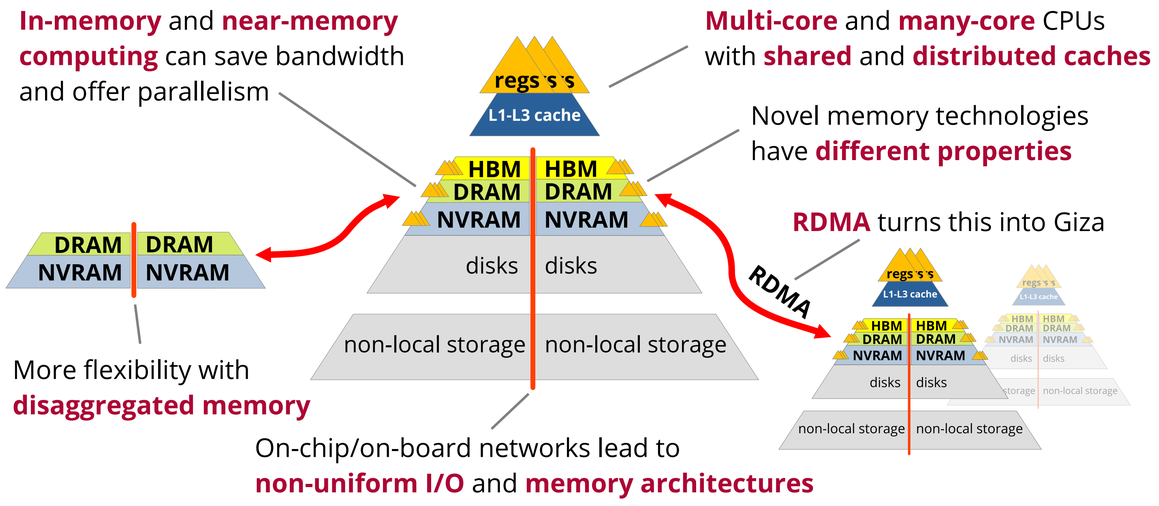
SMAUG

System-Level Modeling and Optimized Use of Disruptive Memory Technologies

UNIVERSITÄT OSNABRÜCK

Embedded Software Systems

Memory Hierarchy with DMTs? Complex!



As memory hierarchies are becoming increasingly complex, system software needs a **hardware/software model** to manage resources efficiently.

Requirements / Decisions to Make

Relevant for different software layers: OS, DB, Compiler, …

Mapping decision

- **Data** to memory region
- **Control flows** to CPU or other processing elements (NMC, PIM)

Optimization goals

- Performance, energy, WCET, memory, aging, …

Input data

- **HW** component properties (performance, energy consumption, …)
- **Application** profiles (R/W patterns, atomicity requirements, …)
- **System** (bottlenecks, e.g. interconnect and PCIe, parallelism, concurrency, …)

Precise system-level models are crucial for the optimized use of disruptive memory technology!

State of the Art / Models

Analyses/models of disruptive memory technology

NVM
[Izraelevitz '19, Wang '20, Pohl '17, George '20, Scargall '20, Köppen '19]
RDMA
[MacArthur '12, Kalia '16, Nelson '19, Wei '21, Shen '20, Qiu '18]
NMC/PIM
[Singh '19, Hsieh '16, Mutlu '20, Khan '20, Corda '21, Lee '20]
HBM
[Das '20, Laghari '17]

- **Isolated performance analyses** (best case, benchmarks)
- **Level of detail** only suitable for **offline** usage

Whole-system models for OS decision making

Barrelfish SKB
[Schüpbach '12, Baumann '09]
MCTOP
[Chatzopoulos '17]
hwloc
[Broquedis '10]

- Almost **no support** for **disruptive memory technology**, yet
- Focus on NUMA (latencies, bandwidth)
- No application model, no prediction support

Approach / SMAUG Project

Phase 1: Models

- Set up a **test bed** for disruptive memory technologies
  - Hardware: Cluster that supports NVRAM, RDMA, PIM, HBM, NMC
  - Software: Linux and MxCluster (extended MxKernel for bare-metal tests)
- Detailed **measurements**
  - Bottlenecks, synchronization effects, access patterns, …
- Creation of **models** for HW components, systems, and applications
- Model **validation** based on system simulator
- Early experiments with model-based **management strategies**

Phase 2: Management

- **Systematic analysis** of resource management strategies
- **Integration** into **MxCluster operating system** and evaluation

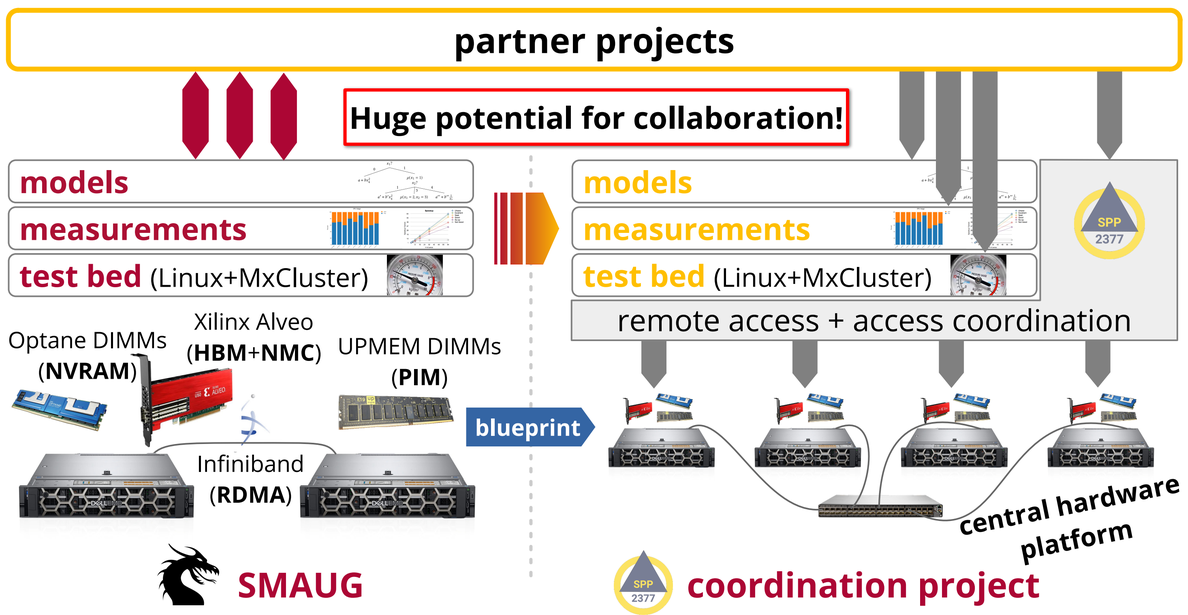
Milestones in Phase 1

Work structured in three full iterations:
3x (test bed → modeling → measurements → validation)

NVRAM (latest gen. Optane DCPMMs)
RDMA (Mellanox Infiniband)
PIM (UPMEM DIMMs)
NMC & HBM (Xilinx Alveo U50/280)

		year 1				year 2				year 3				
work package		Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	MM
WP1	platform model	■	■	■			■			■				5
WP2	measurements		■	■			■			■				4
WP3	testbed	■■	■	■			■			■				6
WP4	SW model				■■			■			■			4
WP5	SW analysis							■■			■■			4
WP6	validation					■■			■■			■■■		7
WP7	optimized use					■							■■	3
WP8	collaboration & dissemination				■				■				■	3
	total	3	3	3	3	3	3	3	3	3	3	3	3	36

Role within the Priority Program



SMAUG plans to build models for the central hardware platform that all participating projects could use.

References

Baumann, A. et al. 2009. The multikernel: a new OS architecture for scalable multicore systems. Proceedings of the ACM SIGOPS 22nd symposium on Operating Systems Principles - SOSP '09 (Big Sky, Montana, USA, 2009), 29.
Broquedis, F. et al. 2010. hwloc: A Generic Framework for Managing Hardware Affinities in HPC Applications. 18th Euromicro Conference on Parallel, Distributed and Network-based Processing (Feb. 2010), 180–186.
Chatzopoulos, G. et al. 2017. Abstracting Multi-Core Topologies with MCTOP. Proceedings of the Twelfth European Conference on Computer Systems (Belgrade, Serbia, Apr. 2017), 544–559.
Corda, S. et al. 2021. NMPO: Near-Memory Computing Profiling and Offloading. 24th Euromicro Conference on Digital System Design (DSD) (2021), 259–267.
Das, R. et al. 2020. How to Manage High-Bandwidth Memory Automatically. Proceedings of the 32nd ACM Symposium on Parallelism in Algorithms and Architectures (New York, NY, USA, Jul. 2020), 187–199.
George, J.S. et al. 2020. go-pmem: Native Support for Programming Persistent Memory in Go. 2020 USENIX Annual Technical Conference (USENIX ATC 20) (2020), 859–872.
Hsieh, K. et al. 2016. Transparent Offloading and Mapping (TOM): Enabling Programmer-Transparent Near-Data Processing in GPU Systems. ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA) (Jun. 2016), 204–216.
Izraelevitz, J. et al. 2019. Basic Performance Measurements of the Intel Optane DC Persistent Memory Module. arXiv:1903.05714 [cs]. (Aug. 2019).
Kalia, A. et al. 2016. Design Guidelines for High Performance RDMA Systems. Proceedings of the 2016 USENIX Annual Technical Conference (USENIX ATC 16) (2016), 437–450.
Khan, K. et al. 2020. A Survey of Resource Management for Processing-In-Memory and Near-Memory Processing Architectures. Journal of Low Power Electronics and Applications. 10, 4 (Dec. 2020), 30. DOI:https://doi.org/10.3390/jlpea10040030.
Köppen, M. et al. 2019. Cache-Line Transactions: Building Blocks for Persistent Kernel Data Structures Enabled by AspectC++. Proceedings of the 10th Workshop on Programming Languages and Operating Systems (New York, NY, USA, Oct. 2019), 38–44.
Laghari, M. and Unat, D. 2017. Object Placement for High Bandwidth Memory Augmented with High Capacity Memory. Proceedings of the 29th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD) (Oct. 2017), 129–136.
Lee, D. et al. 2020. Optimizing Data Movement with Near-Memory Acceleration of In-memory DBMS. Proceedings of the 22nd International Conference on Extending Database Technology (EDBT). (Apr. 2020), 371–374. DOI:https://doi.org/10.5441/002/edbt.2020.35.
MacArthur, P. and Russell, R.D. 2012. A Performance Study to Guide RDMA Programming Decisions. Proceedings of the 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (Jun. 2012), 778–785.
Mutlu, O. et al. 2020. A Modern Primer on Processing in Memory. CoRR. abs/2012.03112, (2020).
Nelson, J. and Palmieri, R. 2019. Understanding RDMA Behavior in NUMA Systems. Proceedings of the IEEE/ACM International Symposium on Code Generation and Optimization (CGO) (Feb. 2019), 273–274.
Pohl, C. et al. 2017. A Cost Model for Data Stream Processing on Modern Hardware. ADMS@ VLDB (2017), 8.
Qiu, H. et al. 2018. Toward Effective and Fair RDMA Resource Sharing. Proceedings of the 2nd Asia-Pacific Workshop on Networking (New York, NY, USA, Aug. 2018), 8–14.
Scargall, S. 2020. Introducing the Persistent Memory Development Kit. Programming Persistent Memory: A Comprehensive Guide for Developers. S. Scargall, ed. Apress. 63–72.
Schüpbach, A.L. 2012. Tackling OS Complexity with Declarative Techniques. ETH Zurich.
Shen, D. et al. 2020. Distributed and Optimal RDMA Resource Scheduling in Shared Data Center Networks. IEEE INFOCOM 2020 - IEEE Conference on Computer Communications (Jul. 2020), 606–615.
Singh, G. et al. 2019. NAPEL: Near-Memory Computing Application Performance Prediction via Ensemble Learning. Proceedings of the 56th ACM/IEEE Design Automation Conference (DAC) (Jun. 2019), 1–6.
Wang, Z. et al. 2020. Shuhai: Benchmarking high bandwidth memory on FPGAS. 2020 IEEE 28th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM) (2020), 111–119.
Wei, X. et al. 2021. Characterizing and Optimizing Remote Persistent Memory with RDMA and NVM. Proceedings of the 2021 USENIX Annual Technical Conference (USENIX ATC 21) (2021), 523–536.

Marcel Köppen and Olaf Spinczyk {marcel.koeppen,olaf}@uos.de

Embedded Software Systems Group · Computer Science Institute · Osnabrück University · ess.cs.uos.de