Chair of Decentralized Systems Engineering
Department of Informatics

Programming model for hybrid persistent memory systems

General information	**Advisor**	Dimitrios Stavrakakis
	Email	dimitrios.stavrakakis@tum.de
	Date	17.01.2022
Type	Master / Bachelor / Guided Research	
Description	Persistent memory[1] is a recently added layer in the system stack of data centers. It resides on the memory bus and provides read and write access in byte granularity with latencies close to DRAM. Along with these features, it is also a non-volatile storage medium. This unique combination of features has opened up a way for a new programming model targeting to manipulate this medium and reap its benefits. Having that in mind, Intel developed PMDK[2], a collection of libraries and tools for system administrators and application developers to simplify managing and accessing persistent memory devices. However, integrating PM in existing systems requires significant effort and knowledge from the side of the programmer. There exist widely-used systems that have been adapted to use PM[3, 4], but they remain in a premature state and are not extensively used in production. They are mainly rewritten based on the PM programming model and incorporate manual optimizations, which are error-prone and tedious tasks.	

Therefore, in this project, we are aiming to design a compiler-based hybrid programming model that will integrate PM into existing systems developed for conventional DRAM-storage system stacks while abstracting out the PMDK programming model details to ease programmability.

To achieve this, we will rely on the LLVM/clang compiler, design compiler passes and clang plug-ins. In this way, the programmer will be able to access and manage PM simply by providing his intentions to the compiler. Then, the compiler should be able to consider these intentions and transparently adapt the program to use PMDK functionalities in the backend.

Keywords

Persistent memory, DRAM, LLVM, clang, PMDK, portability, programming framework

Goals

Concrete outcomes

1. LLVM compiler passes & clang plug-ins to abstract out the programming details of PM programming from the developer (programmability)
2. Port of existing applications to use PM based on the proposed hybrid programming framework (portability)
3. Performance evaluation of ported systems against their hand-optimized counterpart (performance)

Prerequisites

Compulsory

- Basic operating system knowledge
- Good C/C++ knowledge
- Good understanding of memory & storage hierarchy

Preferred

- Experience with PM programming based on PMDK
- Experience with LLVM passes & clang plug-ins

References

1. https://docs.pmem.io/persistent-memory/getting-started-guide/introduction
2. https://github.com/pmem/pmdk
3. https://github.com/pmem/pmem-redis
4. https://github.com/pmem/pmem-rocksdb

5. https://www.intel.com/content/www/us/en/developer/tools/oneapi/overview.html

Application process

Please send an email to the advisor including the following:

- Email subject: "Thesis application (DSE)"
- CV
- A copy of your transcript(s)
- A **motivation statement**, please include samples of your work that you are proud of (e.g., major projects, open-source contributions, Github page, etc.) and/or writing samples (e.g., your technical blog, project reports, etc.)