technische universität dortmund

Bachelor/Master Thesis

Hypervisor-based Memory Access Analysis and Non-Volatile Memory Simulation

Kuan-Hsun Chen
Prof. Dr. Jian-Jia Chen
Otto-Hahn-Str. 16
Technische Universität Dortmund
Email: kuan-hsun.chen@tu-dortmund.de
17.06.2019

Byte-addressable non-volatile memories (NVMs) recently have emerged to serve as main memory due to the features of low leakage power, high density, and low unit costs. However, such NVMs has a significantly low endurance count of the memory cells compared to the classic DRAM. To mitigate this problem, several techniques in the literature have proposed wear-leveling algorithms in different granularities. To evaluate these approaches, cycle accurate system simulations (e.g., gem5 and NVMain) can capture the resulting memory access behavior, but require very long simulation times usually.

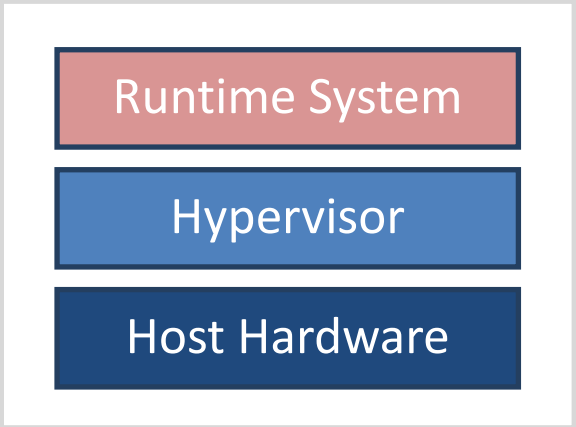

Figure 1: Overview of hypervisor-based simulator.

Alternatively, we would like to develop a hypervisor (see Figure 1), counting memory accesses, which can be executed on real hardware with a bare-metal runtime system. This hypervisor has to use a special memory configuration (MMU configuration) to trap every memory access from the underlying guest system to the hypervisor. The hypervisor can inspect the related instructions to see if the who issued them, count the memory accesses to different memory locations, and provide this information to the guest system in a dedicated hypercall.

In this thesis, students first should study the data sheet of the target platform and collect details about the hypervisor extension. After setting up the startup environment, e.g., bootloader configurations, students should implement a simple hypervisor, which only starts an in-house runtime system. Then the hypervisor should be extended by the required memory configuration and hypercalls to trap every memory access from the executed runtime system. At the end, the hypervisor-based simulator should be evaluated by reasonable experiments, e.g., applying an in-house designed weal-leveling algorithm. Students should note that, the involved source code this thesis will be publicly released and should be fully documented to comply the rationale of open-source software development.

Other suggestions and related topics are also welcome. Please do not hesitate to make an appointment.

Required Skills:

- Knowledgeable of C and C++ programming
- Willing to program bare-metal
- Computer architecture knowledge is required

Acquired Skills after the thesis:

- Knowledge about modern CPU execution models and the Hypervisor extensions.
- Knowledge about ARM CPUs and the ARM instruction set