Resilience by Codesign (and not as an Afterthought)

Christian Engelmann: engelmannc@ornl.gov
Oak Ridge National Laboratory, Oak Ridge, TN, USA

Topics: Modeling and simulation, codesign methodologies

Challenge: Resilience, i.e., obtaining a correct solution in a timely and efficient manner, is one of the key challenges in extreme-scale high-performance computing (HPC). Extreme heterogeneity, i.e., using multiple, and potentially configurable, types of processors, accelerators and memory/storage in a single computing platform, will add a significant amount of complexity to the HPC hardware/software ecosystem.

The notion of correct computation and program state assumed by users and application developers today, which has been based on binary bit-level correctness, will no longer hold for processing elements based on some emerging technologies, such as neuromorphic computing elements. The diverse set of compute and memory components in future heterogeneous systems will require novel hardware and software resilience solutions. Errors and failures reported by heterogeneous hardware will need to be handled by the appropriate software component to enable efficient masking, recovery, and avoidance with little burden on the user. Similarly, errors and failures reported by the software running on heterogeneous hardware need to be equally efficiently handled with little burden on the user.

This requires a new approach, where resilience is holistically provided by the HPC hardware/software ecosystem. The key challenge is to codesign extreme heterogeneous HPC systems with (1) wide-ranging resilience capabilities in architecture, system software, programming models, libraries, and applications, (2) interfaces and mechanisms for coordinating resilience capabilities across diverse hardware and software components, (3) appropriate metrics and tools for assessing resilience, and (4) an understanding of the performance, resilience and energy trade-off that eventually results in well-informed system design choices.

The current state of practice for HPC resilience is global application-level checkpoint/restart, a single-layer approach that burdens the user with employing a resilience strategy at extreme coarse granularity (the job level). Part of the current state of practice for HPC resilience are also hardware solutions at extreme fine granularity, such as SECDED ECC for main memory, caches, registers and architectural state, Chipkill for main memory, and redundant power supplies and voltage regulators. RAS management systems are deployed for monitoring and control. The state of resilience research is more advanced and includes a number of technologies, such as fault-tolerant programming (fault-tolerant MPI, re-execution of failed tasks and containment domains), proactive fault tolerance using migration of computation away from components that are about to fail, and resilient solvers with recovery, compensation or self-stabilization properties. Recent work made inroads in understanding the fault, error and failure models of HPC systems. Some work in understanding the performance/energy and performance/resilience trade-offs exists as well. Other recent work pioneered the concept of design patterns for a structured approach to HPC resilience.

Hardware/software HPC codesign for resilience is mostly nonexistent at this point! There are a few concepts, models and tools, investigating and comparing individual resilience technologies and their performance/resilience trade-offs, such as for checkpoint/restart and redundancy. There are no design space exploration tools investigating the performance, resilience and energy trade-offs of different compute node or HPC system hardware/software designs. As a result, HPC resilience research solutions are not adopted in practice, as it is unclear if their benefits warrant adoption costs. Another result is the inability to mitigate unexpected reliability issues in HPC systems with the employed resilience technologies. A prime example is the impact of unexpected GPU failures on ORNL's Titan. The system was never designed to handle the resulting MTBF of 2 hours, requiring replacement of 11,000 out of 18,688 GPUs.

Opportunity: Coordinated cross-layer and adaptive resilience solutions can offer efficient error and failure masking, recovery, and avoidance at the appropriate hardware or software component and compute or data granularity. While the various heterogeneous compute and memory components will have hardware resilience mechanisms, software-based solutions to fill gaps in detection, masking, recovery, and avoidance of errors and failures will require coordination between the multiple layers of the system by design. Based on the underlying execution model and intrinsic resilience features of the hardware, the various components in an extreme heterogeneous system may be organized into predefined protection domains. Coordinated resilience solutions will handle errors and failures in specific components and granularities where it is most appropriate to do so and in coordination with the rest of the system, which prevents errors from propagating and failures from cascading beyond the protection domains.

This approach also removes some of the complexity that is introduced by extreme heterogeneity. Adaptive strategies can leverage the unique capabilities of heterogeneous protection domains, since the performance, resilience and energy profiles of each domain are different. Programming models and runtime environments may dynamically configure an application to use specific components in the heterogeneous system based on performance, resilience and energy costs. For example, critical computation may be executed on more resilient components; computation on less resilient components may be checked for errors with computation on more resilient components. Critical data may be stored solely or at least backed up on more resilient storage. Holistic cross-layer and adaptive resilience essentially provides efficient end-to-end resilience by design for computation and data.

Resilience needs to become an integral part of the HPC hardware/software ecosystem through codesign, such that the burden for resilience is on the system by design and not on the operator or user as an afterthought. Understanding the performance, resilience and energy trade-off is key to solving the resilience challenge for extreme heterogeneity, which is to design a reliable system within a given cost budget to achieve an expected performance. Design choices are based on a detailed understanding of this trade-off, which is HPC system and HPC application specific. Future research in hardware/software HPC codesign for resilience needs to address the following aspects:

- Develop an understanding of the error and failure characteristics of hardware and software components.
- Identify protection domains, interfaces and mechanisms of resilience capabilities in hard- and software.
- Design interfaces and mechanisms for coordinating resilience capabilities and quality of service requirements across hardware and software components.
- Define uniform metrics for assessing performance, resilience and energy across heterogeneous components to enable design trade-offs.
- Create design space exploration tools to understand the performance, resilience and energy trade-offs between different node and system designs.

Timeliness or maturity: The state of research for HPC resilience is rich in mechanisms that can be utilized. However, a longer-term and coordinated codesign effort is required to enable wide-ranging resilience capabilities in practice and to make them an integral part of the HPC hardware/software ecosystem. Research in defining, communicating and matching HPC resilience capabilities with quality of service requirements is required as we transition to extreme heterogeneity, including creating best practices and standards for resilience. Recent work in fault models, trade-offs and resilience design patterns can form the basis for solving the challenges. However, more research in (1) uniform metrics, (2) performance/resilience/energy trade-offs and (3) design space exploration tools is still required.

Simply put, if resilience by design is not done now, in the early stages of extreme heterogeneity, the current state of practice for HPC resilience, global application-level checkpoint/restart, will remain the same for decades to come due to the high costs of adoption of alternatives later on. The prime example for this is MPI, for which, 25 years after its first standardization, resilience is still not part of the MPI standard, despite 20 years of research in fault-tolerant MPI, numerous research prototypes and a 10-year discussion in the MPI standardization body. PVM, MPI's predecessor, was fault tolerant in 1993! In contrast to the MPI standardization effort 25 years ago, the current state of research for HPC resilience is far beyond the current state of practice. The existing knowledge, experience and prototypes serve as a foundation for making resilience an integral part of the HPC hardware/software ecosystem.