martensstraße 1
91058 Erlangen

Administration, Management, and Public Relations
Dr. Sandra Mattauch
Stefanie Kugler
Lehrstuhl für Informatik 12
Friedrich-Alexander-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 2018.

The CRC/Transregio InvasIC is funded by the Deutsche Forschungsgemeinschaft since July 2010. It 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 most exciting event in 2018 certainly was the DFG review meeting in January. In May 2018, we finally received the official news that our research ideas for a third funding phase were received so positively by the reviewers that InvasIC will continue being funded until the year 2022.

At this point, we would like to thank all of our members for their enormous effort and contributions.

2018 was also an outstanding year concerning the professional success of several members of the CRC/Transregio: Among others, Hananeh Aliee and Johannes Götzfried received awards for their doctoral theses. Also, Frank Hannig was awarded the Habilitation. Moreover, Jürgen Teich was elevated Fellow of the IEEE. Ulf Schlichtmann and Jürgen Teich were elected members of the National Academy of Science and Engineering (acatech). Finally, members of InvasIC also won several best paper awards in 2018.

We would like to thank all members of the CRC/Transregio InvasIC and all our partners from industry and academia for many fruitful discussions and support. We do hope that you will enjoy reading about the progress achieved in 2018, 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

Our CRC/Transregio systematically investigates the novel paradigm of *invasive computing* for designing and programming future parallel computing systems. For systems with 1,000 or more cores on a chip, *resource-aware programming* is of utmost importance to obtain high *utilisation* as well as *computational* and *energy* and *power efficiency*. With this goal in mind, invasive computing was introduced to provide a programmer explicit handles to specify and argue about *resource requirements* desired or required in different phases of execution: In an *invade* phase, an application asks the operating system to allocate a set of processor, memory and communication resources to be *claimed*. In a subsequent *infect* phase, the parallel workload is spread and executed on the obtained claim of resources. Finally, if the degree of parallelism should be lower again, a *retreat* operation frees the claim again, and the application resumes a sequential execution. To support this idea of *self-adaptive* and *resource-aware programming*, not only new programming concepts, languages, compilers, and operating systems need to be developed, but also revolutionary architectural changes in the design of MPSoCs (multiprocessor systems-on-a-chip) to efficiently support invasion, infection, and retreat operations.[1] This includes new architectural concepts for a dynamic processors, interconnects, and memory reconfiguration, to give some examples.

Necessity and First Achievements

As predicted at the start of our journey in 2010, we will see more than 1,000 processor cores integrated on a single chip in 2022.[2] Yet, programming such large-scale processor systems is a nightmare if resource awareness is a must and certain execution qualities must be guaranteed. Using invasive computing, a programmer may specify resource requirements and, if available, the application will obtain as many exclusive resources to deliver a desired quality of execution. This *dynamic* and *application-driven isolation* is unique. Starting off from scratch in terms of invasive processor hardware, language, compiler, and operating system, we have genuinely fostered the *fundamentals of invasive computing*

[1] This focus on investigations on invasive MPSoCs has inspired us to give our CRC the acronym InvasIC, see `http://www.invasic.de` for more details.

[2] Some GPU devices already having surpassed this number today!

in the first funding phase: These include the definition of programming language elements for invasion primitives as well as constraints to argue about number, types, and state of resources that may be invaded (the *invasive command space*, project area A). A first language based on the programming language X10 by IBM as well as a compiler for translation of invasive X10 programs (project area C) onto invasive multi-tile architectures (investigated by project area B) and a run-time system (*i*RTSS) for managing their execution is available. Invasive applications exploiting different types of processor and communication resources of an invasive network-on-chip (*i*NoC) have shown considerable gains in resource utilisation and efficiency in comparison with their non-invasive counterparts.

Recent Challenge: Predictability. Or: Sharing is Not Caring!

By the fact that resources are temporally claimed (by default) in an exclusive manner, interference by other applications due to resource sharing may be reduced if not avoided completely. This *isolation*, combined with *run-to-completion* as the default mode of thread execution and bandwidth guarantees on communication links, allow us to provide predictable quality-of-service (QoS) also for communication. In the current funding phase, we played out this ace systematically by tackling (a) *predictability* of (b) *multiple execution qualities* of parallel invasive programs and including their (c) *mapping optimisation*. Our recent findings include new language constructs to define so-called *requirements* on desired, respectively amended qualities of execution. Addressed qualities include performance (e. g. execution time, throughput, etc.), *security* and *fault tolerance*. Through the analysis of application requirements from different domains including stream processing and malleable task applications, not only efficiency but also predictable execution qualities can now be demonstrated for applications stemming from robotics, imaging, as well as HPC. As another new yet very important facet of invasive computing, a particular focus of the second funding phase was devoted to the problem of *dark silicon* and *energy*- and *power-efficient computing*.

The Missing Link: Beating Run-Time Uncertainties and Run-Time Requirement Enforcement

The isolation gained by invasive computing is essential to establish *composability*. This, in turn, paves the way for an independent and static analysis of individual program qualities in dependence of only resource claim properties, giving an unprecedented gain in predictability. Yet, even if this **-predictability*[3] (boundedness of any of the above non-functional properties through the invasion of resources) can be shown to hold, (a) the *effective bounds* (either lower or upper) as well as (b) their *variability* might still be too big or too coarse to be desirable or affordable in practical application fields such as embedded real-time control. Also, claiming resources exclusively might keep these either *underutilised* (in case of low application workload demands) or *inefficiently used* (e. g. when running a claim always at maximal processor speeds) in order to safely guarantee timing bounds also for the worst-case input.

Our third funding phase is therefore dedicated to the missing link: Beating the uncertainty caused by variation of *program input*, *machine state* and *environment* at run time. The envisioned solution: *Run-time requirement enforcement*. Formally, we want to investigate *hybrid techniques* combining (a) static analysis of the robustness of desired qualities in dependence of input and state fluctuations and (b) systematic generation of suitable *run-time requirement enforcers (RRE)* (additional code that either *locally* or *globally* observes and controls the satisfaction of requirements in respective *corridors* at run time). This also includes the generation of necessary program-specific *run-time requirement monitors (RRM)*. With these techniques, we want to reach our final goals and vision formulated already at the beginning of our mission: *Invasive computing will be a—if not the—vehicle for providing* resource awareness *for a mixture of* best-effort *and* predictable quality *applications. We do believe huge application and business fields in embedded systems will become accessible for multicore technology through the foundations of invasive computing.*

[3] See [Tei17] for proper definitions.

2 Participating University Groups

Friedrich-Alexander-Universität Erlangen-Nürnberg

Lehrstuhl für Hardware-Software-Co-Design

- Prof. Dr.-Ing. Jürgen Teich
- PD Dr.-Ing. Frank Hannig
- Dr.-Ing. Stefan Wildermann

Lehrstuhl für IT-Sicherheitsinfrastrukturen

- Prof. Dr.-Ing. Felix Freiling

Lehrstuhl für Verteilte Systeme und Betriebssysteme

- Prof. Dr.-Ing. Wolfgang Schröder-Preikschat
- Dr.-Ing. Timo Hönig

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
- Dr.-Ing. Daniel Müller-Gritschneder

Lehrstuhl für Integrierte Systeme

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

Lehrstuhl für Rechnertechnik und Rechnerorganisation

- Prof. Dr. Michael Gerndt

Lehrstuhl für Wissenschaftliches Rechnen

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

Universität Bremen

Arbeitsgruppe für Informatikmethoden zur adaptiven Steuerung in Produktion und Logistik

- Prof. Dr. Nicole Megow

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 and to decide on the further roadmap of specific hardware for invasive computing.

Four working groups **Run-Time Requirement Monitoring and Enforcement**, **Memory Models, Architecture and Management**, **Benchmarking and Evaluation** and **Power and Thermal Aspects** 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. G. Snelting, Prof. Dr.-Ing. J. Teich*	A1
	Characterisation and Analysis of Invasive Algorithmic Patterns *Prof. Dr. M. Bader, Dr.-Ing. S. Wildermann*	A4
	Scheduling Invasive Multicore Programs Under Uncertainty *Prof. Dr. N. Megow*	A5
B: Architectural Research	Adaptive Application-Specific Invasive Micro-Architectures *Dr.-Ing. L. Bauer, Prof. Dr.-Ing. J. Becker, Prof. Dr.-Ing. J. Henkel*	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
	Generation of Distributed Monitors and Run-Time Verification of Invasive Applications *Dr.-Ing. D. Müller-Gritschneder, Prof. Dr.-Ing. U. Schlichtmann*	B4
	Invasive NoCs and Memory Hierarchies for Run-Time Adaptive MPSoCs *Prof. Dr.-Ing. J. Becker, Prof. Dr. sc. techn. A. Herkersdorf*	B5
C: Compiler, Simulation, and Run-Time Support	Invasive Run-Time Support System (*i*RTSS) *Dr.-Ing. L. Bauer, Prof. Dr.-Ing. J. Henkel, Dr.-Ing. T. Hönig, Prof. Dr.-Ing. W. Schröder-Preikschat*	C1
	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, Prof. Dr.-Ing. G. Snelting*	C5
D: Applications	Invasive Software-Hardware Architectures for Robotics *Prof. Dr.-Ing. T. Asfour, Prof. Dr.-Ing. W. Stechele*	D1
	Invasive Computing and HPC *Prof. Dr. M. Bader, Prof. Dr. H.-J. Bungartz, Prof. Dr. M. Gerndt*	D3
Z: Administration	Validation and Demonstrator *Prof. Dr.-Ing. J. Becker, PD Dr.-Ing. F. Hannig, Dr.-Ing. T. Wild*	Z2
	Central Services *Prof. Dr.-Ing. J. Teich*	Z
WG: Working Groups	Run-Time Requirement Monitoring and Enforcement *Prof. Dr.-Ing. F. Freiling, Dr.-Ing. T. Hönig, Dr.-Ing. D. Müller-Gritschneder*	WG1
	Memory Models, Architecture and Management *Prof. Dr. sc. techn. A. Herkersdorf, Prof. Dr.-Ing. W. Schröder-Preikschat, Prof. Dr.-Ing. G. Snelting*	WG2
	Benchmarking and Evaluation *Prof. Dr. M. Gerndt, Prof. Dr.-Ing. W. Stechele*	WG3
	Power and Thermal Aspects *Prof. Dr.-Ing. J. Henkel, Prof. Dr. N. Megow, Dr.-Ing. S. Wildermann*	WG4

4 Research Projects

A1: Basics of Invasive Computing

Gregor Snelting, Jürgen Teich

Joachim Falk, Frank Hannig, Pouya Mahmoody, Behnaz Pourmohseni, Tobias Schwarzer, Maximilian Wagner, Stefan Wildermann

The goal of Project A1 is to develop the theoretical foundations for *Run-Time Requirement Enforcement (RRE)* of invasive programs and to investigate the formal tractability of invasive X10 programs. The research in Project A1 focuses on (a) establishing the theory and semantics of RRE, (b) development of central and distributed enforcement techniques, (c) development of strict and loose enforcement techniques in support of hard and soft non-functional requirements, respectively, and (d) on applying theorem provers to formalise the semantics of invasive X10 programs. In the following, the results of our research in these areas in 2018 are presented.

Run-Time Requirement Enforcement (RRE)

Uncertainties in the input and execution environment of a program may cause variations in its non-functional properties, e. g. latency or power consumption. Often, multiple properties are required to stay within a user-specified interval (*requirements*) or are to be optimised (*objectives*) despite input and environment uncertainties. In this context, the term *run-time requirement enforcement (RRE)* denotes any technique that may be employed to enforce the satisfaction of requirements. In the following, we define an extension of the basic definition of RRE in [Tei17] to include also the optimisation of additional objectives:

Definition *Given a program p with input space I and state space of its execution environment Q. Let $O = R \cup G$ denote a set of non-functional properties of execution for p where R denotes the set of properties with requirements, each specified by an interval $[LB_o(p, Q, I), UB_o(p, Q, I)]$ where $o \in R$, and G denotes a set of objectives to be minimised.* ***Run-time***

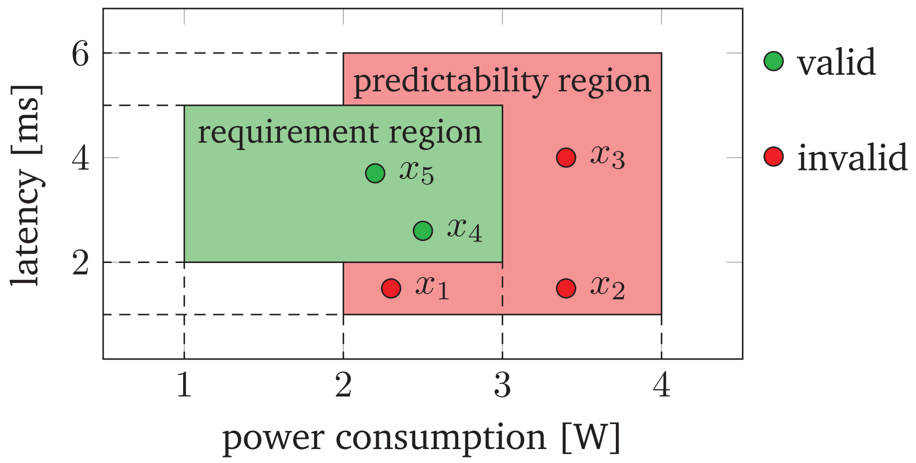


Figure 4.1: Space of two non-functional properties, power and latency, of an exemplary program. The predictability region (red box) and requirement region (green box) are given as well. Any observable design point, e. g. x_{1-5}, always lies within the predictability region. A *valid* design point, e. g. x_4 or x_5, always lies within the requirement region.

requirement enforcement *of p shall*

$$\begin{aligned}
&\forall(q,i) \mid q \in Q, i \in I : \\
\text{min.} \quad & G(p,q,i) = \left(o_1(p,q,i), ..., o_{|G|}(p,q,i)\right)^T \qquad \text{where } o_i(p,q,i) \in G \\
\text{such that} \quad & \forall o \in R : \; o \in [LB_o, UB_o]
\end{aligned}$$

A ***run-time requirement enforcement technique*** *hereby refers to any run-time technique that is capable of identifying and effectuating design points to be a)* valid *(w. r. t. the requirements) and b)* optimal *(w. r. t. the objectives) for each environment state $q \in Q$ and input $i \in I$ at run time.*
Example Consider a program p with a power requirement of $[1\,W, 3\,W]$, a latency requirement of $[2\,ms, 5\,ms]$, and an objective energy consumption to be minimised under variation of input $i \in I$ and core power mode $q \in Q$. The uncertainty analysis of p reveals a latency predictability (see **-predictability* definition[1]) of $L(p,Q,I) = [1\,ms, 6\,ms]$ and a power predictability of $P(p,Q,I) = [2\,W, 4\,W]$ under variation of $q \in Q$ and $i \in I$ (red box in Figure 4.1). For an exemplary input $i \in I$, Figure 4.1 shows power-latency trade-offs of five observable design points x_{1-5} under variation of power mode $q \in Q$. The green box in Figure 4.1 represents the so-called *requirement region* confined by the requirements. Let the energy consumption of design point x_i be calculated as

[1] J. Teich et al. "Language and Compilation of Parallel Programs for *-Predictable MPSoC Execution using Invasive Computing". In: *Proceedings of the 10th IEEE International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. Lyon, France, Sept. 21–23, 2016, pp. 313–320. DOI: 10.1109/MCSoC.2016.30.

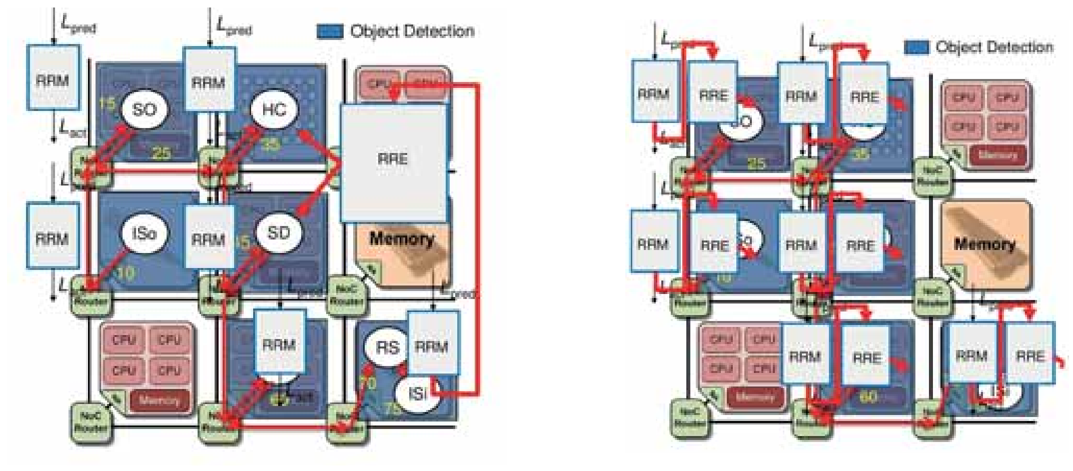


Figure 4.2: Example of a centralised (left) and distributed (right) run-time requirement enforcement for an object detection application comprising 7 actors. In centralised enforcement, a central RRE unit enforces the satisfaction of requirements based on its global knowledge on the application and monitored information gathered by the monitors (RRM). In distributed enforcement, a local RRE is generated per tile to enforce the satisfaction of a respective requirement sub-corridor.

$E_{x_i}(p,q,i) = P_{x_i}(p,q,i) \times L_{x_i}(p,q,i)$. Taking also the objective energy into account, an effective RRE technique would ensure that p will be executed on design point x_4 rather than x_5 once input $i \in I$ is observed or predicted, since $E_{x_4}(p,q,i) = 6.5\,\text{mJ}$ and $E_{x_5}(p,q,i) = 8.1\,\text{mJ}$. This can be effectuated by selecting the power mode corresponding to x_4 upon input $i \in I$. In the absence of the energy objective, both design points x_4 and x_5 would be equally desired.

Run-Time Requirement Enforcement Techniques

According to [Tei17], RRE techniques can be classified w. r. t. two aspects. First, in terms of the deployment and scope of knowledge for the enforcer(s), we distinguish between centralised and distributed enforcement techniques. In *centralised* enforcement, a single enforcer is generated for the whole program to realise the enforcement based on its global knowledge on the current state of the program and claimed resources, see Figure 4.2 (left). In *distributed* enforcement, multiple so-called local enforcers are generated, each enforcing a *sub-region* in the requirement region for one part of the program, e. g. an actor or a tile, based on local knowledge on the respective program part and hardware region, see Figure 4.2 (right). In the second classification, we distinguish between strict and loose enforcement techniques w. r. t. the strictness of the requirements. *Strict* enforcement techniques employ formal verification methodologies to guarantee that a violation of the

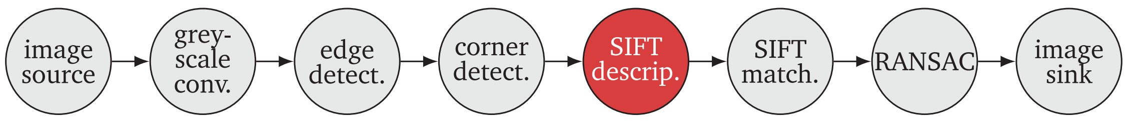


Figure 4.3: Actor graph of an object detection algorithm chain with SIFT-based feature description and matching with a highly input-dependent workload variation.

given user requirements will never occur, and therefore, become essential in case of hard requirements. *Loose* enforcement techniques, on the other hand, allow for a temporary violation of the requirements, and therefore, are relevant in case of soft requirements.

Distributed Enforcement of Soft Timing Requirements: First Results

This section presents first results of our investigations towards run-time requirement enforcement of invasive programs. As a use case, we consider an object detection program composed of eight actors which process input images in a pipelined fashion, see Figure 4.3. The execution latency of the SIFT description, SIFT match, and RANSAC actors depends on the content of the input image and increases proportionally to the number of corners in the input image which are identified by the corner detection actor. The execution latency of the remaining actors does not depend on the contents of the input image.

For our enforcement case study, we consider the SIFT description actor with a latency requirement $[LB_L, UB_L]$ and the single objective energy consumption E to be minimised given an uncertain space of input images reflected by $I = [N_{\min}, N_{\max}]$ where $N_{\min}$ and $N_{\max}$ denote the minimum and maximum number N of corners that may exist in an input image that are responsible for creating a high variability in workload to be processed per image. Thus, the enforcement problem is formulated as:

$$\forall N \in [N_{\min}, N_{\max}] : \min\ E(p, N) \mid L(p, N) \in [LB_L, UB_L] \qquad (4.1)$$

One solution to incorporate enforcers already at the program level is to integrate a so-called *enforcer actor* into the actor graph model, see Figure 4.4 for illustration. Whereas enforcers cannot change the input space I to counteract workload variation, they can affect the state space Q of a program's environment. In our case, the SIFT description actor is mapped to a 5-core tile with 20 power modes to allow dynamic voltage and frequency scaling. Now, for proper enforcement of the above latency requirement, we assume that for each input image, the workload of the actor can be distributed across up to four cores while the last core

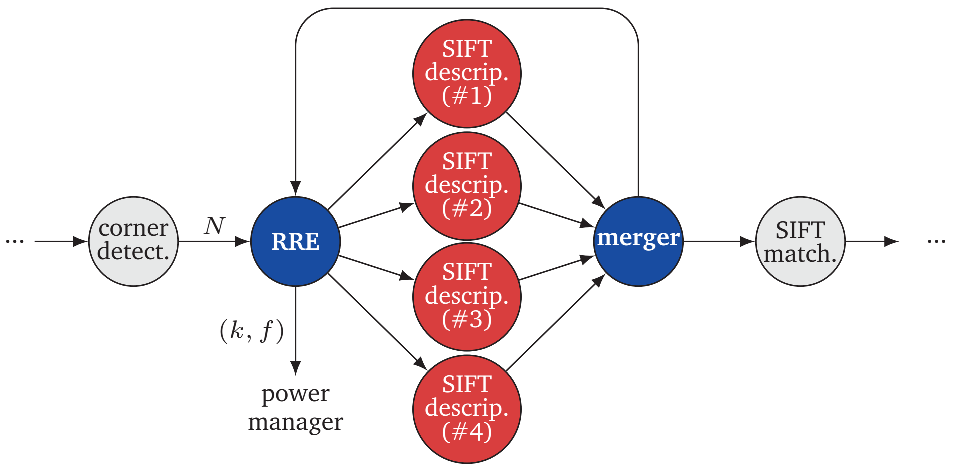


Figure 4.4: Modifications introduced in the actor graph from Figure 4.3 to introduce the enforcement support for the SIFT description actor.

is reserved to execute the enforcer. At run time, the enforcer selects a) the power mode of the tile and b) the number of active cores to tune the latency and energy consumption of the actor per input image to realise the enforcement as described in Eq. (4.1). The enforcer thereby controls the power management unit. The enforcer also c) distributes the workload among the parallel instances of the actor on the active cores and merges their results after the processing which is then passed to the next actor, i. e. SIFT matching.

We use Eq. (4.2) to calculate the latency of the SIFT description actor when processing $N \in [N_{\text{min}}, N_{\text{max}}]$ input image corners on $k \in [1, 4]$ cores running at frequency f (representing a power mode).

$$L(N, k, f) = L(1, 1, f_{\text{max}}) \cdot \left\lceil \frac{N}{k \cdot e(k)} \right\rceil \cdot \frac{f_{\text{max}}}{f} \tag{4.2}$$

Here, $L(1, k, f_{\text{max}})$ denotes the average latency of processing $N = 1$ corner using $k = 1$ core at maximum frequency $f = f_{\text{max}}$ which is derived by simulation using InvadeSIM[2] for a set of 9,100 input images from different domains. In Eq. (4.2), $e(k) \in (0, 1]$ denotes the efficiency of workload distribution on k active cores which we assume to be $e(k) = 1$. Accordingly, the maximum number of image corners that can be processed within the latency bound UB_L is calculated using Eq. (4.3).

$$N_{\text{max}}(k, f) = \left\lfloor UB_L \cdot \frac{f}{f_{\text{max}}} \cdot \frac{k \cdot e(k)}{L(1, 1, f_{\text{max}})} \right\rfloor \tag{4.3}$$

[2] 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. DOI: `10.1145/2744769.2744840`.

A1

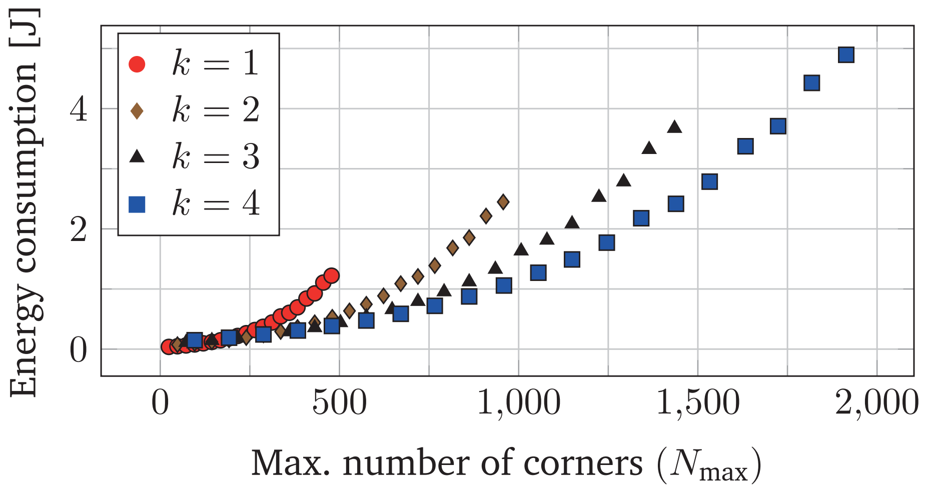


Figure 4.5: Energy consumption and maximal number N_{max} of corners that can be computed within an upper latency bound of UB_L by the enforced SIFT description actor for various configurations (k, f) with $k \in [1, 4]$ denoting the number of enforced active cores and $f \in \{f_1, ..., f_{20}\}$ denoting their clock frequency.

Finally, given the power consumption of $k = 1$ core when operating at frequency f, denoted by $P(1, f)$, the energy consumption of the actor for processing an image with N corners on k cores running on frequency f is calculated using Eq. (4.4).

$$E(N, k, f) = L(N, k, f) \cdot P(1, f) \cdot k \qquad (4.4)$$

To identify the best (k, f) configurations for enforcement of the latency requirement while minimising energy, we iterate the design space of $\{(k, f) \mid k \in [1, 4] \wedge f \in \{f_1, ..., f_{20}\}\}$, and for each configuration (k, f), evaluate a) the maximum number of corners N_{max} that can be processed within the latency upper bound UB_L using Eq. (4.3) and b) the respective energy consumption using Eq. (4.4). Figure 4.5 illustrates the energy and N_{max} of the explored (k, f) configurations. We extract the Pareto-optimal configurations as listed in Table 4.1 which are then used by the enforcer at run time to select the optimal (k, f) configuration for each input image according to its number N of corners.

To implement the enforced variant of the SIFT description actor, we adapt the actor graph from Figure 4.3 with the modifications depicted in Figure 4.4. In the adapted actor graph, the output of the corner detection actor (which also extracts the number N of corners in the current image) is first provided to the *RRE actor* which selects the optimal (k, f) configuration according to Table 4.1 and distributes the workload between the active instances of the SIFT description actor. Once the execution of the instances is finished, the *merger actor* merges

Table 4.1: (k, f) configurations minimising energy consumption in dependence of the number N of corners in an input image.

N	k	f	N	k	f	N	k	f	N	k	f
23	1	f_1	287	4	f_3	671	4	f_7	1,342	4	f_{14}
47	2	f_1	359	3	f_5	766	4	f_8	1,438	4	f_{15}
119	1	f_5	383	2	f_8	862	4	f_9	1,532	4	f_{16}
143	3	f_2	431	3	f_7	959	4	f_{10}	1,632	4	f_{17}
167	1	f_7	479	4	f_5	1,005	4	f_{11}	1,724	4	f_{18}
215	3	f_3	503	3	f_7	1,149	4	f_{12}	1,818	4	f_{19}
239	2	f_5	575	4	f_6	1,246	4	f_{13}	1,913	4	f_{20}

the results from the parallel instances, forwards the merged data to the SIFT matching actor, and notifies the RRE actor to be ready to modify the (k, f) configuration for the next image if necessary.

Formalisation of Invasive X10

Apart from the formal verification of RRE techniques, Isabelle/HOL shall be used to verify requirements on parallel applications formally. Assuming a claim invasion succeeds, we can use the specified constraints and requirements in proofs about the application running on the claim. We want to assess non-functional properties and bounds, like memory consumption (tile-local and global memory), communication times, and load. For this, an operational semantics of X10 is needed upon which to build additional models for invasive computing: constraints, requirements, claims, and infect/invade/retreat all need to be modelled along with their semantics to have a basic formal model of invasive X10.

As a first step, one might assume that being based on Java historically, X10 could share the Java Memory Model. While it is true that X10 can be compiled to JVM bytecode, however, there are compelling reasons why X10 should be regarded on its own. First and foremost, X10 has a more structured approach to multiprocessing, only allowing parent activities to wait for their children, which (for instance) already makes circular waiting situations impossible. Next, it has built-in semantics for its *partitioned global address space* (PGAS) model, where access to data rooted at another shared memory domain (which X10 calls *places*) implies a deep copy of the corresponding object graph. Thus, an operational semantics of X10 based on the semantics of Java would have to define the semantics of these basic language primitives in terms of their (potentially complex) implementation on the JVM, thereby certainly losing conciseness and probably losing formal tractability.

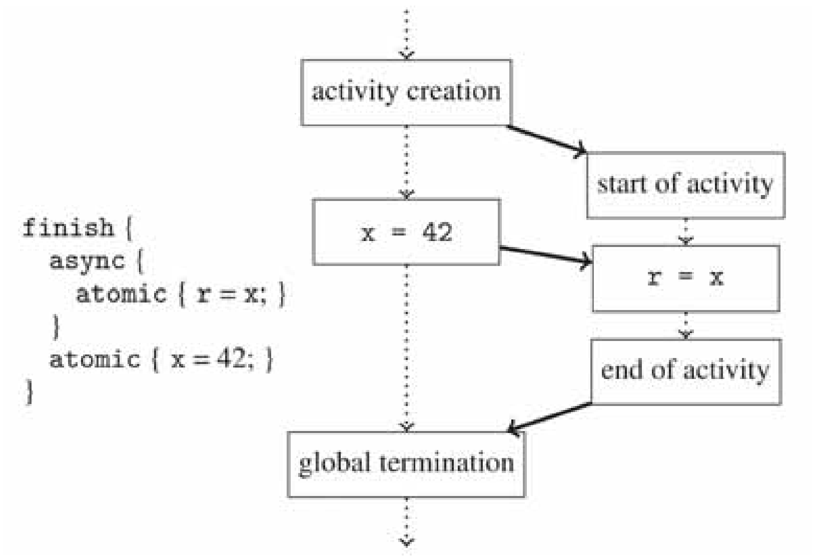


Figure 4.6: An example execution of an activity as written in the code on the left. Each box is an action. The thick arrows are *synchronise-with* edges and the dotted arrows show *program order*. The transitive closure of both together form the *happens-before* relation. Taken from [Zwi16].

Our model of X10 has to define which guarantees it grants in the case of data-race freedom and in the presence of data races. One common choice for programming languages is to guarantee *sequential consistency* for data-race-free programs:

Definition *A program is sequentially consistent if the result of any execution is the same as if the operations of all the processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program.*

Once the basic guarantees of the memory model have been formally modelled, the resulting concepts of parallelism can be investigated. For instance, Zwinkau[3] proposed a sequentially consistent memory model and defines a variant of the *happens-before* relation for X10 programs strongly based on the structured model of parallelism employed by X10. An example of the different relations introduced by Zwinkau can be seen in Figure 4.6.

Further Results

Together with Tulika Mitra and Lothar Thiele, Jürgen Teich has edited a special issue on Time-Critical Systems Design in IEEE Design & Test [MTT18]. In the context of our work on hybrid application mapping

[3]A. Zwinkau. “A Memory Model for X10”. In: *Proceedings of the 6th ACM SIGPLAN Workshop on X10* (Santa Barbara, CA, USA). X10 2016. ACM, 2016, pp. 7–12. DOI: 10.1145/2931028.2931031.

(HAM) [Wei+18a], we have investigated in [Ric+18] techniques for finding a run-time mapping of a distributed application given by an actor graph using parallelisation techniques for SAT-solvers as well as exploiting architecture symmetries, see also [Sch+18a]. [Hen+18] describes an approach that exploits adaptive techniques to guardbanding to provide higher resource utilisation without risking any thermal violation. Still, as a degradation of an application in terms of performance may not be tolerable for programs with hard real-time requirements, concepts for dynamic Real-time Mapping Reconfiguration (RMR) are proposed based on the previously described HAM approach. Finally, the dissertation of Andreas Weichslgartner has been extended and accepted as a book by Springer [Wei+18b]. This comprehensive book is one of the first on the general topic of invasive computing.

Publications

[Hen+18] J. Henkel, J. Teich, S. Wildermann, and H. Amrouch. “Dynamic Resource Management for Heterogeneous Many-Cores”. In: *Proceedings of the International Conference on Computer-Aided Design (ICCAD)* (San Diego, CA, USA). ACM, Nov. 5–8, 2018, 60:1–60:6. DOI: 10.1145/3240765.3243471.

[MTT18] T. Mitra, J. Teich, and L. Thiele. “Guest Editors’ Introduction: Special Issue on Time-Critical Systems Design”. In: *IEEE Design and Test of Computers* 35 (2018), pp. 5–7. DOI: 10.1109/MDAT.2018.2796037.

[Ric+18] V. Richthammer, T. Schwarzer, S. Wildermann, J. Teich, and M. Glaß. “Architecture Decomposition in System Synthesis of Heterogeneous Many-Core Systems”. In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[Sch+18a] T. Schwarzer, A. Weichslgartner, M. Glaß, S. Wildermann, P. Brand, and J. Teich. “Symmetry-eliminating Design Space Exploration for Hybrid Application Mapping on Many-Core Architectures”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37.2 (Feb. 2018), pp. 297–310. DOI: 10.1109/TCAD.2017.2695894.

[Sou+18] É. Sousa, M. Witterauf, M. Brand, A. Tanase, F. Hannig, and J. Teich. “Invasive Computing for Predictability of Multiple Non-functional Properties: A Cyber-Physical System Case Study”. In: *Proceedings of the 29th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Milan, Italy). IEEE, July 10–12, 2018. DOI: 10.1109/ASAP.2018.8445109.

[Tei17] J. Teich. *Run-Time Monitoring and Enforcement of Non-functional Program Properties of Invasive Programs: Terms and Definitions*. Technical Report 01-2017. Erlangen, Germany: Hardware/-Software Co-Design, Friedrich-Alexander-Universität Erlangen-Nürnberg, Department of Computer Science, Jan. 2017.

[Tei18a] J. Teich. “Hybrid Application Mapping for NoC-Based MPSoCs with Guarantees on Timing, Reliability and Security”. Invited Talk University of New South Wales, Australia. July 31, 2018.

[Tei18b] J. Teich. “Methodologies for Application Mapping for NoC-Based MPSOCs”. Keynote, Adaptive Many-Core Architectures and Systems workshop, York, UK. June 14, 2018.

[Tei18c] J. Teich. “Run-Time Application Mapping in Many-Core Architectures”. Invited Talk National University of Singapore. Aug. 24, 2018.

[Wei+18a] A. Weichslgartner, S. Wildermann, D. Gangadharan, M. Glaß, and J. Teich. “A Design-Time/Run-Time Application Mapping Methodology for Predictable Execution Time in MPSoCs”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 17.5 (Nov. 2018), 89:1–89:25. DOI: `10.1145/3274665`.

[Wei+18b] A. Weichslgartner, S. Wildermann, M. Glaß, and J. Teich. *Invasive Computing for Mapping Parallel Programs to Many-Core Architectures*. Springer, Jan. 15, 2018. DOI: `10.1007/978-981-10-7356-4`.

A4: Characterisation and Analysis of Invasive Algorithmic Patterns

Michael Bader, Stefan Wildermann

Tobias Schwarzer, Behnaz Pourmohseni, Alexander Pöppl, Joachim Falk

Project A4's mission is to explore and establish application characterisation in the invasive computing paradigm and to investigate and evaluate how algorithms and applications may exploit and profit from the resulting predictability features.

A major challenge is that our study of invasive proxy applications in Phase II and the experiences gained within the embedded system and HPC communities reveal that application execution is getting increasingly dynamic:

- Workload scenarios and execution phases of an application vary strongly, such that a static worst- or average-case characterisation for a single scenario becomes inadequate.
- Performance behaviour on modern hardware is highly uncertain, as external influences, differences in manufacturing and measures to enforce power limits or energy budgets affect quality numbers in an unforeseeable way.
- Hardware faults are expected to occur more often thus making resources temporally or, due to manufacturing variability and ageing, even permanently unavailable.

The enforcement of non-functional requirements for the execution of one operating point, as investigated in Project A1, concentrate on modifications of the claimed resources, and thus always happen within a corridor defined by the respective operating point. Project A4 expands this approach beyond the boundaries of a single operating point by considering algorithmic adaptations of the application as well as modifications of the resource claim. Our goal is to provide optimised and robust application execution in the context of above aspects by characterising and enforcing adaptation of the application behaviour (e. g. algorithmic variants), the application structure (i. e. the actor model),

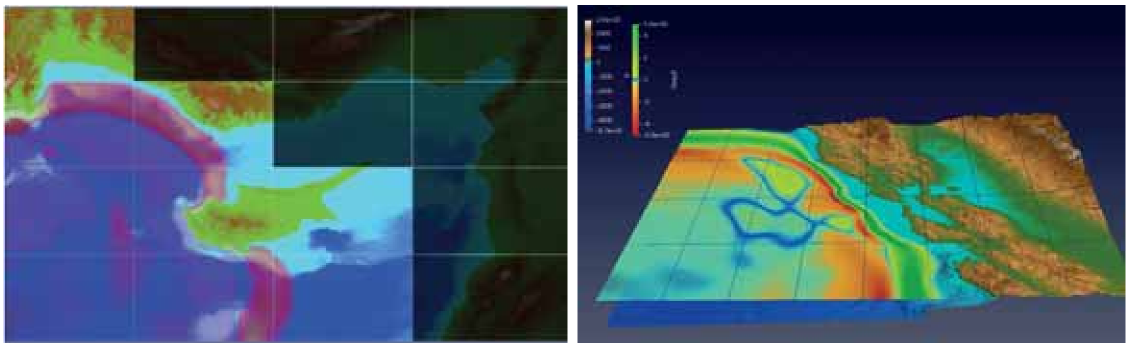


Figure 4.7: Tsunamis used to showcase SWE-X10 and Pond. The left figure shows the scenario used in the demonstration on the prototype platform, an artificial tsunami in the Western Mediterranean Sea. The right figure depicts a sample scenario computed in Pond on a single node, an artificial tsunami in the vicinity of the San Francisco Bay Area.

and the set of allocated resources by means of re-invasion and run-time requirement enforcement (RRE) to such variances and fault scenarios.

Shallow Water on a Deep Technology Stack SWE-X10, the tsunami proxy application developed in Project A4 was used to demonstrate the invasive application design flow as well as the benefits of co-designing hardware and software in concert with most of the other projects of the CRC. At the core was the acceleration of the f-Wave Riemann solver using an *i*-Core special instruction. Using this, and a triple buffering scheme using the tile-local memory, we were able to speed up the computation of a single patch by a factor of 5 [Pöp+18]. To optimise the overall application performance, a simulative Design Space Exploration (DSE) was employed. Using InvadeSIM, we determined operating points that optimise performance for a given hardware configuration. The results of this collaboration were used as one of the two large demonstrations in the DFG review in January 2018, see also the report section of Project Z2.

Pond: An actor-based shallow water code for UPC++ While X10 is a powerful and expressive language, its prototypical nature brings limitations in terms of potential outreach and available tooling. To make the actor-based programming model available to a broader HPC community, we implemented Actor-Upcxx, and the shallow water proxy application *Pond* on top of it. This development was part of a collaboration with the group of Scott Baden at Berkeley Lab (lead developer of the UPC++ programming model), which included two research visits (both in 2018) of Alexander Pöppl in Berkeley (see Figure 4.7 showing Tsunami simulations using both implementations).

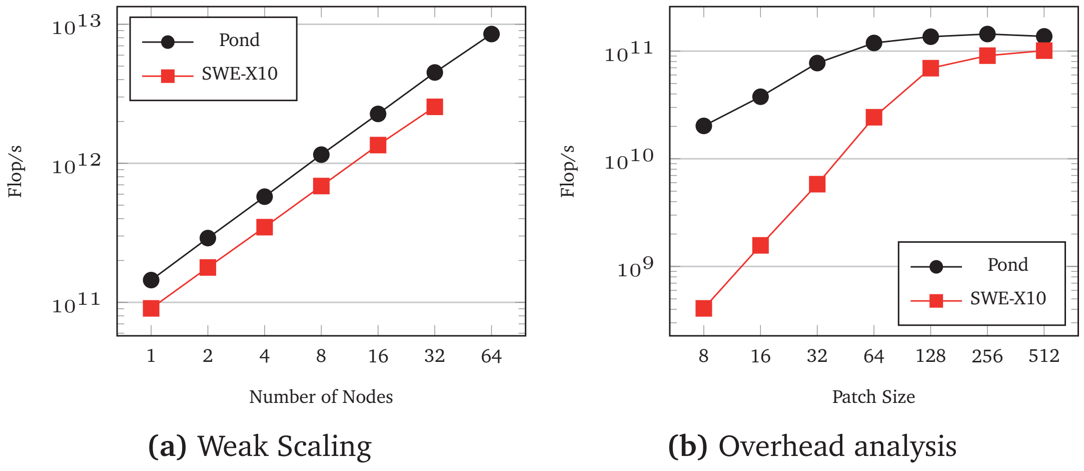


Figure 4.8: Performance comparison between SWE-X10 and Pond

Actor-Upcxx implements the same semantics as actorX10: Actors are triggered by changes in their ports. Whenever another actor writes to a connected channel or reads from it, the actor gets notified and its `act` method is called. Internally, the actor framework uses UPC++ remote procedure calls (RPCs) and local procedure calls (LPCs) with callbacks to insert the data into the channel, to obtain it, and to notify the port on the other side.

Compared to X10, we were able to get improved performance and less management overhead. Figure 4.8a shows a weak scaling test on the Haswell nodes of the Cori supercomputer. For the test, we scaled from 1 to 64 nodes, with a working set of 512×512 grid cells per logical core, resulting in a total load of 4,096×4,096 cells per node. It can be seen that both applications scale linearly with the number of nodes, but Pond consistently outperforms SWE-X10.

In a second test, we looked at framework overhead. Here, we kept the actor graph, i. e. the number of actors and their connections, constant, but varied the individual actors' workloads by changing the number of total grid points for the simulation, from 512×512 down to 8×8. The test was again conducted on Cori, on a single node (see Figure 4.8b). Here, the difference in overhead is clearly visible. While SWE-X10 is competitive for large patch sizes, performance declines for smaller patch sizes (esp. $\leq$ 128×128), whereas Pond performance declines to a much lesser degree and remains useable for small patch sizes. However, features like local time stepping, lazy activation, or mesh refinement are not implemented in Pond.

Invasive Algorithmic Patterns

In SWE-X10, to date, the size of a time step has been fixed at application start-up based on the simulation domain and the initial displacement. Actors then used this time step to compute the updates to their unknowns. This time stepping scheme becomes problematic once scenarios exploit strong mesh refinement of if there are large inundation areas to be computed. During wetting and drying, there are large variations in the wave speeds computed during the net update computation. Via the Courant-Friedrichs-Lewy condition, these variations in wave speeds translate to time step size limits for the respective mesh cells. Similarly, refined mesh cells require smaller time steps.

In a global time stepping mechanism, all grid cells share a global time step size, which requires all simulation actors to coordinate with a coordination actor that determines the largest allowed time step size after each net update computation. In contrast, using a local time step size for each actor, so-called *local time stepping*, at the same time removes the global coordination, but also poses additional challenges on the actor approach. To simplify the implementation, we chose to limit the time step sizes to fixed multiples: $2^n t_{\text{orig}}, n \in \mathbb{Z}$. This ensures that the patches regularly end up at the same time step again.

Scenario-aware hybrid application mapping

An overview of our hybrid application mapping (HAM) flow is illustrated in Figure 4.9. Rather than mapping onto concrete resources of the architecture, we cluster tasks and assign these *task clusters* onto *resource types* or on *resource classes*[4]. Furthermore, for all messages sent from one task cluster to another one, we assign a maximal allowed hop length $hops$ of the NoC route and the amount of time slots bw that shall be reserved on the links along this route. Based on compositional timing analysis or simulation, it is possible to determine the latency, throughput, and power consumption for such *equivalence classes* of mappings–equivalent in terms of quality number, but also resource and power needs.

Design Space Exploration (DSE) We developed and apply DSE [Wei+18b; Wei+18a] following a hybrid application mapping approach to evaluate multiple of such mappings, optimising for their resource requirements and further objectives like, e. g. power consumption. The result is thus

[4]In case of a heterogeneous manycore platform, each combination of resource type and possible power mode form one class.

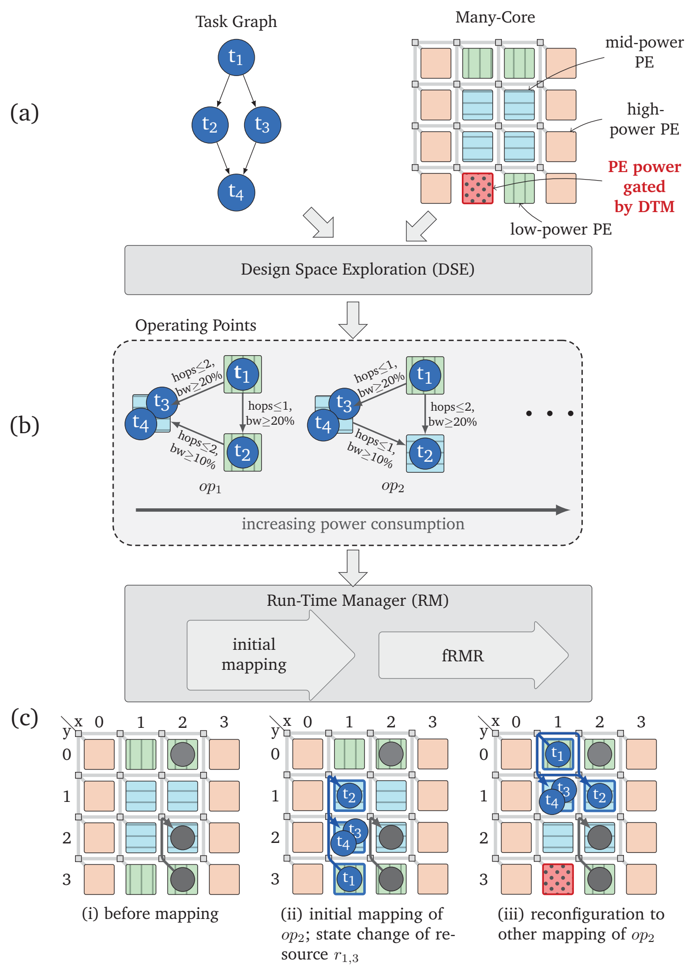


Figure 4.9: Hybrid application mapping (HAM) flow including the Real-time Mapping Reconfiguration (RMR) on a NoC-based target platform.

a set of Pareto-optimal (or at least non-dominated) mapping classes, which we denote *operating points*. Figure 4.9(b) illustrates such a set of operating points for the example task graph shown in Figure 4.9(a). Operating point op_1 with the lowest power consumption requires two low-power PEs and one mid-power PE interconnected by NoC routes as specified according to the directed edges between them. Task sets $\{t_1\}$, $\{t_2\}$, and $\{t_3, t_4\}$ form the three task clusters which are then to be assigned to the respective resource classes at run time.

Increasing the scalability of DSE Schwarzer et al. [Sch+18a] propose new methods to enhance the scalability of DSE when applied for hybrid application mapping. The approaches SElim and SMT remove symmetries from the search space and SMT even applies learning strategies to remove infeasible design options from the DSE's search space over time. Figure 4.10 shows the results of one experiment considering DSE for a 24×24 symmetric NoC.

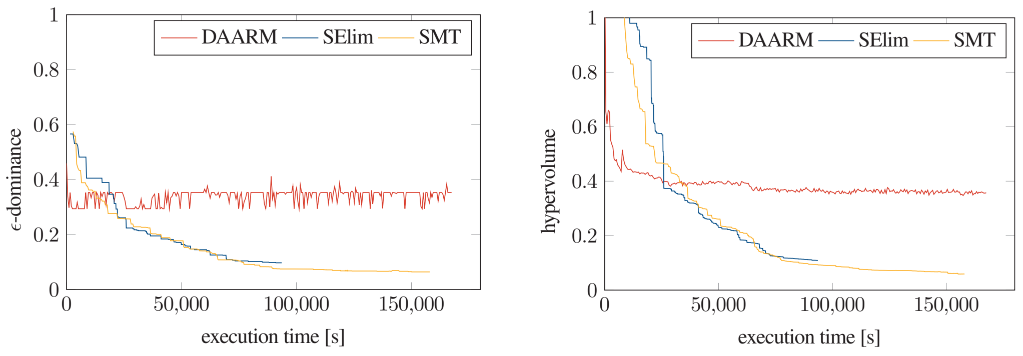


Figure 4.10: Shown are the achieved ϵ-dominance (left) and hypervolume (right) values of the three approaches known as DAARM, SElim, and SMT over the execution time for the auto-industry application evaluated on a 24×24 symmetric NoC. While giving all approaches the same amount of optimisation time, DAARM converges faster but to considerably inferior quality sets. All approaches employ *archives* to store the set of best found, non-dominated design points at each time step. However, the memory is limited so when there are too many non-dominated design points, some of the points in the archive have to be removed for the archive to fit into the available memory. DAARM, in particular, produces many non-dominated design points, and thus, has to apply such strategies quite often. Removing design points from the archive also means that the ϵ-dominance and hypervolume can worsen again.

Furthermore, Richthammer et al. [Ric+18] has shown how to enhance scalability of DSE even further by decomposing the search space according to symmetries in the architecture.

Support for re-invade

Once such an application is then dispatched for execution, the information from the DSE is handed over to the Run-time Management (RM) as illustrated in Figure 4.9(c). It is now the task of RM to find an initial concrete mapping according to one of these operating points, which we call *embedding*. Furthermore, during application execution, it could become necessary for the run-time system to dynamically switch between operating points of an application (e. g. freeing occupied/hotspot resources). This work package considers such adaptation techniques for providing (a) resource-aware, (b) fault-aware, and (c) workload-aware transition between operating points.

Run-time Embedding Determining an embedding involves solving a complex constraint system. However recently, Schwarzer et al. have shown that it is possible to determine feasible embeddings at run time within a few milliseconds also for manycore systems with more than 100 PEs [Sch+18b]. Figure 4.11 shows the distribution of all execution

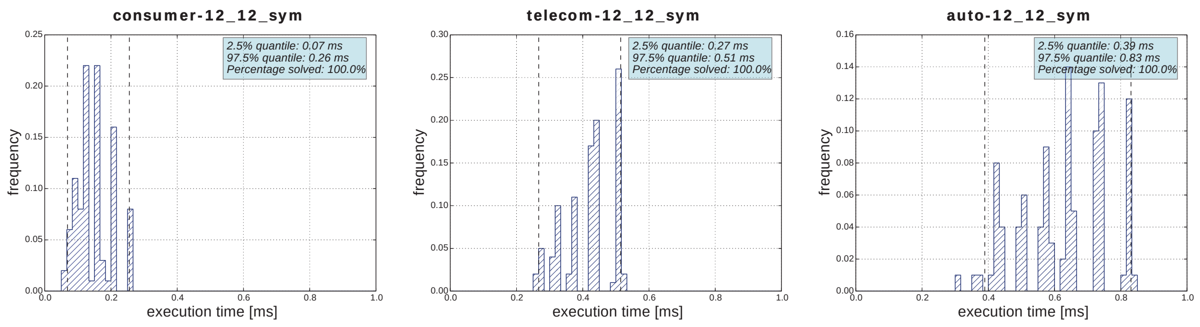


Figure 4.11: Distributions of solving times [ms] for run-time embedding of applications from the E3S benchmark suite on a *12* $\times$ *12-random* NoC.

times measured for targeting a 12$\times$12 NoC and for executing test cases from the E3S benchmark suite. We used the InvadeSIM simulator to perform the measurements. Here, the solver was executed on PEs of the NoC with processor frequencies set to 1,000 MHz. The figure shows that on such processor cores, feasible run-time mappings may be determined in the order of 1 ms using our approach. We believe that this amount of time is affordable when admitting a new application to a manycore architecture.

Reconfiguration Analysis Transition between operating points involves the allocation of new resources as well as synchronised and consistent migration of one or multiple actors to the new resource claim (acc. to the operating point mapping). In [Pou+17], we propose a formal analysis for operating point transition which (a) determines efficient migration routes for migrating actors via the NoC with minimal allocation overhead and migration latency, and (b) analytically derives upper-bound guarantees on the reconfiguration latency. It is thus possible for the run-time system to take the transition costs (in space and time) between operating points into account when performing run-time decisions. This will particularly also serve as basis for our RRE strategies in Phase III.

Power/Temperature-aware Run-time Management Thermal and reliability awareness are the biggest challenges of managing the execution on manycore platforms. In [Hen+18], we discussed reliability optimisation for thermally-constrainted manycore systems. We demonstrated that adaptive guardbanding techniques to overcome ageing effects are required to maximise resource utilisation without thermal violations. The experimental results show that such techniques may lead to a performance degradation of applications executed on the modified core.

While this could be (temporally) tolerated for best-effort applications, it is not tolerable for applications with hard real-time requirements as a performance degradation could lead to deadline misses. Therefore, we have introduced a mechanism for dynamic Real-time Mapping Reconfiguration (RMR) based on the results on HAM and reconfiguration analysis as described above. As Figure 4.9(c) is motivating, RMR enables to dynamically adapt the mapping of a real-time application at run time with the goal of freeing cores that are affected by thermal management. Reconfiguration takes place without violating the real-time constraints. The now freed cores can afterwards be utilised by best-effort applications, turned off for thermal management, or modified by adaptive guardbanding techniques, see project B3's report for further details.

Publications

[Hen+18] J. Henkel, J. Teich, S. Wildermann, and H. Amrouch. "Dynamic Resource Management for Heterogeneous Many-Cores". In: *Proceedings of the International Conference on Computer-Aided Design (ICCAD)* (San Diego, CA, USA). ACM, Nov. 5–8, 2018, 60:1–60:6. DOI: `10.1145/3240765.3243471`.

[Pöp+18] A. Pöppl et al. "Shallow Water Waves on a Deep Technology Stack: Accelerating a Finite Volume Tsunami Model using Reconfigurable Hardware in Invasive Computing". In: *Euro-Par 2017: Proceedings of the 10th Workshop on UnConventional High Performance Computing (UCHPC 2017)*. Ed. by D. B. Heras et al. Lecture Notes in Computer Science (LNCS). Santiago de Compostela, Spain: Springer International Publishing, 2018, pp. 676–687.

[Pou+17] B. Pourmohseni, S. Wildermann, M. Glaß, and J. Teich. "Predictable Run-Time Mapping Reconfiguration for Real-Time Applications on Many-Core Systems". In: *Proceedings of the International Conference on Real-Time Networks and Systems (RTNS)*. Outstanding paper award. Grenoble, France: IEEE, 2017. DOI: `10.1145/3139258.3139278`.

[Ric+18] V. Richthammer, T. Schwarzer, S. Wildermann, J. Teich, and M. Glaß. "Architecture Decomposition in System Synthesis of Heterogeneous Many-Core Systems". In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[Sch+18a] T. Schwarzer, A. Weichslgartner, M. Glaß, S. Wildermann, P. Brand, and J. Teich. “Symmetry-eliminating Design Space Exploration for Hybrid Application Mapping on Many-Core Architectures”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37.2 (Feb. 2018), pp. 297–310. DOI: 10.1109/TCAD.2017.2695894.

[Sch+18b] T. Schwarzer et al. “On the Complexity of Mapping Feasibility in Many-Core Architectures”. In: *Proceedings of Multicore/Many-core Systems-on-Chip (MCSoC-2018)*. Sept. 12–14, 2018.

[Wei+18a] A. Weichslgartner, S. Wildermann, D. Gangadharan, M. Glaß, and J. Teich. “A Design-Time/Run-Time Application Mapping Methodology for Predictable Execution Time in MPSoCs”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 17.5 (Nov. 2018), 89:1–89:25. DOI: 10.1145/3274665.

[Wei+18b] A. Weichslgartner, S. Wildermann, M. Glaß, and J. Teich. *Invasive Computing for Mapping Parallel Programs to Many-Core Architectures*. Springer, Jan. 15, 2018. DOI: 10.1007/978-981-10-7356-4.

A5: Scheduling Invasive Multicore Programs Under Uncertainty

Nicole Megow

Bertrand Simon

Project A5 is a new project that joined in the third funding phase. It started in September 2018. We report here on the project goals and first preliminary results.

The goal of Project A5 is to develop algorithms that can handle uncertain input data. We design and mathematically analyse algorithms for scheduling and resource management using the *invasive computing paradigm*. Our methods shall give performance guarantees concerning the predictability and also quantify how hardware and software requirements affect the performance and predictability. This may reveal also optimisation potential in the systems architecture or algorithms.

This project is of foundational character and aims for theoretical guarantees by building on methods from algorithms theory and mathematical optimisation. Our main focus lies on *provable worst-case guarantees* and *coping with uncertainty*. When predictability is crucial, e. g. in safety-critical applications, most multicore systems still rely on single-core usage. The reason is that the current state-of-the-art approaches in real-time scheduling do not capture the difficulties in scheduling parallel workloads in a predictable way. In this project, we will develop algorithms with guarantees on predictability and resource utilisation as is required to exploit the full power of parallel computing and particularly the benefits of invasive computing also for safety-critical applications. We expect to develop algorithms that are not only efficient from a theoretical standpoint but can be efficiently implemented in practice.

Speed-scaling for power management

Speed scaling (or frequency/voltage scaling) is the main technique for power management, in both academic research and practice. It involves *dynamically* changing the speed of a processor which is allowed by current microprocessors. Major research questions ask for dynamic algorithms that determine a schedule for a given set of tasks and also decide at which speed $s \geq 0$ the processor(s) shall run at any time. Running a processor at a certain speed requires a certain amount of power. Power is typically modelled as a monomial (convex) function of speed, $P(s) = s^\alpha$ with a small constant $\alpha > 1$. Given an energy budget E, we ask for the optimal power (and thus speed) distribution and corresponding schedule that minimises some scheduling cost function, e. g. makespan, sum of completion times/flow time, lateness. We may also ask for the converse: given a bound on some scheduling cost function, compute the optimal power distribution and schedule that minimises energy consumption.

We currently focus on the problem of scheduling a set of tasks with precedence constraints on multiple processing units while minimising the makespan under a given energy budget. For this particular problem, a polynomial-time 2-approximation algorithm is already known, which means that the makespan obtained by this algorithm is guaranteed to be at most twice the optimal makespan under the same energy budget. To put this result into perspective, under some conjectures, no polynomial-time algorithm can have a substantially better guarantee. Nevertheless, this algorithm requires solving a *convex program* – an operation which, although polynomial, may be too expensive to be carried out in practice. One of our goals will be to design an algorithm having the exact same guarantees, but requiring much less computing time. We noticed that this problem shares some properties with studies conducted in a different context. This link may be particularly useful to design a low-complexity algorithm assuming that the precedence constraints belong to a particular class of graphs, a series-parallel graph.

In collaboration with Project A1, we explore how theoretically guaranteed speed-scaling scheduling algorithms can be used in invasive computing.

Real-time scheduling theory

The field of real-time scheduling theory is concerned with scheduling problems that arise specifically in the design and implementation of real-time and embedded computer application systems. Such systems operate recurrent workloads and infinite task sequences, which is a crucial difference to standard scheduling problems considered by the algorithms and mathematical programming community. Real-time scheduling research is naturally practically oriented and an immense body of literature deals with models and solution methods which are experimentally evaluated. A comparatively small sub-community investigates real-time task systems from a theoretical point of view aiming at complexity results and provable performance guarantees as they are fundamental in algorithmics. We will focus on *sporadic task systems* to be scheduled on several identical machines. In such a system, each task has a period which corresponds to the minimum separation between the release of two successive operations of this task. A task system therefore implicitly defines an infinite set of instances. The future release dates depending on external events. Scheduling decisions have to be made without knowing when the next occurrences will be released. In critical systems, it is required that all instances will be scheduled without deadline miss. Deciding whether this property is true of a task system is then of utter importance at design time. An interesting question which remained open until a few years ago is the following: If all instances of a task system can be scheduled when the release times are known in advance, is it possible to schedule these instances without this knowledge? The answer is negative, but it remains to quantify the power of knowing the release dates in advance. Such studies exist for arbitrary instances, but not for sporadic task systems. In another direction, we will also consider harmonic task sets, in which the tasks' periods divide each other. The answer to the above question is not yet known for such systems.

Security in invasive scheduling

In collaboration with Project C5, we investigate security in invasive scheduling. Scheduling decisions (i. e. at a given time, deciding which tasks are executed) can be used by malicious users as a source of covert communication. If the scheduling policy is known, two tasks that may not be allowed to communicate for security reasons can indeed adapt their behaviour in order to exchange information. Depending on the scheduling policy, the amount of information that can be communicated may differ. We aim at analysing this property for common scheduling policies and propose secure strategies.

Thermal-aware mapping

In collaboration with Project B3, we study thermal-aware mapping. Executing a job on a chip component increases its temperature and the one of nearby components. This effect depends on multiple factors such as the power consumed by a core, the local thermal conductivity or the ambient temperature. If the temperature of a component exceeds a certain threshold, dynamic thermal management techniques are triggered in order to avoid overheating. Triggering these techniques results in performance losses and must then be avoided. We will consider the theoretical aspects of the problem consisting of mapping tasks to components and deciding a maximum power that each component is allowed to consume while ensuring that no component exceeds the threshold temperature. This problem has already been studied in Project B3, but a deeper theoretical understanding is required.

B1

B1: Adaptive Application-Specific Invasive Micro-Architectures

Lars Bauer, Jürgen Becker, Jörg Henkel

Marvin Damschen, Tanja Harbaum, Fabian Lesniak

Project B1 investigates mechanisms that provide run-time adaptivity: in the microarchitecture (μArch) and by using a run-time–reconfigurable fabric. We propose concepts and methods that allow invading the reconfigurable fabric and μArch within the invasive core (*i*-Core). The *i*-Core is an integral part of invasive architectures as investigated in project area B. Therefore, we integrated our *i*-Core prototype as well as support hardware into the new demonstration platform (proFPGA) and provided a tool flow for partial run-time reconfiguration. In the following, we briefly describe our last results of the second funding phase and the recent activities of the third funding phase.

Auto-SI and universal accelerator

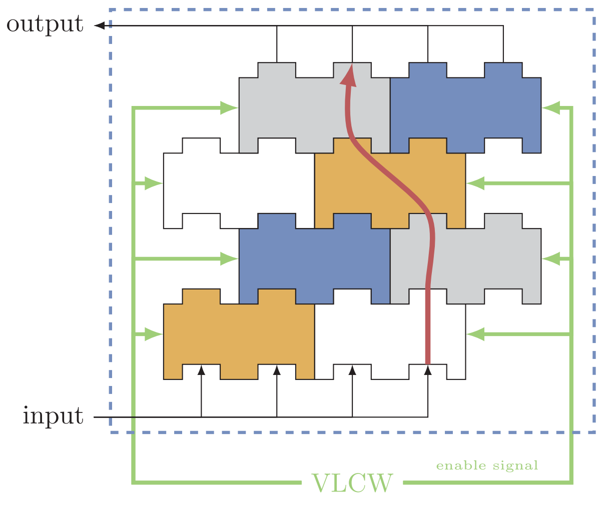


Figure 4.12: Idea of a universal accelerator

In the last year of the second funding phase, we developed an automatic loop detection and hardware acceleration approach for an adaptive reconfigurable processor. The Auto-SI approach is implemented and integrated into the microarchitecture of the *i*-Core. In the first experiments, a transparent speedup up to the factor 9.5× is demonstrated on sequences of ADD instructions, with an additional overhead of 22% resource consumption. The relation shown between overhead and speedup (43:1) of these first results is very promising [Har+17]. To further improve the acceleration and the capacity of the *i*-Core, we envision an accelerator with a universal structure which combines several atoms. Figure 4.12 shows the working principle of a universal accelerator. This accelerator exploits the maximum size of one atom container and combines several algorithms with similar data paths. To analyse the feasibility and potential of this approach, we implemented several hardware accelerators of image processing algorithms, which calculate a convolution. In a next step, one common accelerator is implemented, which combines four different of these algorithms and activate the used convolution matrix by an enable signal. First results shows resource savings of roughly 20% and a low increased latency of 2.5% of the universal accelerator in comparison to the single accelerators.

Worst-Case Execution Time Optimisation

One of the major topics since Phase II of InvasIC is predictability. In Project B1, we presented our WCET analysis for runtime-reconfigurable processors in last year's report (published in [DBH17b]). It enabled us to analyse tasks that reconfigure Special Instructions (SIs) at run time for worst-case execution time (WCET) guarantees. As the reconfigurable area, which is needed to configure SIs, is constrained, the question of which SIs to configure for an optimised WCET guarantee was posed. Therefore, we targeted the problem of *selecting WCET-optimising sets of SIs* for computational kernels[5]. Our approach selects subsets from a bigger set of possible SI implementations for a constrained reconfigurable area, with the aim of optimising the task's WCET bound. The main problem in selecting WCET-optimising SIs is *the instability of the worst-case path*, i. e. when reducing the latency of the worst-case path by inserting an SI, a completely different path can become the new worst-case path. Therefore, WCET bound estimation is an integral part of WCET-optimising SI selection.

First, we introduced *SI super blocks*. SI super blocks are a concept that is introduced to enable static WCET optimisation using reconfigurable SIs on a finished binary. They begin with a conditional before every SI which jumps to functionally-equivalent software code when the SI is not implemented in hardware. During WCET optimisation, SI super blocks capture all the information about each implementation alternative for the respective SI that should be invoked: reconfiguration delay, WCET and reconfigurable container demands. One of the implementation alternatives is always the software implementation which does not

[5]M. Damschen, L. Bauer, and J. Henkel. "Extending the WCET Problem to Optimize for Runtime-Reconfigurable Processors". In: *ACM Trans. on Archit. and Code Optim.* 13.4 (Dec. 2016), 45:1–45:24.

introduce any reconfiguration delay or area demands. Effectively, a CFG is obtained that is parameterised by the chosen implementation for each SI using SI super blocks. The result of the selection is a matrix $y \in \{0,1\}^{|\mathcal{SI}| \times M}$ that maps each SI $k \in |\mathcal{SI}|$ to an implementation $j \in M$, where $|\mathcal{SI}|$ is the total number of SIs and M the maximum number of implementations per SI, respectively. On top of worst-case path analysis using Implicit Path Enumeration Technique (IPET) which is an ILP problem to find the maximum execution time, the WCET-optimising selection needs to find the selection y that minimises the maximum execution time. Additionally, e. g. reconfigurable area constraints need to be met. Even if the reconfigurable area was infinitely large, it is not necessarily beneficial to select the highest-performance implementation for each SI: it might also incur the highest reconfiguration delay. The objective function of the WCET-optimising selection and its constraints formulate an NP-hard combinatorial optimisation problem (IPET already is NP-hard).

B1

An optimal solution to the WCET-optimising SI selection can be obtained using a branch and bound algorithm that generates all possible selections of SI implementations and prunes branches that do not fit onto the reconfigurable area [DBH16]. The problem with this approach is that the WCET of each possible selection needs to be evaluated. The number of possible selections grows exponentially in the number of reconfigurable containers A of the reconfigurable fabric and the number of SIs used in the task under optimisation. Thus, we designed a greedy algorithm that proceeds as follows:

1. Start with an empty reconfigurable area (choose the software implementation $j = 0$ for all SIs)
2. Obtain a WCET estimate for the current selection y
3. For each SI k, calculate the profit on *current* worst-case path of "upgrading" from selected implementation j to the next implementation $j + 1$ in ascending order of reconfigurable container demand $a_{k,j}$. The profit accounts for the latency reduction of a candidate and reconfiguration overhead.
4. If exists, select upgrade $j + 1$ (instead of j) for SI k with highest positive profit (increasing allocated reconfigurable containers by at least one, possibly changing the worst-case path) and go to 2., terminate otherwise

Instead of requiring a number of WCET estimates that grows exponentially in the number A of accelerators that fit onto the reconfigurable

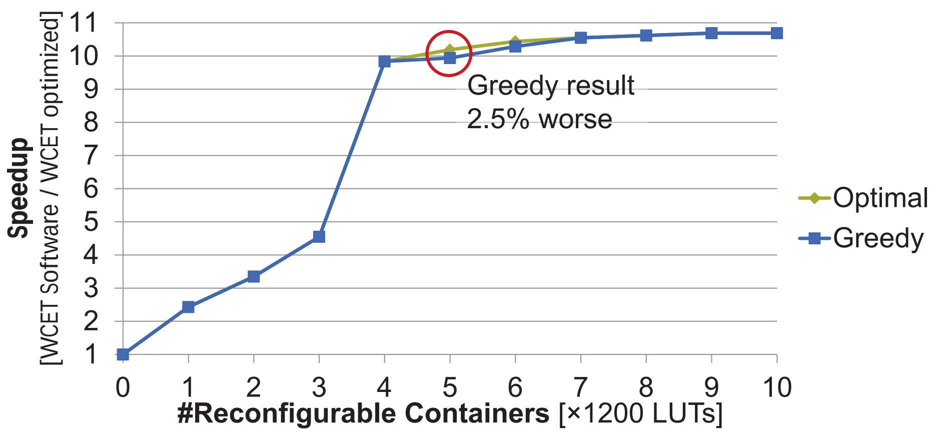


Figure 4.13: Resulting speedup of WCET-optimising SI selection when optimising H.264's Encode Macroblock kernel compared to executing without acceleration

fabric and the number of SIs used in the task under optimisation, this greedy algorithm performs at most A WCET estimates.

The optimal branch and bound as well as heuristic selection algorithms were implemented on 'processors' within the open-source WCET estimation framework OTAWA. In the following, results are shown for a reconfiguration bandwidth of 400 MB/s, running the CPU at 400 MHz (which the LEON3 processor is advertised as running at) and the reconfigurable fabric at 100 MHz (which corresponds to the current *i*-Core design on Xilinx Virtex-7).

Execution time results of the optimisation approach applied to the most complex kernel of the H.264 Encoder "Encode Macroblock" showed that reconstructing the CFG from the binary and analysing all basic blocks already takes around 4.6 seconds. While the runtime of the optimal algorithm first increases exponentially when the amount of area is increased, but then flattens out because branch and bound finds more opportunities to prune the search space, the runtime of the greedy algorithm first seemed to stay constant. It rises slightly, however, until an area of 10 accelerators. The amount of WCET estimates that need to be performed to find the final result grows linearly in the available area. But on average, an additional estimate adds only 40ms to the algorithm's runtime.

Figure 4.13 shows that the speedup results of the SI selections obtained by the greedy algorithm compared to execution without any accelerators are on par with the optimal solution for most cases. However, for an area of 5 and 6 accelerators, the speedup obtained by greedy is up to 2.5% worse than the speedup of the optimal solution. Figure 4.14 shows a simplified example of the case in which greedy performs suboptimal. In this example, the current worst-case path (left

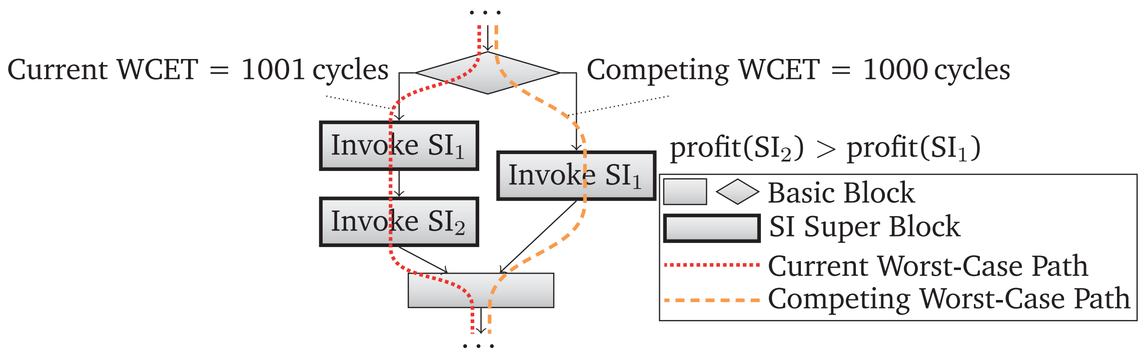

Current WCET = 1001 cycles
Competing WCET = 1000 cycles
Invoke SI$_1$
Invoke SI$_2$
Invoke SI$_1$
profit(SI$_2$) > profit(SI$_1$)
Basic Block
SI Super Block
Current Worst-Case Path
Competing Worst-Case Path

Figure 4.14: Greedily selecting the SI with the biggest profit on the *current* worst-case path (SI_2) is not always optimal. The worst-case path changes during optimisation

path through the CFG) contains two SI super blocks that could be upgraded. SI_2 provides a bigger profit than SI_1, therefore, greedy will select SI_2. However, no matter how big the profit of SI_2 actually is on the current worst-case path the WCET will only be reduced by a single cycle, because the right path through the CFG will immediately become the new worst-case path and reduce the WCET from 1001 cycles to 1000 cycles. The optimal solution would have been to select SI_1. Even though it provides less profit for the current worst-case path, it would also reduce the competing worst-case path and thus provide a bigger benefit for the total WCET. The greedy algorithm cannot make this choice, because it is impossible to foresee the next worst-case path after optimising the execution time of the current worst-case path.

Prototyping and Integration

We are continuously extending and improving the *i*-Core prototype and integrate new features into the joint InvasIC hardware prototype. Together with virtually all other InvasIC projects, we were able to demonstrate an *i*-Core-accelerated actorX10-based version of the 'Shallow Water Equations' application (SWE-X10) during the review of Phase II of InvasIC [Pöp+17]. The demonstration showed how all layers (from hardware until application) work together in an invasive computing system. The demonstrated prototype contained 16 tiles, of which 4 were *i*-Core tiles.

Next Steps in Plan

We are continuing our efforts to provide an adaptable and at the same time predictable processor architecture. In Phase III of InvasIC, we will (among other topics) investigate how runtime enforcement can enable a high resource utilisation while maintaining WCET guarantees (usually conflicting goals). Furthermore, we will investigate approximate accelerators for *i*-Core and introduce information leakage protection for reconfigurable SIs.

Intra-tile Memory Hierarchy Reorganization

Currently, the intra-tile memory hierarchy is organised in a classic bus topology. While this bus system is sufficient for small tiles with a low number of cores, we expect it to become a bottleneck when opting for more complex tiles. Furthermore, Quality of Service requirements are hard to fulfil due to the nature of the shared medium. Thus, we are investigating other interconnect topologies in collaboration with Project B5. One approach that is being examined is a Network-in-a-Network. It incorporates a network similar to the NoC developed in Project B5, but adapted to the needs of intra-tile communication. It will be investigated whether such a design is feasible. To support choosing an appropriate infrastructure, we are working on monitoring bus traffic. We expect that analysis to provide results helping to understand further requirements to the interconnect architecture. Besides, the opportunities of near-data processing (NDP) are investigated. As the intra-tile memory hierarchy needs to allow NDP in the first place, we are exploring the dependencies of both approaches concurrently.

Publications

[DBH17a] M. Damschen, L. Bauer, and J. Henkel. “CoRQ: Enabling Runtime Reconfiguration Under WCET Guarantees for Real-Time Systems”. In: *IEEE Embedded Systems Letters (ESL)* 9.3 (2017), pp. 77–80. DOI: 10.1109/LES.2017.2714844.

[DBH17b] M. Damschen, L. Bauer, and J. Henkel. “Timing Analysis of Tasks on Runtime Reconfigurable Processors”. In: *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)* 25.1 (Jan. 2017), pp. 294–307. DOI: 10.1109/TVLSI.2016.2572304.

[Har+17] T. Harbaum et al. “Auto-SI: An Adaptive Reconfigurable Processor with Run-time Loop Detection and Acceleration”. In: *30th IEEE International System-on-Chip Conference (SOCC)*. IEEE, Sept. 2017, pp. 224–229. DOI: 10.1109/SOCC.2017.8226027.

[Pöp+17] A. Pöppl et al. “Shallow Water Waves on a Deep Technology Stack: Accelerating a Finite Volume Tsunami Model Using Reconfigurable Hardware in Invasive Computing”. In: *Workshop on UnConventional High Performance Computing (UCHPC), Santiago de Compostela, Spain, August 28-29, 2017, Revised Selected Papers*. 2017, pp. 676–687. DOI: 10.1007/978-3-319-75178-8_54.

[Ros+18] E. Rossi, M. Damschen, L. Bauer, G. Buttazzo, and J. Henkel. “Preemption of the Partial Reconfiguration Process to Enable Real-Time Computing with FPGAs”. In: *ACM Trans. on Reconfig. Technol. and Syst. (TRETS)* 11.2 (July 2018), 10:1–10:24. DOI: 10.1145/3182183.

B2: Invasive Tightly-Coupled Processor Arrays

Jürgen Teich

Andreas Becher, Marcel Brand, Frank Hannig, Faramarz Khosravi

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 loop applications from diverse areas such as scientific computing, digital signal and image processing. In terms of latency and throughput, TCPAs are predictable in the number of cycles when executing a loop application in parallel.

Reconfigurable Memory Architecture

Heterogeneous MPSoC architectures containing reconfigurable accelerators like TCPAs have the advantage of providing flexibility, power efficiency, and high performance. However, high-performance computing on TCPAs is only possible by providing data at a high bandwidth, which may lead to a bottleneck. To mitigate this problem, we presented a reconfigurable memory architecture in [Sou+17]. Here, buffers can be configured at run time to select between different schemes for memory access, i. e. random access memory or pixel buffers. Figure 4.15 gives an overview of the reconfigurable I/O buffer architecture connecting a Multiprocessor System-on-Chip (MPSoC) to a TCPA. The memory structure has several memory banks, which consist of one or several dual-ported RAMs (DPRAMs). Each port of a DPRAM is fully synchronous with independent clocks, thus can be employed as an interface for data transfer and clock domain transition between an MPSoC interconnect and the processor array.

We showcased the benefits of the approach by prototyping a TCPA tile consisting of a RISC processor and a TCPA connected by an Advanced High Performance Bus on a Xilinx Virtex-7 2000T FPGA. Our hardware solution needs only 0.7% slice registers and 6.9% LUTs to provide parallel memory accesses to up to 32 Processing Elements (PEs).

Moreover, less than 100 clock cycles are needed to reconfigure the buffer architecture.

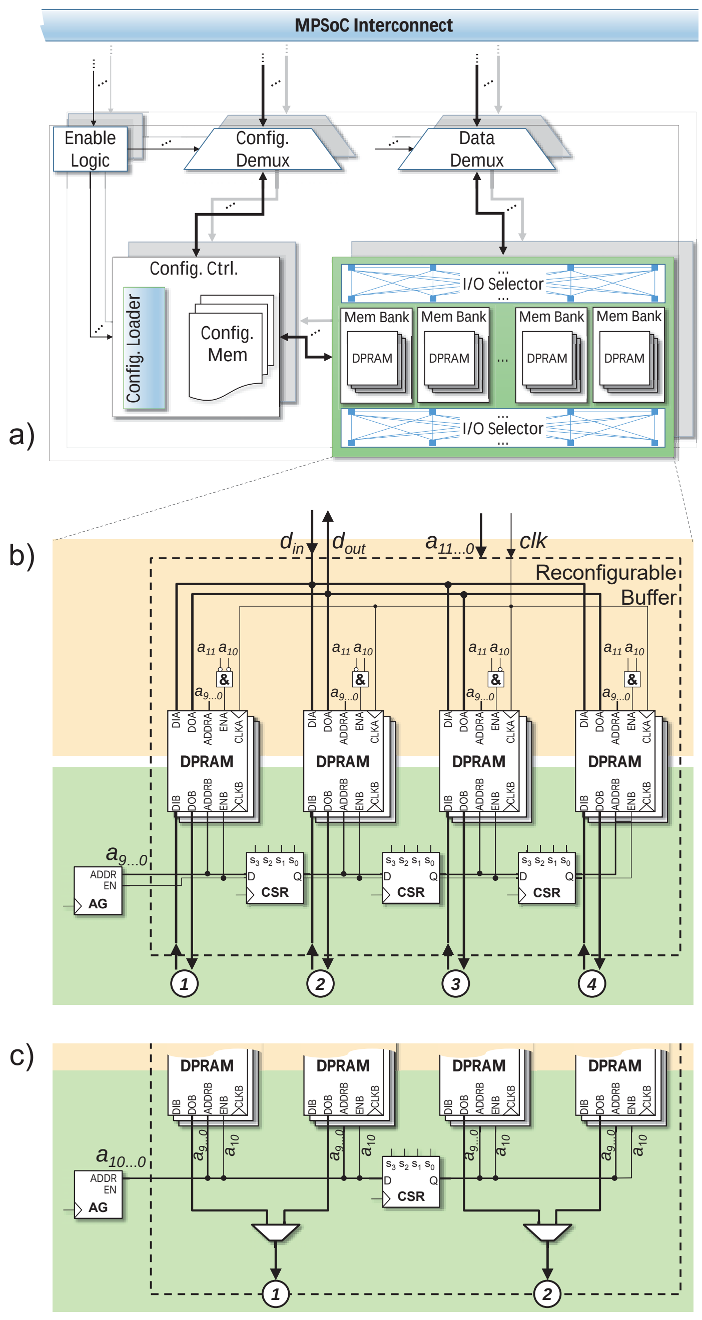


Figure 4.15: (a) The top level of the TCPA memory architecture. A configuration controller is responsible for reconfiguring the architecture at run time. Demultiplexers are used to control the data exchange of different buffer structures. (b) and (c) Examples of an I/O buffer architecture with four memory banks. The buffer's memory banks may be configured into different modes which trade off individual buffer sizes with the number of independent ports to the peripheral cores of the TCPA.

Orthogonal Instruction Processing

We also proposed a novel type of processor architecture and way of instruction processing called Orthogonal Instruction Processing (OIP). In [Bra+17a], we presented the fundamentals of OIP. Contrary to Very Long Instruction Word (VLIW) decoding, the main idea of OIP is to decode the sub-instruction words of each Functional Unit (FU) instead orthogonally by providing an individual instruction sequencer to each FU. Figure 4.16 shows the internal structure of a processor architecture for OIP.

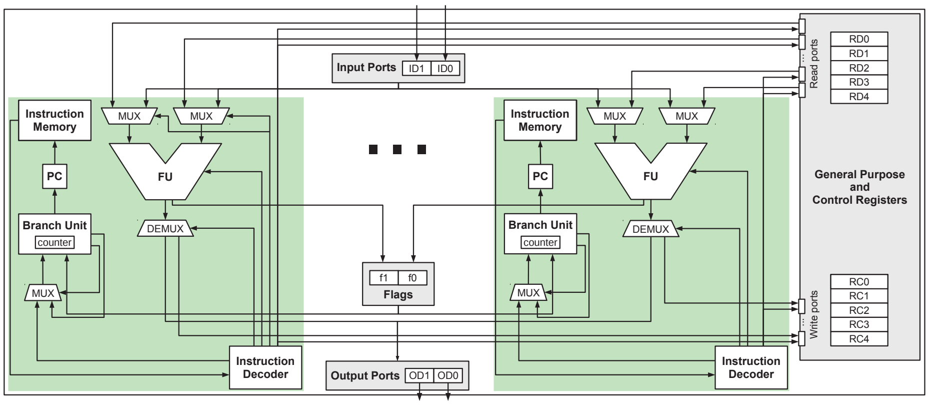


Figure 4.16: Processor architecture for Orthogonal Instruction Processing (OIP). Contrary to a VLIW processor, each Functional Unit (FU) of an OIP processor has its own control structure consisting of an instruction memory, a program counter, and a branch unit. This allows to run a sub-program on each FU independently, while still sharing flags and registers.

One major advantage that we expected from precomputations was that it is possible to severely reduce the overall machine code size of VLIW programs as generated from parallel loop programs as investigated in Project C3. In [Bra+17b], we showed analytically as well as experimentally that, compared to a conventional VLIW processor, the savings in instruction memory size easily compensate the overhead of one separate branch unit needed for each FU (see Figure 4.17). For the analysis, a mathematical model of hardware costs of an OIP processor was used to compare to a conventional VLIW processor. In addition, we investigated the code size in bits, the size of the instruction memory in bits (memory size), as well as the minimum number of instructions needed per FU's instruction memory (memory length) of selected representative programs implemented for the new processor architecture. Figure 4.17 shows the relative code size, instruction memory size and length of each benchmark implemented for an OIP processor, as well as the resulting hardware costs of that processor compared to an implementation of a conventional VLIW processor. Indeed, the instruction memory requirements decrease in average to about half for the considered benchmarks out of domains like linear algebra, digital signal processing, or video/image processing. As the figure also shows, due to severe instruction memory savings, an OIP processor has no higher overall hardware costs than a conventional VLIW processor.

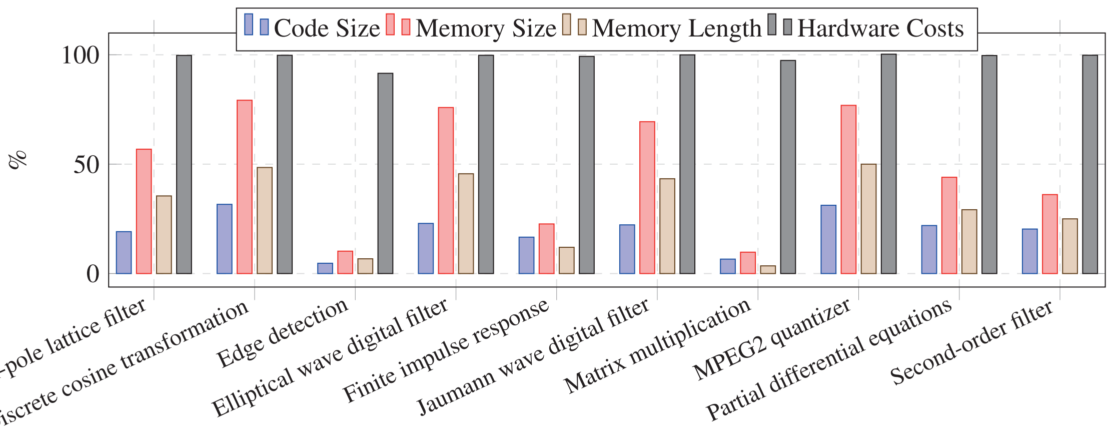


Figure 4.17: Relative code and memory sizes, memory length and hardware costs of an OIP processor compared to a VLIW counterpart.

Anytime Instructions

The concept of *anytime instructions* is based on the idea to process basic arithmetic operations such as additions, subtractions, multiplications, and divisions at a programmable accuracy. Programmable accuracy may be achieved by explicitly specifying the number of computed bits of a result inside the instruction itself but still executing the operation at full precision, i. e. the number of bits of the result representation is not decreased. Through such instructions of resulting variable latency, design trade-offs between execution time and accuracy may be realised. This feature may become essential for systems that want to profit from approximate computing at the level of processor instructions. Current general-purpose processors do not permit to exploit this trade-off in the instruction set. Subsequently, this concept is best applied to operations that are calculated most-significant bit first, thereby keeping a potentially produced error minimal. This is based on the assumption that an error in the least-significant bits of a completed anytime instruction has the least impact on the value of its result. For example, let 5 and 16 be represented each by a single-precision floating-point number, then the division $5/16 = 0.3125$ typically needs 24 (M+1) clock cycles plus one for normalisation. However, the same result can already be achieved after only three cycles for computing three mantissa bits plus one for normalisation, yielding an error of 0% but a latency improvement of 625% – a clear motivation to pursue *anytime instructions*.

Based on IEEE standard floating-point formats, we are currently developing a novel processor architecture with FUs supporting anytime instructions. These will be used as processing elements (PEs) within TCPAs. These PEs are able to compute an exact result if needed but can also produce approximate results to achieve potential latency and power reductions. The degree of approximation is thereby specified in the instruction itself. Thus, a division specified by an anytime instruction

```
DIV_a  target  dividend  divisor
```

will just compute a most-significant bits of a floating-point mantissa division, trading off time and energy against accuracy.

FloaTCPAs - Floating Point Functional Units

Moreover, we started the investigation of complexity and implementation of floating-point support for PEs of a TCPA, widening the range of application domains suited for TCPAs. The floating-point support ranges over functional units for floating-point addition, multiplication, division compliant to the IEEE 754 standard, as well as a conversion unit between fixed and floating-point formats, enabling also mixed PE architectures. Such capability will open TCPAs to target a myriad of scientific applications that require high precision in their calculations. Aside from offering pure floating-point support, this is the first step for realising also *invadable-precision floating-point TCPA architectures* in the third funding phase.

Finally, at the end of the second funding phase, we investigated schemes for fault tolerance (DMR/TMR) on TCPAs as well as also different power-aware resource management. A central goal of demonstration of the CRC/Transregio in Phase II was to show the gain of *-predictability[6] obtained by invasive computing. In this context, a distributed inverse pendulum control application was implemented on the FPGA-based demonstrator platform. This demonstrator served to show that using invasion, the isolation of applications on a multicore platform

[6] J. Teich et al. "Language and Compilation of Parallel Programs for *-Predictable MPSoC Execution using Invasive Computing". In: *Proceedings of the 10th IEEE International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. Lyon, France, Sept. 21–23, 2016, pp. 313–320. DOI: 10.1109/MCSoC.2016.30.

enables (a) real-time, (b) fault tolerance, and finally (c) security guarantees, see [Sou+18; Bra+19] for details. Moreover, in 2018, E. Sousa successfully defended his PhD in Project B2 with a dissertation on the topic *Memory and Interface Architectures for Invasive Tightly Coupled Processor Arrays* [Sou18]. Another highlight is that the dissertation of Alex Tanase has been extended and accepted for publication as a book by Springer [THT18]. Finally, Prof. Teich has edited a special issue on Time-Critical Systems Design in *IEEE Design & Test* [MTT18].

Publications

[Bra+17a] M. Brand, F. Hannig, A. Tanase, and J. Teich. “Efficiency in ILP Processing by Using Orthogonality”. In: *Proceedings of the 28th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Seattle, WA, USA). IEEE, July 10–12, 2017, p. 207. DOI: 10.1109/ASAP.2017.7995282.

[Bra+17b] M. Brand, F. Hannig, A. Tanase, and J. Teich. “Orthogonal Instruction Processing: An Alternative to Lightweight VLIW Processors”. In: *Proceedings of the IEEE 11th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)* (Seoul, Republic of Korea). IEEE Computer Society, Sept. 18–20, 2017, pp. 5–12. DOI: 10.1109/MCSoC.2017.17.

[Bra+19] M. Brand, M. Witterauf, É. Sousa, A. Tanase, F. Hannig, and J. Teich. “*-Predictable MPSoC Execution of Real-Time Control Applications Using Invasive Computing”. In: *Concurrency and Computation: Practice and Experience* (Feb. 2019). DOI: 10.1002/cpe.5149.

[MTT18] T. Mitra, J. Teich, and L. Thiele. “Guest Editors’ Introduction: Special Issue on Time-Critical Systems Design”. In: *IEEE Design and Test of Computers* 35 (2018), pp. 5–7. DOI: 10.1109/MDAT.2018.2796037.

[SHT18] C. Schmitt, F. Hannig, and J. Teich. “A Target Platform Description Language for Parallel Code Generation”. In: *Workshop Proceedings of the 31st GI/ITG International Conference on Architecture of Computing Systems (ARCS)* (Braunschweig). Berlin: VDE VERLAG GmbH, Apr. 9–12, 2018, pp. 59–66.

[Sou18] É. Sousa. “Memory and Interface Architectures for Invasive Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 20, 2018.

[Sou+18] É. Sousa, M. Witterauf, M. Brand, A. Tanase, F. Hannig, and J. Teich. “Invasive Computing for Predictability of Multiple Non-functional Properties: A Cyber-Physical System Case Study”. In: *Proceedings of the 29th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Milan, Italy). IEEE, July 10–12, 2018. DOI: 10.1109/ASAP.2018.8445109.

[Sou+17] É. Sousa, A. Tanase, F. Hannig, and J. Teich. “A Reconfigurable Memory Architecture for System Integration of Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the International Conference on Reconfigurable Computing and FPGAs (ReConFig)* (Cancun, Mexico). IEEE, Dec. 4–6, 2017. DOI: 10.1109/RECONFIG.2017.8279768.

[THT18] A. Tanase, F. Hannig, and J. Teich. *Symbolic Parallelization of Nested Loop Programs*. ISBN: 978-3-319-73908-3. Springer, Feb. 2018.

B3: Power-Efficient Invasive Loosely-Coupled MPSoCs

Jörg Henkel, Andreas Herkersdorf

Nguyen Anh Vu Doan, Heba Khdr, Martin Rapp, Mark Sagi, Thomas Wild

The overall goal of Project B3 is to optimise power/energy efficiency under power density and temperature constraints. Within the paradigm of invasive computing, the goal is to ensure that invaded *claims* remain thermally reliable while maintaining the ability that *teams* can invade and execute *i*-lets to *infect* new resources. The pursued objectives are:

Objective 1: Improve power/energy efficiency under power density and temperature constraints.

Objective 2: Develop an adaptive and self-aware system for run-time power, energy, and thermal management.

Objective 3: Refine the run-time model and online estimation of power density and temperature for invasive computing.

The related scientific challenges include the maximisation of the performance under given temperature constraints or under a given energy budget, and minimising the energy consumption or peak power under a certain performance requirement. The intelligent collection and aggregation of relevant data for resource management from all over the SoC is another challenge that we address by means of powerful on-chip on-the-fly data analysis infrastructure. These goals require deep insight into the system, which often is tackled by design-time models. However, these static design-time models cannot reflect the spatial and temporal uncertainties and variations that come from the environment, the hardware, the input data, or from the workloads/applications. Therefore, we aim to achieve the objectives without relying on design-time models.

Performance Optimisation under Power and Temperature Constraints

Manycores with S-NUCA architecture exhibit non-uniform average LLC latency among cores. We are the first to characterise and exploit this observation in [PH18b] to increase the performance. In [RPH18], we propose a technique to increase the performance of S-NUCA manycores that operate under a thermal constraint. We show the trade-off between the thermally safe power budget and the average LLC latency when mapping threads to cores. This technique does not rely on design-time application models, but instead follows an application-agnostic approach.

In [Pat+18], we propose a QoS-aware power management technique that operates under a thermal design power (TDP) constraint. This technique is based on the observation that total power consumption of many independent applications that execute in parallel follows a normal distribution. Therefore, a centralised fine-grained control of every core is not required, but a stochastic control can be employed, reducing the overhead while simultaneously decreasing the fraction of applications that miss the QoS requirement. This technique considers temporal uncertainty by modelling it as a stochastic process.

Besides power and temperature, we consider in [KAH18c; KAH18a; KAH18b] the ageing of the chip. In [KAH18c], we propose *AgRM*, a resource management technique that optimises the performance under an ageing constraint. To achieve this goal, we present a new ageing-aware design space exploration that translates the temperature and the supply voltage to the amount of ageing quantified as ΔV_{th}. This new design space enables *AgRM* to optimise performance within the feasible design space that satisfies the ageing constraint. Contrarily, state-of-the-art techniques only utilise a limited design space because it employs fixed bounds of temperature and voltage to satisfy the ageing constraint. Compared to the state-of-the-art, *AgRM* achieves significant performance gain.

After enforcing an ageing constraint, there is a need to apply guardbands on the system to compensate for the timing delays that are induced by ageing. There are two guardband types; Frequency guardband and voltage guardband. State-of-the-art techniques select at design time the guardband type and it will stay fixed throughout the chip lifetime. However, we propose in [KAH18a] to select the guardband type at run time according to the running workload. Particularly, if the workload is leading to high power consumption, and thereby a thermal violation might occur, we select a frequency guardband to avoid additional power consumption that would be resulting from adding a voltage guardband. Otherwise, if the workload is leading to low power consumption, then a voltage guardband can be selected to avoid performance losses that would be resulting from a frequency guardband.

Some of these (and other) research efforts are also summarised in [Khd+18], [Pag+18a] and [Hen+18].

Online Data Analysis to Support Power/Thermal Management

Manycore resource management techniques rely on accurate and fine-grained power/thermal information. Directly measuring the power consumption of processor components is cost prohibitive. We investigate a representation learning algorithm based on independent component analysis to accurately estimate power consumption during run-time using on-chip activity data. We apply independent component analysis to transform a large set of activity data into a representation with lower dimensionality and higher information density, so-called feature vectors. The power consumption related to these feature vectors is then derived with a regression analysis, leading to power features. Such power feature representations show both visible power activity and hidden power activity. This means we are able to distinguish between power consumption due to computations and power consumption due to data communication which is hidden in the activity data. The distinction provides power proxies for communication as well as higher accuracy in attributing power consumption to specific applications. We also investigate how power features can be used to identify power-inefficient system states. Our current approach is to execute individual applications on the system without other applications interfering while tracing their power features. When multiple applications run in parallel, performance and power efficiency degrades due to shared resource interference. By comparing the (non-interfering) power features with run-time power features, we obtain a metric of the system's power inefficiency. Power management can use this metric to redistribute power or reschedule applications to minimise the system inefficiencies.

Machine Learning in Power/Thermal Management

Machine learning has shown to be able to replace design-time models by either building models at run time or enabling model-free operation. In [Pag+18b], we present a survey of existing techniques that employ machine learning for resource management for power/thermal efficiency. We explore how different machine learning techniques, like supervised learning, unsupervised learning, and reinforcement learning, are used in resource management of single-core, homogeneous multicore, and heterogeneous multicore processors. We further classify the techniques based on their optimisation objective/constraints (performance, power, energy, temperature), and based on the employed optimisation knobs (task allocation, power management and voltage-/frequency scaling).

Evaluation of Resource Management Techniques

In [PH18a], we propose a simulation framework that combines performance, power, and temperature simulation of multicore and manycore processors. This framework allows simulating the execution of multi-threaded multi-program workloads on a multi/manycore processor with full modelling of shared-resource contention. We use this framework in the evaluation of our proposed techniques.

Dissertations

In May 16th 2018, Anuj Pathania received his PhD (Dr.-Ing.) from the Faculty of Informatics, Karlsruhe Institute of Technology (KIT). The title of his dissertation is "Scalable Task Schedulers for Many-Core Architectures" [Pat18].

In July 4th 2018, Heba Khdr received her PhD (Dr.-Ing.) from the Faculty of Informatics, Karlsruhe Institute of Technology (KIT). The title of her dissertation is "Resource Management for Multicores to Optimize Performance under Temperature and Aging Constraints" [Khd18].

Organisation of Scientific Events and Public Dissemination

Project B3 organised a special session on energy efficiency titled "Managing Heterogeneous Many-Cores for High-Performance and Energy-Efficiency" at the IEEE/ACM 37th International Conference on Computer Aided Design (ICCAD), 2018. More information at
`http://iccad.com/event_details?id=263-5-E`.

Publications

[Hen+18] J. Henkel, J. Teich, S. Wildermann, and H. Amrouch. "Dynamic Resource Management for Heterogeneous Many-Cores". In: *Proceedings of the International Conference on Computer-Aided Design (ICCAD)* (San Diego, CA, USA). ACM, Nov. 5–8, 2018, 60:1–60:6. DOI: `10.1145/3240765.3243471`.

[Khd18] H. Khdr. "Resource Management for Multicores to Optimize Performance under Temperature and Aging Constraints". Dissertation. Chair of Embedded Systems, Department of Informatics, Karlsruhe Institute of Technology, Germany, 2018.

[KAH18a] H. Khdr, H. Amrouch, and J. Henkel. "Dynamic Guardband Selection: Thermal-Aware Optimization for Unreliable Multi-Core Systems". In: *Transactions on Computers (TC)* (2018).

[KAH18b] H. Khdr, H. Amrouch, and J. Henkel. “Aging-Aware Boosting”. In: *IEEE Transactions on Computers (TC)* (2018). DOI: `10.1109/TC.2018.2816014`.

[KAH18c] H. Khdr, H. Amrouch, and J. Henkel. “Aging-Constrained Performance Optimization for Multi Cores”. In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[Khd+18] H. Khdr, S. Pagani, M. Shafique, and J. Henkel. “Dark Silicon Aware Resource Management for Many-Core Systems”. In: *Advances in Computers: Dark Silicon and Future of On-chip Systems*. Elsevier, 2018.

[Pag+18a] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. *Advanced Techniques for Power, Energy, and Thermal Management for Clustered Manycores*. Springer, 2018.

[Pag+18b] S. Pagani, S. M. PD, A. Jantsch, and J. Henkel. “Machine Learning for Power, Energy, and Thermal Management on Multi-core Processors: A Survey”. In: *Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2018).

[Pat18] A. Pathania. “Scalable Task Schedulers for Many-Core Architectures”. Dissertation. Chair of Embedded Systems, Department of Informatics, Karlsruhe Institute of Technology, Germany, 2018.

[PH18a] A. Pathania and J. Henkel. “HotSniper: Sniper-Based Toolchain for Many-Core Thermal Simulations in Open Systems”. In: *Embedded Systems Letters (ESL)* (2018).

[PH18b] A. Pathania and J. Henkel. “Task scheduling for many-cores with S-NUCA caches”. In: *Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2018, pp. 557–562.

[Pat+18] A. Pathania, H. Khdr, M. Shafique, T. Mitra, and J. Henkel. “QoS-Aware Stochastic Power Management for Many-Cores”. In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[RPH18] M. Rapp, A. Pathania, and J. Henkel. “Pareto-Optimal Power- and Cache-Aware Task Mapping for Many-Cores with Distributed Shared Last-Level Cache”. In: *International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE. 2018.

B4: Generation of Distributed Monitors and Run-Time Verification of Invasive Applications

Ulf Schlichtmann, Daniel Müller-Gritschneder

Alexandra Listl, Marcel Mettler, Bing Li, Li Zhang

Introduction

The breakdown of Dennard scaling has resulted in the fabrication of Multi-Processor Systems-on-Chips (MPSoCs), which satisfy the ever-growing demand for performance, while continuously reducing the chip size. Another consequence is the increasing power densities that arise, which critically influence the chip temperatures and accelerate device degradation due to ageing. This problem becomes more critical as the number of cores in MPSoCs increases since the temperature of a processing core not only depends on its own power consumption, but also on the activity and power consumption of neighbouring cores. Run-time power management can be utilised to counter these reliability threats and increase the lifetime of a system. For the development of run-time power management strategies, monitoring data for power, temperature and ageing is required. Project B4 evolves the monitors, which were investigated in the past two funding phases, into a distributed monitoring system capable of supporting run-time verification of user-defined application properties such as latency, throughput, power, reliability and security. The run-time verification chain consists of probes, property checkers and event handlers. Probes are blocks that trace system events or states. Property checkers analyse the probed trace to generate the verdict whether a certain property is violated or validated. Hardware property checkers may be watchdog timers, and software property checkers may be additional software code to check the values of variables. Hybrid property checkers may be a software code that compares the value of two hardware timers. The verdict is forwarded to the event handler that can trigger a reaction or generate a log message. A key event handler will be the run-time requirement enforcers (RREs) of Project Project A1 and Project A4. These RREs will enable a much closer control of non-functional program properties.

Currently, Project B4 is evaluating distributed monitoring concepts for hardware-supported run-time verification. To express the run-time verification properties, different variants of temporal logic are analysed according to the requirements arising in embedded applications. For this, Project B4 actively participates in the working group on *Run-time Requirement Monitoring and Enforcement*.

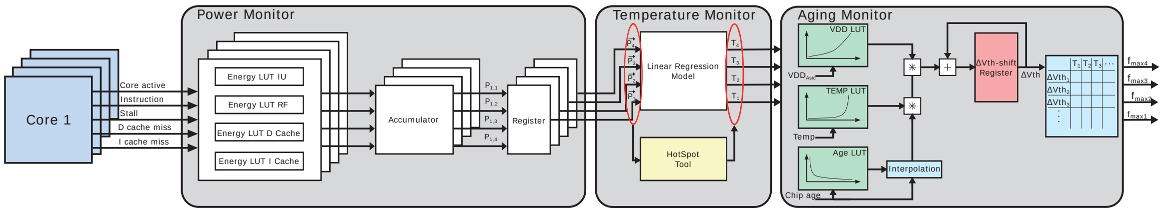


Figure 4.18: Improved eTAPMon including ageing emulation

eTAPMon Emulator

In [Lis+18], we proposed a real-time power, temperature and ageing monitor system (eTAPMon) for FPGA prototypes of MPSoCs. The monitor system emulates data characterised from the target ASIC design. The emulation approach models the behaviour of ASIC power monitors based on an instruction-level energy model, temperature monitors based on a linear regression model obtained from thermal offline simulations and ageing monitors based on a critical path model to compute the decreasing timing margin due to ageing. An accelerated ageing emulation is possible to predict aged ASIC behaviour. Hence, this FPGA emulation enables the early evaluation of run-time power management strategies.

SRAM Aging

On-Chip SRAMs are an integral part of safety-critical MPSoCs. At the same time, however, they are also most susceptible to reliability threats such as Bias Temperature Instability (BTI), originating from the continuous trend of technology shrinking. BTI leads to significant performance degradation, especially in the Sense Amplifiers (SAs) of SRAMs, where failures are fatal, since the data of a whole column is destroyed. As BTI strongly depends on the workload of an application, the ageing rates of SAs in a memory array differ significantly and the incorporation of workload information into ageing simulations is vital. Especially in safety-critical systems, precise estimation of application-specific reliability requirements to predict the memory lifetime is a key concern.

In [Lis+19], we present a workload-aware ageing analysis for On-Chip SRAMs that incorporates the workload of real applications executed on a processor. According to this workload, we predict the performance degradation of the SAs in the memory. We integrate this ageing analysis into an ageing-aware SRAM design exploration framework that generates and characterises memories of different array granularity to select the most reliable memory architecture for the intended application. We show that this technique can mitigate SA degradation significantly depending on the environmental conditions and the application workload.

Post-silicon Clock Tuning

At submicron manufacturing technology nodes, process variations affect circuit performance significantly. To counter these variations, engineers are reserving more timing margins to maintain yield, leading to an unaffordable overdesign. Most of these margins, however, are wasted after manufacturing, because process variations cause only some chips to be slow, while other chips can easily meet given timing specifications. To reduce this pessimism, less timing margins may be reserved and failed chips can be tuned with clock buffers to meet timing specifications. With this clock tuning technique, critical paths can be balanced with neighbouring paths in each chip specifically. Consequently, chips with timing failures can be rescued and the yield can thus be improved. This is especially useful in high-performance designs, e. g. high-end MPSoCs. In [Zha+18b], we propose a method to learn the buffer locations with a Sobol sequence iteratively and reduce buffer ranges afterwards to achieve a performance improvement up to 26%. This method can be used to adapt chips in invasive computing systems according to performance requirements. To apply post-silicon clock tuning, path delays in chips with timing failures should be measured to gather timing information for clock reconfiguration. However, prior methods for delay measurement rely on path-wise frequency stepping, which requires much time from expensive testers. In [Zha+18a], we propose an efficient delay test framework to solve this problem by testing only representative paths with delay alignment taking advantage of the existing tunable buffers in the circuit. This efficient test method can reduce the number of frequency stepping iterations by more than 94%, so that clock tuning in invasive MPSoCs can be implemented efficiently to adapt chips to meet performance requirements.

Publications

[Lis+18] A. Listl, D. Mueller-Gritschneder, F. Kluge, and U. Schlichtmann. "Emulation of an ASIC Power, Temperature and Aging Monitor System for FPGA Prototyping". In: *International On-Line Testing Symposium (IOLTS)*. July 2018.

[Lis+19] A. Listl, D. Mueller-Gritschneder, S. R. Nassif, and U. Schlichtmann. "SRAM Design Exploration with Integrated Application-Aware Aging Analysis". In: *Proceedings of Design, Automation and Test in Europe Conference (DATE), March 2019*. accepted for publication. 2019.

[Zha+18a] G. L. Zhang, B. Li, J. Liu, Y. Shi, and U. Schlichtmann. "Design-Phase Buffer Allocation for Post-Silicon Clock Binning by Iterative Learning". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. Vol. 37. 2. 2018.

[Zha+18b] G. L. Zhang, B. Li, Y. Shi, J. Hu, and U. Schlichtmann. "EffiTest2: Efficient Delay Test and Prediction for Post-Silicon Clock Skew Configuration under Process Variations". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2018). Early Access Paper, doi: 10.1109/TCAD.2018.2818713.

B5: Invasive NoCs and Memory Hierarchies for Run-Time Adaptive MPSoCs

Jürgen Becker, Andreas Herkersdorf

Nidhi Anantharajaiah, Leonard Masing, Sven Rheindt, Akshay Srivatsa

In invasive computing architectures, the invasive NoC (*i*NoC) makes up the basic communication infrastructure among all tiles, enabling and supporting invasive concepts from the application level down to hardware realisation. However, for inter-tile communication, not only data transport alone is a key issue, but also aspects of memory accessibility and the availability of data where they are processed play an important role. Therefore, current research within Project B5 concentrates on mechanisms to optimise the data exchange among tiles by introducing a specific asynchronous circuit-switched overlay network or generalising the *i*NoC towards hierarchical topologies to improve performance as well as scalability. Further, investigations are carried out on providing a cache-coherent view among a runtime-configurable subset of tiles within so-called coherence regions. In order to reduce the costs of memory accesses, it is also studied how computation and the location of data to be processed can be brought closer to each other.

Region-Based Cache Coherence

Embedded system applications, with their inherently limited parallelism, rarely use all cores of large manycore architectures. Further, global coherence spanning all tiles does not scale well. Therefore, we proposed a region-based cache coherence (RBCC) concept that confines hardware coherence support to a selectable set of tiles [Sri+17]. These so-called coherence regions can be dynamically configured at run time based on application requirements using a coherency region manager (CRM) module. We synthesised the CRM on our hardware platform as part of a 16-tile 64-core multi-FPGA design (Figure 4.19). As an example benchmark, we executed the feature extraction task of a video streaming application (developed by Project D1) both in a shared memory (enabled by RBCC) and a message passing mode. The demonstration revealed

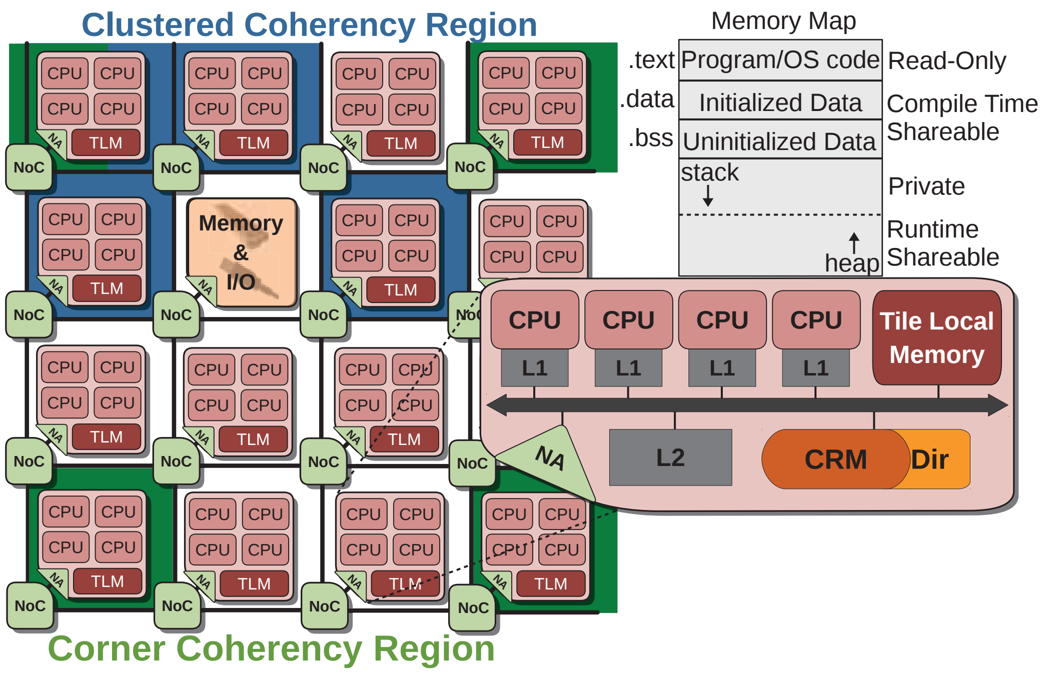


Figure 4.19: A 4×4 InvasIC architecture with multiple coherence regions

B5

an application acceleration of up to 42% using shared memory mode compared to the message-passing-based implementation. Hardware resource utilisation numbers show that by using the RBCC concept, directory (BRAM) resources are reduced by 57% compared to global coherence for a region size of 4-tiles. The tile-local memory (TLM) does not entirely contain shareable data but also tile-local OS data, core instructions, core stacks, etc., which are tile-private and do not need to be kept coherent. Therefore, we introduced *RBCC-malloc()* as a further extension to RBCC that transparently tailors coherence to shareable application working sets within the regions at run time. Current research in this area focuses on developing architecture-aware replacement policies for sparse directories and exploring the context switching penalties (for flushing caches, resetting directory entries, etc.) during run time coherence region reconfigurations.

Remote Near-Memory Acceleration

Hitting a wall is not a pleasant thing. Computer systems faced and still face many walls in the last decades. Tiled architectures help to tackle the memory and power wall by physically distributing processing nodes and memory. Although this measure reduces access hot-spots and increases the overall memory bandwidth, performance does not likewise scale due to data-to-task dislocality. The so-called locality wall is the next crucial barrier for scalability to 100's and 1000's of cores.

The newest trend to tackle this issue is data-task migration and processing in or near memory. Related work mostly considers 3D-stacked architectures currently. We focus on an orthogonal approach and research these aspects in tiled architectures. Figure 4.20 highlights the different aspects. Data can be migrated close to the tasks that work on them or vice versa. Additionally, those architectures add the dimension of remote near-memory operations that get offloaded to dedicated hardware near the memory where the data resides.

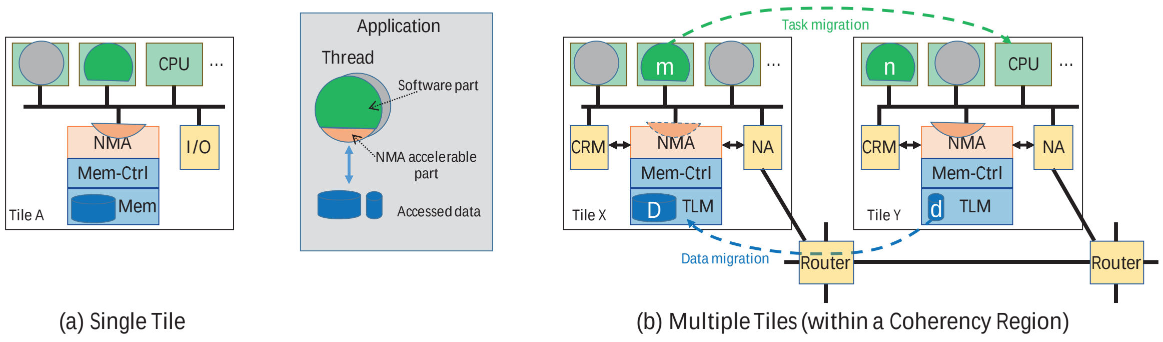


Figure 4.20: Near-memory accelerators (NMA) and data migration as approaches to increase data locality

Our previous work dealt with the important aspect of efficient inter-tile communication in distributed architectures. Efficient task spawning [Zai+17] [Zai18] paved the beginning of this path. We further proposed complex inter-tile atomic operations in [Rhe+18]. We showed that especially remote operations highly profit from dedicated hardware implementations that operate near the memory where the data structure resides. Our approach, therefore, implements often used and highly concurrent tasks as dedicated hardware.

Additionally, we investigate efficient queue mechanisms for these architectures and are currently developing near-memory accelerators that deal with crucial system software operations to achieve performance improvements for a wide range of applications. Complementary to bringing tasks closer to their data, we are starting to investigate efficient mechanisms for data migration. These aspects will be studied in cooperation with Project C1 and Project C3.

In-NoC-Circuits

Modern NoCs often offer multiple NoC layers, typically tailored towards a specific task. However, this can lead to poor resource utilisation, especially if the application domain is not fully known in advance or the execution contains a lot of dynamically changing computations. As an effort towards multiple NoC layers in the *i*NoC, an asynchronous circuit-switched layer was added. In contrast to the previously existing regular packet-switched layer, this layer allows connections that offer a significantly reduced latency, even over long hop distances. However, as in any circuit-switching scheme, the connections need to be established first and block resources which cannot be used for other flows of traffic. In order to overcome this limitation, we proposed the concept of so-called In-NoC-Circuits (INCs) [Mas+18], which do not start and end in a tile (i. e. the network adapter), but rather stretch between two routers. This allows multiple communication partners to benefit from the same circuit and the low latencies it provides. A 4×4 architecture that was extended by the INC feature is shown in Figure 4.21. We

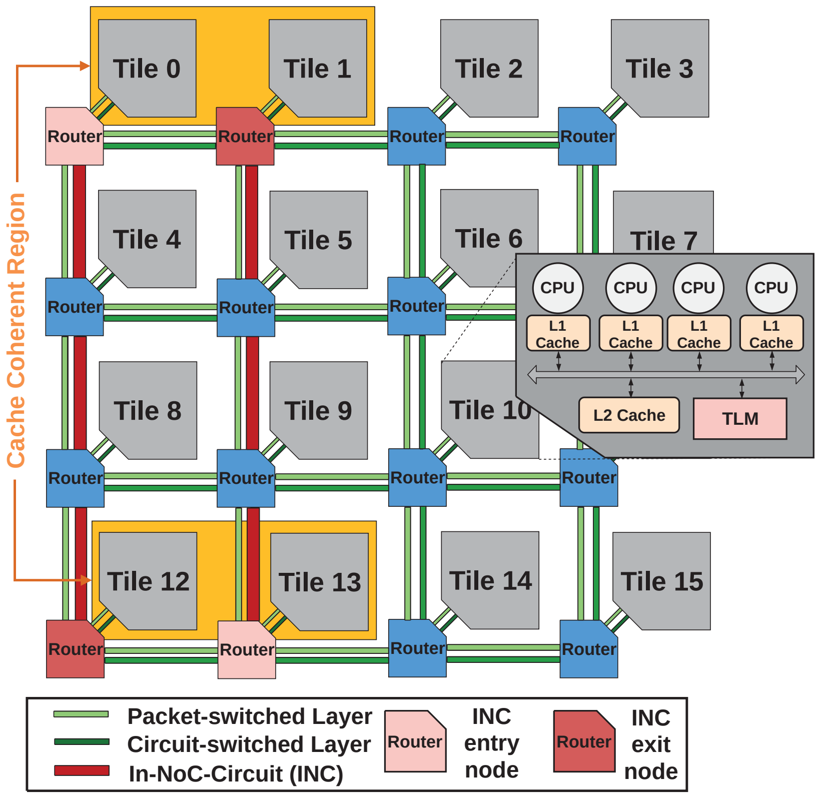


Figure 4.21: A 4×4 InvasIC architecture with In-NoC-Circuits

investigated this novel feature in the context of RBCC, since coherence traffic benefits from low latency exchanges among a coherence region. Furthermore, the regions may often come in clusters, which specifically benefit from the ability to use the same circuit for communication between the clusters. Depending on the selected coherence region, latency improvements of up to 45% on average could be achieved in a 4×4 meshed design with even higher potential for larger meshes. The additional resource cost for this feature amount to 18% more LUT, 16% more FF for the INC extension and another 13% LUT, 5% FF for an optional feature that automatically detects and establishes promising INC connections.

Hierarchical NoC

During the first two funding phases, a basic meshed *i*NoC architecture with special features was developed. Next, we plan to investigate how the techniques present in the *i*NoC for giving QoS guarantees can be applied to novel architectural layouts and techniques like hierarchical topologies and adaptive routing algorithms. We extend the network by incorporating multiple layers forming a Hierarchical *i*NoC (H*i*NoC). An example of a 5×5 H*i*NoC is illustrated in Figure 4.22. This is done to improve performance, as there are multiple paths available on the different layers and depending on the availability of the routers and links, the packet can choose the shortest path between nodes. Additionally, it will help increasing fault tolerance by providing redundancy due to the layered structure, where single layers can also be disabled on demand. We investigate cases where not all layers are enabled at the same time. This allows us to manage power/performance trade-offs or also reserve a full layer for guaranteeing QoS for an application. By allowing packets to use several layers and different paths to the destinations at runtime,

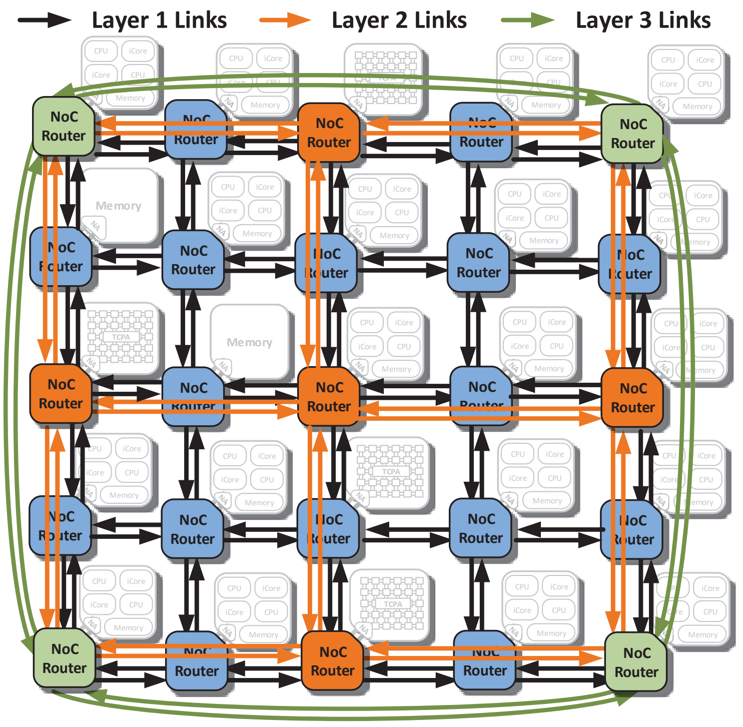


Figure 4.22: 5×5 Hierarchical *i*NoC

care must be taken to ensure there are no deadlocks. One method to achieve this is implementing deadlock-free routing algorithms. Another method which is investigated is adding features to the router such that a packet can escape a deadlock when it occurs. Care must also be taken to ensure there is no message-dependent deadlock, which occurs when a generation of one message depends on the consumption of another (e. g. request-response dependency). One method which ensures this is using the strict ordering concept with virtual networks. In this technique, packets of different message classes travel in different virtual channels, ensuring there are no message-dependent deadlocks. We plan to enhance this established concept further in the context of H*i*NoC.

Publications

[Fri17] S. Friederich. “Automated Hardware Prototyping for 3D Networks on Chips”. Dissertation. Institut für Technik der Informationsverarbeitung, Karlsruhe Institute of Technology (KIT), May 23, 2017.

[Mas+18] L. Masing, A. Srivatsa, F. Kreß, N. Anantharajaiah, A. Herkersdorf, and J. Becker. “In-NoC Circuits for Low-Latency Cache Coherence in Distributed Shared-Memory Architectures”. In: *2018 IEEE 12th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. IEEE, Sept. 2018. DOI: `10.1109/mcsoc2018.2018.00033`.

[Rhe+18] S. Rheindt, A. Schenk, A. Srivatsa, T. Wild, and A. Herkersdorf. “CaCAO: Complex and Compositional Atomic Operations for NoC-based Manycore Platforms”. In: *ARCS 2018 - 31st International Conference on Architecture of Computing Systems*. Braunschweig, Germany, 2018.

[Sri+17] A. Srivatsa, S. Rheindt, T. Wild, and A. Herkersdorf. “Region Based Cache Coherence for Tiled MPSoCs”. In: *2017 30th IEEE International System-on-Chip Conference (SOCC)*. Sept. 2017.

[Zai+17] A. Zaib et al. “Efficient Task Spawning for Shared Memory and Message Passing in Many-core Architectures”. In: *Journal of Systems Architecture (JSA)* (2017). DOI: `10.1016/j.sysarc.2017.03.004`.

[Zai18] M. A. Zaib. “Network on Chip Interface for Scalable Distributed Shared Memory Architectures”. Dissertation. München: Technische Universität München, 2018.

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

Lars Bauer, Jörg Henkel, Timo Hönig, Wolfgang Schröder-Preikschat

Gabor Drescher, Christoph Erhardt, Tobias Langer, Sebastian Maier, Anuj Pathania, Florian Schmaus, 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 multicore/manycore machines.

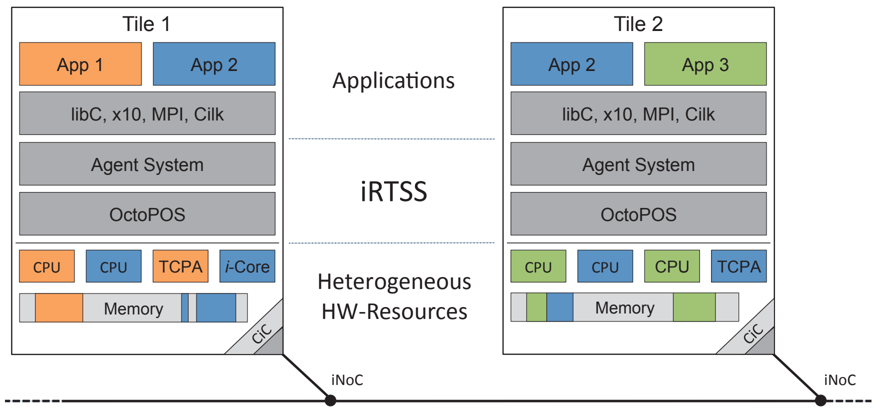


Figure 4.23: *i*RTSS architecture on a multi-tile system. Colours indicate invaded resources.

Architectural Overview

Figure 4.23 provides a high-level view of the current *i*RTSS architecture. Key elements are OctoPOS,[7] the parallel operating system (POS) that implements the *mechanisms* of *i*RTSS to make all capabilities of the underlying hardware available to higher (software) levels, and the

[7]Prefix 'Octo' indicates the 8th generation within a particular family of special-purpose operating systems.

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 runtime libraries for various kind of invasive-parallel applications and the OctoPOS kernel.

The OctoPOS Kernel

OctoPOS makes the computing platform (LEON, x86-64) accessible to processes as a "virtual machine" that functions either as native ("satellite") neighbouring to or as guest ("planet") managed by a host operating system (Linux). 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 *unfiltered* way.

Cleanup and Consolidation: The operating system work in the year under review was characterised by clearing up and consolidating. This involved our work on synchronising concurrent processes [RS18a; RS18b], measuring latency in interrupt handling [Her+18], WCET analysis [Wäg+18; Eic+18], and energy demand analysis [Hön+18a; Hön+18b; Eib+18]. In addition, some development work has been completed, mainly to improve the stability of the demonstrator system. This included the implementation of a (custom) garbage collector for X10 to replace the (generic) Boehm variant used so far, *i*NoC hardware and driver enhancements, an adjustment of the MPI layer to make use of pull DMA, local DMA, byte alignment, and hardware-managed queues (Figure 4.24), dynamic OctoPOS configuration based on system informa-

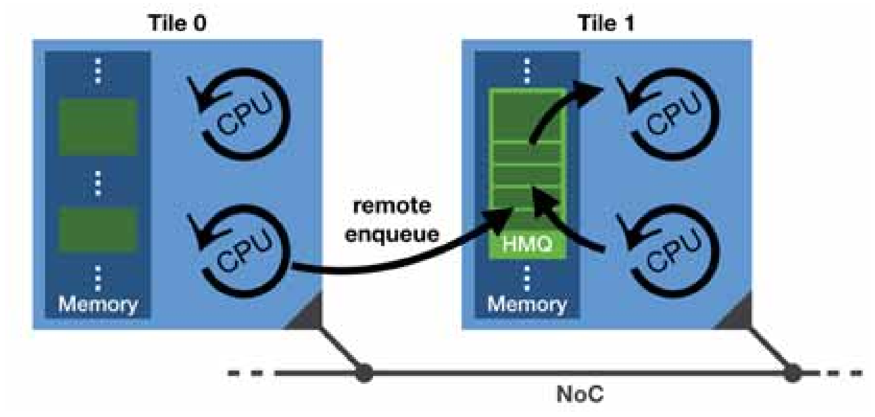


Figure 4.24: Hardware-managed queue. A mechanism to carry out inter-tile interactions between concurrent processes without software intervention.

tion provided by hardware (tile count/layout, memory locations/layout, hardware revision, etc.), work on the distributed network stack for an improved user-level I/O interface, and bug fixing.

System-Event Handling: A key issue in the current phase is the provision of efficient mechanisms for operating feedback loops that incorporate system events and states into application flows. In order to propagate those events to user-level execution contexts, OctoPOS will facilitate the concept of a so-called *squad* (Figure 4.25) as a general abstraction for pairing the *synchronous* control flow of an *i*-let, the *lead*, with (at least) one *potentially asynchronous* control flow, an *aide*. Both control-flow entities together form a *team tuple*, which then is

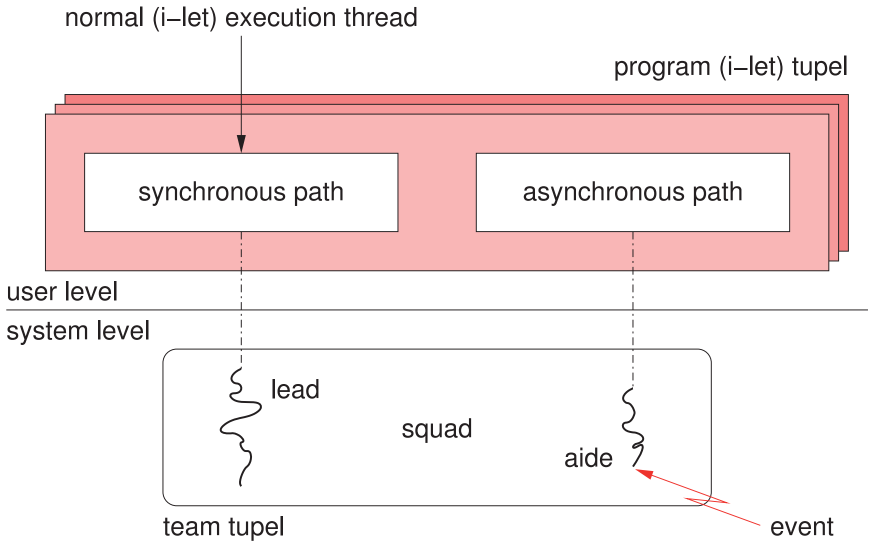


Figure 4.25: The *squad* design pattern

handled by OctoPOS as a single unit of processing. The work focused on considering how such teams could be included as entities in the OctoPOS kernel.

Memory Consistency: Another new working area in the current phase is virtual shared memory (VSM), a feature that, in functional terms, provides transparent access to the different main-memory subsystems of the PGAS model (partitioned global address space) as defined for invasive computing. Here, the work is still very early and focused primarily on a review of the state of research in the operating system area and initial considerations of suitability for multicore systems. The investigations delivered input to the working group on “Memory Models, Architecture and Management”.

Energy Demand Analysis: Energy is considered as a first-class operating system resource and, thus, must be considered for energy-efficient run-time decisions by OctoPOS. In particular, as to the *enforcement* of run-time properties, Project C1 explores the adaptation of the squad concept to enforce energy demand of *i*-lets. During the early phase of the third funding phase, Project C1 has provided research results to the working groups on “Run-Time Requirement Monitoring and Enforcement” and “Power and Thermal Aspects”. Currently, the exploration on building reliable energy models for applicable, invasive hardware platforms continue at the hand of analysis methods that are based on machine-learning techniques.

The Agent System

The task of the *i*RTSS Agent System (AS) is to provide a decentralised, scalable resource management for invasive applications. Per application, agents negotiate among each other for computational resources by evaluating given constraints (e. g. comparing speedup curves) to optimise the performance of their applications.

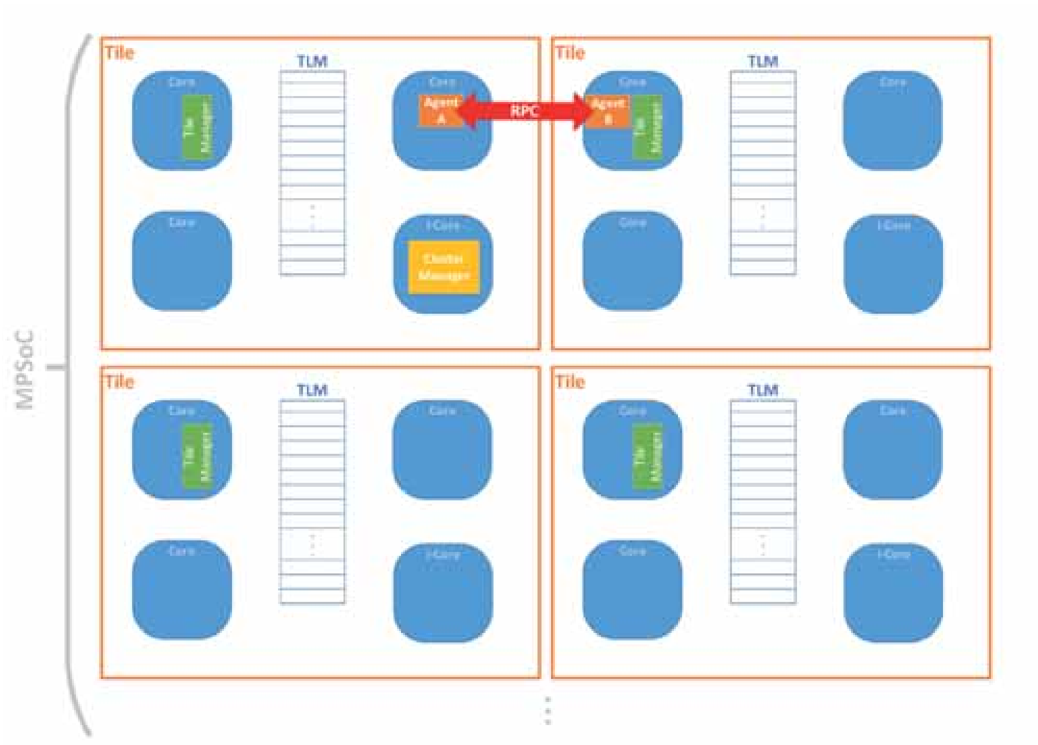


Figure 4.26: The new decentralised Agent infrastructure with Tile Managers and Cluster Managers.

Decentralised Agent System Infrastructure: Considerable effort went into a significantly improved decentralised implementation of the AS with a new underlying software architecture. The interfaces to the applications and the fundamental concepts that were developed in Phases I and II remained the same, i. e. it is compatible. The new infrastructure focuses on a scalable implementation for OctoPOS that helps to support the AS-internal operations, e. g. communication between agents, finding the communication partners, and (distributed) constraint solving. To allow for improved parallel execution and scalability, we implemented all data structures needed for the new infrastructure as lock-free data structures (lists and hash maps) for *i*RTSS.

Figure 4.26 shows the main new components of the decentralised agent infrastructure, i. e. the *Tile Managers* and *Cluster Managers*. Agents are tile-local entities that act on behalf of their application. The data that represents an agent is stored on a particular tile (ideally on a tile within the claim that is managed by the agent) and it can be migrated to another tile. Therefore, when the application wants to communicate with its agent then it first has to find the tile that currently stores that data structure. Similar, if another agent wants to communicate (via RPCs) with this agent to bargain about resources, then it also has to find it first. The new Tile Managers help in doing so. They exist on every tile and keep track about all agents that reside on that tile (i. e. that have their data structures stored on this tile). Agents that migrate to/from this tile register/unregister from the Tile Manager. Additionally, agents that migrate to other tiles leave a forwarding address behind that is stored in the Tile Manager. If an application/agent tries to communicate with an agent that once resided in a particular tile but recently migrated away, then the Tile Manager might be able to tell to where that agent migrated to. But only the last few migrations are stored and in case the Tile Manager does not know about a particular agent, then it has to be searched. The Cluster Manager comprises several tiles (the particular number is compiler-time configurable) and knows which agents reside in a cluster. This helps searching for a particular agent, instead of querying all tiles individually.

DCOP-based constraint solver: We developed a new distributed constraint solver that is based on the Distributed Constraint Optimization Problem (DCOP). A variety of algorithms exist to solve a DCOP formulation. Preliminary tests showed the *Maximum Gain Message* (MGM) algorithm to be a well-suited candidate for our requirements. MGM is an agent-based local search algorithm that is *incomplete* (i. e. it is not guaranteed to find an optimal solution) and that comes with an *anytime property* (i. e. it gradually improves its solution and the best solution found so far can be extracted at any time). Starting with MGM as a baseline, we analysed its performance and behaviour when applied to our specific scenario of manycore resource management and we conceived multiple novel heuristics that improve the execution time and solution quality significantly, altogether building the foundation of our new *Resource Management MGM* (RESMGM) algorithm.

It is executed on every agent involved in a particular resource bargaining decision and communication takes place via asynchronous messages (RPCs) between *neighbouring* agents, i. e. agents that have a constraint between them. The algorithm uses two types of messages that are iteratively exchanged between neighbouring agents. The *ok?*-message contains an agent's current *variable assignment* (i. e. which resources it aims to use in its claim and the resources that it knows that another agent aims to use in its claim). The agents that receive this message add that knowledge to their own variable assignment. After having received the *ok?*-messages from all neighbours, an agent uses its local view to calculate its current constraint cost and determines for which assignment of its variables it can minimise it. The agents subsequently communicate their improvements via *improve*-messages to all their neighbours. To avoid oscillations, only the agent with the maximum improvement among its neighbours is actually allowed to change its variable assignment. In each iteration, the solution quality of the system improves monotonously. The algorithm can be aborted at any time and returns the best solution found so far, or, if in any iteration no agent was able to make an improvement, then the algorithm terminates and each agent involved in the negotiation knows its new claim.

Publications

[Eib+18] C. Eibel, C. Gulden, W. Schröder-Preikschat, and T. Distler. "Strome: Energy-Aware Data-Stream Processing". In: *Proceedings of the 18th International Conference on Distributed Applications and Interoperable Systems (DAIS '18)* (Madrid, Spain). Lecture Notes in Computer Science (LNCS). Springer, June 2018, pp. 40–57.

[Eic+18] C. Eichler, T. Distler, P. Ulbrich, P. Wägemann, and W. Schröder-Preikschat. "TASKers: A Whole-System Generator for Benchmarking Real-Time-System Analyses". In: *Proceedings of the 18th International Workshop on Worst-Case Execution Time Analysis (WCET 2018)* (Barcelona, Spain, July 3–6, 2018). July 3, 2018, 6:1–6:12. DOI: 10.4230/OASIcs.WCET.2018.6.

[Her+18] B. Herzog, L. Gerhorst, B. Heinloth, S. Reif, T. Hönig, and W. Schröder-Preikschat. "INTSPECT: Interrupt Latencies in the Linux Kernel". In: *Proceedings of the 2018 Brazilian Symposium on Computing Systems Engineering (SBESC '18)* (Salvador, Brazil, Nov. 6–9, 2018). Nov. 2018, pp. 1–8.

[Hön+18a] T. Hönig, C. Eibel, A. Wagenhäuser, M. Wagner, and W. Schröder-Preikschat. "How to Make Profit: Exploiting Fluctuating Electricity Prices with Albatross, A Runtime System for Heterogeneous

HPC Clusters”. In: *Proceedings of the 8th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS 2018)* (Tempe, AZ, USA). ACM, June 12, 2018, Article No. 4. DOI: `10.1145/3217189.3217193`.

[Hön+18b] T. Hönig, C. Eibel, A. Wagenhäuser, M. Wagner, and W. Schröder-Preikschat. “Making Profit with Albatross: A Runtime System for Heterogeneous High-Performance-Computing Clusters”. In: *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing (HPDC’18)* (Tempe, AZ, USA). Poster session. ACM, June 13, 2018, pp. 11–12. DOI: `10.1145/3220192.3220457`.

[RS18a] S. Reif and W. Schröder-Preikschat. “A Predictable Synchronisation Algorithm”. In: *Proceedings of the 23rd Annual Symposium on Principles and Practice of Parallel Programming (PPoPP ’18)* (Vienna, Austria, Feb. 24–28, 2018). Poster session. ACM, Feb. 2018, pp. 415–416. DOI: `10.1145/3178487.3178533`.

[RS18b] S. Reif and W. Schröder-Preikschat. *Predictable Synchronisation Algorithms for Asynchronous Critical Sections*. Tech. rep. CS-2018-03. Erlangen, Germany: Friedrich-Alexander-Universität Erlangen-Nürnberg, Department Informatik, Mar. 2018. DOI: `10.25593/issn.2191-5008/CS-2018-03`.

[Wäg+18] P. Wägemann, C. Dietrich, T. Distler, P. Ulbrich, and W. Schröder-Preikschat. “Whole-System Worst-Case Energy-Consumption Analysis for Energy-Constrained Real-Time Systems”. In: *Proceedings of the 30th Euromicro Conference on Real-Time Systems (ECRTS ’18)* (Barcelona, Spain, July 3–6, 2018). July 2018, 24:1–24:25. DOI: `10.4230/LIPIcs.ECRTS.2018.24`.

C3: Compilation and Code Generation for Invasive Programs

Gregor Snelting, Jürgen Teich

Jorge A. Echavarria, Andreas Fried, Sebastian Graf, Frank Hannig, 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.

In the second funding phase, a major focus of research has been the quest for (higher) predictability of non-functional aspects of invasive parallel program execution, i. e. performance, fault tolerance, and security. The first part of this report will be on these aspects related to compiler and compilation.

In the starting third funding phase, our major research focus will be on run-time requirement enforcement of multiple non-functional execution qualities such as user-specified performance and energy corridors. This includes (a) constant latency/throughput loop processing, (b) approximate loop processing including language extensions to specify imprecise loop variables. (c) a symbolic code generator for TCPA targets shall be developed to support the symbolic loop parallelisation techniques developed in the previous funding phases. Moreover, we will investigate (d) compiler optimisations for the *i*-Core, (e) optimisations for FPGAs, and (f) discovery of invasive programming patterns.

We now shortly summarise results on individual aspects of the first six months of the third funding phase.

Compiler synthesis

The instruction set architectures (ISAs) of current processors are extended at an ever-increasing pace. In addition, we see the advent of *application-specific, reconfigurable* processors (such as the *i*-Core), which add new instructions for each application that runs on them.

A compiler should be able to support ISA extensions as quickly as possible. In order to support a new instruction, the compiler needs a model of it during instruction selection, i. e. the translation from the compiler's intermediate representation to machine language. This model usually takes the form of a set of *IR patterns* that describe for which IR constructs the new instruction may be substituted.

Maintaining the set of patterns by hand is a tedious and error-prone process, which we have demonstrated by analysing the instruction selectors of both GCC and clang.

We have therefore developed a tool to automatically generate IR patterns for a set of machine instructions, ranging from just a new ISA extension to a sizeable portion of the x86 instruction set [BFH18]. Our tool takes as input the semantics of both the compiler's IR operations and the machine instructions. Then, it synthesises an exhaustive set of IR patterns for each instruction. We can thus be sure that the compiler is able to identify every opportunity to use a new instruction.

In particular, we support synthesising patterns for instructions which access memory by using an efficient encoding of the relevant subset of the address space.

Compiler optimisations targeted at FPGAs

Conventional compilers are tailored to produce machine code that runs efficiently on general purpose CPUs. However, in order to use accelerators such as the *i*-Core efficiently, the compiler needs to be aware of the performance models of FPGAs as well as CPUs, and needs to optimise for both targets.

A prime example of these different optimisation targets is bit-width analysis: On a CPU, knowing the precise bit-width of a value is of no use, since the CPU can usually compute any simple operation on a full word in one cycle. On the other hand, knowing the bit-width of a value while synthesising hardware for a reconfigurable accelerator will always help to conserve FPGA resources.

Our preliminary research shows that there are in fact opportunities for bit-width reduction. Figure 4.27 shows the bit-width distribution of a simple example program. We can see that a majority of values declared as 32 bit wide in the source code do not actually need as many bits in the algorithm at hand.

Our next step will be to take a more detailed look at the interface between code running on the CPU and code running on an accelerator. While optimisations are well-understood for both domains individually, we expect that there is an opportunity for performance improvement

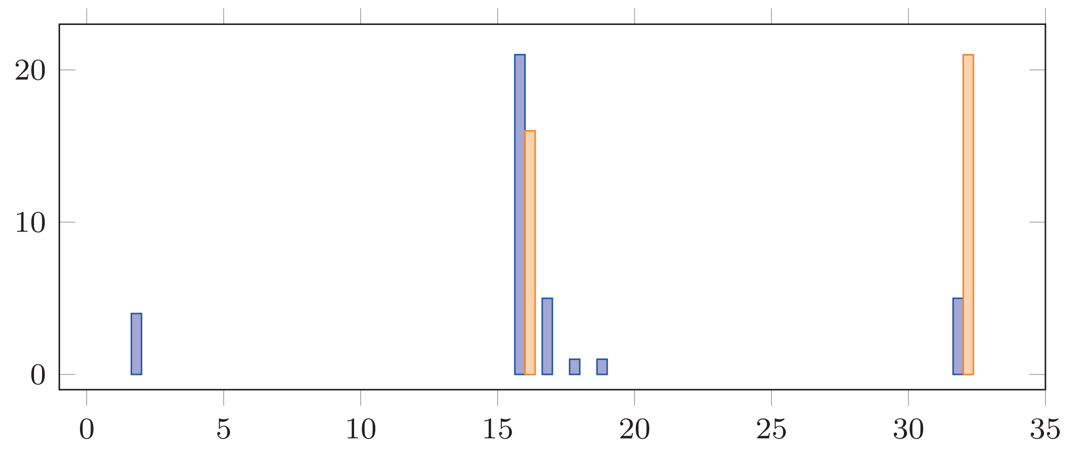


Figure 4.27: A comparison of the distribution of bit-widths as declared in the source code (orange) and as actually needed for the computation (blue) of an example program. Note that the bit-width analysis has deduced that most values declared as 32 bits wide actually require little more than 16 bits to compute.

when compiler and hardware synthesiser are able to communicate more analysis information between themselves.

C3

Predictability of multiple non-functional properties

We demonstrated the advantages of invasive computing as an enabler for predictability of multiple non-functional properties in the context of a cyber-physical system design [Sou+18; Bra+19]. In particular, we presented the application and multiprocessor implementation of a reliable and time-predictable acceleration of object detection algorithms for hard real-time control of an inverted pendulum (see Figure 4.28).

Two aspects make this case study interesting: First, since the camera provides frames at a fixed rate of 50 Hz, missing a single frame will may render the pendulum uncontrollable, making this a hard real-time application. Second, we implemented this computationally intensive, distributed signal processing application on our FPGA-based prototype. Because the prototype runs at only 33 MHz, meeting the required latency bound of 20 ms by pure computing power is impossible.

Project C3, in particular, was responsible for mapping the employed object recognition pipeline onto the TCPA provided by the prototype system, as well as for compiling the overall application for the prototype system such that the given latency requirements of 20 ms per frame were met.

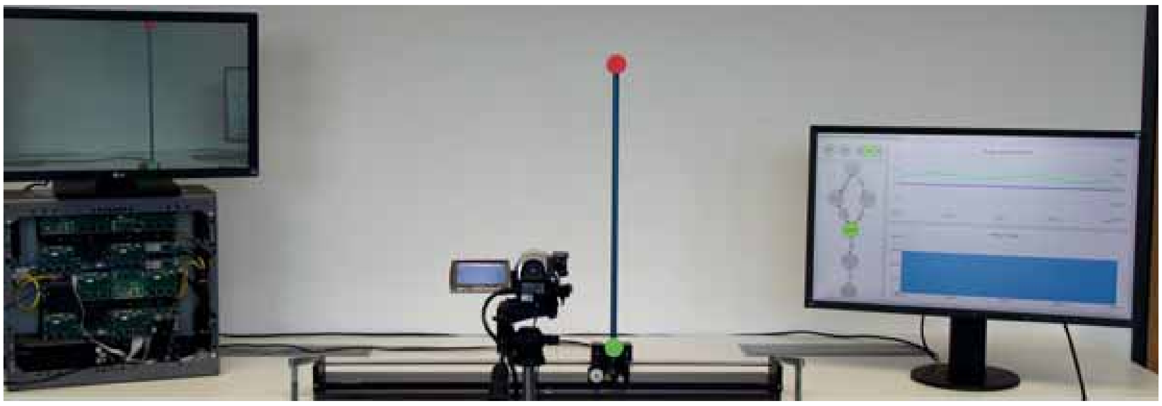


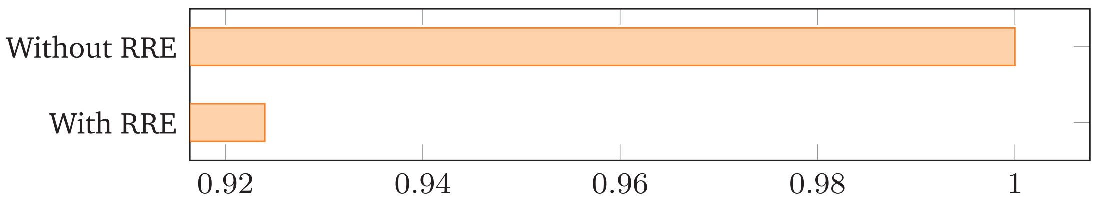
Figure 4.28: Video-based control of an inverted pendulum using a red and green marker attached to the rod. Each frame, the positions of both markers are detected using an object detection pipeline and the angle between them is calculated. The angle of the rod and position on the track are then used to control the motor of the cart, balancing the pendulum.

Run-time requirement enforcement on TCPAs

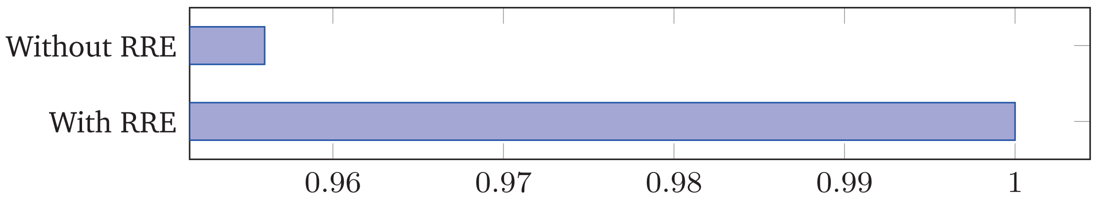
Loop bounds are often unknown until run time, making it difficult to analyse non-functional properties such as latency or energy at compile time. Similarly, static allocations of processing resources to loop computations might be too conservative with respect to given performance requirements, or not optimal with respect to energy consumption. With invasive computing, this problem is exacerbated since the number of available processing elements only becomes known at run time.

To still satisfy requirements when accelerating loop nests under this uncertainty of loop bounds and mapping, we formalised an approach to run-time requirement enforcement of loops on TCPAs: At run time, select a mapping among a set of candidates that satisfies a given set of requirements while optimising secondary objectives. Because the candidate search space of suitable mappings might be prohibitively large to evaluate at run time, we further introduce two approaches to reduce its cardinality: (1) architecture-specific reduction by solving for parts of the mapping from the requirements and (2) design-time reduction by finding a k-subset of mappings that maximises the number of loop bounds where the requirements are satisfied.

We implemented these first ideas on run-time requirement enforcement for TCPAs and demonstrated their effectiveness with a case study, where a Sobel filter with a latency requirement and energy as its secondary objective. The case study shows the effectiveness of our approach: We can satisfy given latency requirements while easily saving up to 10% in energy (see Figure 4.29) compared to using a single, static mapping.



(a) Normalised mean of energy



(b) Normalised coverage

Figure 4.29: Using run-time requirement enforcement for a Sobel filter loop nest (a) can lower the estimated average energy consumption and (b) improves the coverage (that means the percentage of loop bounds where there is a mapping that satisfied the given requirements) of the loop bound space.

Miscellaneous

C3 Finally, the dissertation of Alexandru Tanase on Symbolic Loop Parallelization has been extended and accepted by Springer as a book publication [THT18].

Publications

[Bra+19] M. Brand, M. Witterauf, É. Sousa, A. Tanase, F. Hannig, and J. Teich. "*-Predictable MPSoC Execution of Real-Time Control Applications Using Invasive Computing". In: *Concurrency and Computation: Practice and Experience* (Feb. 2019). DOI: 10.1002/cpe.5149.

[BFH18] S. Buchwald, A. Fried, and S. Hack. "Synthesizing an Instruction Selection Rule Library from Semantic Specifications". In: *Proceedings of 2018 IEEE/ACM International Symposium on Code Generation and Optimization*. CGO '18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3168821.

[Qia+18] B. Qiao, O. Reiche, F. Hannig, and J. Teich. "Automatic Kernel Fusion for Image Processing DSLs". In: *Proceedings of the 21st International Workshop on Software and Compilers for Embedded Systems (SCOPES)* (St. Goar, Germany). ACM, May 28–30, 2018, pp. 76–85. DOI: 10.1145/3207719.3207723.

[Rei18] O. Reiche. “A Domain-Specific Language Approach for Designing and Programming Heterogeneous Image Systems”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 5, 2018.

[SHT18] C. Schmitt, F. Hannig, and J. Teich. “A Target Platform Description Language for Parallel Code Generation”. In: *Workshop Proceedings of the 31st GI/ITG International Conference on Architecture of Computing Systems (ARCS)* (Braunschweig). Berlin: VDE VERLAG GmbH, Apr. 9–12, 2018, pp. 59–66.

[Sou+18] É. Sousa, M. Witterauf, M. Brand, A. Tanase, F. Hannig, and J. Teich. “Invasive Computing for Predictability of Multiple Non-functional Properties: A Cyber-Physical System Case Study”. In: *Proceedings of the 29th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Milan, Italy). IEEE, July 10–12, 2018. DOI: `10.1109/ASAP.2018.8445109`.

[THT18] A. Tanase, F. Hannig, and J. Teich. *Symbolic Parallelization of Nested Loop Programs*. ISBN: 978-3-319-73908-3. Springer, Feb. 2018.

C5: Security in Invasive Computing Systems

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

Simon Bischof, Johannes Götzfried, Pieter Maene, Furkan Turan

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.

In the current third funding phase, we wish to provide mechanisms that can reliably and provably enforce security properties that have been requested by applications at run time even if attacker assumptions are violated, thereby increasing the assumption coverage. Furthermore, we plan to finally extend intra-tile to inter-tile security for integrity and confidentiality.

We took first steps towards enforcing security properties for violated attacker assumptions by introducing a software and control flow integrity architecture (to be combined with static information flow control in collaboration with Project C3, work package C5.1). Furthermore, we started on developing a hardware-based remote (and destructive) reset mechanism to check and (re-)enforce the integrity of computations on remote tiles (in collaboration with the Mercator, work package C5.6).

Attacker Model

Our attacker model consists of four hierarchy levels at which an attacker can operate and execute software. An *X10-attacker* corresponds to an attacker who can run programs written in X10. These programs can be statically checked through a trusted X10 compiler, strongly inhibiting malicious behaviour, 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. Finally, *physical attacks* are the most powerful ones which are considered to be technically difficult to perform, but also to defend against. Physical attacks are not considered in the project.

Results

We mapped the existing landscape of trusted computing architectures to further refine our solutions and be able to design the best possible trusted computing architecture [Mae+18a] for invasive computing platforms. Attackers target many different types of computer systems in use today, exploiting software vulnerabilities to take over the device and make it act maliciously. Reports of numerous attacks have been published, against the constrained embedded devices of the Internet of Things, mobile devices like smartphones and tablets, high-performance desktop and server environments, as well as complex industrial control systems. Trusted computing architectures give users and remote parties like software vendors guarantees about the behaviour of the software they run, protecting them against software-level attackers. We defined the security properties offered by current architectures, and presented detailed descriptions of state-of-the-art hardware-based attestation and isolation architectures from academia and industry (see Figure 4.30). Furthermore, we compared their designs with respect to the security properties and architectural features they offer.

Based on our results of systematically examining existing trusted computing architectures, we developed a scalable approach for current heterogeneous MPSoCs. In order to protect the intellectual property of code running on MPSoCs as well as the confidentiality of sensitive data processed, applications have to be isolated from each other [Mae+18b]. Traditional memory protection and memory management units provide such isolation, but rely on operating system support for their configuration. However, modern operating systems tend to be vulnerable and cannot guarantee confidentiality when compromised. We presented a hardware-based security architecture, complementary to traditional memory protection mechanisms, ensuring code and data confidentiality through transparent encryption, even when the system software has been exploited. Our solution relies on its zero-software trusted computing base to protect against system-level attackers and also supports secure shared memory. We implemented our solution based on the LEON3 softcore processor, including toolchain extensions for developers. Our FPGA-based evaluation shows minimal cycle overhead at the cost

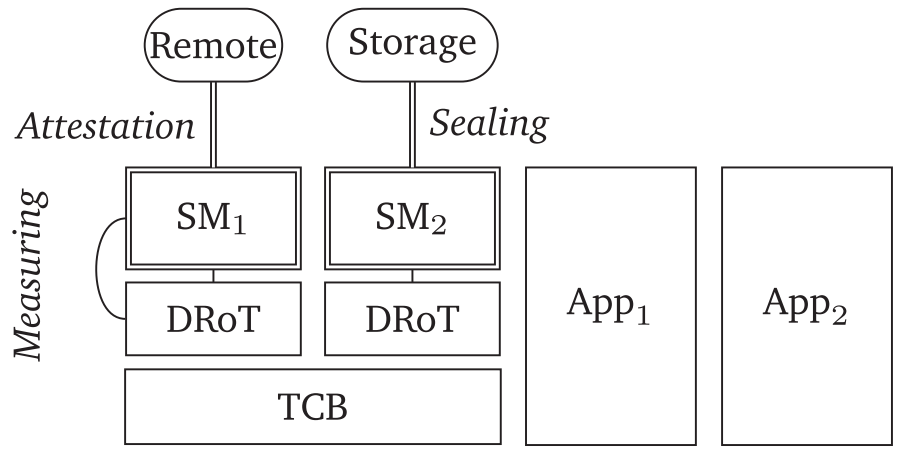


Figure 4.30: A state-of-the-art Protected Module Architecture (PMA) runs multiple Software Modules (SM) side by side, along with one or more unprotected applications. The Trusted Computing Base (TCB) ensures that the internal state of an SM is protected from any other software running on the system. The measurement of an SM establishes a Dynamic Root of Trust (DRoT). The result can be used to attest the state of the module to a remote verifier. By sealing data, the SM can send it securely to untrusted storage.

of a reduced maximum frequency.

Isolation alone is not sufficient to protect against attackers violating our attack assumptions. Applications running on MPSoCs are still sensitive to software vulnerabilities and their exploitation is easily possible. As a countermeasure, we propose a hardware-based security architecture, which in addition to isolation gives certain software integrity and control flow integrity guarantees [Cle+17]. Our architecture allows the processor to defend against a large number of attacks, including code injection, code reuse, and fault-based attacks on the program counter. In addition, the architecture also defends against software copyright infringement and reverse engineering. All protection mechanisms are enforced in hardware using cryptographic techniques. We are the first to propose a mechanism to enforce control flow integrity at the finest possible granularity using cryptographic techniques. The proposed architectural features have been implemented on the LEON3 microprocessor. However, the modified core requires that its software conforms to a strict format. To this end, we additionally designed and implemented a software toolchain to compile source code that adheres to the formatting rules. Several benchmarks were compiled with our toolchain, and were executed on the modified core running on an FPGA, showing an average total execution time overhead of 106% compared to an unmodified LEON3 core. Our hardware evaluation shows a significant clock speed reduction.

A PDG $G = (N, \to)$ is classified correctly, if

1. $\forall n \in N\colon cl(n) \geq \bigsqcup_{m \to n} cl(m)$
2. $\forall n, m \in N\colon MHP(n, m) \wedge c = cda(n, m) \wedge \exists c' \in path(c, n), cl(c') = H \implies cl(n) = H$
3. $\forall n \in I : ucl(n) = cl(n)$ and $\forall n \in O : ucl(n) \geq cl(n)$

Figure 4.31: The new RLSOD criterion, describing when a Program Dependency Graph can be correctly classified by cl, which implies that the program does not interfere. Rule 1 handles sequential dependencies, rule 2 handles nondeterminism arising from scheduling and rule 3 ensures that the classification respects the input and output security levels ucl given by the user.

To increase the performance of control flow integrity solutions, we propose to use static information flow control (IFC) to determine critical points which need to be checked and measured during run time, while the other parts of the application can be considered secure. Our work will leverage our JOANA system, which uses a new, groundbreaking algorithm for probabilistic noninterference [Bis+18] (see Figure 4.31) and provides IFC for full multithreaded Java. Before we implement it for X10, we will implement a proof-of-concept CFI in Java and combine it with JOANA.

Isolation and control flow integrity solutions will only be used if existing software can be protected. Therefore, we present a tool suite that protects the confidentiality and integrity of code by shifting selected functions into TEEs [Laz+18]. Our approach works entirely on binary-level and does not require the adaption of source code projects or build environments, nor does it require compiler-level patches. Programmers provide a list of ELF symbols pointing to the functions that should be protected. After post-processing an ELF binary with our tool suite, the selected functions are not present in cleartext anymore. Only after attesting to a remote party that the loading enclave behaves with integrity, the functions are decrypted, but remain inside the enclave protected against reverse engineering. By allowing programmers to move selected function into TEEs, without patching their source code, we provide a convenient way to enable TEEs in existing projects while preserving the flexibility for a fine-grained security and performance trade-off. We evaluated our tool using a real-world gaming application, confirming the practicability of our approach for existing projects. With our tool suite, we overcome the limitation of a fragmented TEE landscape using a unified API.

Traditionally, information flow control checks for non-interference as security property. This results in a binary answer: Either the secret inputs of the program cannot have any influence on the program output or the security analysis finds that there may be such an influence, however slight. In the latter case, the program is rejected as insecure. In certain cases, however, such influence cannot be prevented since it is part of that program's functionality. With quantitative IFC, one can deduce bounds on how many bits a program can reveal, giving a much more fine-grained answer than traditional IFC. As a part of a master's thesis, a basic quantitative IFC algorithm is currently being integrated into JOANA. It uses a data flow analysis to calculate constant bits and dependency relations between different bits in the program. This results in a dependency graph on the bit level. A minimal cut is then computed to obtain a minimal number of bits which determine the output of the program. Thus, we get a sound approximation of the leakage. For if-statements, we increase precision in the then- and else-branch by using guarantees we get from knowing that the if-condition must have been true or false, respectively. By using techniques based on summary edges, the analysis is interprocedural, including arbitrary recursion. We expect the thesis to be completed by the end of 2018.

Summary

The invasive computing paradigm offers applications the possibility of dynamically spreading their computation in a multicore/multiprocessor system in a resource-aware way. If applications are assumed to act maliciously, many security problems arise. We argue that all our solutions for isolating applications as well as for guaranteeing software and control flow integrity are able to enforce security on invasive computing architectures.

Outlook

As the next steps, we will continue the work started on work packages C5.1, C5.2 and C5.6 (see above). Furthermore, we will focus on work package C5.5 (virtual shared memory encryption). To this end, we want to provide a possibility to shift existing functions into encrypted containers by leveraging secure encrypted virtualisation provided by recent general purpose CPUs. Integrated into the InvasIC x86 port (in collaboration with Project C1) it will contribute to inter-tile security for confidentiality.

Publications

[Bis+18] S. Bischof, J. Breitner, J. Graf, M. Hecker, M. Mohr, and G. Snelting. “Low-Deterministic Security For Low-Deterministic Programs”. In: *Journal of Computer Security* 26 (3 2018), pp. 335–366. DOI: 10.3233/JCS-17984.

[Cle+17] R. de Clercq, J. Götzfried, D. Übler, P. Maene, and I. Verbauwhede. “SOFIA: Software and Control Flow Integrity Architecture”. In: *Computers & Security* 68 (2017), pp. 16–35. DOI: 10.1016/j.cose.2017.03.013.

[Laz+18] T. Lazard, J. Götzfried, T. Müller, G. Santinelli, and V. Lefebvre. “TEEshift: Protecting Code Confidentiality by Selectively Shifting Functions into TEEs”. In: *Proceedings of the 3rd Workshop on System Software for Trusted Execution*. SysTEX ’18. Toronto, Canada: ACM, 2018, pp. 14–19. DOI: 10.1145/3268935.3268938. URL: https://www1.cs.fau.de/teeshift.

[Mae+18a] P. Maene, J. Götzfried, R. de Clercq, T. Müller, F. Freiling, and I. Verbauwhede. “Hardware-Based Trusted Computing Architectures for Isolation and Attestation”. In: *IEEE Transactions on Computers* 67.3 (2018), pp. 361–374. DOI: 10.1109/TC.2017.2647955.

[Mae+18b] P. Maene, J. Götzfried, T. Müller, R. de Clercq, F. Freiling, and I. Verbauwhede. “Atlas: Application Confidentiality in Compromised Embedded Systems”. In: *IEEE Transactions on Dependable and Secure Computing* (2018). DOI: 10.1109/TDSC.2018.2858257.

D1: Invasive Software–Hardware Architectures for Robotics

Tamim Asfour, Walter Stechele

Dirk Gabriel, Fabian Paus

The main research topic of Project D1 is the investigation of the benefits of invasive computing for humanoid robotics applications. These applications combine methods and algorithms from the area of computer vision and motion generation with concurrent processes and time-varying resource demands. In order to use existing non-invasive implementations of algorithms, we further investigate how these legacy applications can be automatically ported to utilise the benefits of invasive computing. As an exemplary robotic application, we intend to port parts of the existing perception pipeline for scene analysis and affordance extraction both manually and automatically.

In 2018, we started to develop a prototypical *invasive wrapper*, which translates non-invasive resource requests for memory and threads to claims. Furthermore, we investigated the role of support relations in scene understanding for robotics. Part of the presented algorithm will be used as a case study.

Automatic Porting of Non-invasive Applications

The parametrisation of applications is required in order to perform the execution within timing constraints while the number of available resources and the computational complexity of the processed problem varies. In 2018, we have improved an automated hybrid approach which extracts operating points describing the optimal combinations of resource requirements and parameter settings depending on the complexity of the problem. The points are evaluated within the design space exploration developed by Project A4.

During run time, the set of required resources for an optimal quality of service level is determined first and invaded on the system. If the invade call fails due to too few resources available, the next worse operating point is selected and the request is repeated. The first invade

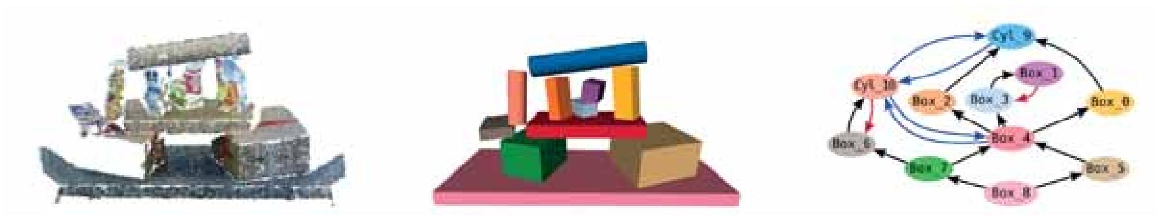


Figure 4.32: Shapes are extracted from a 3D point cloud captured by an RGB-D camera and used as input (left). Given the 3D geometry and pose of each object (middle), we extract binary support relations and represent them as a directed graph (right). Black edges represent certain bottom-up support, blue edges are unknown and red edges are potential top-down support relations.

call which is successful provides the information for the next iteration of the application and therefore the corresponding parameter set is used within the application. Using this approach, we could show that an application fulfils its timing constraints in all situations even after sudden changes.

This approach will be used in the invasive wrapper. In order to support legacy applications without modifying them, we started to implement the pthread library using invasion primitives.

Extraction of Support Relations

Understanding relations between objects and reasoning about manipulation action effects is a necessary skill for a robot working in cluttered environments. To this end, we developed a method to extract physically plausible support relations from perceived depth camera images [Kar+18]. The knowledge about support relations was used to safely execute actions which require bimanual manipulation. One hand was used to secure a potentially unsafe object, while the other hand executes the manipulation action.

A support relation between objects can be defined as follows. Given a set of objects $\mathcal{O}$ represented by their geometric 3D shape, our goal is to extract binary support relations $\mathtt{SUPP}(\cdot,\cdot)$ between each object pair. For two objects $\mathtt{A},\mathtt{B} \in \mathcal{O}$, we denote $\mathtt{SUPP}(\mathtt{A},\mathtt{B})$ iff removing $\mathtt{A}$ causes $\mathtt{B}$ to lose its motionless state, i. e. $\mathtt{A}$ supports $\mathtt{B}$. For instance, $\mathtt{B}$ falls when $\mathtt{A}$ is removed. This definition incorporates physical object interactions and enables us to assess the scene's stability when certain actions are performed. To represent support relations of a given object set $\mathcal{O}$, we define a support graph whose nodes represent objects and whose edges indicate support.

Figure 4.32 shows the different steps of the extraction process. The

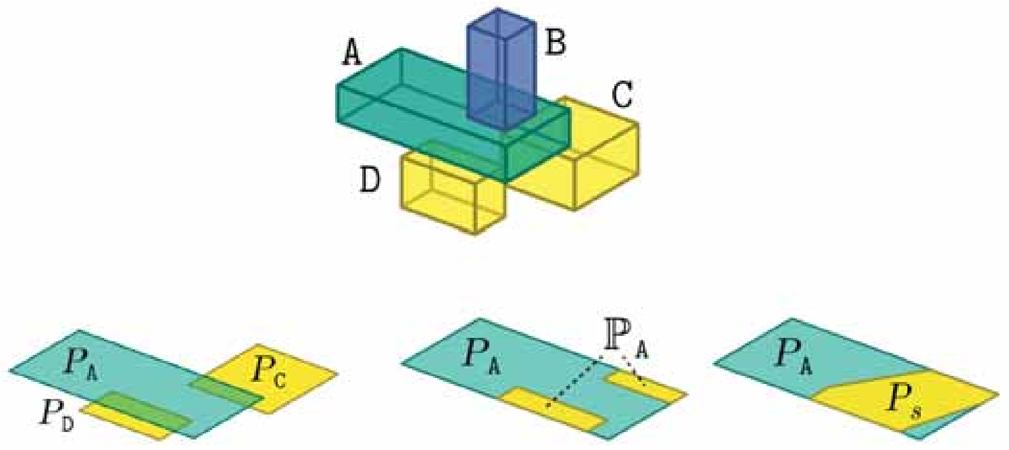


Figure 4.33: Example of support polygon construction. (top) The object constellation. `A` supports `B` and is supported by `C` and `D`. (left) The objects `A`, `C` and `D` are projected onto the ground, creating polygons $P_{\mathtt{A}}$, $P_{\mathtt{C}}$ and $P_{\mathtt{D}}$, respectively. (middle) $P_{\mathtt{C}}$ and $P_{\mathtt{D}}$ are intersected with $P_{\mathtt{A}}$, constructing $\mathcal{P}_{\mathtt{A}}$. (right) The support polygon P_s is the convex hull of the polygons in $\mathcal{P}_{\mathtt{A}}$.

input is a point cloud captured by an RGB-D camera. State-of-the-art segmentation (LCCP) and primitive fitting (RANSAC) methods are used to extract object hypotheses and their geometric features. Based on this geometric representation, we build a graph containing edges for detected support relations.

For safe manipulation, potential top-down support edges are relevant. We extract these edges using support polygon analysis. Figure 4.33 depicts an example of this method. The support area ratio r_s is computed as the proportion of the supported area of `A` to its total area:

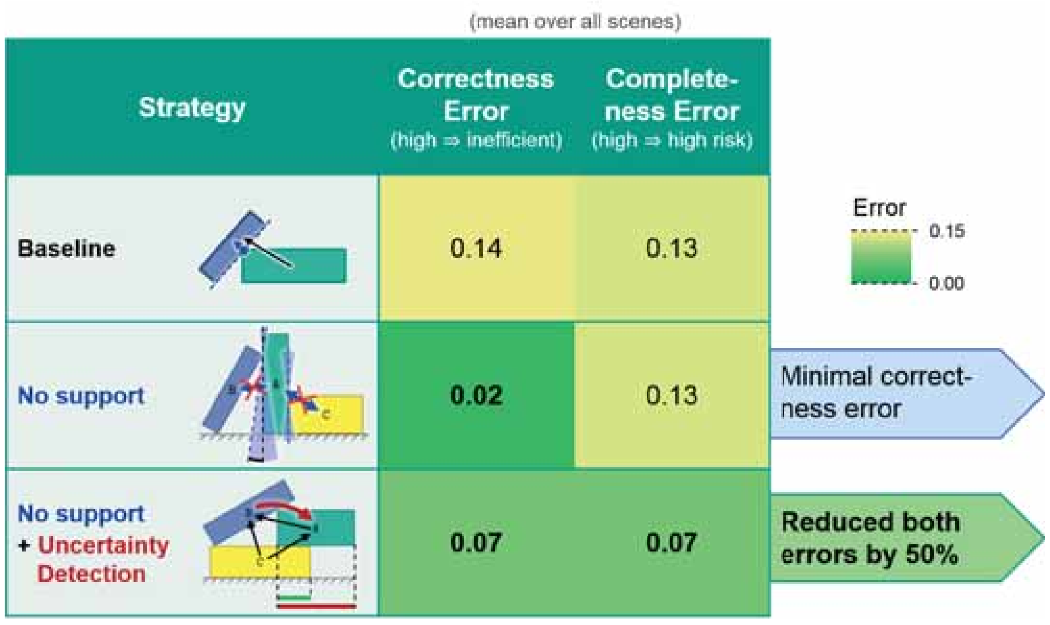
D1

$$r_s = \frac{\text{area}(P_s)}{\text{area}(P_{\mathtt{A}})} \quad . \tag{4.5}$$

Note that $r_s \in [0, 1]$, where $r_s = 1$ if `A` is fully supported, and $r_s = 0$ if `A` is not supported at all, i. e. `A` is floating. We can now define a threshold on r_s at which we assume top-down support exists.

We evaluated different variants of our approach against a state-of-the art implementation. The results are summarised in Figure 4.34. Our implementation and the data used for the evaluation is open-source as well as a script to reproduce the presented results[8].

[8] `https://gitlab.com/h2t/semantic-object-relations/`


(mean over all scenes)

Strategy	Correctness Error (high ⇒ inefficient)	Completeness Error (high ⇒ high risk)	
Baseline	0.14	0.13	
No support	**0.02**	0.13	Minimal correctness error
No support + Uncertainty Detection	0.07	0.07	**Reduced both errors by 50%**

Error
0.15
0.00

Figure 4.34: We evaluated three strategies. The baseline approach is a reimplementation of a state-of-the-art approach. The "No support" strategy assumes that no support exists between objects which are nearly parallel to each other. "Uncertainty Detection" uses the polygon support analysis to detect potential top-down support.

Outlook

In 2019, we plan to use a part of the support extraction algorithm, namely the RANSAC algorithm used for determining object geometry from segmented point clouds, as a first non-invasive application to test the automatic porting. To this end, we will extend the invasive wrapper as necessary, including the implementation of more standard library functionality as well as enhanced multi-threading and memory allocation support.

Publications

[Kar+18] R. Kartmann, F. Paus, M. Grotz, and T. Asfour. "Extraction of Physically Plausible Support Relations to Predict and Validate Manipulation Action Effects". In: *IEEE Robotics and Automation Letters (RA-L)* 3.4 (Oct. 2018), pp. 3991–3998. DOI: 10.1109/LRA.2018.2859448.

D3: Invasive Computing and HPC

Michael Bader, Hans-Joachim Bungartz, Michael Gerndt

Emily Mo-Hellenbrand, Jophin John

The overall goal of Project D3 in Phase III is to show that invasive computing can make a difference in HPC—not only for improving performance and efficiency for individual applications, but also for overcoming some compelling challenges from operating large systems at supercomputing centres.

Invasion for HPC Overview

The biggest concerns of the current supercomputing centres are

- high energy consumption and lack of predictability,
- increasing failure rate due to growing core numbers which in turn aggravate the energy consumption problem,
- increasing complexity for efficiently handling resources due to the introduction of novel manycore architectures.

Invasive computing provides remedies for these problems. Building upon the Elastic MPI (*i*MPI) infrastructure we developed in Phase II, we will investigate using the mechanisms of invasive resource management for power corridor enforcement. This can help to increase predictability and thus, reduce the overall energy cost. To handle hardware failure, we will develop a resource-aware checkpointing infrastructure (*i*Check) for fast adaptive data recovery. We will also investigate the invasion of memory layers with guidance by the applications. This will allow for improved execution despite the increased complexity in resources. On the application side, all these tasks require extreme-scale and realistic simulation codes. We will invasify a set of selected classical HPC applications that were not originally designed for resource elasticity.

Development of Invasive HPC Infrastructure

Finalising the Invasive Infrastructure: We have finalised the implementation of the *i*MPI infrastructure for distributed-memory HPC systems [CG17]. A working prototype of the invasive Resource Manager (*i*RM) along with the application performance monitoring functionality were finalised and were tested on SuperMUC and virtual machine (VM) clusters. Resource-elastic applications developed using *i*MPI were successfully executed on the infrastructure.

Elastic Phase-Oriented Programming (EPOP): EPOP is a programming model designed for simplifying the development of invasive applications. Introducing runtime resource adaptivity to parallel applications is a big challenge because many considerations must be taken into account, e. g. finding a suitable entry point for the joining processes as well as an appropriate exit point for the leaving processes, redistributing data among the new set of resources, synchronising the joining processes with the existing ones, and more. This added complexity in the development of invasive parallel applications motivates a phase-oriented programming concept.

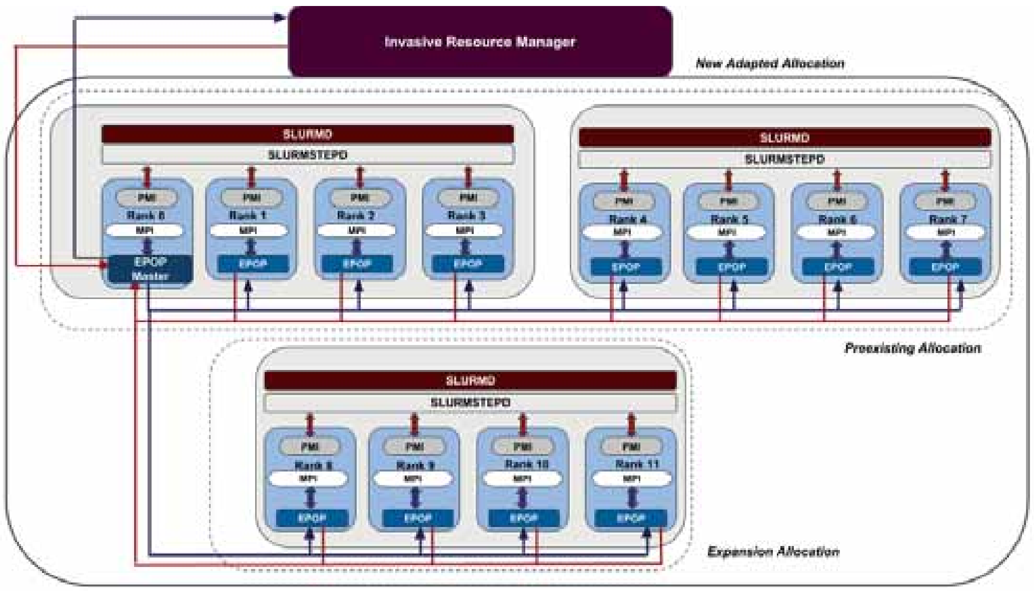


Figure 4.35: Interaction between EPOP and the resource manager.

In the EPOP model, applications are divided into phases that act as an abstraction for providing malleability. Typical parallel applications usually have very distinctive stages with respect to scalability. They usually have an initial phase where initialisation of objects and data takes place, a computation phase(s) where actual computation takes place, and a final phase for writing/saving the computation results and output. According to Amdahl's law, speedup of a program is inhibited by its sequential part. With EPOP, these parallelism inhibiting parts such as the initial and final phases are separated and treated as rigid phases, while the highly parallelisable parts such as the computation phase(s) are treated as resource elastic phases.

EPOP has a user-friendly syntax and provides a simpler programming interface for the *i*MPI library. The programmer can specify in which phases resource adaptation should take place and which routines to call before and during resource adaptation. The programmer is relieved from including the *i*MPI routines in their code. The EPOP driver calls the *i*MPI routines for probing the *i*RM, starting and committing the adaptation at specified points to control the program workflow.

Integration of Apache Spark with *i*RM: The scheduling strategies employed by current HPC systems often result in multiple unutilised nodes. In order to improve system throughput, apart from improving the scheduling strategies, an alternative way is to assign idle nodes to a series of embarrassingly parallel applications that act as "buffers" in an elastic environment. For this purpose, we consider data analytics applications running on Apache Spark. These are embarrasingly parallel machine learning applications used for data streaming such as data enrichment, complex session analysis and batch processing. They are used for obtaining insights on correlation and trends in big data, for marketing purposes and sentimental analysis. As the number of idle nodes changes constantly due to job completions and new launches, these data analytics applications must be able to adapt to the changing set of resources.

We integrated the *i*RM with Apache Spark, a commonly used data analytics infrastructure. We developed communication modules and implemented two scheduling strategies specifically for the data analytics applications—*idle node filling* and *performance-based scheduling*.

With *idle node filling*, all static and elastic MPI applications are first scheduled based on the existing scheduling strategy, then all unutilised nodes (nodes that are not assigned to MPI applications) are assigned to Spark applications. MPI applications are given higher priority in this scenario, and Spark applications are restricted to use the unutilised (idle) nodes only.

Performance-based scheduling is a strategy based on the number of pending tasks in Spark applications. The Spark master sends the number of drivers that are not yet launched and the number of pending

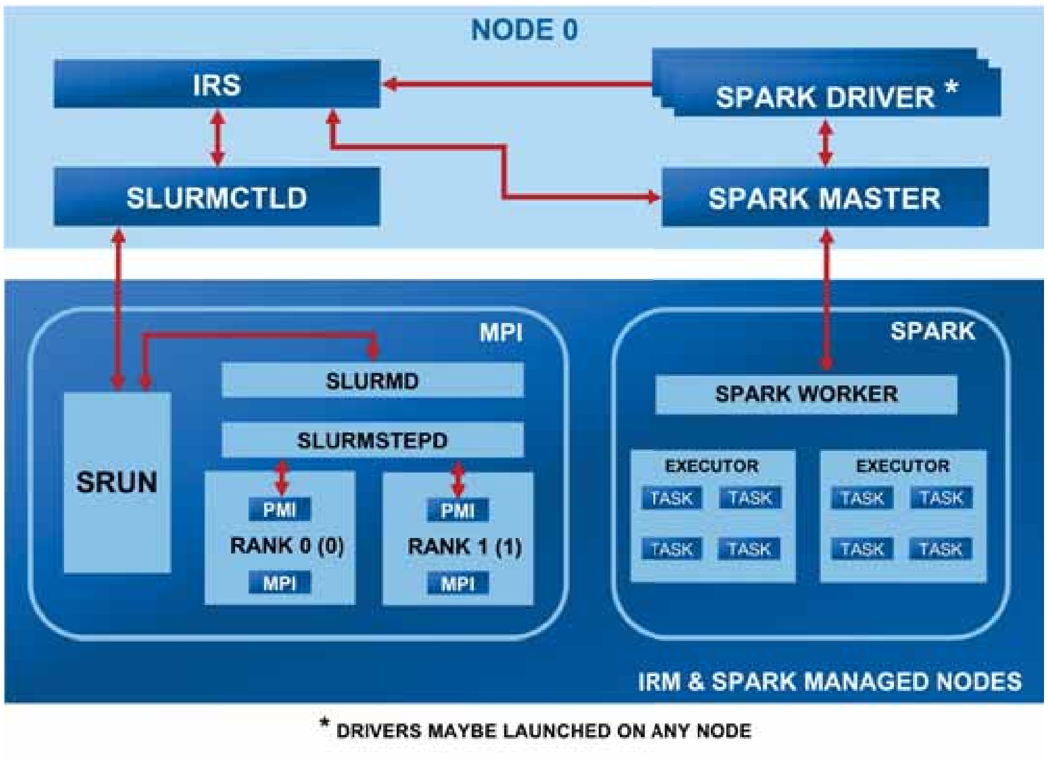


Figure 4.36: Integration of Apache Spark and the invasive resource manager.

tasks of each currently running Spark application to the *i*RM. The new resource assignment for Spark applications is determined based on this information. If the number of waiting drivers increases over time, or the number of pending tasks of a Spark application is above a threshold, more resources are allocated to Spark. On the other hand, if all the Spark applications are completed and there are no more waiting drivers, then resources are taken back from Spark.

Development of Invasive HPC Applications

Invasive Earthquake Simulation: The earthquake simulation code *SeisSol* [Uph+17] is a large-scale, real-world application that was originally developed with MPI. With added resource adaptivity, a large improvement in resource efficiency can be expected in rupture simulations, where most of the simulation time is characterised by the dynamics of the rupture process. We adopt the lazy activation approach explored in Project A4 and activate parallel partitions only when seismic waves reach the partition. Development of the invasive *SeisSol* is currently in progress.

Invasive Molecular Dynamics Simulation: The molecular dynamics simulation code *ls1 MarDyn* is another real-world MPI application that simulates the nucleation processes in large-scale chemical reactors. Nucleation happens in the context of condensation, i. e. when droplets are formed in an over-saturated gas phase (vapor). Such a droplet formation is induced by a local density peak, and it happens in a spontaneous, very heterogeneous and hardly predictable way. Due to the nature of this process, uneven computational workloads are generated across simulation sub-domains. Resource adaptivity can facilitate the dynamicity and mitigate imbalances in workloads. Development of the invasive *ls1 MarDyn* is currently in progress.

Outlook

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

- **Power Corridor Management:** As the next step, we will add the power corridor management capability to the scheduling infrastructure, which can expand or shrink the resources allocated to an application as per the power requirements. The *i*RM will analyse the power usage of an application, create a power model and uses it to predict the maximum and minimum power usage for a time series. The *i*RM will allocate or take back resources to or from applications in an efficient way such that the total power usage will remain in the required power corridor.

- **Invasive Checkpointing System:** Another task is to implement a novel multilevel checkpointing system (*i*Check) for the invasive HPC infrastructure. *i*Check will provide fault tolerance by closely interacting with the *i*RM. The *i*RM will allocate nodes for checkpointing. The checkpointing partition can expand and shrink across nodes according to resource availability. In addition to fault tolerance, *i*Check will also provide automatic data transfer between nodes when an application expands or shrinks. An API will be provided to developers to specify the data movement during resource expansion or reduction. This system will be integrated with EPOP to simplify the programming interfaces.

- **Evaluation and Testing with Invasive Applications:** Performance analysis will be conducted on the invasive *SeisSol* and *ls1 MarDyn* codes to evaluate the impact of resource adaptivity on the application level. Furthermore, these codes will be used for testing the power corridor management and *i*Check's fault-tolerant functionalities. They will eventually adopt the new API provided by *i*Check for automatic data transfer during resource adaptation.

Publications

[CG17] I. A. Comprés Ureña and M. Gerndt. "Towards Elastic Resource Management". In: *Proceedings of the 11th Parallel Tools Workshop*. Sept. 11–12, 2017.

[Uph+17] C. Uphoff et al. "Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake". In: *SC17: The International Conference for High Performance Computing, Networking, Storage and Analysis Proceedings*. ACM. 2017. DOI: 10.1145/3126908.3126948.

Z: Central Services

Jürgen Teich

Ina Derr, Frank Hannig, Jürgen Kleinöder, Stefanie Kugler, Sandra Mattauch

The central activities and services in our CRC/Transregio 89 are coordinated and organised by Project Z. These activities and services are subdivided into two parts:

The first part is administrative support, the organisation of meetings (i. e. internal project meetings and workshops, Doctoral Researcher Retreats (DRR)) and assistance for visits of guest researchers and researchers travelling abroad. Project Z also provides technical support and a multitude of measures improving infrastructure and supporting data management tools for communication and collaboration (subversion repository, mailing lists, Wiki). We continue to support our researchers in facilitating equal career opportunities as well as balancing family and scientific career (e. g. kidsboxes or babysitting service during our annual meetings). In order to promote our doctoral researchers, we also organise courses and trainings. Last but not least, the support of all management activities, financial administration and bookkeeping belong to our central service unit.

The second part of Project Z concerns public relations. This comprises establishing contacts with important research sites as well as the inception of an international Industrial and Scientific Board. Scientific ideas and results of the CRC/Transregio 89 are discussed regularly with guest researchers at our "The InvasIC Seminar" as well as various workshops and conferences. Among others, our website www.invasic.de, promotion material, press releases and media relations are also provided by Project Z. Finally, we also support and organise the publishing process of central publications like the "InvasIC Annual Report". All of our 2018 activities can be found summarised in Part III of this report.

Z2: Validation and Demonstrator

Jürgen Becker, Frank Hannig, Thomas Wild

Nidhi Anantharajaiah, Marcel Brand, Joachim Falk, Leonard Masing, Sven Rheindt, Akshay Srivatsa

The major contribution of Project Z2 is to provide a joint environment for validating the principles of invasive computing. The contributions of the different projects across all project areas are integrated into one platform to demonstrate the advantages of invasive computing such as improved efficiency, resource utilisation, speedup, and *-predictability[9]. In the second funding phase, ProDesign proFPGA systems were set up, containing four Virtex-7 2000T FPGAs, thus allowing for large designs to be prototyped. This prototype has allowed us to implement tiled invasive hardware architectures containing up to 16 multicore (RISC) and manycore (TCPA) tiles, connected via the invasive NoC (*i*NoC), with room for extension and flexibility. To build a working platform, Project Z2 integrated peripherals like the transactor, SSRAM and DDR3 memory controllers into the common demonstrator and established the appropriate tool support for the integration of all the invasive components. By offering continuous support for the demonstrator platforms, Project Z2—in close cooperation with the hardware, operating system, compiler and application projects—was able to provide a wholesome system that integrates the work of all projects.

Demonstrators

At the review meeting in January 2018, two major demonstrators that showcase the advantages of invasive computing were presented. Project Z2 had a leading role towards a successful demonstration of contributions from individual projects by managing the integration efforts in hardware and software. Project Z2 provided the FPGA demonstrator

[9] J. Teich et al. "Language and Compilation of Parallel Programs for *-Predictable MPSoC Execution using Invasive Computing". In: *Proceedings of the 10th IEEE International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. Lyon, France, Sept. 21–23, 2016, pp. 313–320. DOI: 10.1109/MCSoC.2016.30.

platform in the form of the proFPGA multi-FPGA prototyping platform as well as the basic IP blocks and hardware components of the architecture. The platform and prototyping environment was set up in a way which allowed and simplified the integration of all hardware components and fulfils all requirements for the demonstration of the entire invasive hardware/software stack.

One of the presented demonstrators is based on Shallow Water Equations (SWE). The Shallow Water Equations HPC application computes the wave propagation triggered by an earthquake in the ocean. The individual layers of the demonstrator are shown in Figure 4.37.

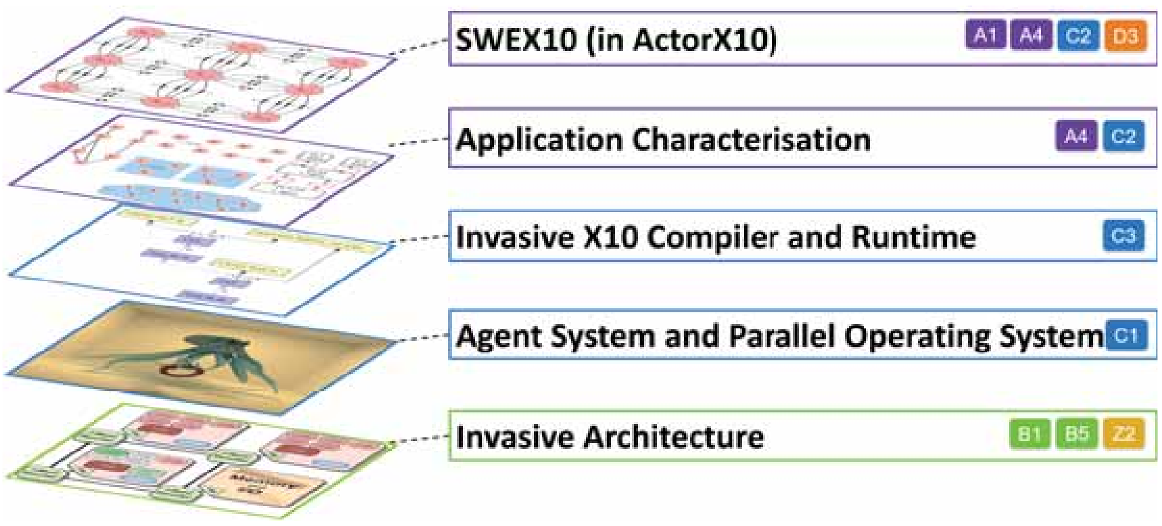

SWEX10 (in ActorX10)
A1 A4 C2 D3
Application Characterisation
A4 C2
Invasive X10 Compiler and Runtime
C3
Agent System and Parallel Operating System
C1
Invasive Architecture
B1 B5 Z2

Figure 4.37: Shallow Water Equations (SWE) application and full demonstrator stack.

The demonstrator shows all the layers working together, from language level (actorX10) and Design Space Exploration (DSE), the middleware consisting of the compiler and operating system down to the hardware level. The demonstrator hosted a 16 tile design of an invasive architecture running on the FPGA prototyping platform. The tiles were connected by the *i*NoC and some of the tiles contained the *i*-Core which improved overall performance by offloading memory intensive parts of the application. The demonstrated scenario used lazy activation, i. e. more and more tiles were invaded and activated as the wave progressed and covered more of the designated area under observation.

Z2

The second major demonstrator showcased *-predictability using invasive computing. The isolation created through an invasion of non-shared resources of an MPSoC was demonstrated here for the application of an inverted pendulum control loop [Sou+18]. The inverted pendulum was controlled by capturing the movements of the pendulum by a standard high-definition camera and then streaming the frame data to the I/O tile of a four tile invasive architecture prototyped on a proFPGA platform, see Figure 4.38.

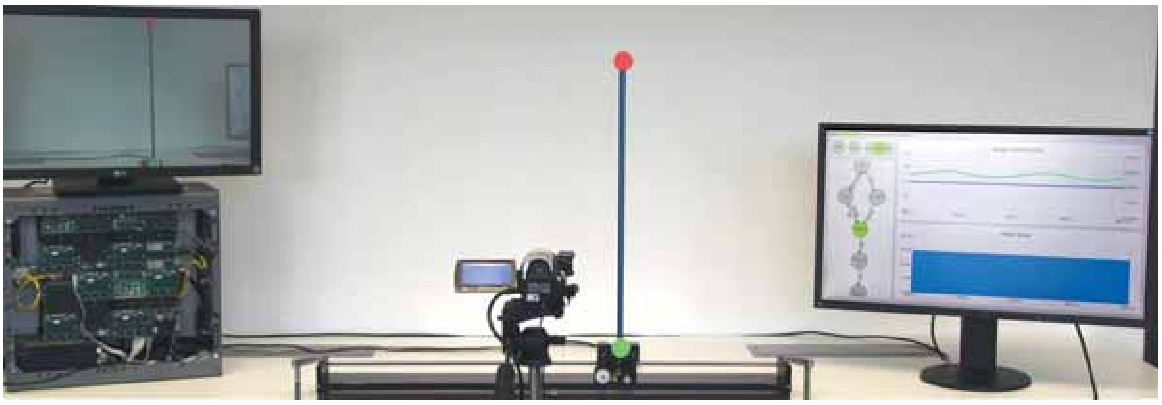


Figure 4.38: Setup of the inverted pendulum real-time demonstrator.

The frame data is periodically copied via DMA transfers to the requesting TCPA tile over the *i*NoC. According to Figure 4.39, a TCPA is invaded to compute a colour segmentation and bounding box algorithm on the frame data scanning for the red and green markers on each end of the pendulum's rod. The position of the markers is then used to calculate the position and angle of the pendulum which in turn activates a PID controller program running on a compute tile. After processing a camera frame, the PID controller updates the power and the direction of the pendulum's motor to balance the rod in an upright position. Each frame has to be processed entirely within 20 ms (50 frames per second) for the inverted pendulum to stay stable [Sou+18]. An application running on a host machine was connected to the invasive tiled architecture via an Ethernet connection to the I/O tile and could be used to track the movement of the pendulum. Apart from demonstrating this distributed hard real-time control being enabled through invasive computing, faults were injected into the application to violate safety properties. Based on our work on fault-tolerant loop replication, we showcased that by switching between different modes of operation like dual modular redundancy (DMR) or triple modular redundancy (TMR), faults may be recognised or even compensated to enable safety properties as well. Finally, it was shown that also security properties can be individually enabled or disabled through invasion. Here, confidentially aspects of IPs such as the PID control algorithm may be enforced through on-the-fly tile-local memory encryption.

This demonstrator is another example of combining the work of several different projects: The architectures provided by Projects B2, B5,

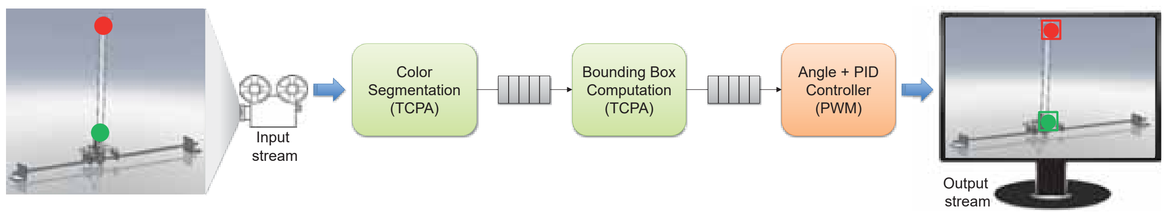

Input stream
Color Segmentation (TCPA)
Bounding Box Computation (TCPA)
Angle + PID Controller (PWM)
Output stream

Figure 4.39: Real-time distributed processing of a 50 Hz camera video stream through invasion of tiles for the different algorithms to be processed within an available time frame of 20 ms.

and Z2 ran an analysed (Project A1) and compiled (Project C3) application on the TCPA, which is configured and continuously provided data by utilising the operating system routines (Project C1). Incorporating the Atlas core [Mae+18b] developed by Project C5 enabled the real-time encryption of confidential programs and data such as the PID control algorithm.

Progress on the Demonstrator Platform

Project Z2 substantially extended the memory subsystem of the demonstrator platform. First, a new SRAM controller was developed and integrated that fully exploits the existing SRAM memory bandwidth. This led to overall performance improvements of the x10-IMSuite benchmarks of up to 45%. Second, the global memory (DDR) was increased from 256 MB to 3.5 GB. This is also an upper bound due to the restriction of a 32 Bit address space. But the extended memory enables the execution of applications with a much bigger memory footprint.

Project Z2 further eased the synthesis flow. A new and highly configurable scripting environment was developed that enables the parallel synthesis of bigger designs. Our multi-FPGA implementation of a 4x4-tile design spreads over 4 Virtex-7 2000T FPGAs that can be synthesised in parallel. Several options can be enabled or disabled for every design like the usage of a DDR or debugging components, etc. This flow eases the cooperation of doctoral researchers and students from the three universities.

System-level testing and debugging played a vital role in the integration and progression of the demonstrator platform. Project Z2 provides an environment that allows performing holistic tests of project specific components and their interplay with the whole demonstrator in an automated fashion. A bare metal test environment is provided that is able to run on the FPGA prototype as well as in a hardware simulator. The environment additionally uses a test suite (provided by Project C1) with real-world applications (like the NAS parallel benchmarks or the x10-IMSuite) running on top of *i*RTSS. Tracing the outputs for error messages helps to categorise them and giving first hints where to focus the further testing on.

Both the synthesis flow as well as the test environment are building blocks and the first step towards the continuous integration and testing environment described below.

Continuous Integration and Testing

In order to further ease and automate the seamless integration of new components and features into our prototype, Project Z2 plans the establishment of a continuous integration and testing environment. As this is common practice for any software development project, we want to establish it into the hardware design flow. When new features are integrated, the design will be synthesised in an automated fashion and an extensive test suite consisting of microbenchmarks as well as real applications benchmarks will continuously be executed.

Validation and Exploration based on Simulation

The system-level simulator InvadeSIM[10,11], which was developed by Project C2, has been one major and operative instrument in the previous funding phases. Particularly in project areas A and D, it has been used as a testbed for invasive programming concepts as well as a basis for architecture exploration and application mapping. The research achievements on modelling and simulation of invasive applications and architectures have resulted in a successful PhD defence of Sascha Roloff in July 2018 [Rol18]. While Project C2 ended, InvadeSIM still has a vital role in the third funding phase, e. g. for first test and integration of investigated run-time requirement enforcement (RRE) techniques. Therefore, its development and extension will be supported by Project Z2. For this purpose, all developments have been structured in one version-control system (*invasicsdk* Git). Here, the integral parts are: (a) InvadeSIM, the high-level simulator for manycore architectures;

[10] 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. DOI: `10.1145/2744769.2744840`.

[11] S. Roloff, F. Hannig, and J. Teich. “High Performance Network-on-Chip Simulation by Interval-based Timing Predictions”. In: *Proceedings of the 15th IEEE/ACM Symposium on Embedded Systems for Real-time Multimedia (ESTIMedia)* (Seoul, Republic of Korea). ACM, Oct. 15–20, 2017, pp. 2–11. DOI: `10.1145/3139315.3139320`.

simulates any X10 program on such architectures, (b) InvadeVIEW, a client-server visualization tool for InvadeSIM, (c) several libraries (e. g. actorX10—a library for actor-based X10 program development, X10CV—an OpenCV wrapper), and (d) example applications. As an example, Project A1 currently validates its theories and techniques on enforcement of non-functional properties of program execution using the simulator infrastructure. For further details, we refer to Project A1 in this report.

Outlook and Public Dissemination

In the third funding phase, Project Z2 is continuing to support all projects, particularly in their work on RRE of non-functional properties as well as run-time verification of properties and constraints. Therefore, Project Z2 will provide appropriate extensions to emulate the power dissipation and temperature behaviour of architectural components. These extensions contain power and temperature models of the components that are capturing the dissipation of static and dynamic power of the components when they would have been implemented on a given ASIC technology. To support the evaluation of run-time monitoring (RRM) and run-time enforcement (RRE) mechanisms such as the concept of leads and aides, Project Z2 will provide probes within invasive architecture prototypes that allow assessing metrics to be optimised or enforced and visualise them to an external observer. Further, several performance-related extensions are already planned for the joint demonstrator platforms, e. g. increasing the SRAM size per tile requiring a further adaption of the memory controller as well as the address map for utilising all available SRAM banks of our proFPGA systems. In cooperation with Project B5, we also plan to implement DMA copies from remote address space to local address space, which will, in turn, ease the design and address management of larger tiled architectures, e. g. ones including TCPAs. Finally, Project Z2 will contribute to the dissemination of the CRC by promoting the accomplished innovations and the research achievements. Here, apart from presentations at events like the Long Night of Sciences (“Lange Nacht der Wissenschaften”, a biannual happening in Erlangen/Nuremberg), we will present one of our joint demonstrators—the invasive inverted pendulum—at DATE 2019.

Publications

[Mae+18b] P. Maene, J. Götzfried, T. Müller, R. de Clercq, F. Freiling, and I. Verbauwhede. “Atlas: Application Confidentiality in Compromised Embedded Systems”. In: *IEEE Transactions on Dependable and Secure Computing* (2018). DOI: 10.1109/TDSC.2018.2858257.

[Rol18] S. Roloff. “Modeling and Simulation of Invasive Applications and Architectures”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 19, 2018.

[Sot+18] E. Sotiriou-Xanthopoulos, L. Masing, S. Xydis, K. Siozios, J. Becker, and D. Soudris. “OpenCL-based Virtual Prototyping and Simulation of Many-Accelerator Architectures”. In: *ACM Trans. Embed. Comput. Syst.* 17.5 (Sept. 2018), 86:1–86:27. DOI: 10.1145/3242179.

[Sou+18] É. Sousa, M. Witterauf, M. Brand, A. Tanase, F. Hannig, and J. Teich. “Invasive Computing for Predictability of Multiple Non-functional Properties: A Cyber-Physical System Case Study”. In: *Proceedings of the 29th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Milan, Italy). IEEE, July 10–12, 2018. DOI: 10.1109/ASAP.2018.8445109.

5 Working Groups

WG1: Run-Time Requirement Monitoring and Enforcement

Coordinators: Felix C. Freiling, Daniel Müller-Gritschneder, Timo Hönig

The focus of this newly founded working group lies on the joint investigation of run-time measures for monitoring and enforcement of requirements on non-functional properties such as timing, power, temperature, reliability and security. The enforcement of such requirements is a central research challenge of the CRC with high practical relevance. In the domain of real-time systems, strict guarantees on the execution times of tasks are required. Designers often are forced to add overly pessimistic safety margins to assure these guarantees. These margins can be relaxed if the embedded system supports the monitoring and enforcement of these properties during run-time. Other examples, for which run-time monitoring and enforcement will aid in making better use of the system resources, include systems with reliability or security requirements. Additionally, also thermal management strategies, which gain increasing attention as we move in the age of dark silicon, require accurate monitoring data from the chip. Next to embedded computing, also applications in the high-performance computing (HPC) domain can profit from run-time requirement enforcement. HPC centres often need to satisfy certain power corridors, which were negotiated with the power grid providers. The enforcement of such corridors requires constant monitoring and run-time adjustments.

Activities

The working group just started its activities in the third funding phase. In the first meetings of the working group, different definitions and the terminology in the field of run-time monitoring and enforcement were discussed. Differences between terminologies in different communities were identified. Additionally, the participants identified the relevant research questions for the field of invasive computing such as the investigation of distributed vs. centralised monitoring and enforcement strategies, the implementation in an invasive MPSoC in hardware and/or software, the classes of requirements and properties to support monitoring and enforcement as well as the investigation of strategies for run-time requirement enforcement.

The working group organised a workshop on run-time verification on Sept. 26th, 2018 in Erlangen. During the workshop, three invited experts from the field—Felix Klaedtke from NEC Laboratories Europe, Heidelberg, as well as Srdan Krstic and Dmitriy Traytel from the Institute for Information Security of ETH Zürich—gave a tutorial on run-time verification theory and methods. In this course, they contrasted techniques developed independently in different areas of computer science (see Figure 5.1): *model checking*, *monitoring*, and *testing*. Overall, the tutorial formalised runtime monitoring as the task to "compute a verdict on a trace regarding a specification". The tutorial covered the associated questions: What is a trace? How is the system instrumented to get a trace? What is a specification for a property? How is the verdict computed? Solutions were shown with case studies and algorithms based on the formalisation of properties with Linear Temporal Logic, First-order Temporal Logic and Metric Temporal Logic.

Results

As an outcome of the activities during the early stage in the third funding phase, the working group contributed a glossary to the overall project. The glossary establishes a common definition of the core terminology for *run-time verification* and *run-time requirement enforcement*, with the latter being a shared building block and common theme for all projects of InvasIC in the current funding phase. More specifically, the core terminology for run-time requirement enforcement builds a common understanding on how to adapt methodologies of the different projects in a consistent manner.

Model Checking	**Monitoring**	**Testing**
in system model	**in** system trace	**in** real system
in specification	**in** specification	**in** test cases
out counterexample (if any)	**out** pass or fail	**out** pass or fail

Figure 5.1: Differences between model checking, monitoring and testing.

WG2: Memory Models, Architecture and Management

Coordinators: Lars Bauer, Andreas Herkersdorf, Wolfgang Schröder-Preikschat, Gregor Snelting

Memory access interference caused by concurrent applications and I/O processes execution may evolve to become the dominating performance (and predictability) bottleneck for manycore processor platforms. Hence, a holistic approach to memory organisation, considering the entire memory hierarchy from L1 caches over distributed tile-local SRAM memories to globally shared external SDRAM, has to involve multiple domains of expertise within the invasive computing paradigm. This includes programming models and language (i. e. the X10 memory model) implementation as well as the underlying middleware software, operating system, and hardware architecture of the processing platform.

Figure 5.2 depicts the spectrum of different intra- and inter-tile memory accesses within an invasive computing architecture. Hence, topics of this working group include questions on caches (invadeable caches, region-based vs. global coherence, cache management, non-cacheable address ranges), memory coherency, strong/weak memory consistency

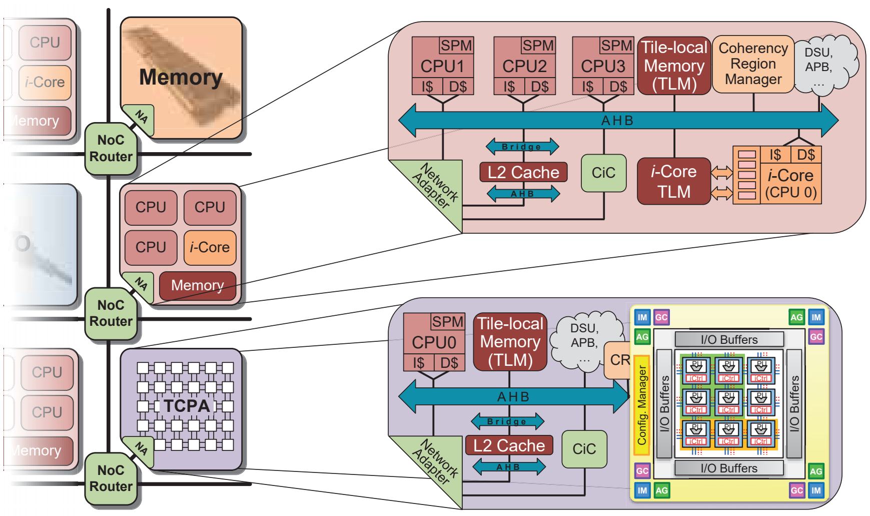


Figure 5.2: Spectrum of different intra- and inter-tile memory accesses within the invasive computing architecture

models, memory isolation, DMA via the *i*NoC, near-memory computing. Predictability of memory accesses is a high priority topic item on our agenda with mutual interest among several projects. It is to a high degree also interrelated to aspects of the high-capacity, low-latency, guaranteed services *i*NoC interconnect infrastructure.

Predictability is maximised if the memory model is formalised and provides provable guarantees. Based on the formalisation of the Java memory model[1] and the existing informal X10 memory model[2], it is thus planned to model the invasive memory model in the machine prover Isabelle, and provide proofs about coherence and consistency.

At the kick-off meeting in the year under review, the following activities were identified for the working group (with the relevant project names in brackets, from left to right in descending leadership): memory map for shared memory greater than 1 GB (C1, Z2), OctoPOS support for POSIX threads (D1, C1, B5), region-based cache coherence (RBCC) and TSO memory model (B5, C1, C3), cross-tile (remote) atomic machine instructions similar to *compare-and-swap*, *fetch-and-*Φ, *load-link/store-conditional*, and others (B5, C1, C3), virtual shared memory (C1, C5) and suitable new-memory support for it (B5, C1) as well as for X10 graph cloning (B5, C3) and for BLAS (B5, D3, A4), object-aware X10 garbage collection (C1, C3), software-defined hardware-managed queues (B5, C1), formal memory model for X10 (A1), intra-tile memory hierarchy (B1, Z2), design time characterisation of memory accesses (A4), and memory invasion (D3, C1).

This list of partially already started activities impressively illustrates the inter-disciplinary collaborations among different projects which are necessary to tackle the memory architecture and management challenges. Despite being early in Phase III, the collaboration already yielded a joint submission to a conference between Projects B5 and C1.

[1] A. Lochbihler. “A Machine-Checked, Type-Safe Model of Java Concurrency: Language, Virtual Machine, Memory Model, and Compiler”. Dissertation. Fakultät für Informatik, Karlsruher Institut für Technologie, Germany, 2012.

[2] A. Zwinkau. “A Memory Model for X10”. In: *Proceedings of the 6th ACM SIGPLAN Workshop on X10* (Santa Barbara, CA, USA). X10 2016. ACM, 2016, pp. 7–12. DOI: 10.1145/2931028.2931031.

WG3: Benchmarking and Evaluation

Coordinators: Michael Gerndt, Walter Stechele

The goal of the Working Group Benchmarking and Evaluation is to identify potential demonstration scenarios for invasive computing, with a focus on runtime enforcement of non-functional properties in the current third funding phase. We are considering project-specific results, as well as overall demonstration, on the three target platforms, i. e. FPGA-based MPSoC, commercial x64 multicore, and High Performance Computing systems. Application scenarios are based on the D projects on robotics and HPC.

A specific focus is on the exploitation of invasive computing methods after the end of the third funding phase. We plan to identify and collect success metrics for the exploitation of invasive methods in academia and industry, indicating the impact on the scientific community.

During the bi-annual meeting in October 2018, WG3 started with a collection of planned project-specific results, targeting basic mechanisms of invasion (project area A), invasive hardware extensions (project area B), the invasive run-time system (project area C), and invasive applications (project area D).

In project area A, there is a focus on worst-case guarantees under memory and communication restrictions, considering uncertain application variability. In project area B, the focus is on heterogeneous hardware and memory hierarchy, including instruction set extensions, two-dimensional processor arrays, near-memory accelerators, run-time monitoring and power management. The focus in project area C is on the invasive run-time system on multiple target platforms, including heterogeneous NoC-based MPSoC, commercial Xeon-Phi cluster, and HPC machines, considering automatic code generation and security challenges. Project area D has a focus on speculative resource planning, porting of robotic legacy code on invasive platforms, HPC programming abstractions and power corridor enforcement.

The initial collection of potential results from the individual project will be filtered and extended to get a better overview what technologies might be demonstrated at the end of the project. The next meetings will focus on gathering more information about these results and to discuss about potential demonstration scenarios. These scenarios might come from the application projects but also from all other projects in the research centre. To make the results accessible for all interested partners, the working group will maintain and continuously extend the list with more information during the course of the project.

WG4: Power and Thermal Aspects

Coordinators: Jörg Henkel, Nicole Megow, Stefan Wildermann

A limiting factor for high performance has been, is, and will be power consumption. There are various aspects for this limit:

1. The power density represents a physical limit as the amount of power that can be dissipated at a certain chip area is limited by the maximum temperature a circuit can stand without the risk of accelerated circuit degradation or even immediate irreversible damage.
2. The energy efficiency determines how much computation (or communication) can be accomplished with a certain amount of energy. Especially in energy-limited embedded applications it is the goal to make as much as possible use of a limited amount of energy.
3. Power and energy under real-time constraints: while reasons 1 and 2 are already hard to accomplish, the problem grows more complex when real-time comes into play. For example, if a real-time tasks needs to complete at a certain time, boosting might be a preferable means. That, however, will increase peak temperature and put the circuits under non-sustainable high stress. A multi-objective optimisation strategy may be needed. In general, a thorough investigation of the trade-offs is a primary goal.
4. Investigating how various scheduling and allocation algorithms match or can be adapted to invasive computer architectures in order to achieve a high efficiency.

InvasIC has various projects (among them Project A5, Project B3, Project C1) that focus on one or more aspects of power and energy with respect to aspects 1, 2, 3 and 4. The goal of WG4 is to bring these various goals under one umbrella in order to:

- Coordinate these various aspects such that in various phases during execution on an invasive multicore architecture, the applied power and energy means at different components (OS, architecture, application software, etc.) are targeted towards the same goals and contradictory control loops are avoided.
- Develop power and energy models that can be used by all projects that deal with the topic.
- Identify which aspects of power, energy and temperature analysis and modelling should be addressed at design time and which should be modelled as uncertainties when making scheduling and application mapping decisions (at run time). Of particular interest is to investigate in how far dynamic resource reservation by means of invasive computing and novel ideas about run-time requirement enforcement and run-time verification can help to reduce such uncertainties.

From an organisational point of view, it is the goal to target two working group meetings per year. It is also the goal to organise a session at an international event to present and discuss the very goals of this WG4 with international experts working on similar topics.

First Meeting The first meeting of the new working group WG4 Power and thermal aspects took place on December 6, 2018 in Erlangen. The goal of our meeting was to identify:

- Which tools (models, algorithms, foundations,...) for power, energy, temperature, timing analysis are required by projects in the CRC?
- Which tools (models, algorithms, foundations,...) for power, energy, temperature, timing analysis are currently developed in InvasIC?

We found the following topics which will be relevant for our next meetings:

- Thermal-aware application mapping
- High-performance computing: Relationship between job scheduling and run-time resource management
- Ageing monitoring (temperature, power, ...)

- Power models
- Application models and benchmarks (for evaluation)
- Machine learning (to learn application, power, temperature, etc. models)

Special Session Prof. Henkel organised the ICCAD Special Session "Managing Heterogeneous Many-cores for High-Performance and Energy-Efficiency". The special session considered a holistic approach to the broad topic of heterogeneous architectures. We presented our work on thermal and reliability awareness for executing applications with hard real-time requirements on manycore platforms [Hen+18]. Furthermore, it was possible to discuss the aspects tackled in our working group with various experts who joined the special session.

Publications

[Hen+18] J. Henkel, J. Teich, S. Wildermann, and H. Amrouch. "Dynamic Resource Management for Heterogeneous Many-Cores". In: *Proceedings of the International Conference on Computer-Aided Design (ICCAD)* (San Diego, CA, USA). ACM, Nov. 5–8, 2018, 60:1–60:6. DOI: 10.1145/3240765.3243471.

III

Events and Activities

Summary

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

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

In the following sections, we summarise the major events Project Z organised such as the DFG Review Meeting (Section 6), Internal Meetings (Section 7), Training Courses and Tutorials (Section 8) as well as further InvasIC Activities (Section 9) and Awards (Section 10). Last but not least, we present the current composition of the Industrial and Scientific Board in Section 11.

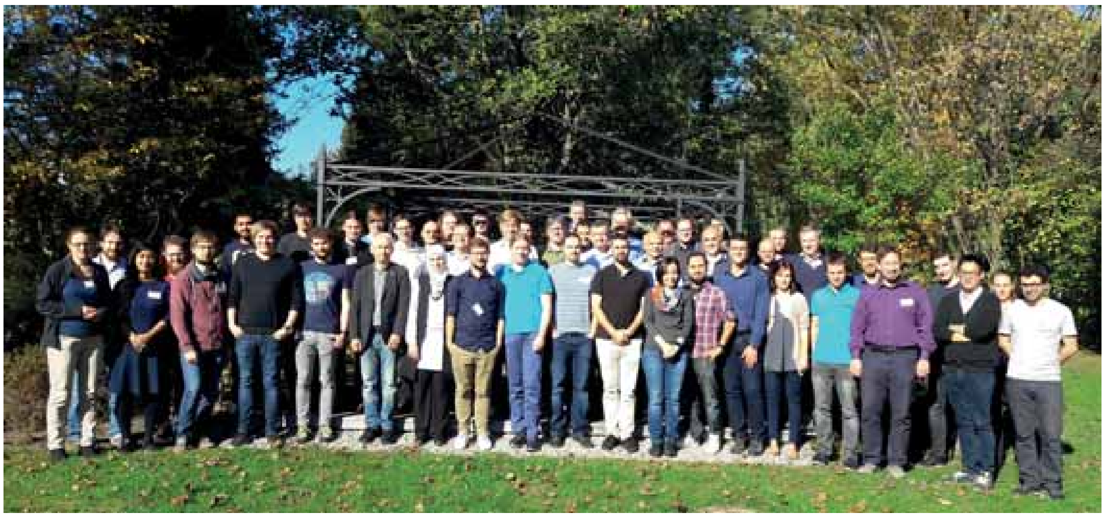


Figure 5.4: At the annual meeting in Adelsried, October 2018

6 DFG Review

In January 2018, the proposal of a third funding phase of our CRC/Transregio 89 was successfully reviewed by the German Research Foundation (DFG) and 10 peer experts.

Already during 2017, all members were involved in preparing the funding proposal and the review. At the end of 2017 and at the beginning of 2018, two rehearsals took place in Karlsruhe where researchers from all projects met to prepare for the DFG review: talks were given, posters presented and the demonstrators were set up and rehearsed.

Figure 6.1: At the DFG review meeting in Karlsruhe, January 2018

The review started in the morning of January 24 with a plenary meeting of the review panel. In the following session, the four Project Area Coordinators presented results in short talks and the doctoral researchers provided further information during a poster session. The doctoral researchers also introduced two demonstrators to the reviewers:
One platform was developed to demonstrate the predictability and compliance of non-functional program properties, in particular execution time, fault tolerance and security requirements through invasive computing using the example of a camera-based acquisition and control of an inverse pendulum on a corporately developed invasive multi-tile architecture. The second demonstrator served to show complex invasive applications in a continuous design flow. A shallow water equation

Figure 6.2: Doctoral researchers presenting one of the demonstrators at the DFG review meeting in Karlsruhe, January 2018

application was used as to illustrate the resource-aware computing from the language (ActorX10) to simulation-based DSE, compilers and *i*RTSS (OctoPOS, agents) through to execution on FPGA prototypes. The first day ended with a closed-to-public meeting of the review panel.

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

In May 2018, the Grants Committee finally decided to fund our CRC/-Transregio 89 for another term of 4 years!

7 Internal Meetings

Collaboration between the researchers of the three sites Karlsruhe, Munich, and Erlangen is essential for the success of the CRC/Transregio 89 – InvasIC. In 2018, researchers met at the following opportunities (list not being exhaustive):

Event	Date	
Rehearsal II	Jan. 12, 2018, Karlsruhe	In Karlsruhe, researchers from all projects met to prepare the DFG review meeting.
Review	Jan. 24/25, 2018, Karlsruhe	In January, the DFG Head Office, the review panel and researchers from all projects of InvasIC met for the DFG review meeting in Karlsruhe to present the results of the second funding phase as well as the research program of the third funding phase in plenary talks, posters and demonstrators.
WG Runtime Requirement Monitoring and Enforcement	Sept. 25/26, 2018, Erlangen	Members from the working group met in Erlangen to attend a tutorial by Felix Klaedtke (NEC), Srdan Krstic (ETHZ) and Dmitriy Traytel (ETHZ) on runtime enforcement.
Annual Meeting 2018	Oct. 11/12, 2018, Adelsried	At the annual meeting, the status quo of projects and the working groups were summarised.
Doctoral Researcher Retreat	Oct. 24–26, 2018, Adelsried	The doctoral researchers met in Adelsried to discuss progress and further challenges of the third funding phase.
WG Power and Thermal Aspects	Dec. 6, 2018, Erlangen	The goal of this meeting was to discuss and establish a common toolset to model, simulate, and measure power, temperature and energy in the projects.
WG Memory Models and Architectures	Dec. 13, 2018, Munich	The Members of WG2 met to have a personal exchange among memory-related issues within our CRC/Transregio.

8 Training Courses and Tutorials

The following internal workshops and training courses were organised by Project Z to give the doctoral researchers of InvasIC the opportunity to strengthen their soft skills, train their key qualifications, and improve their knowledge on topics related to invasive computing. Based on multiple requests and the wish of our doctoral researchers, InvasIC has organised workshops at all three sites of the Transregio on topics selected by the doctoral researchers.

Event	Date	
Workshop Publish or Perish	Aug. 23, 2018, Karlsruhe	The seminar was organised for young researchers to learn about bibliometric methods (Journal Impact Factor, Hirsch Index).
Workshop Research Data Management	Nov. 15/16, 2018, Munich	Members of InvasIC were invited to join a two-day workshop to gain new skills in research data management and learn how to manage their data.
Workshop Good Scientific Practice	Nov. 15/16, 2018, Erlangen	The workshop was organised with the objective of knowing and understanding the basic rules and values of the responsible conduct of research in all its stages, according to local, national and international regulations and guidelines.

Publish or Perish

A workshop named “Publish or Perish” was held by Dr. habil. Alexander Schiller. The seminar presented the most well-known bibliometric methods (Journal Impact Factor, Hirsch Index) and critically evaluated their meaningfulness. The participants learned to perform simple bibliometric analyses themselves and were given an insight into what they should pay attention for when writing a publication.

Research Data Management

Research data are an essential basis for scientific work. The long-term safeguarding and provision of research data contributes to the confirmability and quality of scientific work and opens up important opportunities for further research. It is therefore important to take a closer look at this aspect in research work. In a two-day workshop in

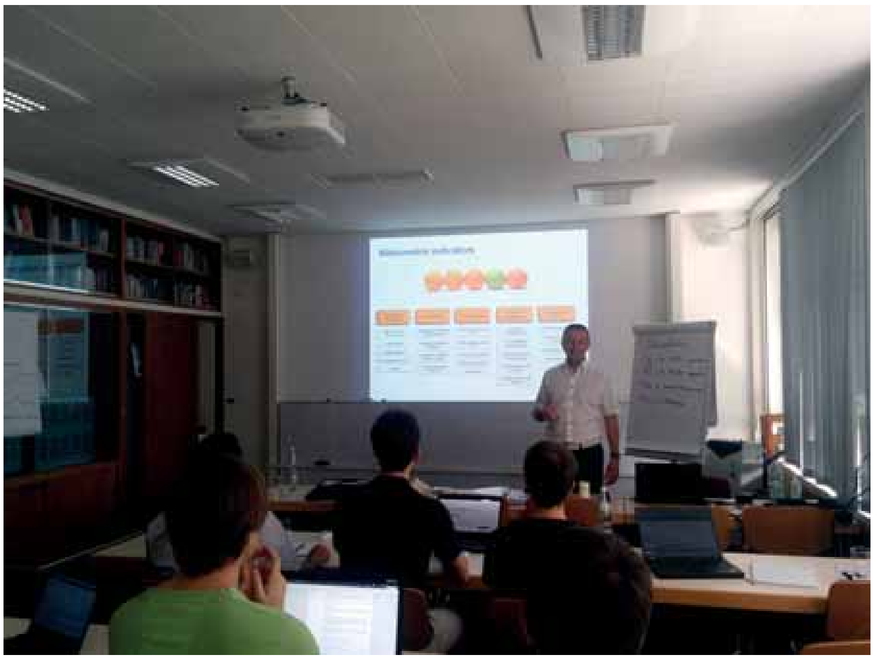


Figure 8.1: Publish or Perish, Karlsruhe, August 2018

cooperation with the university library at TUM, the participants received a first overview on research data management. Data documentation and organisation, long-term availability and archiving as well as publication options and selection of the archive were discussed. At the end of the workshop, the basics of creating a research data management plan were exemplified.

Figure 8.2: Research Data Management, Munich, October 2018

Good Scientific Practice

The third workshop organised in 2018 by Project Z on “Good Scientific Practice” aimed at getting to know and to understand the basic rules and values of the responsible conduct of research in all its stages, according to local, national and international regulations and guidelines. The participants explored the differences and grey areas between good scientific practice, questionable research practice and scientific misconduct. Furthermore, they learned how misconduct can be recognised and prevented, and how it should be addressed and dealt with in case it occurs, and what damage it can cause if handled improperly.

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". Additionally, members of InvasIC gave talks and seminars at important research sites and conferences ("Invited Talks") or organised conferences and workshops ("Organised Conferences and Workshops") on the topics of Invasive Computing.
The InvasIC Seminar is a series of talks given at one of the three sites ("InvasIC Seminar"). Videos of the respective talks are provided at our website `http://www.invasic.de`.

Figure 9.1: Prof. Christophe Bobda giving a talk at the InvasIC Seminar

Figure 9.2: Professor Shao-Yun Fan together with Professor Ulf Schlichtmann

InvasIC Seminar

Place and Date	Title	Speaker
Erlangen, Mar. 16, 2018	Self-Awareness for Heterogeneous MPSoCs: A Case Study using Adaptive, Reflective Middleware	Prof. Nikil Dutt (University of California, Irvine)
Munich, Mar. 26, 2018	Opportunities and Challenges of Silicon Photonics for Computing Systems	Prof. Jiang Xu (Hong Kong University of Science and Technology)
Erlangen, June 6, 2018	Hardware Isolation Framework for Security Mitigation in FPGA-Based Cloud Computing	Prof. Christophe Bobda (University of Arkansas)
Erlangen, June 15, 2018	Cross Media File Storage with Strata	Prof. Simon Peter (The University of Texas at Austin)
Erlangen, July 5, 2018	AnyDSL: A Partial Evaluation Framework for Programming High-Performance Libraries	Prof. Sebastian Hack (Universität des Saarlandes)

able Execution Platform:
Chip Based – NoC-HMP
eneous Multi-core Processor (NoC-HMP)
twork-on-chip interconnect

Figure 9.3: Prof. Zoran Salcic giving a talk at the InvasIC Seminar

Place and Date	Title	Speaker
Erlangen, July 12, 2018	Interference analysis with models for multi-core platform, AURIX TC27x - the case study	Wei-Tsun Sun PhD (IRT Antoine de Saint Exupéry)
Munich, July 17, 2018	Memristive Electronics	Prof. Sung-Mo (Steve) Kang (University of California, Santa Cruz)
Erlangen, July 17, 2018	Designing Static and Dynamic Software Systems - A SystemJ Perspective	Prof. Zoran Salcic (The University of Auckland)
Munich, Aug. 31, 2018	Routability Prediction in Early Placement Stages using Convolution Neural Networks	Assoc. Prof. Shao-Yun Fang (National Taiwan University of Science and Technology)
Munich, Sept. 20, 2018	Comparing Voltage Adaptation Performance between Replica and In-Situ Timing Monitors	Prof. Masanori Hashimoto (Osaka University)
Munich, Nov. 16, 2018	Algebraic Statistical Static Timing	Dr. Sani Nassif (Radyalis, USA)

iddle end:
ansformations /
ode „Optimization"
• Lower high-level (domain-specific) constructs with lower level constructs
• Pattern match constructs and rewrite them with oth
• Classical (scalar) opts
ck end:
ichine-specific
de generation
• Backend translates into HW language
17

Figure 9.4: Professor Sebastian Hack giving a talk at the InvasIC Seminar

Invited Talks

Place and Date	Title	Speaker
Pune, India, Jan. 6, 2018 31st International Conference on VLSI Design	Keynote: Power Density and Reliability in Embedded On-Chip Systems	Prof. J. Henkel (KIT)
Florianópolis, Brazil, Feb. 16, 2018 Software/Hardware Integration Lab (LISHA)	Talk: Predictability Issues in Operating Systems	Prof. W. Schröder-Preikschat (FAU)
Berkeley, USA Mar. 16, 2018 Lawrence Berkeley National Laboratory	Talk: Shallow Water on a Deep Technology Stack — Actor-Based Tsunami Simulation using Invasive Computing	A. Pöppl (TUM)
Shonan Village, Japan, Mar. 27, 2018 NII Shonan Meeting on Resilient Machine-to-Machine Communication	Talk: Predictability Issues in Operating Systems	Prof. W. Schröder-Preikschat (FAU)

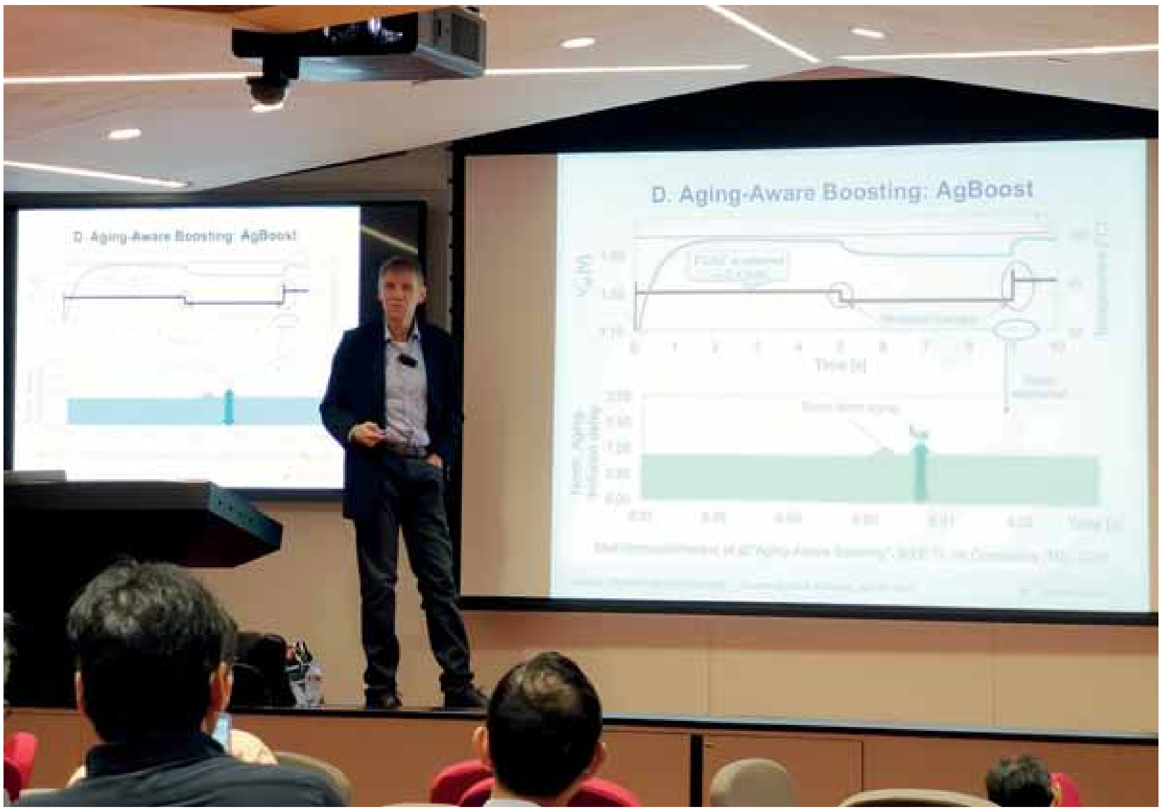


Figure 9.5: Prof. Jörg Henkel giving a keynote talk at the IEEE Computer Society Annual Symposium on VLSI, Hong Kong

Place and Date	Title	Speaker
Potsdam, Germany, Apr. 20, 2018 Hasso-Plattner-Institut (HPI)	Talk: Adaptive Memory Protection for Many-Core Systems	Prof. W. Schröder-Preikschat (FAU)
Amherst, USA, May 30, 2018 University of Massachusetts Amherst	Talk: Building a Runtime System for Heterogeneous HPC Clusters to Exploit Dynamic Electricity Pricing	T. Hönig (FAU)
Austin, USA, June 1, 2018 University of Texas at Austin	Talk: Linking Energy Awareness with Cost Effectiveness: Considering Fluctuating Electricity Prices for Operating Heterogeneous HPC Systems	T. Hönig (FAU)
Salt Lake City, USA, June 5, 2018 University of Utah	Talk: Energy-Aware System Software for Operating Heterogeneous HPC Systems in the Age of Dynamic Electricity Pricing	T. Hönig (FAU)
York, UK, June 14, 2018 Adaptive Many-Core Architectures and Systems Workshop	Keynote: Methodologies for Application Mapping for NoC-Based MPSOCs	Prof. J. Teich (FAU)
Nürnberg, Germany, June 28, 2018 Zollhof University Innovation Day	Talk: Die Multicore-Challenge meistern mit Invasive Computing	M. Brand (FAU)
Hannover, Germany, June 29, 2018 Leipniz Universität	Talk: Predictability Issues in Operating Systems	Prof. W. Schröder-Preikschat (FAU)
Hong Kong, China July 10, 2018 IEEE Computer Society Annual Symposium on VLSI	Keynote: Power Density and Circuit Aging – System-Level Means for Mitigation	Prof. J. Henkel (KIT)
Sydney, Australia, July 31, 2018 University of New South Wales (UNSW)	Talk: Hybrid Application Mapping for NoC-Based MPSoCs with Guarantees on Timing, Reliability and Security	Prof. J. Teich (FAU)
Curitiba, Brazil, Aug. 7, 2018 Technological Federal University of Paraná	Talk: Predictability Issues in Operating Systems	Prof. W. Schröder-Preikschat (FAU)

Place and Date	Title	Speaker
Singapore, Aug. 23, 2018 Nanyang Technological University (NTU)	Talk: Mixed Static/Dynamic Application Mapping for NoC-Based MPSoCs with Guarantees on Timing, Reliability and Security	Prof. J. Teich (FAU)
Singapore, Aug. 24, 2018 National University of Singapore (NUS)	Talk: Run-Time Application Mapping in Many-Core Architectures	Prof. J. Teich (FAU)
San Diego, USA Nov. 6, 2018 Special Session at the International Conference On Computer Aided Design (ICCAD)	Talk: Dynamic Resource Management for Heterogeneous Many-Cores	Prof. J. Teich (FAU) and Prof. J. Henkel (KIT)

Organised Conferences and Workshops

Dagstuhl Seminar 18101 “Scheduling”

Prof. Dr. Nicole Megow (UB), Prof. Dr. Magnús M. Halldórsson (Reykjavik University, IS) and Prof. Dr. Clifford Stein (Columbia University, US) organised and coordinated the Dagstuhl Seminar on “Scheduling” that took place March 4-9, 2018 at Dagstuhl Castle.
The Seminar on “Scheduling” brought together part of the community of algorithmic researchers who focus on scheduling, and part of the community of algorithmic researchers who focus on networking in general, and resource management within networks in particular. The primary objective of the seminar was to expose each community to the important models, problems and techniques from the other community, and to facilitate dialogue and collaboration between researchers. The program included 22 invited main talks including an inspiring talk on practical applications at ABB Corporate Research, 8 short spot-light talks, two open problem sessions in the beginning of the week, and ample unstructured time for research and interaction. The overall atmosphere among the 44 participants was very interactive.

© SCHLOSS DAGSTUHL – LZI GMBH
licensed under Creative Commons License CC BY-NC-ND

Figure 9.6: Participants of the Dagstuhl Seminar 18101

A highlight of the seminar was a joint Wednesday-session with the Dagstuhl Seminar 18102 “Dynamic Traffic Models in Transportation Science”. It was a fortunate coincidence that both seminars were scheduled in parallel. Indeed, questions related to networks, scheduling and resource sharing arise naturally in traffic control and transportation science. It was an inspiring secondary outcome of the workshop to realise this strong overlap in interests which led to interesting discussions between researchers of the different communities.

Special Session on “Managing Heterogeneous Many-Cores for High-Performance and Energy-Efficiency”

From November 5-8, 2018 the International Conference on Computer Aided Design (ICCAD) took place in San Diego, USA. Prof. Dr.-Ing. Jörg Henkel (KIT) organised a special session on “Managing Heterogeneous Many-Cores for High-Performance and Energy-Efficiency”. This special session presented the state-of-the-art in the form of three talks covering resource management, online learning and communication architectures for heterogeneous many-cores. Prof. Dr.-Ing. Jörg Henkel (KIT) and Prof. Dr.-Ing. Jürgen Teich (FAU) jointly presented a talk on joint research results on the topic of “Dynamic Resource Management for Heterogeneous Many-Cores”.

Public Relations

Film "Invasive Computing for Dummies"

"A picture is worth a thousand words", moving pictures might be worth even more. With this motivation, we developed a short introductory video to explain the basics of invasive computing to any human being. The video is available on our homepage `http://www.invasic.de`.

Figure 9.7: Introductory video on Invasive Computing available at `http://www.invasic.de`

Panel Discussion "Digitalisation - Chance or Risk?"

Prof. Dr.-Ing. Felix Freiling participated in a panel discussion on "Digitalisation - Chance or Risk?" on February 9, 2018 at the Foyercafe of the Markgrafentheater, Erlangen. The discussion captured the positive and negative side effects of digitalisation.

University Innovation Day

ZOLLHOF – Tech Incubator is a new home for tech startups and digital innovators located in Nuremberg. On June 28, 2018 the newest technologies, solutions and business cases that scientist have worked on so far were displayed at its University Innovation Day. Marcel Brand represented InvasIC with his talk on "Die Multicore-Challenge meistern mit Invasive Computing".

Day of the Technical Faculty

On November 16, the Day of the Technical Faculty 2018 of the Friedrich-Alexander-University in Erlangen took place. Selected research projects of the departments and projects of the PhD students (Dr. Hananeh Aliee) had the opportunity to present their work in a poster exhibition. InvasIC joined the day of the technical faculty to present its current research. Dr.-Ing. Timo Hönig, Dr.-Ing. J. Götzfried and Sebastian Maier presented a poster and a demonstrator explaining the idea of Invasive Computing. The very interested audience involved the presenting scientists in exciting discussions and conversations.

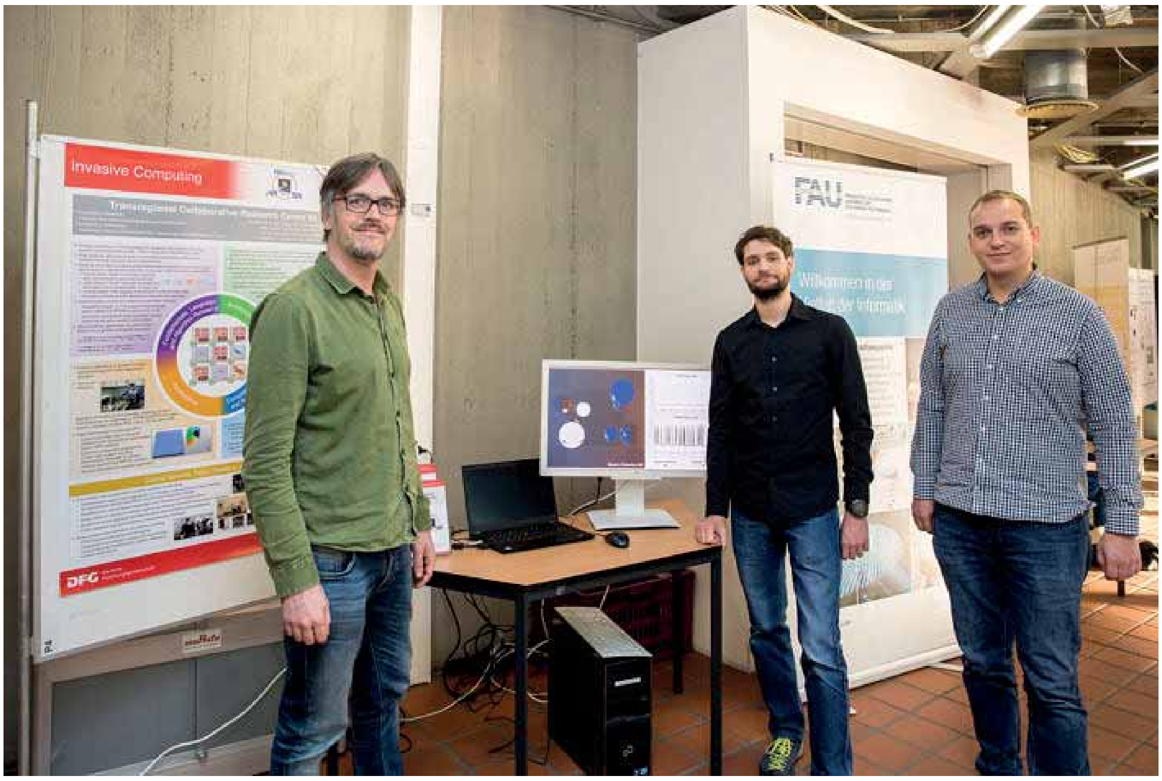


Figure 9.8: Dr.-Ing. Timo Hönig, Dr.-Ing. J. Götzfried and Sebastian Maier at the Day of the Technical Faculty

10 Awards

Fellow of the IEEE and Member of the National Academy of Science and Engineering

The IEEE Board of Directors, at its November 2017 meeting, elevated our coordinator, Prof. Dr.-Ing. Jürgen Teich (FAU) to IEEE Fellow, effective 1 January 2018, with the following citation: "for contributions to hardware/software co-design for embedded systems". IEEE Fellow is a distinction reserved for select IEEE members whose extraordinary accomplishments in any of the IEEE fields of interest are deemed fitting of this prestigious grade elevation.

Figure 10.1: Prof. Dr.-Ing. Jürgen Teich, Fellow of the IEEE

In addition, Ulf Schlichtmann and Jürgen Teich were elected member of the Deutsche Akademie der Technikwissenschaften (acatech) in 2018. The members of the Academy are outstanding scientists and scholars from the fields of engineering and the natural sciences, medicine as well as from the humanities and the social sciences and admitted on the basis of their scientific achievements. About 500 members of acatech currently work together in projects with experts from science and industry. They are also involved in thematic networks of the academy, discussing specific topics of technical sciences and overarching issues with technological political background.

CAST Award IT Security 2017 for Moritz Eckert

Moritz Eckert (FAU) ranked first with his Bachelor thesis in the CAST Förderpreis IT-Sicherheit 2017. Within his work "Cache-Timing Side-Channel Attacks against Intel's SGX", he implemented the prime practical cache attack against an AES implementation within a secure SGX-enclave.

Best Paper Award for Prof. Dr.-Ing. Felix Freiling

The paper "A Standardized Corpus for SQLite Database Forensics" by Sven Schmitt, Felix Freiling and Sebastian Nemetz received the Best Paper Award at this year's Digital Forensics Research Conference Europe (DFRWS EU 2018). The award was handed over during the conference which was held March 21-23, 2018 in Florence, Italy. The paper presents a standardised corpus of SQLite files that can be used to evaluate and benchmark analysis methods and tools.

Awarding Habilitation Certificate

Congratulations to Dr.-Ing. habil. Frank Hannig (FAU) for receiving his habilitation certificate for the field of "Domain-specific and Resource-aware Computing".

Figure 10.2: PD Dr.-Ing. habil. Frank Hannig

Awards for Doctoral Thesis of Dr.-Ing. Hananeh Aliee

Dr.-Ing. Hananeh Aliee won the doctoral award of the Women's Representative of FAU for her dissertation "Reliability Analysis and Optimization of Embedded Systems using Stochastic Logic and Importance Measure". The award was presented by Prof. Dr. Barbara Kappes on the occasion of the "Day of the Technical Faculty".

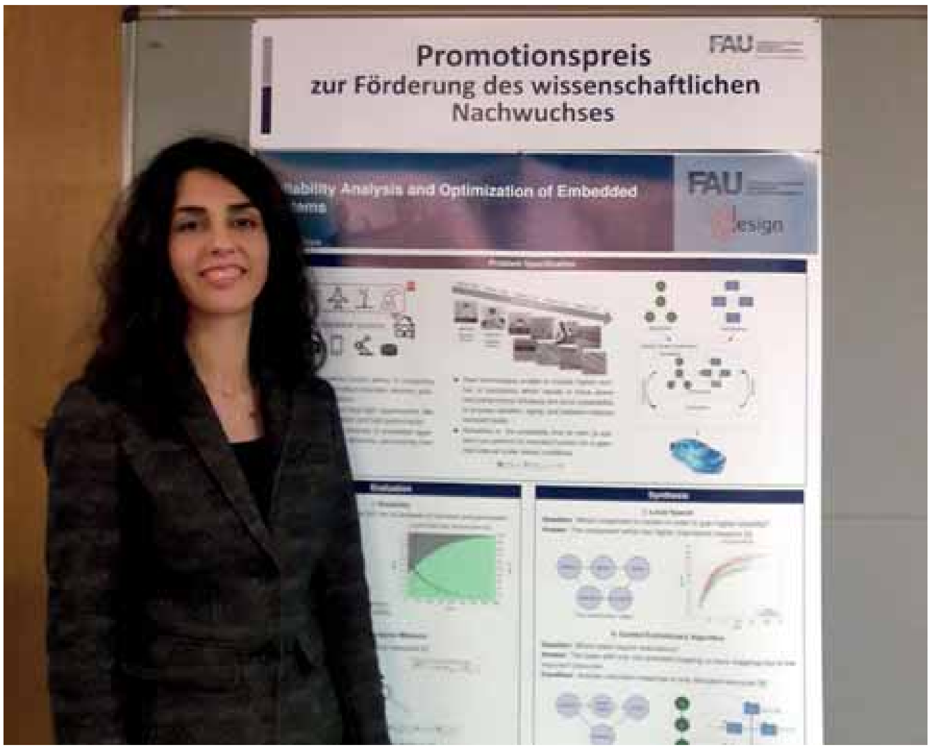

Promotionspreis
zur Förderung des wissenschaftlichen Nachwuchses
FAU
Reliability Analysis and Optimization of Embedded Systems

Figure 10.3: Dr.-Ing. Hananeh Aliee

Best Paper Award for Dr.-Ing. Johannes Götzfried

The paper "TEEshift: Protecting Code Confidentiality by Selectively Shifting Functions into TEEs" by Titouan Lazard, Johannes Götzfried, Tilo Müller, Gianni Santinelli, and Vincent Lefebvre received the Best Paper Award at this year's Workshop on System Software for Trusted Execution (SysTEX'18). The award was handed over during the workshop which was held on October 15, 2018 in Toronto, Canada. The paper presents a tool suite that protects the confidentiality and integrity of code by shifting selected functions into TEEs.

11 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-Hans Roth (IBM Böblingen)

Dr. Patricia Sagmeister (IBM Rüschlikon)

Intel

Hans-Christian Hoppe (Intel Director of ExaCluster Lab Jülich)

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)

BETTEN & RESCH Patent & Trademark Attorneys

Prof. Dr. Christoph von Praun
(European Trademark Attorney)

IAV – Automotive Engineering

Elmar Maas (IAV, Gifhorn)

Xilinx

Michaela Blott (Xilinx, Dublin)

12 Publications

[Bis+18] S. Bischof, J. Breitner, J. Graf, M. Hecker, M. Mohr, and G. Snelting. “Low-Deterministic Security For Low-Deterministic Programs”. In: *Journal of Computer Security* 26 (3 2018), pp. 335–366. DOI: 10.3233/JCS-17984.

[Bra+17a] M. Brand, F. Hannig, A. Tanase, and J. Teich. “Efficiency in ILP Processing by Using Orthogonality”. In: *Proceedings of the 28th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Seattle, WA, USA). IEEE, July 10–12, 2017, p. 207. DOI: 10.1109/ASAP.2017.7995282.

[Bra+17b] M. Brand, F. Hannig, A. Tanase, and J. Teich. “Orthogonal Instruction Processing: An Alternative to Lightweight VLIW Processors”. In: *Proceedings of the IEEE 11th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)* (Seoul, Republic of Korea). IEEE Computer Society, Sept. 18–20, 2017, pp. 5–12. DOI: 10.1109/MCSoC.2017.17.

[Bra+19] M. Brand, M. Witterauf, É. Sousa, A. Tanase, F. Hannig, and J. Teich. “*-Predictable MPSoC Execution of Real-Time Control Applications Using Invasive Computing”. In: *Concurrency and Computation: Practice and Experience* (Feb. 2019). DOI: 10.1002/cpe.5149.

[BFH18] S. Buchwald, A. Fried, and S. Hack. “Synthesizing an Instruction Selection Rule Library from Semantic Specifications”. In: *Proceedings of 2018 IEEE/ACM International Symposium on Code Generation and Optimization*. CGO ’18. New York, NY, USA: ACM, 2018. DOI: 10.1145/3168821.

[Cle+17] R. de Clercq, J. Götzfried, D. Übler, P. Maene, and I. Verbauwhede. “SOFIA: Software and Control Flow Integrity Architecture”. In: *Computers & Security* 68 (2017), pp. 16–35. DOI: 10.1016/j.cose.2017.03.013.

[CG17] I. A. Comprés Ureña and M. Gerndt. “Towards Elastic Resource Management”. In: *Proceedings of the 11th Parallel Tools Workshop*. Sept. 11–12, 2017.

[DBH17a] M. Damschen, L. Bauer, and J. Henkel. “CoRQ: Enabling Runtime Reconfiguration Under WCET Guarantees for Real-Time Systems”. In: *IEEE Embedded Systems Letters (ESL)* 9.3 (2017), pp. 77–80. DOI: 10.1109/LES.2017.2714844.

[DBH17b] M. Damschen, L. Bauer, and J. Henkel. “Timing Analysis of Tasks on Runtime Reconfigurable Processors”. In: *IEEE Transactions on Very Large Scale Integration Systems (TVLSI)* 25.1 (Jan. 2017), pp. 294–307. DOI: `10.1109/TVLSI.2016.2572304`.

[Eib+18] C. Eibel, C. Gulden, W. Schröder-Preikschat, and T. Distler. “Strome: Energy-Aware Data-Stream Processing”. In: *Proceedings of the 18th International Conference on Distributed Applications and Interoperable Systems (DAIS ’18)* (Madrid, Spain). Lecture Notes in Computer Science (LNCS). Springer, June 2018, pp. 40–57.

[Eic+18] C. Eichler, T. Distler, P. Ulbrich, P. Wägemann, and W. Schröder-Preikschat. “TASKers: A Whole-System Generator for Benchmarking Real-Time-System Analyses”. In: *Proceedings of the 18th International Workshop on Worst-Case Execution Time Analysis (WCET 2018)* (Barcelona, Spain, July 3–6, 2018). July 3, 2018, 6:1–6:12. DOI: `10.4230/OASIcs.WCET.2018.6`.

[Fri17] S. Friederich. “Automated Hardware Prototyping for 3D Networks on Chips”. Dissertation. Institut für Technik der Informationsverarbeitung, Karlsruhe Institute of Technology (KIT), May 23, 2017.

[Har+17] T. Harbaum et al. “Auto-SI: An Adaptive Reconfigurable Processor with Run-time Loop Detection and Acceleration”. In: *30th IEEE International System-on-Chip Conference (SOCC)*. IEEE, Sept. 2017, pp. 224–229. DOI: `10.1109/SOCC.2017.8226027`.

[Hen+18] J. Henkel, J. Teich, S. Wildermann, and H. Amrouch. “Dynamic Resource Management for Heterogeneous Many-Cores”. In: *Proceedings of the International Conference on Computer-Aided Design (ICCAD)* (San Diego, CA, USA). ACM, Nov. 5–8, 2018, 60:1–60:6. DOI: `10.1145/3240765.3243471`.

[Her+18] B. Herzog, L. Gerhorst, B. Heinloth, S. Reif, T. Hönig, and W. Schröder-Preikschat. “INTSPECT: Interrupt Latencies in the Linux Kernel”. In: *Proceedings of the 2018 Brazilian Symposium on Computing Systems Engineering (SBESC ’18)* (Salvador, Brazil, Nov. 6–9, 2018). Nov. 2018, pp. 1–8.

[Hön+18a] T. Hönig, C. Eibel, A. Wagenhäuser, M. Wagner, and W. Schröder-Preikschat. “How to Make Profit: Exploiting Fluctuating Electricity Prices with Albatross, A Runtime System for Heterogeneous HPC Clusters”. In: *Proceedings of the 8th International Workshop on Runtime and Operating Systems for Supercomputers (ROSS 2018)* (Tempe, AZ, USA). ACM, June 12, 2018, Article No. 4. DOI: `10.1145/3217189.3217193`.

[Hön+18b] T. Hönig, C. Eibel, A. Wagenhäuser, M. Wagner, and W. Schröder-Preikschat. “Making Profit with Albatross: A Runtime System for Heterogeneous High-Performance-Computing Clusters”. In: *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing (HPDC’18)* (Tempe, AZ, USA). Poster session. ACM, June 13, 2018, pp. 11–12. DOI: `10.1145/3220192.3220457`.

[Kar+18] R. Kartmann, F. Paus, M. Grotz, and T. Asfour. “Extraction of Physically Plausible Support Relations to Predict and Validate Manipulation Action Effects”. In: *IEEE Robotics and Automation Letters (RA-L)* 3.4 (Oct. 2018), pp. 3991–3998. DOI: `10.1109/LRA.2018.2859448`.

[Khd18] H. Khdr. “Resource Management for Multicores to Optimize Performance under Temperature and Aging Constraints”. Dissertation. Chair of Embedded Systems, Department of Informatics, Karlsruhe Institute of Technology, Germany, 2018.

[KAH18a] H. Khdr, H. Amrouch, and J. Henkel. “Dynamic Guardband Selection: Thermal-Aware Optimization for Unreliable Multi-Core Systems”. In: *Transactions on Computers (TC)* (2018).

[KAH18b] H. Khdr, H. Amrouch, and J. Henkel. “Aging-Aware Boosting”. In: *IEEE Transactions on Computers (TC)* (2018). DOI: `10.1109/TC.2018.2816014`.

[KAH18c] H. Khdr, H. Amrouch, and J. Henkel. “Aging-Constrained Performance Optimization for Multi Cores”. In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[Khd+18] H. Khdr, S. Pagani, M. Shafique, and J. Henkel. “Dark Silicon Aware Resource Management for Many-Core Systems”. In: *Advances in Computers: Dark Silicon and Future of On-chip Systems*. Elsevier, 2018.

[Laz+18] T. Lazard, J. Götzfried, T. Müller, G. Santinelli, and V. Lefebvre. “TEEshift: Protecting Code Confidentiality by Selectively Shifting Functions into TEEs”. In: *Proceedings of the 3rd Workshop on System Software for Trusted Execution*. SysTEX ’18. Toronto, Canada: ACM, 2018, pp. 14–19. DOI: `10.1145/3268935.3268938`. URL: `https://www1.cs.fau.de/teeshift`.

[Lis+18] A. Listl, D. Mueller-Gritschneder, F. Kluge, and U. Schlichtmann. “Emulation of an ASIC Power, Temperature and Aging Monitor System for FPGA Prototyping”. In: *International On-Line Testing Symposium (IOLTS)*. July 2018.

[Lis+19] A. Listl, D. Mueller-Gritschneder, S. R. Nassif, and U. Schlichtmann. “SRAM Design Exploration with Integrated Application-Aware Aging Analysis”. In: *Proceedings of Design, Automation and Test in Europe Conference (DATE), March 2019*. accepted for publication. 2019.

[Mae+18a] P. Maene, J. Götzfried, R. de Clercq, T. Müller, F. Freiling, and I. Verbauwhede. “Hardware-Based Trusted Computing Architectures for Isolation and Attestation”. In: *IEEE Transactions on Computers* 67.3 (2018), pp. 361–374. DOI: 10.1109/TC.2017.2647955.

[Mae+18b] P. Maene, J. Götzfried, T. Müller, R. de Clercq, F. Freiling, and I. Verbauwhede. “Atlas: Application Confidentiality in Compromised Embedded Systems”. In: *IEEE Transactions on Dependable and Secure Computing* (2018). DOI: 10.1109/TDSC.2018.2858257.

[Mas+18] L. Masing, A. Srivatsa, F. Kreß, N. Anantharajaiah, A. Herkersdorf, and J. Becker. “In-NoC Circuits for Low-Latency Cache Coherence in Distributed Shared-Memory Architectures”. In: *2018 IEEE 12th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)*. IEEE, Sept. 2018. DOI: 10.1109/mcsoc2018.2018.00033.

[MTT18] T. Mitra, J. Teich, and L. Thiele. “Guest Editors’ Introduction: Special Issue on Time-Critical Systems Design”. In: *IEEE Design and Test of Computers* 35 (2018), pp. 5–7. DOI: 10.1109/MDAT.2018.2796037.

[Pag+18a] S. Pagani, J.-J. Chen, M. Shafique, and J. Henkel. *Advanced Techniques for Power, Energy, and Thermal Management for Clustered Manycores*. Springer, 2018.

[Pag+18b] S. Pagani, S. M. PD, A. Jantsch, and J. Henkel. “Machine Learning for Power, Energy, and Thermal Management on Multi-core Processors: A Survey”. In: *Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)* (2018).

[Pat18] A. Pathania. “Scalable Task Schedulers for Many-Core Architectures”. Dissertation. Chair of Embedded Systems, Department of Informatics, Karlsruhe Institute of Technology, Germany, 2018.

[PH18a] A. Pathania and J. Henkel. “HotSniper: Sniper-Based Toolchain for Many-Core Thermal Simulations in Open Systems”. In: *Embedded Systems Letters (ESL)* (2018).

[PH18b] A. Pathania and J. Henkel. “Task scheduling for many-cores with S-NUCA caches”. In: *Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE. 2018, pp. 557–562.

[Pat+18] A. Pathania, H. Khdr, M. Shafique, T. Mitra, and J. Henkel. "QoS-Aware Stochastic Power Management for Many-Cores". In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[Pöp+17] A. Pöppl et al. "Shallow Water Waves on a Deep Technology Stack: Accelerating a Finite Volume Tsunami Model Using Reconfigurable Hardware in Invasive Computing". In: *Workshop on UnConventional High Performance Computing (UCHPC), Santiago de Compostela, Spain, August 28-29, 2017, Revised Selected Papers*. 2017, pp. 676–687. DOI: 10.1007/978-3-319-75178-8_54.

[Pöp+18] A. Pöppl et al. "Shallow Water Waves on a Deep Technology Stack: Accelerating a Finite Volume Tsunami Model using Reconfigurable Hardware in Invasive Computing". In: *Euro-Par 2017: Proceedings of the 10th Workshop on UnConventional High Performance Computing (UCHPC 2017)*. Ed. by D. B. Heras et al. Lecture Notes in Computer Science (LNCS). Santiago de Compostela, Spain: Springer International Publishing, 2018, pp. 676–687.

[Pou+17] B. Pourmohseni, S. Wildermann, M. Glaß, and J. Teich. "Predictable Run-Time Mapping Reconfiguration for Real-Time Applications on Many-Core Systems". In: *Proceedings of the International Conference on Real-Time Networks and Systems (RTNS)*. Outstanding paper award. Grenoble, France: IEEE, 2017. DOI: 10.1145/3139258.3139278.

[Qia+18] B. Qiao, O. Reiche, F. Hannig, and J. Teich. "Automatic Kernel Fusion for Image Processing DSLs". In: *Proceedings of the 21st International Workshop on Software and Compilers for Embedded Systems (SCOPES)* (St. Goar, Germany). ACM, May 28–30, 2018, pp. 76–85. DOI: 10.1145/3207719.3207723.

[RPH18] M. Rapp, A. Pathania, and J. Henkel. "Pareto-Optimal Power- and Cache-Aware Task Mapping for Many-Cores with Distributed Shared Last-Level Cache". In: *International Symposium on Low Power Electronics and Design (ISLPED)*. IEEE. 2018.

[Rei18] O. Reiche. "A Domain-Specific Language Approach for Designing and Programming Heterogeneous Image Systems". Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 5, 2018.

[RS18a] S. Reif and W. Schröder-Preikschat. "A Predictable Synchronisation Algorithm". In: *Proceedings of the 23rd Annual Symposium on Principles and Practice of Parallel Programming (PPoPP '18)*

(Vienna, Austria, Feb. 24–28, 2018). Poster session. ACM, Feb. 2018, pp. 415–416. DOI: 10.1145/3178487.3178533.

[RS18b] S. Reif and W. Schröder-Preikschat. *Predictable Synchronisation Algorithms for Asynchronous Critical Sections*. Tech. rep. CS-2018-03. Erlangen, Germany: Friedrich-Alexander-Universität Erlangen-Nürnberg, Department Informatik, Mar. 2018. DOI: 10.25593/issn.2191-5008/CS-2018-03.

[Rhe+18] S. Rheindt, A. Schenk, A. Srivatsa, T. Wild, and A. Herkersdorf. "CaCAO: Complex and Compositional Atomic Operations for NoC-based Manycore Platforms". In: *ARCS 2018 - 31st International Conference on Architecture of Computing Systems*. Braunschweig, Germany, 2018.

[Ric+18] V. Richthammer, T. Schwarzer, S. Wildermann, J. Teich, and M. Glaß. "Architecture Decomposition in System Synthesis of Heterogeneous Many-Core Systems". In: *55th ACM/EDAC/IEEE Design Automation Conference (DAC 2018)* (San Francisco, CA). June 24–28, 2018.

[Rol18] S. Roloff. "Modeling and Simulation of Invasive Applications and Architectures". Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 19, 2018.

[Ros+18] E. Rossi, M. Damschen, L. Bauer, G. Buttazzo, and J. Henkel. "Preemption of the Partial Reconfiguration Process to Enable Real-Time Computing with FPGAs". In: *ACM Trans. on Reconfig. Technol. and Syst. (TRETS)* 11.2 (July 2018), 10:1–10:24. DOI: 10.1145/3182183.

[SHT18] C. Schmitt, F. Hannig, and J. Teich. "A Target Platform Description Language for Parallel Code Generation". In: *Workshop Proceedings of the 31st GI/ITG International Conference on Architecture of Computing Systems (ARCS)* (Braunschweig). Berlin: VDE VERLAG GmbH, Apr. 9–12, 2018, pp. 59–66.

[Sch+18a] T. Schwarzer, A. Weichslgartner, M. Glaß, S. Wildermann, P. Brand, and J. Teich. "Symmetry-eliminating Design Space Exploration for Hybrid Application Mapping on Many-Core Architectures". In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 37.2 (Feb. 2018), pp. 297–310. DOI: 10.1109/TCAD.2017.2695894.

[Sch+18b] T. Schwarzer et al. "On the Complexity of Mapping Feasibility in Many-Core Architectures". In: *Proceedings of Multicore/Many-core Systems-on-Chip (MCSoC-2018)*. Sept. 12–14, 2018.

[Sot+18] E. Sotiriou-Xanthopoulos, L. Masing, S. Xydis, K. Siozios, J. Becker, and D. Soudris. “OpenCL-based Virtual Prototyping and Simulation of Many-Accelerator Architectures”. In: *ACM Trans. Embed. Comput. Syst.* 17.5 (Sept. 2018), 86:1–86:27. DOI: `10.1145/3242179`.

[Sou18] É. Sousa. “Memory and Interface Architectures for Invasive Tightly Coupled Processor Arrays”. Dissertation. Hardware/Software Co-Design, Department of Computer Science, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany, July 20, 2018.

[Sou+18] É. Sousa, M. Witterauf, M. Brand, A. Tanase, F. Hannig, and J. Teich. “Invasive Computing for Predictability of Multiple Non-functional Properties: A Cyber-Physical System Case Study”. In: *Proceedings of the 29th Annual IEEE International Conference on Application-specific Systems, Architectures and Processors (ASAP)* (Milan, Italy). IEEE, July 10–12, 2018. DOI: `10.1109/ASAP.2018.8445109`.

[Sou+17] É. Sousa, A. Tanase, F. Hannig, and J. Teich. “A Reconfigurable Memory Architecture for System Integration of Coarse-Grained Reconfigurable Arrays”. In: *Proceedings of the International Conference on Reconfigurable Computing and FPGAs (ReConFig)* (Cancun, Mexico). IEEE, Dec. 4–6, 2017. DOI: `10.1109/RECONFIG.2017.8279768`.

[Sri+17] A. Srivatsa, S. Rheindt, T. Wild, and A. Herkersdorf. “Region Based Cache Coherence for Tiled MPSoCs”. In: *2017 30th IEEE International System-on-Chip Conference (SOCC)*. Sept. 2017.

[THT18] A. Tanase, F. Hannig, and J. Teich. *Symbolic Parallelization of Nested Loop Programs*. ISBN: 978-3-319-73908-3. Springer, Feb. 2018.

[Tei17] J. Teich. *Run-Time Monitoring and Enforcement of Non-functional Program Properties of Invasive Programs: Terms and Definitions*. Technical Report 01-2017. Erlangen, Germany: Hardware/-Software Co-Design, Friedrich-Alexander-Universität Erlangen-Nürnberg, Department of Computer Science, Jan. 2017.

[Tei18a] J. Teich. “Hybrid Application Mapping for NoC-Based MPSoCs with Guarantees on Timing, Reliability and Security”. Invited Talk University of New South Wales, Australia. July 31, 2018.

[Tei18b] J. Teich. “Methodologies for Application Mapping for NoC-Based MPSOCs”. Keynote, Adaptive Many-Core Architectures and Systems workshop, York, UK. June 14, 2018.

[Tei18c] J. Teich. “Run-Time Application Mapping in Many-Core Architectures”. Invited Talk National University of Singapore. Aug. 24, 2018.

[Uph+17] C. Uphoff et al. “Extreme Scale Multi-Physics Simulations of the Tsunamigenic 2004 Sumatra Megathrust Earthquake”. In: *SC17: The International Conference for High Performance Computing, Networking, Storage and Analysis Proceedings*. ACM. 2017. DOI: 10.1145/3126908.3126948.

[Wäg+18] P. Wägemann, C. Dietrich, T. Distler, P. Ulbrich, and W. Schröder-Preikschat. “Whole-System Worst-Case Energy-Consumption Analysis for Energy-Constrained Real-Time Systems”. In: *Proceedings of the 30th Euromicro Conference on Real-Time Systems (ECRTS ’18)* (Barcelona, Spain, July 3–6, 2018). July 2018, 24:1–24:25. DOI: 10.4230/LIPIcs.ECRTS.2018.24.

[Wei+18a] A. Weichslgartner, S. Wildermann, D. Gangadharan, M. Glaß, and J. Teich. “A Design-Time/Run-Time Application Mapping Methodology for Predictable Execution Time in MPSoCs”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 17.5 (Nov. 2018), 89:1–89:25. DOI: 10.1145/3274665.

[Wei+18b] A. Weichslgartner, S. Wildermann, M. Glaß, and J. Teich. *Invasive Computing for Mapping Parallel Programs to Many-Core Architectures*. Springer, Jan. 15, 2018. DOI: 10.1007/978-981-10-7356-4.

[Zai+17] A. Zaib et al. “Efficient Task Spawning for Shared Memory and Message Passing in Many-core Architectures”. In: *Journal of Systems Architecture (JSA)* (2017). DOI: 10.1016/j.sysarc.2017.03.004.

[Zai18] M. A. Zaib. “Network on Chip Interface for Scalable Distributed Shared Memory Architectures”. Dissertation. München: Technische Universität München, 2018.

[Zha+18a] G. L. Zhang, B. Li, J. Liu, Y. Shi, and U. Schlichtmann. “Design-Phase Buffer Allocation for Post-Silicon Clock Binning by Iterative Learning”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*. Vol. 37. 2. 2018.

[Zha+18b] G. L. Zhang, B. Li, Y. Shi, J. Hu, and U. Schlichtmann. “EffiTest2: Efficient Delay Test and Prediction for Post-Silicon Clock Skew Configuration under Process Variations”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* (2018). Early Access Paper, doi: 10.1109/TCAD.2018.2818713.