Our Perspectives

Jian-Jia Chen and Joerg Henkel

Research and development in the last decades have led to a silicon process that has been expected to become inherently undependable in the near future when migrating towards new technologies. The special priority program (SPP) 1500 funded by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) in 2010–2016 and the Variability Expedition funded by the National Science Foundation (NSF) in 2010–2015 made a joint effort to explore design challenges of *Power Consumption, Reliability, Interference, and Manufacturability* under such a design requirement.

The exploration started with a vision to go beyond simply developing fault-tolerant systems that monitor the device at run-time and react to error detection. Instead, the design should consider error as a design constraint and develop methodologies to achieve *resilience* at the presence of errors. Under such a design principle, error is inevitable and the error rate should be a tradeoff against performance.

This book summarizes the achievements of the SPP 1500 partners, the Variability Expedition partners, and their collaborators. After telling the successful stories in the previous chapters, this chapter provides a summary of our perspectives of the exploration and a short outlook of future.

One important perspective to achieve resilience at the presence of faults is to *quantitatively* define resilience and errors and use the resilience in a *cross-layer* manner. Specifically, the RAP model summarized in chapter “RAP Model–Enabling Cross-Layer Analysis and Optimization for System-on-Chip Resilience” provides a milestone to help annotate how variability related to physical faults can be expressed

J.-J. Chen
TU Dortmund, Dortmund, Germany
e-mail: jian-jia.chen@tu-dortmund.de

J. Henkel (✉)
Karlsruhe Institute of Technology, Karlsruhe, Germany
e-mail: henkel@kit.edu

at higher abstraction levels. RAP is a result of several working group meetings and collaborative efforts among SPP 1500 partners. It has been also used as a demonstrator in several projects. We believe that RAP is an initial step towards good abstractions that can be used to model the faulty hardware and its impact on the software. It may be possible that the probabilistic information encoded in RAP is not precise enough for further optimizations. We envision that a more flexible and more precise model to correctly quantify the resilience will be needed in the future. It may be a set of models that can be configured depending on the required accuracy level.

One possible way to analyze system-level resilience in a modularized manner is to enable compositional reliability analysis. The support of composition and decomposition is important for modularized analysis and can be used to model uncertainties in both functional and non-functional properties. The modularity provided in chapter “EM Lifetime Constrained Optimization for Multi-Segment Power Grid Networks” extends the existing compositional performance analysis (CPA) (or real-time calculus) to handle reliability. We believe that there is a great potential to utilize the concept in the system design. However, to achieve composition and decomposition, rules to bound the approximation errors of composition and/or decomposition would be needed. The automatic design of efficient and effective rules is essential for compositional reliability analysis.

In many of the results of the SPP 1500 and Variability Expedition partners, cross-layer and interactive optimization has been explored. Unlike the classic multi-layered approach, in which each layer *passively* takes the input from the higher/lower layers, the cross-layer approach applies *active* optimization routines across multiple layers. Since the system-level resilience cannot be optimized unless all the layers are optimized, such a cross-layer approach has been used as interfaces between different layers. An overview of the (coarse-grained) layers and their interactions can be found in Fig. 10 in chapter “RAP Model-Enabling Cross-Layer Analysis and Optimization for System-on-Chip Resilience”.

To validate the research results, *fault injection* through instrumentation, emulation, and simulation has been developed and used. Fault injection is an important routine that should be deployed before fault detection. The computational effort of fault injection can become a bottleneck. Proper models and tools for fault injections are important contributions of the research partners. For example, the FPGA fault injection tool in chapter “Dependability Aspects in Configurable Embedded Operating Systems” can be used to emulate the entire SoC with specific faults. Different fault injection scenarios can be found in chapter “Lightweight Software-Defined Error Correction for Memories”. Despite its importance, to the best of our knowledge, there is no integrated tool that can be used for benchmarking the quality and (intended) consequence of fault injection. Although there have been several attempts to provide an integrated tool from the partners in SPP1500 for different types of fault injection, the diverse scenarios in the cross-layer settings made the integration very difficult. We envision that fault injection tools that can be configured and applied for different layers and scenarios can be developed in the near future so that cross-layer design and optimization can be further modularized and deployed.

When the design considers error as a design constraint, the system has to be *adaptive* to react (or even be proactive) according to the faults and errors to achieve the targeted resilience. Adaptive methods in physical, micro-architecture, architecture (ISA), compiler, and operating systems are explored and discussed. It has been demonstrated in several research results that adaptivity should be applied across layers. For example, the error semantics in chapter “Soft Error Handling for Embedded Systems using Compiler-OS Interaction” in the software development process provides the information in the compilation needed for the operating systems to be adaptive according to faults. Moreover, the annotation of multiple execution versions in chapter “Cross-Layer Dependability: From Architecture to Software and Operating System” provides a means to the run-time system to execute different versions according to the reliability condition. Furthermore, the dependability aspects can be further configured in operating systems as demonstrated in chapter “ASTEROID and the Replica-Aware Co-scheduling for Mixed-Criticality”. We strongly believe that adaptivity is a key insight. However, the reported achievement is based on ad hoc treatments for well-defined scenarios. It will be very practical and impactful to explore automatic adaptivity so that suggestions of proper means can be provided to the designers for achieving high resilience.

The adaptive handling of errors and faults naturally makes the timing behavior dynamic over time. When there is no fault, an embedded system functions correctly with respect to the specified timing. However, when there are faults, the embedded system may not function correctly anymore since some jobs may be aborted or may miss their deadlines. Therefore, it is of importance to explore both functional and timing correctness. If all jobs have to meet their deadlines, the hardware may have to be over-dimensioned. If some jobs can be allowed to miss their deadlines when faults are present, the system designer just has to ensure that all the desired timing behavior can be verified offline. Such dynamic timing requirements can be modeled as mixed criticality. When the system does not suffer from any fault, it is at the low-criticality mode. When the system suffers from some faults, it is promoted to the high-critical mode. Such a treatment has been presented in chapter “Dependability Aspects in Configurable Embedded Operating Systems”. An alternative is to explore the probability (or miss rate) of deadline misses, presented in chapter “Cross-Layer Dependability: From Architecture to Software and Operating System”. Although the above treatments are successful, they are not originally from the resilience perspectives. It remains open whether the timing requirements to achieve system resilience should be treated as the first-class design objective. More specifically, although dynamic timing behavior and requirements are considered, they are not directly related to resilience. Moreover, the tradeoffs of the timing requirement and system resilience in the presence of faults are still in the infant stage and require more research efforts to reach a conclusion.

We believe that the successful stories in the previous chapters and the perspectives presented in this chapter provide cornerstones for the design of dependable systems on unreliable hardware. Based on the foundation established by the partners, designs which consider faults/errors as a design constraint will be continued in different directions, including physical, micro-architecture, architecture (ISA), compiler, and operating systems layers, and, *most importantly*, in a **cross-layer** manner.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.