Journal of Systems Architecture 72 (2017) 80–92

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Fine grained, direct access file system support for storage class memory

Yi Wang[a,b], Tianzheng Wang[c], Duo Liu[d], Zili Shao[f,*], Jingling Xue[e]

[a] Beijing Advanced Innovation Center for Imaging Technology, China
[b] College of Computer Science and Software Engineering, Shenzhen University, China
[c] Department of Computer Science, University of Toronto, Canada
[d] College of Computer Science, Chongqing University, China
[e] School of Computer Science and Engineering, University of New South Wales, Sydney, Australia
[f] Embedded Systems and CPS Laboratory, Department of Computing, The Hong Kong Polytechnic University, Hong Kong

ARTICLE INFO

Article history:
Received 14 January 2016
Revised 24 May 2016
Accepted 18 July 2016
Available online 19 July 2016

Keywords:
Storage class memory
Memory management
Block device support
Phase change memory
Byte-addressable direct I/O,

ABSTRACT

New storage class memory (SCM) technologies, such as phase change memory (PCM) and memristors, are not only byte-addressable like DRAM but also non-volatile like traditional hard disk drives. SCM modules can be placed side-by-side with DRAM on the memory bus, available to memory instructions issued by the CPU. This shift thus engenders a new "DRAM-SCM" storage architecture, which promises near-DRAM secondary storage access speed at several orders of magnitude faster than magnetic disk or flash memory. Utilizing SCM as a secondary storage device will have a profound impact on memory hierarchy design, requiring new architectural and operating system support.

In this paper, we adopt PCM in the DRAM-SCM storage architecture and present *BSS* to provide file system-independent Block device Support for Storage class memory. To ensure backward compatibility and high performance, BSS provides a block device interface found in traditional hard disk drives and allows existing file systems to be built on top of itself without any modifications. BSS is designed to directly access the PCM through memory instructions and bypass traditional disk caches that are intended to reduce seek time.

The DRAM-SCM architecture and BSS are prototyped in QEMU and the Linux kernel, respectively. Validation using benchmarks reveals that both work together well to exploit significant advantages of SCM. Compared to traditional hard disk drives, our approach boosts the write/read performance by up to 204x for large files and achieves comparable performance for small ones.

1. Introduction

Storage systems have long been a major I/O bottleneck in modern computer systems. To eliminate this bottleneck, solid-state, non-volatile memory technologies, have been proposed to replace traditional hard disk drives. *Storage class memory* (SCM) represents a new class of data storage and memory devices, which blur the distinction between memory and storage.

Despite its promising benefits (e.g., low power consumption, non-volatility and high density), SCM still suffers from the lack of a unified storage architecture to fully realize its potentials. As existing operating systems are designed to cater for the traditional DRAM-hard disk architecture with a strict distinction between memory and storage, considerable efforts on architectural and operating system support are required to facilitate smooth access to SCM as secondary storage.

In this paper, we present *BSS* to provide file system-independent Block device Support for Storage class memory. BSS provides a block device interface to the file system and bypasses the existing page cache to achieve direct I/O accesses to SCM, with the resulting advantage that *no changes are required to be made to the file system or the hardware implementation of SCM.* Consequently, existing file systems can be used on top of BSS directly without any modifications. File system users can continue to enjoy advanced file system features, such as scalability and performance enhancements for large files, without actually noticing that the underlying storage medium is SCM rather than hard disk drives.

This paper offers both architectural and operating system support for SCM. For architectural support, we propose a design of

* Corresponding author.
E-mail address: cszlshao@comp.polyu.edu.hk (Z. Shao).

http://dx.doi.org/10.1016/j.sysarc.2016.07.003
1383-7621/

the DRAM-SCM storage architecture and adopt phase change memory (PCM) in this DRAM-SCM architecture. In our architecture, both DRAM and PCM are attached to the memory bus directly through the DDR interface. DRAM is used as primary storage while PCM is intended for secondary storage. The design of memory management unit (MMU) is done so that the memory partitions, which represent memory regions covered by DRAM and PCM modules, can be reported to the operating system.

With the DRAM-SCM architecture, SCM can be accessed through traditional memory instructions. For secondary storage, the access of physical space is based on the sector. The sector is fixed in the secondary storage device, and it does not require further address mappings. Then the process of virtual-to-physical memory address translation for the access to SCM becomes redundant. Therefore, we propose a hybrid CPU mode to facilitate data access to SCM. In the proposed hybrid mode, the addresses for DRAM are still interpreted as virtual ones using the virtual mode. The access to SCM will be conducted through new memory instructions. The SCM addresses are interpreted as physical addresses in MMU. This will prevent the unnecessary address translations in SCM.

In addition to the aforementioned architectural support, we have also provided operating system support to SCM. In order to utilize SCM as a secondary storage device, one possible solution would be to design a new file system specifically for SCM. However, a new file system requires a tremendous amount of design efforts and will only become mature after a long period of time. As a result, our solution is different. BSS maintains backward compatibility by providing a block device interface usually implemented by traditional hard disk drives for file systems. In addition, BSS directly utilizes the new instructions for data transfer and bypasses the traditional disk cache to achieve fast direct I/O access to SCM.

This paper makes the following contributions:

- We propose a file system-independent block device interface for SCM, which provides backward compatibility and transparently supports the integration of SCM in current computer systems.
- We bypass traditional disk caches to allow fast direct I/O access to SCM, so that secondary storage can be accessed identically as main memory.
- We propose a new hybrid CPU mode for the DRAM-SCM architecture and introduce two new privileged instructions to provide fine-grained, physical access to SCM.

We have implemented the DRAM-SCM architecture on top of QEMU [1], a full-system emulator, by adding new memory regions, maps, and instructions for PCM. The new memory map that contains both DRAM and PCM regions will be reported to the operating system by the BIOS. In addition, we have also built a prototype of BSS in the Linux kernel. In our experimental validation and benchmarking, we strive to demonstrate both the performance advantages and feasibility of the DRAM-SCM architecture and BSS. To the best of our knowledge, this is the first work to provide a comprehensive working design and prototype for SCM as secondary storage, with both architectural and operating system support.

The rest of this paper is organized as follows. We first present an overview of storage class memory in Section 2. We discuss the related work in Section 3. The designs of the DRAM-SCM architecture and BSS are presented in Sections 4 and 5, respectively. Section 6 introduces our implementation details. We evaluate BSS on top of the DRAM-SCM architecture in Section 7. Finally, Section 8 concludes this paper and discusses future work.

Table 1
Comparing DRAM, NAND flash and PCM.

Attributes	DRAM	PCM	NAND
Non-Volatile	No	Yes	Yes
Erase Required	Bit	Bit	Block
Software	Simple	Simple	Complex
Power	~W/GB	100 → 500mW/die	~100mW/die
Write Latency	~20-50ns	~2-3μs	~100μs
Write Energy	~0.1nJ/b	< 1nJ/b	0.1-1nJ/b
Read Latency	50ns	50-100ns	10-25μs
Read Energy	~0.1nJ/b	$\ll$1nJ/b	$\ll$1nJ/b
Idle Power	~W/GB	$\ll$0.1W	$\ll$0.1W
Endurance	∞	10^8	$10^5 \rightarrow 10^4$
Data Retention	ms	Not f (cycles)	f (cycles)

2. Background

The term *storage class memory* refers to a type of storage technology that is non-volatile and spacious like traditional hard disk drives but with performance and interfaces similar to those of main memory. It is a type of solid-state, resistive storage medium that blurs the boundaries between storage and memory by being low-cost, fast, and non-volatile [2].

Resistive memories, such as phase-change memory (PCM) [3–5], Ferroelectric RAM (FeRAM) [6], spin-transfer torque RAM (STT-RAM) [7–12], and memristors [13,14], use atomic arrangements to set the resistance of memory cells to store information [15]. Unlike DRAM and flash memory, resistive memories store data by using resistivity instead of electrical charge. Resistive memories are typically byte-addressable, non-volatile, and have better scalability compared to flash memory. Although resistive memories are currently only available in small capacity, the scaling trends predict that larger and cheaper resistive memories will quickly become available in the near future. The emerging three-dimensional architecture also boost the adoption of resistive memories [16,17]. They will replace most of semiconductor memories in current computer systems. Therefore, resistive memories are promising candidates to implement SCM.

Among these resistive memories, PCM is one of the promising candidates to replace hard disk drives and closest to high-density commercial deployment. PCM cells can be organized into an array structure like that of DRAM [18]. Thus, it is possible to manufacture PCM modules that operate in the same way as exiting DRAM modules [19]. For instance, 20 nm, 8 Gbit DDR-compatible PCM modules announced by Samsung are now available in the market for embedded mobile devices. The PCM chip can provide a program bandwidth of 40 MB/s [20], making it an ideal candidate for future storage systems. This paper focuses on the architectural and operating system support for PCM-based SCM.

Table 1 shows the access latency and endurance properties of DRAM, NAND flash memory, and PCM. Compared to DRAM, PCM handles read operations at the same or half speed but write operations about 100 times slower. In addition, PCM excels in idle power but costs 10 times more write energy. Thus the cache hierarchy is needed in future systems to improve write performance for PCM. Compared to NAND flash memory, PCM has much better access latency while achieving a similar energy consumption level.

Although raw PCM is much more endurable (10^8) compared to flash memory (10^4 for MLC NAND flash and 10^5 for SLC NAND flash), it still suffers from the limited maximum number of write cycles per cell [21,22] Numerous attempts at the architectural level have been made to tackle this problem [23–25], some of which provide guarantees to prolong the lifetime of PCM [26–29]. In this paper, we assume that the endurance problem has been taken care at the architectural level. This work focuses only on providing architectural and OS designs to adopt PCM in modern computer sys-