CHAPTER 6

Conclusion

Upcoming PMs aim to combine the byte-addressability and performance of DRAM, and the durability of traditional storage devices such as hard disks and SSDs. PMs may be accessed over a byte-addressable interface that is significantly faster than the block interface required to access traditional storage media. Unfortunately, the existing hardware, compiler, and software systems are not fully equipped to fully avail the full performance that the PMs offer.

This book surveyed a large class of works that have emerged both in the industry and the academic literature to integrate the PMs in hardware systems, compiler frameworks, and software applications. Correct recovery requires that PM operations are ordered to PM; this book detailed memory persistency models that prescribe the ordering constraints on PM operations. We defined strict and relaxed models formally, and also discussed the mechanisms proposed in the literature to implement these models in hardware. Further, this book described the logging mechanisms required to enable failure atomicity for operations that are larger than an individual persist. We detailed undo-, redo-, and shadow-logging mechanisms built in hardware that ensure that either all or none of the updates are visible to recovery in case of a failure. Finally, this book detailed several programming and software libraries that enable easier PM programming. It described software transactions, file systems, language persistency models, and testing frameworks designed for PM programming.

V. Gogte et al., *A Primer on Memory Persistency*