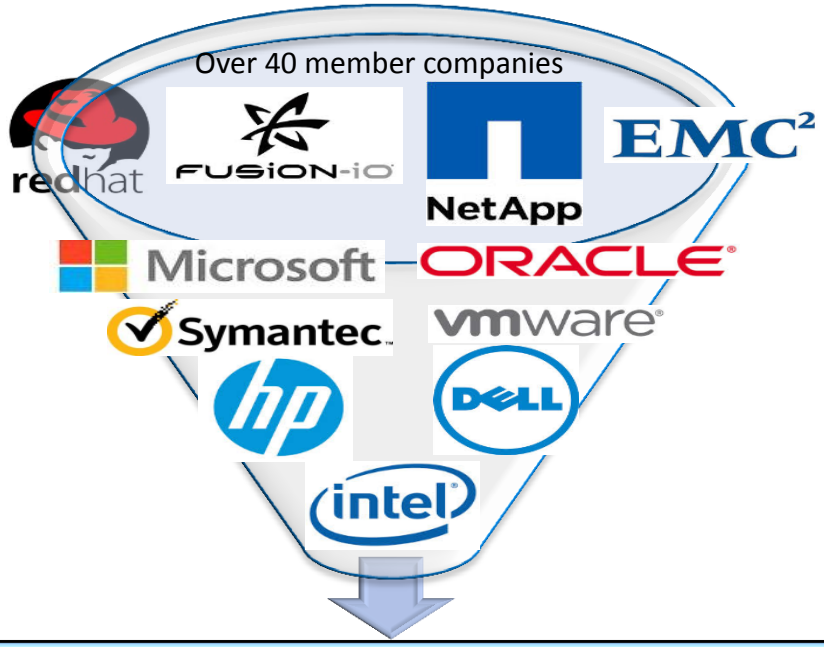
Persistent Memory
Byte-Addressable Non-Volatile Memory

Andy Rudoff

Principal Engineer, Intel Corporation

June 2014

SNIA NVM Programming TWG



SNIA

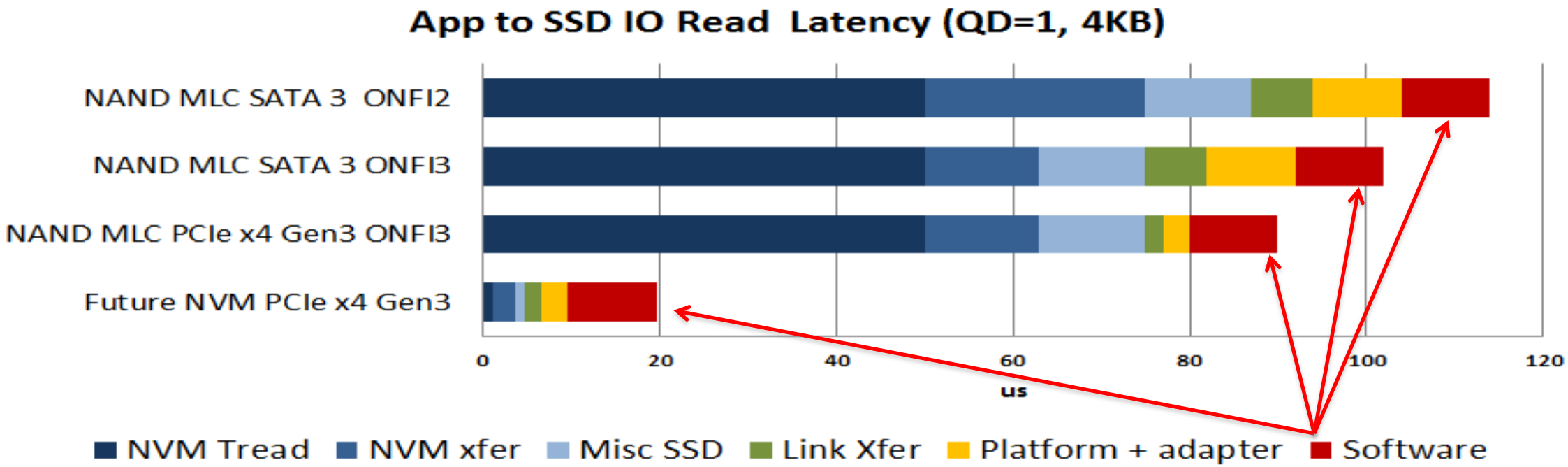
SNIA TWG Focus

- 4 modes of NVM programming
  - NVM PM Volumes, NVM PM Files, NVM block, NVM file,
- Ensure APIs cover the needs of the ISVs

Unified NVM Programming model

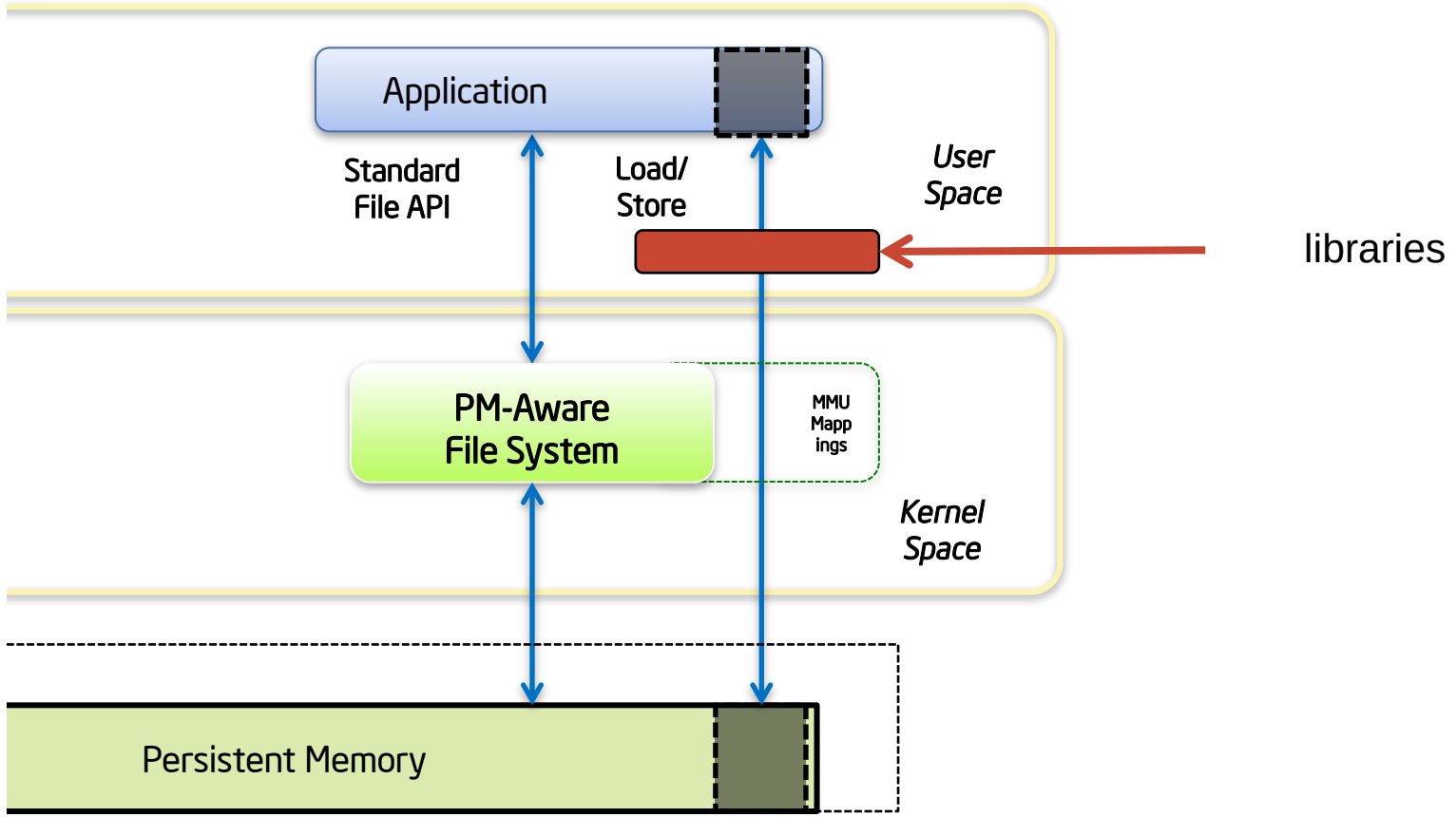
- First version of the spec out for public review

The NVM Software Stack



Building on the Basic PM Model

- NVM.PM.FILE programming model “surfaces” PM to application
- Still somewhat raw at that point
- **Build on it with additional libraries**
- May eventually turn into language extensions



NVM Library Modes

PMEMTRN: Persistent Memory Transactional

- Malloc broken into steps to make transactional
- Interruption-safe transactions for PM
- NVML routines all start with pmemtrn_ prefix

PMEMBLK: Persistent Memory carved into blocks

- Pool is divided up into a specific chunk size
- Single block writes to the pool are atomic
- NVML routines all start with pmemblk_ prefix

VMEM: Volatile Memory Allocator

- Use PM as volatile memory via malloc/free-like calls
- Leverage capacity
- Don't bother flushing for durability
- Pool "resets" on application restart
- NVML routines all start with vmem_ prefix

PMEMLOG: Log file (append-mostly)

- Common use case, write mostly
- Append operation very cheap
- Read through (for log shipping) also optimized
- NVML routines all start with pmemlog_ prefix

Summary

- Persistent Memory is coming…
- This is a game changer, similar to the explosion of cores
- There are the hard new programming problems
- Opportunities in both **Scale Up** and **Scale Out**