An Overview on MEMS-based Storage, Its Research Issues and Open Problems

Yifeng Zhu
Department of Computer Science and Engineering
University of Nebraska – Lincoln

Abstract

A disruptive new storage technology based on Microelectromechanical Systems (MEMS) is emerging as an exciting complement to the memory hierarchy. This study reviews and summarizes the current research about integrating this new technology into computer systems from four levels: device, architecture, system and application. In addition, several potential research issues in MEMS storage are identified, including (1) exploiting idle read/write tips to perform prefetching, (2) reversal access to save seek time, (3) fault-tolerance design inside storage devices, (4) power consumption modeling, (5) reevaluation of existing disk-type I/O optimization algorithms.

1. Introduction

While magnetic disks have dominated on-line secondary storage for over 30 years, the performance of computer systems has suffered from the increasing gap between RAM and disks in latency, bandwidth and cost. The increase in this RAM-to-disk gap, estimated at a rate of 50% per year [1], has been caused primarily by the physical characteristics of disk drives, such as the limitation of disk rotational speed, bit density (which is constrained by super-paramagnetic effect [2][3]) and read-write head technology. To overcome these limitations, researchers in academia and industry are developing disruptive new storage devices, among which, MEMS-based (micro-electromechanical systems) storage [4][5][6] is poised to be the most promising one. This new technology is expected to be commercially available in 2004 [7] and will offer a new set of characteristics that bridge the increasing gap in the memory hierarchy. The duty of computer scientists and engineers is then to integrate these new devices into memory hierarchy and optimize their performances for most applications. The challenge, however, lies in the significant difference between MEMS storage devices and disk devices.

This study starts with a brief description of MEMS storages in Section 2. Then Section 3 investigates the current research work trends in four levels of abstractions: device, architecture, system and application. Potential research issues and open problems in MEMS storage are identified and summarized in Section 4. Finally Section 5 concludes the study.

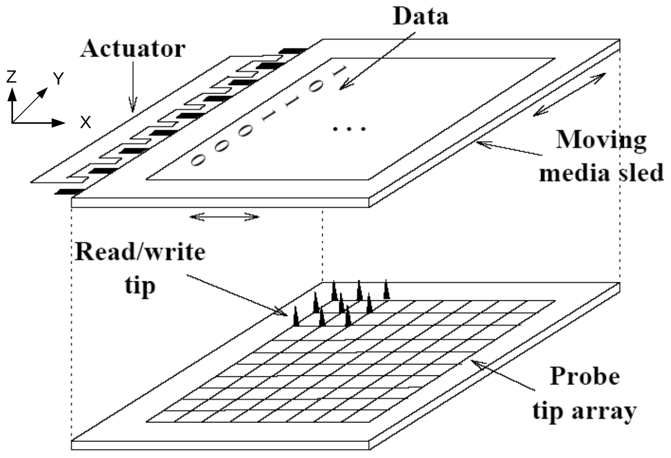

Figure 1: A high-level diagram of a MEMS-based storage chip [8]. Every side of the media sled has an actuator, and a read/write tip is placed at all crossing points in the 2-D tip array. This media sled is spring-mounted above the stationary tip array and can be moved a small distance (up to 10% of actuator length) in X-axis and Y-axis by actuators.

2. MEMS-based Storage Basics

This section describes the basic operation mechanism in MEMS-based storage devices and compares their physical characteristics with those of conventional disk drives.

2.1 Basic Read/Write Operations

MEMS storage has a MEMS structure to mechanically move its storage media. Specifically, a MEMS structure includes multiple tiny mechanical structures ranging 10-1000 μm in size. Fabricated on the surface of silicon wafer, these miniature structures can slide, bend, or deflect in response to electrostatic or electromagnetic forces from nearby actuators [9].

A MEMS storage device consists of an array of stationary read/write probe tips and a movable rectangular storage media sled, as shown in Figure 1. Unlike disks, the media sled does not rotate but moves simultaneously in two independent directions, X and Y, to place the probe tips right above the first destination bit. Then, the sled continues to move in the Y-axis at a constant velocity to read or write consecutive bits.

To cover a large media area, the MEMS devices usually need a large number of probe tips. Since the reciprocating X and Y motions of the media sled are typically bounded to 10% of the actuator length, one probe tip can only access around 1% of the media surface with a capacity of tens of megabytes [4]. To break the capacity limitation, an array of probe tips are used and distributed evenly above the media surface. In addition, the media sled is logically divided into small rectangular regions, each accessed by one probe tip exclusively. For power and heat considerations, a limited number of tips (200-2000) can be active simultaneously. Consequently the access area increases from 1% to 50%, a percentage comparable to that used by disks [1].

2.2 Data Layout

Integrating MEMS storage into a current computer system requires a disk-like interface on these devices. Accordingly, the address space of MEMS storage needs to be organized to facilitate the development of this interface. Figure 2 shows a typical address scheme that uses terminologies similar to disks. Like disks, data is not byte-addressable. The smallest unit of data in a MEMS-based storage device is called a tip sector, consisting of its servo information for positioning, the actual data content and its own error correction information. Each tip sector is identified by the triple <x, y, tip>,

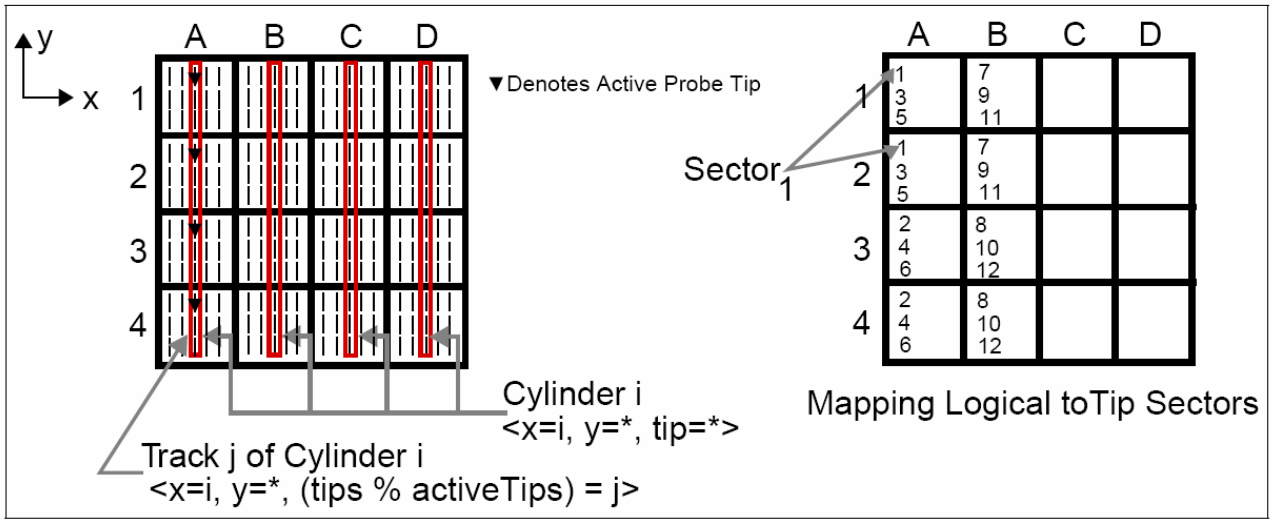

Figure 2: Cylinders, Tracks, and Sectors [10]. All the columns with the same X coordinate form a cylinder. Track j consists of tip sectors within a cylinder that are accessible by the concurrently active tips. In this example, tip A1, A2, A3, A4 can be active simultaneously and activeTips = 4. The tips are linearly indexed (A1 = 0, B1=1, C1=2, etc.). Each logical sector, shown on the right side, is striped across two tip sectors.

where <x, y> are coordinates within the rectangular region accessible by <tip>. Note that each rectangle has its own origin. A ***cylinder*** consists of all tip sectors that are accessible by any tip without moving the sled along the X-axis; all the tip sectors in a cylinder share the same value for <x>. A ***track*** is a subset of a cylinder consisting of all tip sectors that can be accessed by simultaneously active tips. A track is divided into multiple ***logical sectors***. To improve the throughput and avoid hot spots, a logical sector is striped across multiple tip sectors. Ref. [11] designs a SCSI-type interface with a special Logical-Block-Number-to-Physical-Block-Number (LBN-to-PBN) mapping that fully exploits the internal parallelism among read/write tips. A work-in-progress [12] shows that the disk-like layout may be detrimental to application performance and it suggests a cylinder to be rectangular lines instead of straight lines. To reduce service time, Ref. [13] proposes a serpentine layout where data is accessed from +Y to –Y and from –Y to +Y alternatively in neighboring tracks.

2.3 Strengths and Weaknesses

By using MEMS structures to move the media sled beneath a large array of stationary probe tips and employing a disk-like interface, MEMS storage is an exciting new technology with many advantages over conventional disks.

- **Small volume and low power dissipation:** Fabricated on silicon chips using standard CMOS processes, MEMS storage takes limited space and little power. For example, IBM promises to release their MEMS storage with a size of a postage-stamp in 2005 [5].

- **Ultra-low and stable latency:** Due to the absence of spindle rotation time, short distance seeking (less than 3mm [14]) and lightweight media sled, the positioning delays in MEMS storage are much smaller than those in disks. Table 1 shows that such new devices will be faster than the future disk by one order of magnitude. Another benefit of MEMS storage is that the latency is more predictive than disks, even under random workload. Disks spend much larger amount of time to locate destination data by moving disk heads and rotating platters, thus making its response time unstable.
- **High throughput:** While the transfer rate for individual probe tip is only 100kb/s – 1Mb/s [15], the aggregate data rate for a large number of simultaneously active probe tips can reach 1 GB/s, two orders of magnitude higher than the rate of current disks, as shown in Table 1.
- **Low cost:** MEMS storage with capacity of 1-10GB is expected to have the lowest price per byte, while the cheapest storage under 100MB and above 100 GB is offered by nonvolatile RAM and disks, respectively [16].

In spite of the above advantages, MEMS storage is not expected to completely replace the nonvolatile RAM and disks for at least two reasons. First its performance is worse than nonvolatile RAM and its price is higher than large capacity disks. Second, its reliability remains unpredictable.

3. Current Research Directions

During the past three years, a series of studies have been conducted to revisit many aspects of the computing systems to integrate the new and innovative MEMS storage technology. As shown in Figure 3, studies have been conducted on four levels of abstractions: device, architecture, system and application. The following is a summery of these studies.

	RAM	Atlas 10K	Future Disk	G1 Model	G2 Model	G3 Model
RPM	N/A	10,025	20, 000	N/A	N/A	N/A
Bandwidth (*GB/s*)	3.2 – 6.4	0.025	0.10 – 0.15	0.23	0.8	1.14
Avg. access time (*ms*)	$(2.5 – 5)\times10^{-6}$	10.83	3 – 4	1 – 2	0.8 – 1.0	0.4 – 0.7
Density (GB/cm^2)	N/A	0.14	0.36	4	6.25	11.1

Table 1: Comparisons between RAM, a state-of-art disk, a future disk in 2005, and three generations of CMU MEMS storage [13]. G1 is a conservative estimation of what is fabricated within next year. G2 is an enhanced one over G1 with bidirectional access, higher tip transfer rate, better servo system and higher density. G3 is targeted as high-end storage.

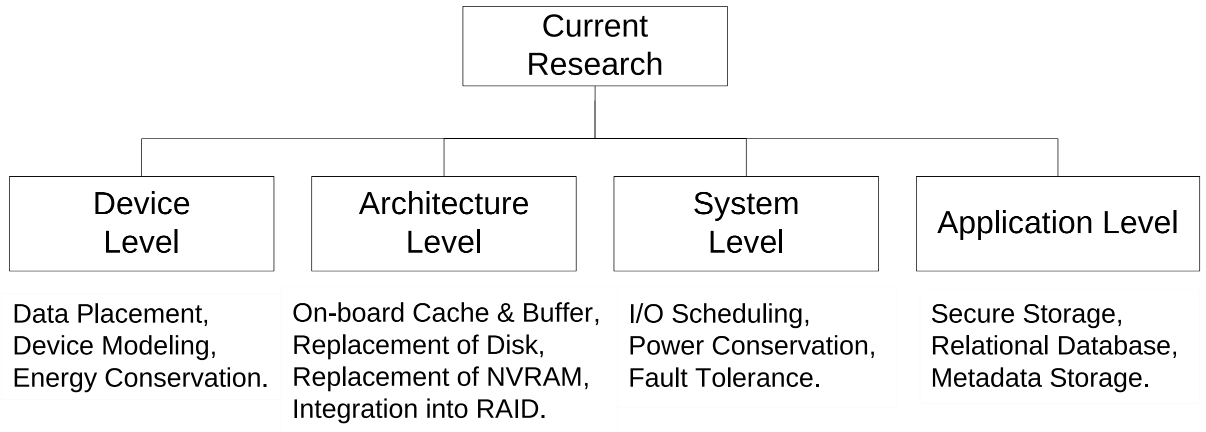
Current Research
Device Level
Architecture Level
System Level
Application Level
Data Placement, Device Modeling, Energy Conservation.
On-board Cache & Buffer, Replacement of Disk, Replacement of NVRAM, Integration into RAID.
I/O Scheduling, Power Conservation, Fault Tolerance.
Secure Storage, Relational Database, Metadata Storage.

Figure 3: An overview of the research directions

3. 1. Device Level

While material scientists and mechanical engineers are concerned with the physical design and the implementation of MEMS storage devices, computer scientists and engineers focus on the performance optimization, performance modeling and power conservation.

3.1.1 Dynamic Data Placement

Dynamic data placement is a conventional load balance technique that is gradually offloaded from the applications to on-board processors in disks. With accurate knowledge of disk layout and geometrical distribution of user data, the offloading can achieve a higher efficiency. For disks, there are three well-known placement policies: (1) Organ-pipe placement [17][18], (2) Camel placement [19], and (3) Simulation Annealing [20]. Organ pipe placement dynamically rearranges the disk cylinders such that the most frequently accessed files are placed in the middle tracks to reduce mean seek time optimally. Camel placement positions the most frequently accessed data around the middle of two intervals divided by the middle disk cylinder. Simulation Annealing is an iteratively swapping technique that clusters highly correlated data and stores them contiguously on disk platters.

In MEMS storage, data placement is also important to minimize the number of data accesses. MEMS storage has a higher acceleration near the centermost point where the resistant force from the suspension springs is relatively smaller. As a result, it tends to have quicker seek operations in the center subregion than in the outer subregion within each tip region. Based on this characteristic, Ref. [21] proposes a bipartite data placement scheme: small data is placed in the centermost subregions and long sequential data is placed in outer subregions. This optimization technique is based on the fact that the service time is dominated by the seek time for small random requests and by the transfer time for large sequential request. Ref. [21] also shows that Organ-pipe placement is not optimal in MEMS storage. The efficiency of Camel placement and simulation annealing in MEMS storage remains unevaluated and can be potential research issues.

3.1.2 Device Modeling

Since MEMS storage devices do not exist yet, it is critical to build accurate models to gain insights into their performance. While some references [22] model such devices at the electrical gate level to analyze the data transfer rate, our survey concentrates on the mechanical level because mechanical motions determine the latency and the system-end throughput.

Similar to disks, MEMS storage needs to put the media right beneath the read/write tips. However, it differs from disks by abandoning the rotation mechanism and moving the media rectilinearly. Thus the service time in MEMS storage includes only two elements, the seek time and the transfer time, as shown in Equation 1, while the disk models have an extra element, i.e., the rational latency.

$$t_{service} = t_{seek} + t_{transfer} \quad (1)$$

The media motions in X and Y are independent and thus a seek time is the greater one between X and Y. Thus

$$t_{seek} = \max(t_{seek\_x},\ t_{seek\_y}) \quad (2)$$

A seek operation along X involves three phases: acceleration, deceleration, and oscillation damping while a seek in Y only consists of acceleration and deceleration. Oscillation damping exists only in the seeks along X because the probe tips have to reduce their velocity to zero and keep their X coordinate on a specific value. The acceleration and deceleration force comes from the actuators and the suspension springs that always move the media sled towards the centermost position. The transfer time is related to the number of active tips, the request sizes, data layout schemes and the actual data distribution [23][24].

Currently there are two major approaches to compute t_{seek} and $t_{transfer}$: detailed simulation and analytical modeling. In the former approach, the simulator emulates the physical behaviors of such devices, performs I/O events in an order specified in traces and calculates the time for every operation. In the latter approach, the analytical models extract the statistical characteristics from traces to estimate the average response time without simulating trace events. Ref. [25], [26] and [27] follow the first approach by building the interaction of the spring force and actuator force into physical equations according to Newton's three laws of motions. As the spring force is proportional to its displacement, Ref. [25] simplifies the equations by using piecewise-constants to approximate the spring force. We call this model ***Piecewise Model***, which has been recently implemented in a storage simulator named Memulator [28]. Ref. [26] proposes a ***String Model*** that removes the approximation and precisely models the continuously changing force from the strings. Ref. [27] further improves the string model by incorporating an optimal control policy on the actuator system as Equation 3.

$$m\frac{d^2x(t)}{dt^2} + \lambda\frac{dx(t)}{dt} + kx(t) = F(t) \quad (3)$$

where $x(t)$ is the position of the sled at time t, m is the mass of the sled, λ is the damping coefficient and k is the coefficient of the elasticity of the restoring force from springs, and $F(t)$ is the external force created by the MEMS structures with their controllers. Ref. [24] follows the second approach and we call it ***Statistical Model***. In this model, t_{seek} is computed as the weighted average of the mean seek time in X and Y, shown in Equation 4:

$$t_{seek} = a \times t_{seek\_x} + b \times t_{seek\_y} \qquad (4)$$

where $a(b)$ is the probability that a seek time in X(Y) is greater than the one in Y(X). The values of *a* and *b* can be estimated from traces.

3.1.3 Power Conservation

Minimizing power dissipation in storage systems is an important issue, especially for mobile computing. It is expected that MEMS storage will consume much less power than disks, which consume 20–54% of the total energy [29][30]. Efforts to reduce energy in the MEMS devices can be made at both the device level and the system level. We focus on the device level here and Section 3.3.2 will investigate relevant research at the system level.

At the device level, the basic design is to provide different power modes. IBM microdrives have five operational modes that turn off various amount of hardware resource to achieve different tradeoffs between performance and power dissipation [31]. These disk drives judiciously make transitions between the various modes based on the history of the command burst, the command history, the desired performance level, and the energy costs in each mode. Similarly, MEMS storage devices can be designed with multiple power modes. Ref. [21] and [32] take a simple approach and assume that MEMS storage devices have two modes: an active mode and an idle mode. Their devices aggressively reduce power consumption by switching into the idle mode whenever the I/O queue is empty. This greedy scheme is justified by the fact that small latency and energy penalty of a mis-switch is paid on future requests in MEMS storage.

3.2. Architecture Level

From the architecture point of view, the current research on MEMS storage focuses on how to use it in a cost-effective way. There are huge gaps in performance, price and reliability between NVRAM and disks. The NVRAM-to-disk latency gap has already reached 6 orders of magnitude (10 *ms vs* 50 *ns*) and continues to expand by around 50% per year [1]. In terms of price per byte, NVRAM costs nearly 1000 times more than disks [7]. In addition, the mean-time-to-failure of NVRAM is only about 15K hours while disks have already achieved a million hours.

MEMS storage will offer a set of characteristics to narrow the above gaps. The possible placements of MEMS storage in the memory hierarchy are (1) working beneath RAM and replacing disks, (2) working as buffer and cache between RAM and disks; (3) replacing all disks in RAIDs; (4) working in parallel with a disk drive; (5) replacing partial disks in a conventional disk array. Table 2 presents the performance and cost comparisons among these placements, measured by running actual file system traces over simulators. Although MEMS storage is not likely to replace disks completely due to the capacity and price constraints, the research of the first placement, replacing disks, is still very important for certain volume or power sensitive applications, such as embedded systems. In the other placements, MEMS storage can potentially improve the I/O performance significantly, even under the iso-price constraint.

3.3 System Level

During the past 30 years, the designs of operating systems have continuously been tuned to alleviate the I/O bottleneck according to the disk physical characteristics. The advent of MEMS storage makes it a necessity to revisit the I/O management functions in current operating systems, such as I/O scheduling, power management, data management, performance optimization and fault tolerance design.

3.3.1 I/O scheduling

Disk I/O scheduling is a mature research field where many disk scheduling algorithms, such as First-Come-First-Served (FCFS), Circular Look (C-LOOK), Shortest Seek Time First (SSTF), and Shortest Positioning Time First (SPTF), have been extensively studied. Since the physical characteristics of MEMS storage are very different from those of disks, the conventional disk-type algorithms are reevaluated [21] and many new scheduling algorithms are proposed [36] [37] [38].

MEMS storage has two data allocation strategies: ***one-dimensional allocation*** and ***two-dimensional allocation***. With the first strategy, data is placed contiguously along the cylinders and in an I/O access the read/write head moves along a cylinder until the cylinder ends, similar to the disk allocation. With the second strategy, data is placed in equal-distance regions that may cross over to other cylinders; the read/write head stays in one region in a single I/O operation but may visit different cylinders.

Architecture	Approach	Salient Features	Performance

Working as disk write buffer [15]	Runs Piecewise Model on DiskSim [33]	MEMS and disk are placed behind RAM; MEMS handles hit-reads and all writes from RAM while disk handles missed-reads and writes from MEMS.	Achieve hundreds MB/s throughput and eliminate 96% raw write traffic to disks.
Replacing a disk drive with a MEMS device [24]	Runs String Model on DiskSim	MEMS has 320 active tips and bit-wise striping is used.	Average service time ranges 0.5–1 ms, 10 times better than disks.
Replacing a disk drive with a MEMS device [10]	Incorporate DiskSim-based MEMS into SimOS machine simulator [34]	Real application benchmarks are replayed at virtual Alpha 21164 system equipped on SimOS with simulated MEMS device.	Benchmarks complete 3 times faster than state-of-art disk and 2 times faster than future disks in 2005.
Replacing a small NVRAM with a larger MEMS device in disk array [7]	Runs Piecewise Model on Pantheon [35]	MEMS devices are organized as RAID 5 and are used to replace the small-capacity NVRAM.	Reduce response time up to 82% but do not improve throughput. The performance ratio improves 2.1-4.2.
Replacing disk arrays with a bank of MEMS storage devices [7]	Runs Piecewise Model on Pantheon	Under various MEMS-disk price ratios, both iso-capacity and iso-price scenarios are compared.	Improve latency by a factor of 4 -6.5 and throughput by a factor of 4-28. Performance/cost ratio is 2-7 times of disks.
Putting both a bank of MEMS devices in parallel with a log disk [7]	Runs Piecewise Model on Pantheon	MEMS devices are organized as RAID 0. Updates to MEMS array are mirrored in disks as in a log structure fashion.	Offers most consistent performance improvement and best performance/cost ratio. Reduce the latency of disk arrays to 4-16.
Putting a bank of MEMS devices in parallel with disk arrays [7]	Runs Piecewise Model on Pantheon	Use RAID-10 layout. MEMS and disk arrays are both organized as RAID 0 but disk array has larger stripe sizes. Both arrays are mirrored.	No significant improvement in throughput and latency. Performance/cost ratio to conventional arrays is 1.5-0.8

Table 2: Comparisons of different usages of MEMS storage devices in computer systems.

The existing disk-type scheduling algorithms in MEMS storage with the one-dimensional allocation strategy are evaluated in Ref. [21]. It is found that the straightforward adaptations of disk scheduling algorithm into MEMS storage are appropriate, and that SPTF has the lowest average but high variation in response time. In contrast, FCFS has the least variation in response times while exhibiting the highest mean.

For MEMS storage devices with the two-dimensional allocation strategy, some new scheduling algorithms are proposed. While the optimal scheduling in the 2-D space is a NP-complete problem, Ref. [36] and [37] propose an approximation algorithm by constructing the request destinations into a dynamic minimum spanning tree and then serving the requests in the preorder traversal of the tree. They show that this algorithm has a smaller response time than all conventional algorithms under the workload with uniform distribution. Ref. [38] divides the MEMS storage media into a set of zones based on seek time equivalence regions and serves these zones in the C-SCAN order. Within each zone, requests are served in the SPTF order. Their simulation results indicate that this algorithm reduce the variation of response times close to

C-SCAN but has an average response time comparable to SPFT.

3.3.2 Power Conservation

Power conservation can be performed both at the device level and the system level. We have discussed the design of power conservation at the device level in the previous section. At the system level, many useful power conservation strategies for disk drives, such as caching [29], and combination with NVRAM [39], are applicable to MEMS storage to save some seek operations by absorbing a certain amount of write and read requests. According to the physical characteristics of MEMS storage, Ref. [32] proposed two system-level strategies: request merging and subsector access. The first strategy is to merge multiple adjacent requests into a large one and exploit the parallelism to serve this large request in one seek process. The second strategy allows access to partial instead of whole tip sector. However, partial access will lead to the difficulty of cache alignment, a problem not addressed by the authors.

3.3.3 Fault Tolerance

Hardware failure: The MEMS storage devices have thousands of tips and thus there is a high probability of hardware failure. The fault tolerance can be designed at two levels: the ***intra-device*** level and the ***inter-device*** level. Inside a MEMS storage device, data is striped across multiple read/write tips to achieve high aggregate performance. If the error correction code is computed both horizontally and vertically, the missing data due to the media defection or tip failure can be possibly reconstructed. At the inter-device level, MEMS storage devices can be organized as an array and provide RAID style redundancy to tolerate device failure.

MEMS arrays have a faster read-modify-write operation than disk arrays [21]. Read-modify-write is a common operation in RAID structures. For example, a small write in RAID 5 needs two read-modify-write operations: reading the old data and old disparity, then calculating the new disparity and finally writing the new data and new disparity. Read-modify-write in disks suffers one full resolution delay waiting for disk spindles to spin back to the same position while MEMS devices can immediately reverse the direction and significantly reduces the latency.

Software failure: System crashes can generate data inconsistency in file or database systems. To minimize such software state failure, synchronous write is usually used. Due to large latency of synchronous operations on hard drives, systems seldom have enough time to dump all the data in RAM into disks. The low service time in MEMS storage certainly helps reduce software failures during crash. To our best knowledge, no literatures have studied this advantage in details.

3.4. Application Level

The utilizations of MEMS storage in real applications have not been extensively investigated. With the commercial release of these devices in 2004, we expect that more research in the area will show up soon. The following lists three projects that use MEMS storage devices for their specific purposes.

Secure Storage: University of Twente designs MEMS-based storage as “write once disks” for secure storage by exploiting some irreversible process of MEMS storage on patterned magnetic media [40].

Database on MEMS storage: Relational databases usually serialize tabular data onto disks in either the row-major order or the column-major order. However, data is often accessed with an order different from the order in which data is stored. As a result, database applications frequently experience very inefficient I/O service provided by disks. Ref. [41] designs a special layout in MEMS storage for relational databases to allow both efficient row-major and column-major access.

Metadata storage: Ref. [42] examines the efficiency of a scheme where a MEMS storage device is used to store the file system metadata while actual file content resides in disks. They show that this scheme improves the system performance by 28-46%.

4. Discussion of Open Issues

MEMS-based storage is a young technology and there are many research issues that need to be addressed. While the research work presented above is incomplete and immature, we notice that little have been addressed regarding the following five issues.

4.1 Prefetching

Although MEMS storage responds 10 times faster than disk drives, it is still much slower than RAM and processors. Prefetching is an important technique to further alleviate the I/O bottleneck. Unfortunately little attention has been paid to the prefetching issue in MEMS storage. In current disk technology, prefetching is performed at two levels. (1) At the device level, if the firmware detects a read request to be part of some large sequential request, disk drives will continue to fetch a few contiguous blocks in hope to saving future disk accesses. (2) Simultaneously, at the system level, the OS can actively issue prefetching requests by observing the I/O history or by speculatively continuing to execute the code. While both approaches can be applied to the MEMS storage, prefetching in MEMS storage can also be designed in a different way. Recall that MEMS storage has many read/write tips that can be simultaneously active and not all tips are busy in every I/O access. Thus it provides an opportunity for idle tips to prefetch some data while the media moves beneath them. This prefetching is totally free since all tips can work in parallel while penalty is paid in disk drives because prefetching delays the response to other outstanding requests. When the I/O queue is empty, MEMS storage can also prefetch some data at the same media sled position, but from a different tip set than the current active tips, by simply switching to the corresponding tip set without moving the media sled. No seeks are involved in this form of prefetching. The challenge in the new prefetching design is to design a good data clustering strategy to achieve a high prefetch hit ratio.. A possible clustering solution is to examine the I/O history and place data with the highest probabilities of sequential access or with the highest access frequencies under the same set of active tips.

4.2 Reversal Access

We can further modify the design of current MEMS storage to reduce the latency. In disks, data is always read by following the spinning direction. If the destination block passes the disk head, disk has to wait one full resolution to read it since the overhead of reversing the spinning direction is very large. In the current MEMS design, the media can quickly reverse its moving direction and thus the response time is much smaller than that of disks. However, data is read in the direction from its start bit to its end bit in most designs. When data is immediately reassessed, the read/write heads need to traverse the data region to the start bit and then access the data. If MEMS storage allows accessing data in both directions, i.e., data can also be visited in the direction from its end bit to its start bit and the bits are reversed electronically in the firmware cache, then the traversal with a physical length of the re-accessed data on the media can be saved. Since immediate re-access is a frequent pattern, especially in RAID workload [7], the reversal access may have significant improvement on the average latency. The main disadvantage of reversal access is that it complicates the performance modeling and request scheduling as well as hardware design.

4.3 Fault Tolerance

With thousands of read/write tips in one MEMS storage device, fault tolerance is a critical issue for its ultimate market acceptance. To our best knowledge, only CMU has addressed this issue and they used two-dimensional ECC and spare tips to tolerate permanent tip or media failures. However, their devices cannot tolerate two failures in the same row or column. We may pursue a more aggressive fault tolerance scheme, such as mirroring, Information Dispersal Algorithm [44], and RAID 10, to improve the reliability. Then come the questions that computer scientists and engineers need to tackle. What is the mean-time-to-failure of a MEMS storage device in a real environment where all failures are ***NOT*** independent but highly correlated? What is the impact of different redundancy schemes on the throughput and latency? How to maintain the consistency and coherence between redundant data as well as between metadata and data? How to device data allocation to reduce the performance degradation caused by failure components? Additionally, unlike disk arrays where failed disks can be repaired or replaced, the failed tips and media regions cannot be restored in MEMS storage devices. Then how to design a more resilient storage system on these devices that maintains a reasonably “healthy level”?

Another important issue related to fault tolerance is how to efficiently utilize the redundant data to improve the I/O performance. One may approach this problem from two perspectives: avoiding hot spots by sharing loads among tips above redundant data or using a special data placement to reduce the average seek distance.

4.4 Power Consumption Modeling

To study and reduce power dissipation, an efficient and accurate power consumption model is desperately needed since the physical experiments involve a frustrating process, running long traces on real devices with power measurement equipments. After MEMS storage devices are commercially available in 2004 [7], a detailed energy simulator for these devices can be built by measuring the average amount of energy spent in each operation stage. This modeling is not trivial since the mechanical motions in MEMS storage are very complicated.

A power consumption model certainly facilitates the energy aware design. For example, although a MEMS storage device has thousands of read/write tips, only hundreds of them can be active simultaneously due to the power consumption and cooling issues. Additionally, Ref. [23] shows that the performance is proportional to the number of active tips within a certain range. With a power model available, one can design a scheme to dynamically change the number of active tips to optimize the tradeoff between performance and energy saving.

4.5 Reevaluation of Some Disk Optimization Techniques

The I/O management system is continually optimized to reduce the performance gap between disk and processors. While the physical distance between destinations and the current read/write head position determines the access time in both MEMS storage and disks, their access processes and physical characteristics are very different. Accordingly, the efficiency of disk-type optimization algorithms needs to be reexamined. For example, recently Wang *et al* proposes ***eager writing*** [45] that writes new data to free sectors near the disk head’s current location and reorganizes them later. While eager writing reduces the latency of small synchronous writes to disks, it also introduces the overhead of fragmentation management and reorganization. In MEMS storage, the overhead may not justify the adoption of this scheme since MEMS storage has a much smaller seek time.

5. Conclusions

In this study, we discussed MEMS-based storage, a new storage technology with a set of exciting characteristics that bridge the performance and cost gaps between RAM and disks and summarized the current research in this promising technology from four levels: (1) the device level that focuses on the understanding of physical characteristics and designing of interfaces; (2) the architecture level that endeavors to integrate the new devices into the current memory hierarchy in the most cost-effective way; (3) the system level that tries to tune the I/O management subsystem of operating systems to integrate the new devices; and (4) the application level that focuses on how to use MEMS storage efficiently for specific applications.

This paper identified five potential research issues regarding this young storage technology, namely, (1) prefetching, (2) reversal access, (3) fault tolerance, (4) power consumption modeling, and (5) reevaluation of some existing I/O optimization algorithms. We proposed to exploit the parallelism of read/write tips to prefetch data in MEMS storage and design a reversal access strategy to shorten seek distance along data tracks. We suggested using high-degree of fault tolerance design to build an autonomic, resilient storage and provided a possible approach to model power consumption. In the end, we pointed out the necessity of reevaluating some existing I/O optimization algorithms since they were designed for disks and could be detrimental to the performance of MEMS storage.

Acknowledgement

This project was partially supported by a National Science Foundation grant (EPS-0091900), a Nebraska University Foundation grant, and a UNL Academic Priority Grant, for which we are grateful. We would also like to thank other members of the secure distributed information (SDI) group [46] and the research computing facility [47] at the University of Nebraska-Lincoln for their continuous assistance and support.

References

[1] E. Grochowski, IBM leadership in disk storage technology, *IBM Corporation*, 2000.

[2] E. Grochowski, "Future trends in hard disk drives," *IEEE Trans. on Magnetics*, vol. 32, no. 3, pp. 1850-1854, May 1996.

[3] D. A. Thompson and J. S. Best, "The future of magnetic data storage technology," *IBM Journal of Research and Development*, 44(3), May 2000.

[4] L. R. Carley, G. R. Ganger, D. F. Guillou, and D. Nagle, "System design considerations for MEMS-actuated magnetic-probe-based mass storage," *IEEE Trans. on Magnetics*, 37(2 Part 1):657-662, Mar 2001.

[5] P. Vettiger, M. Despont, U. Drechsler, U. Durig, W. Haberle, M. I. Lutwyche, H. E. Rothuizen, R. Stutz, R. Widmer, and G. K. Binnig, "The 'millipede' - more than one thousand tips for future AFM data storage," *IBM Journal of Research and Development*, 44(3):323-340, 2000.

[6] G. Marsh, "Data storage gets to the point," *Materials Today*, Feb 2003.

[7] R. Rangaswami, Z. Dimitrijevic, E. Chang, and K. E. Schauser, "MEMS-based disk buffer for streaming media servers," *IEEE International Conference on Data Engineering*, Bangalore, March 2003.

[8] M. Uysal, A. Merchant, and G. Alvarez, "Using MEMS-based storage in disk arrays," in *Proc. of 2nd USENIX Conference on File and Storage Technologies (FAST)*, April 2003 (Best Award Paper).

[9] K. D. Wise, "Special issue on integrated sensors, microactuators, and microsystems (MEMS)," *Proceedings of the IEEE*, 86(8):1531-1787, August 1998.

[10] J. L. Griffin, S.W. Schlosser, G. R. Ganger, and D. F. Nagle, "Modeling and performance of MEMS-based storage devices," in *International Conference on Measurements and Modeling of Computer Systems (SIGMETRICS)*, pp. 56-65, Santa Clara, California, June 2000.

[11] S. W. Schlosser, J. Schindler, A. Ailamaki, and G. R. Ganger, "Exposing and Exploiting Internal Parallelism in MEMS-based Storage," *Carnegie Mellon University Technical Report CMU-CS-03-125*, March 2003.

[12] Z.N.J. Peterson, S.A. Brandt and D.D.E. Long, "Data placement based on the seek time analysis of a MEMS-based storage device," *A Work in Progress (WIP) at: the Conference on File and Storage Technologies (FAST), USENIX*, 2002.

[13] S. W. Schlosser, J. L. Griffin, D. F. Nagle, and G. R. Ganger, "Designing computer systems with MEMS-based storage," in *9th Int. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS-IX)*, pp. 12-15, Cambridge, Massachusetts, Nov 2000. Published as Operating Systems Review, 34(5):1–12, 2000.

[14] J. F. Alfaro and G. K. Fedder, "Actuation for probe-based mass data storage," in *Proceedings of the 2002 International Conference on Modeling and Simulation of Microsystems*, 2002, Vol. 1, pp. 202-205.

[15] F. Wang and S. Brandt, "Using MEMS device as disk write buffer," *Technical Report*, Department of Computer Science, University of California Santa Cruz, 2001.

[16] Sunny Bains, "An Interview with Rick Carley," *EE Times*.

[17] C. Ruemmler and J. Wilkes, "Disk shuffling," *Technical Report HPL-91-156*. Hewlett-Packard Company, Palo Alto, CA, October 1991.

[18] P. Vongsathorn and S. D. Carson, "A system for adaptive disk rearrangement," *Software Practice and Experience*, 20(3):225-242, March 1990.

[19] A. Vakali and Y. Manolopoulos, "Replication in mirrored disks systems," in *Proceedings of the 2nd East-European Conference on Advanced Databases and Information Systems (ADBIS 98)*, Spring-Verlag, Lecture Notes in Computer Science, #1475, pp. 224-235, 1998.

[20] K. A. Hua, S. D. Lang, and W. K. Lee, "A decomposition-based simulated annealing technique for data clustering," in *Proceedings of the thirteenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pp. 117-128, 1994.

[21] J. L. Griffin, S. W. Schlosser, G. R. Ganger, and D. F. Nagle, "Operating system management of MEMS-based storage devices," in *4th Symp. on Operating Systems Design & Implementation (OSDI)*, pp. 227-242, San Diego, California, Oct 2000.

[22] E. Eleftheriou, T. Antonakopoulos, G. K. Binnig, G. Cherubini, M. Despont, A. Dholakia, U. Dürig, M. A. Lantz, H. Pozidis, H. E. Rothuizen, and P. Vettiger, "Millipede - A MEMS-based scanning-

probe data-storage system," IEEE Transactions on Magnetics, Vol. 39, No. 2, March 2003, pp. 938-945.

[23] M. Sivan-Zimet and T. M. Madhyastha, “Workload based optimization of probe-based storage,” *ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems* (Marina Del Rey, CA, 2002). Published as *ACM SIGMETRICS Performance Evaluation Review*, 30(1):256–257. ACM Press, 2002.

[24] I. Dramaliev and T.M. Madhyastha, "Optimizing probe-based storage," in *2nd USENIX Conference on File and Storage Technologies (FAST),* San Francisco, CA, Mar-Apr 2003.

[25] S. Schlosser, J. Griffin, D. Nagle, and G. Ganger, “Filling the memory access gap: A case for on-chip magnetic storage,” *Technical Report CMU-CS-99-174, Carnegie Mellon University School of Computer Science*, November 1999.

[26] T. M. Madhyastha and K. P. Yang, "Physical modeling of probe-based storage," in *18th IEEE Symp. on Mass Storage Systems*, pp. 207-224, San Diego, California, Apr 2001.

[27] Pu Yang, “Modeling probe-based data storage devices,” *Technical report.* Department of Computer Science, University of California Santa Cruz, June 2000. Master's thesis.

[28] J. L. Griffin, J. Schindler, and S. W. Schlosser, "Timing-accurate storage emulation," in *Proceedings of the Conference on File and Storage Technologies (FAST)*, January 28-30, 2002. Monterey, CA.

[29] F. Douglis, R. Caceres, F. Kaashoek, K. Li, B. Marsh, and J. A. Tauber, "Storage alternatives for mobile computers," in *Proceedings of the 1st Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 25–37, Monterey, CA, Nov. 1994.

[30] K. Li, R. Kumpf, P. Horton, and T. Anderson, "A quantitative analysis of disk drive power management in portable computers," in *Proceedings of the USENIX Technical Conference*, pp 279–291, San Francisco, CA, Jan. 1994.

[31] IBM, Adaptive Power Management for Mobile Hard Drives, *IBM Technical Report*, 1999.

[32] Y. Lin, S. Brandt, D. Long, and E. Miller, "Power conservation strategies for MEMS-based storage devices," *International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*, FortWorth, TX, October 2002.

[33] G. Ganger, B. Worthington, and Y. Patt, "The DiskSim Simulation Environment Version 1.0 Reference Manual," *Technical Report CSE-TR-358-98*, The University of Michigan, Ann Arbor, Feb. 1998.

[34] M. Rosenblum, S. Herrod, E. Witchel, and A. Gupta, "Complete Computer System Simulation: The SimOS Approach," *IEEE Parallel & Distributed Technology*, 3(4), Winter 1995.

[35] J. Wilkes, "The Pantheon storage-system simulator," *Tech. Rep. HPL-SSP-95-14,* Storage Systems Program, Computer Systems Laboratory, Hewlett-Packard Laboratories, Palo Alto, CA, May 1996.

[36] H. Yu, D. Agrawal, and A. E. Abbadi, "Towards the integration of MEMS-based storage in computing systems," *Business Briefing: Data Management and Storage Technology*, 2003.

[37] H. Yu, D. Agrawal, and A. E. Abbadi, "Towards optimal I/O scheduling for MEMS-based storage," in *Twentieth IEEE/Eleventh NASA Goddard Conference on Mass Storage Systems & Technologies (MSST 03),* April 2003, San Diego.

[38] B. Hong, S. A. Brandt, D. D. E. Long, E. L. Miller, K. A. Glocer, and Z. N. J. Peterson, “Zone-based shortest positioning time first scheduling for MEMS-based storage devices,” in *Proceedings of the 11th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '03),* Orlando, FL, 2003.

[39] M. Wu and W. Zwaenepoel, "eNVy: a non-volatile, main memory storage system," in *Proceedings of the 6th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS),* pp. 86–97. ACM, Oct. 1994.

[40] L. Abelmann, C. N. Chong, P. H. Hartel, and C. Lodder, "Design rationale for secure probe storage based on patterned magnetic media," *Technical report TRCTIT-02-42*, Centre for Telematics and Information Technology, Univ. of Twente, The Netherlands, Oct 2002.

[41] H. Yu, D. Agrawal, and A. E. Abbadi, "Tabular placement of relational data on MEMS-based storage devices," *VLDB*, Sept 2003, Berlin Germany.

[42] B. Hong, “Exploring the Usage of MEMS-based Storage as Metadata Storage and Disk Cache in Storage Hierarchy,” *Technical Report*, Department of Computer Science, University of California Santa Cruz. 2003.

[43] L. R. Carley, G. R. Ganger, and D. F. Nagle, "MEMS-based integrated-circuit mass-storage systems," *Communications of the ACM*, November 2000, Vol. 43, No. 11.

[44] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *Journal of the ACM,* vol. 36, no. 2, pp. 335-348, Apr. 1989.

[45] R. Y. Wang, D. A. Patterson, and T. E. Anderson, "Virtual log based file systems for a programmable disk," *Symposium on Operating Systems Design and Implementation,* pp. 29–43. ACM, 1999.

[46] Secure Distributed Information (SDI), http://rcf.unl.edu/~sdi/index.php,

[47] Research Computing Facility (RCF), http://rcf.unl.edu/