I/O Performance Optimization Techniques for Hybrid Hard Disk-Based Mobile Consumer Devices

Young-Jin Kim, *Student Member*, IEEE, Sung-Jin Lee, Kangwon Zhang, Jihong Kim, *Member*, IEEE

Abstract — *A hybrid hard disk employs the advantages of both a hard disk and a NAND flash memory, thus making it a cost-effective fast secondary storage device. In this paper, we improve its I/O performance by combining an intelligent data pinning policy for the flash memory with a caching technique which is aware of access patterns for the flash memory and DRAM. Our proposed techniques reduce the system boot time and application launching time while reducing energy consumption, which is vital in the mobile devices. We built SimHybrid, a flexible trace-driven hybrid hard disk evaluation environment, and used it to demonstrate how a hybrid hard disk can achieve significantly better I/O performance than a traditional hard disk while using much less energy*[1].

Index Terms — hybrid hard disk, I/O performance optimization, boot time, application launching time, energy consumption

I. Introduction

Due to recent innovations, NAND flash memory is now the secondary storage device that competes most closely with hard disks. As a mass storage device, NAND flash memory has many advantages over hard disks such as smaller size, lower-power consumption, and higher shock resistance [1]. The popularity of mobile systems (such as PDAs, PMPs, and MP3 players) which use NAND flash memory as their secondary storage device (e.g., the iPod Nano [2]) reflects these technical advantages of the NAND flash memory.

However, hard disks still have advantages over NAND flash memory in terms of cost and OS support. Additionally, a high storage demand of multimedia devices such as MPEG players favors the use of hard disks as secondary storage because of their higher capacity. Hard disks also have good performance in processing sequential I/Os.

The complementary features of the NAND flash memory and hard disks have motivated several proposals on hybrid storage devices by combining the disks and flash memory. The main goal of the hybrid storage devices is to improve the I/O performance of the storage subsystem with a higher energy efficiency.

These proposals can be grouped in two categories: 1) ones using a flash as a non-volatile cache (NVC) [3, 4, 5, 6], which stores data blocks which are likely to be accessed in the near future, allowing the disk to spin down for longer periods, and 2) the others using flash as a component of heterogeneous secondary storage with data concentration techniques which minimize slower disk traffics [7].

A hybrid hard disk [8, 9, 10], which uses a flash memory as an onboard NVC, is the most well-known example of the first category above. By employing both a flash memory (as an NVC) and a DRAM (as a cache) in addition to a normal hard disk, a hybrid hard disk improves the energy consumption, performance, and reliability over a system with a single hard disk (hereafter called a legacy disk). Because of these characteristics, hybrid hard disks are attractive solutions for meeting major design requirements of mobile consumer devices.

For example, mobile consumer devices are often expected to be instantly available after the power is on, so reducing the boot time is a very important objective. Since the system start-up involves a system boot (after a power-on self test (POST)) by loading a boot sector, if a system is disk-based, the system boot might be delayed by the drive ready time of its disk. In typical mobile disks, reaching to the 'drive ready' state may take about 4 seconds [11]. Furthermore, the processing of boot data incurs many random I/Os as well as sequential ones. (For example, random I/Os occur from meta data updates and sequential ones from loading a boot image.) A disk with a relatively large seek time is ill-suited for random I/Os. By directing random I/Os to a NAND flash memory while sequential I/Os to the disk, a hybrid hard disk can reduce the boot time significantly.

Hybrid hard disks can be also useful in reducing the application launching time. Because of the increasing application complexity, a fast application launching is an important requirement for mobile embedded devices. For example, launching the Windows Media Player on a laptop PC can take up to 10 seconds. Even longer delays, which many users consider unacceptable, may be possible on small mobile devices with limited resources. Pinning time-consuming application-related data into a flash memory can be useful in reducing the application launching time in mobile devices.

Hybrid hard disks can be also effective in reducing the energy consumption of mobile systems, thus extending the battery lifetime. Since hard disks are significant energy

[1] This work was supported by Samsung Electronics Co., Ltd. It was also supported by the Korea Science and Engineering Foundation (KOSEF) through the National Research Lab. Program funded by the Ministry of Science and Technology (No. R0A-2007-000-20116-0). The ICT at Seoul National University provided research facilities for this study.

Young-Jin Kim, Sung-Jin Lee, and Kangwon Zhang are all with the School of Computer Science and Engineering, Seoul National University, 599 Gwanangno, Gwanak-gu, Seoul 151-742, Korea (e-mail: {youngjk, chamdoo, kwzhang} @davinci.snu.ac.kr).

Jihong Kim (corresponding author) is with the School of Computer Science and Engineering, Seoul National University, 599 Gwanangno, Gwanak-gu, Seoul 151-742, Korea (phone: +82-2-880-8792; fax: +82-2-871-4912; e-mail: jihong@davinci.snu.ac.kr)

Contributed Paper
Manuscript received October 15, 2007
0098 3063/07/$20.00 © 2007 IEEE

consumers, redirecting some I/O requests to a flash memory is likely to result in a significant energy saving by putting the disk longer in a low-power mode.

The potential benefits of a hybrid hard disk over a legacy disk for mobile consumer devices, however, can be realized only when several components of a hybrid hard disk operate properly and interact efficiently. These components, shown in Fig. 1, are the disk controller, NVC, DRAM cache, and hard disk itself, all of which are controlled by commands from the host OS through a host interface. More specifically, in order to realize the high I/O performance out of a hybrid hard disk with a lower energy consumption, we need to answer the following two questions:

1) How can a host OS determine which data are valuable, and when should the OS pin the important data into the flash memory?

2) How should the NVC and DRAM cache be managed when both pinned data and unpinned data (i.e., data which are directed to the NVC from normal I/O requests) are mixed?

In this paper, we propose novel I/O performance optimization techniques which provide efficient solutions for these questions. We propose an intelligent pinning policy for the NVC that can be used by the host OS, and develop an access pattern-aware cache management algorithm for the NVC and DRAM cache. We evaluate our proposed techniques using a trace-driven extensive hybrid hard disk evaluation environment. By using I/O traces collected under the Windows XP OS, we show how our optimization techniques can be combined to improve the system boot time, application launching time, and energy consumption.

The remainder of this paper is organized as follows: Section II presents an overview of hybrid hard disks. Section III describes our I/O performance optimization techniques. Section IV covers the hybrid hard disk system simulator. Section V explains how we generate I/O traces while Section VI presents the evaluation results. Section VII describes related work and Section VIII concludes with a summary.

II. Overview of Hybrid Hard Disk

A hybrid hard disk employs an NVC as well as a DRAM cache to maximize the performance of read and write operations. An I/O controller receives SATA commands through a host interface and manages the NVC and the DRAM cache as well as the disk. It has been suggested [10] that the benefits to be expected from a hybrid hard disk are: up to 90% power saving when the disk is powered down; instant read and write operations even when the disk is not rotating; instant read operations even while the disk is spinning for higher I/O rate.

A hybrid hard disk supports non-volatile caching as well as data pinning, unlike traditional caches. This is because a hybrid hard disk is designed to deal with both pinned and unpinned data, which are stored in separate regions of the NVC: the host operating system controls the pinned data while the disk itself controls the unpinned data. New SATA commands to manage pinned data are being standardized in ATA8-ACS [12]. Boot data, resume data, and frequently used applications and files are good candidates to be allocated to the pinned region [8]. The unpinned region is used, together with the DRAM, to cache the data of normal SATA commands.

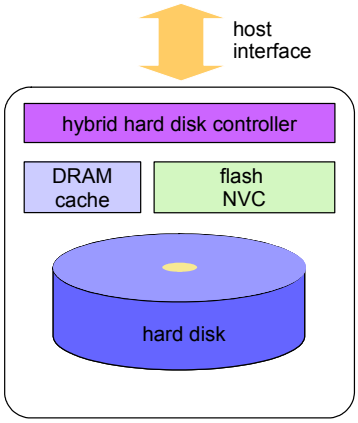

Fig. 1. Overall structure of a hybrid hard disk.

Since the overall system performance is affected by both the data pinning policy and the way caching is shared between the NVC and the DRAM cache, it is necessary to develop these techniques in order to improve I/O performance and to save energy. We aim to develop I/O performance optimization techniques for hybrid hard disks which are appropriate for mobile consumer devices. To evaluate these techniques, we have built a hybrid hard disk system simulator from scratch; it fully emulates the functions of both the host operating system and a hybrid hard disk system.

III. I/O Performance Optimization Techniques for Hybrid Hard Disk Systems

A. Data pinning policy

We consider three candidate data types in developing a data pinning policy over the NVC. Boot data, frequently used data, and favorite user applications, when properly pinned, have a high potential to improve the I/O performance significantly. Fig. 2 shows the management process of the frequently used data case. (For the other data types, we will explain the management policies in Section VI.) As shown in Fig. 2, the proposed pinning policy is based on the access frequencies of logical block address (LBA) groups, which are recorded in an LBA group history and a data map. An LBA group consists of a start LBA, a range (i.e., the I/O size), an age (i.e., the time elapsed since an LBA was accessed), and an access frequency. The aim of this policy is to improve I/O performance by pinning frequently used data into the pinned region in the NVC, which directs the data to the pinned region instead of forwarding it to the disk.

The data pinning policy within the host OS operates as follows:

1) At every I/O request, the number of processed requests is checked to see whether it is larger than N.

2) If an LBA group is accessed its access frequency is increased by one. At every I/O request, the history of accessed LBA groups is updated.

3) If the number becomes larger than *N*, frequently accessed LBA groups are selected after investigating the history. The data map is then updated with these selected LBA groups. Frequently accessed LBA groups are identified by using a weighted average of threshold values for the current and previous periods.

4) If some LBA groups already exist, only their ages and frequencies are updated. Otherwise, the host OS issues SATA commands to add new pinned data (the LBA groups) to the NVC, and to evict aged pinned data from the NVC, if the pinned region is becoming full. The data map also is updated.

B. Access pattern-aware cache management technique

Our algorithm seeks to improve I/O performance by managing the DRAM cache and the unpinned region of the NVC to suit the I/O access patterns, because the relative I/O performances of a disk and an NVC depend on the I/O access patterns, as described previously.

As shown in Fig. 3, our algorithm uses an active and an inactive LRU list to manage the DRAM cache, a mechanism that is similar to page caching in the Linux kernel. Blocks with higher locality are put in the active list and marked at hit, while those with lower locality stay in the inactive list and will be evicted sooner to the NVC or disk. The unpinned region of the NVC is managed with a traditional LRU policy.

When a write arrives at the hybrid disk controller, the algorithm checks whether its access pattern is sequential or not and determines whether to write through the block to the NVC or the disk. Due to the write through policy, the DRAM cache only contains clean blocks. If the write access is random, it will be inserted into the MRU position of the unpinned region of the NVC. Otherwise, the write will be forwarded to the disk. This policy can be expected to ameliorate the poor performance of a disk for random accesses. If the disk is already spun down, the block will be flushed to the NVC regardless of access patterns, with the aim of saving as much energy as possible. When a dirty block is evicted from the NVC, it will be flushed to the disk. But if it is clean it will be discarded.

IV. SIMHYBRID: HYBRID HARD DISK SIMULATOR

Our hybrid hard disk simulation system consists of two parts: 1) A trace-driven hybrid hard disk simulator called SimHybrid, shown in Fig. 4, which models an NVC, a DRAM, a disk, and a hybrid hard disk controller in terms of I/O response time and energy consumption. 2) A host simulator, which mimics the host operating system by monitoring I/O behavior, managing pinned data, and issuing SATA commands to SimHybrid based on real traces.

A. SimFlash: Flash memory simulator

Our NVC model is based on a flash memory simulator called SimFlash, shown in the red dotted region of Fig. 4, which simulates a NAND flash memory and its controller. It

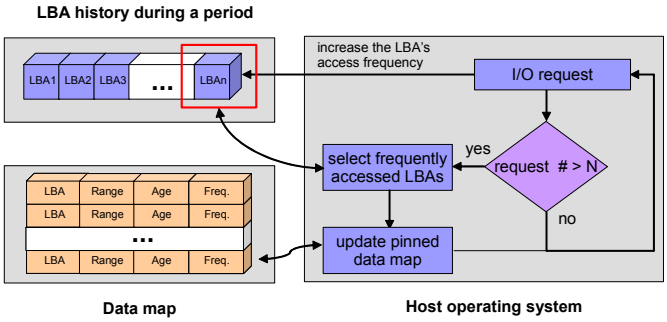

Fig. 2. Operations of a dynamic data pinning policy.

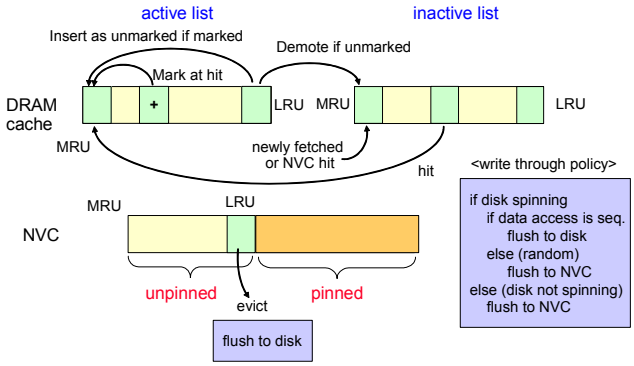

Fig. 3. Proposed cache management algorithm: 2-level cache management with an access pattern-aware flush policy.

includes a flash translation layer (FTL), which consists of an address mapper, a reclamation manager, and a wear leveler. We have implemented page-level, block-level, and hybrid address mapping policies [13]; and greedy and cost-benefit block reclamation policies with hot and cold data separation. We also embodied a swapping policy for wear leveling.

The default flash model is the K9K8G08U0M [14], which is a 1G x 8 bit NAND flash with a 2 KB page size and a 128 KB block size. In experiments, we assigned the capacity of the flash memory to 256 MBs. Using datasheet parameters except for the capacity, we built performance and energy models of the K9K8G08U0M, but parameters of other flash memories can be modeled easily.

B. Hybrid hard disk controller and disk simulator

The hybrid hard disk controller consists of two modules: a host interface and a cache manager. The host interface decodes and processes SATA commands, whether they relate to pinned data in the NVC or not. It also controls all devices (DRAM, flash, and disk) and invokes the cache manager to manage the DRAM cache and the unpinned data in the NVC, using the cache management algorithm described in Section III.

The disk simulator simulates disk I/O accesses to a single disk and estimates its performance and energy consumption using an established disk model and power control policy. We modeled the Samsung HM080HI [11] and used conventional threshold-based power management (TPM). But, any disk model could be used.

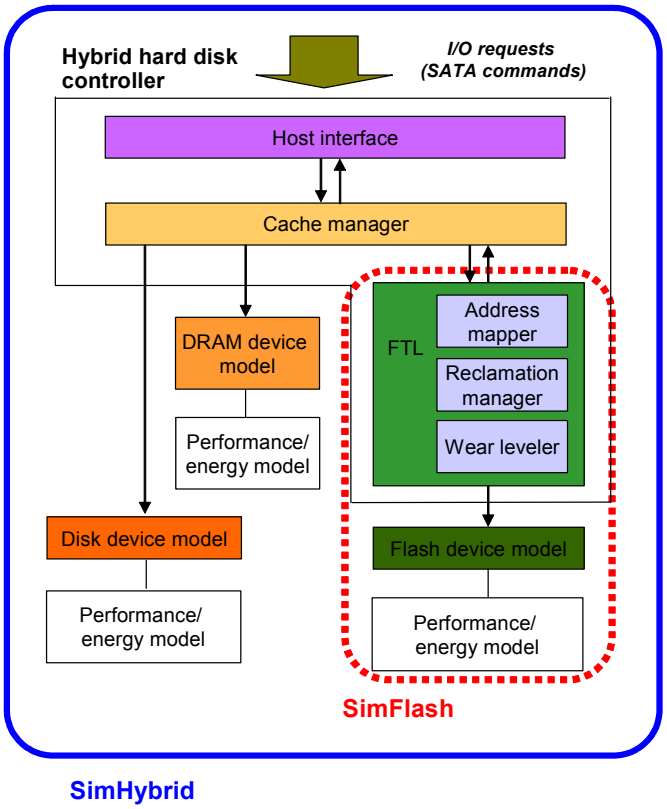

Fig. 4. Overall structure of the SimHybrid: it includes a hybrid hard disk controller, a DRAM model, a disk simulator, and SimFlash. SimFlash consists of a flash memory model and an FTL.

Our disk performance model estimates the total request response time of a disk as the sum of a queue delay, a disk delay, and a service time per request, as follows:

$$T_{total\ response\ time} = \sum T_{queue\ delay} + \sum T_{disk\ delay} + \sum T_{service\ time}.$$

We obtain the average request response time by dividing the total response time by N: which is the total number of requests in the I/O traces. We can also estimate the total elapsed time by summing the response time for each request and the idle time, if any, between successive requests.

$$T_{avg.\ response\ time} = T_{total\ response\ time} / N.$$

Each request to the disk has a service time, which is the time needed to read or write l blocks, starting at block b, and can be estimated as follows:

$$T_{service}(b,l) = T_{seek}(b,b_{last}) + T_{xfer}(l) + T_{avg.\ rotation\ delay}.$$

We used a simple seek time model, based on the relation between the number of sectors per cylinder and the LBA. We used the average value of rotational delay.

The total energy used by the disk can now be modeled as the sum of the energy consumed by the disk in each state and the energy consumed during transition periods:

$$E_{disk} = \sum_j P_j \cdot T_j + \sum_k \sum_l N_{kl} \cdot E_{kl}.$$

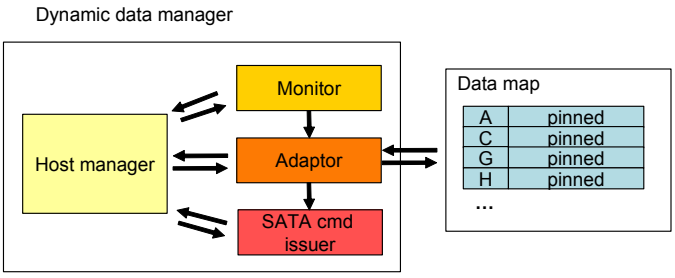

Fig. 5. Overall structure of the host simulator

The disk state j can be active, idle, or standby, and in each state the disk has a different power consumption P_j. T_j is the period during which the disk stays in each power state. E_{kl} is the energy consumed by a transition from state k to state l, and N_{kl} is the number of transitions that the disk makes from state k to state l.

We used ideal power management policies for the remaining devices (i.e., the NAND flash memory and the DRAM), assuming that there are no transition time and energy overheads when these devices are powered down. This is acceptable because the transition time and energy required by the disk overwhelm the overheads of the other devices.

We also implemented a read-ahead mechanism which transfers 128 LBAs (i.e., 64 KB data) at a time from the disk to the DRAM cache when a block miss occurs in the DRAM cache. This read-ahead size can be changed in the simulation input file.

C. *Host simulator*

Fig. 5 shows the overall structure of the host simulator, which mainly consists of a dynamic data manager and a data map. The dynamic data manager is made up of four modules: a host manager, a monitor, an adaptor, and a SATA command issuer. Using real traces collected on a Windows XP platform, the monitor checks whether each I/O request is important (i.e., data that would improve performance if it were pinned) by applying the proposed data pinning policy. When important data, which is typically boot or frequently used data, is identified, the monitor invokes the adaptor, which consults the data map which maintains the information of pinned data which corresponds to the data in the NVC. If there is new data to pin, or old data to evict, the adaptor re-arranges the contents of the data map and invokes the SATA command issuer.

Each entry in the data map consists of an LBA range which follows the ATA8-ACS command specification [13], and attributes which includes a data-type flag, the time that has elapsed since data was pinned, and the frequency with which the data has been accessed. The SATA command issuer issues both normal and NVC-specific SATA commands. The host manager controls the monitor, the adaptor, and the command issuer.

V. Trace Generation

To evaluate our optimization techniques using SimHybrid requires complete I/O traces of both boot and normal procedures on a Windows XP platform. The boot procedure

involves the POST of a main board and a cold booting procedure for the Windows XP kernel loading, during which it is almost impossible to capture I/O operations. We therefore employed a virtual machine tool[2] and obtained I/O traces with a trace logging tool[3] in the host system, while running various applications on Windows XP in the virtual system. In this method disk I/Os in the virtual system reproduces those on the host system (they are addressed to the same file on a virtual disk), but these virtual transactions are isolated from the physical characteristics of the host system. During disk I/O, LBAs on the virtual disk have a constant offset from the file locations, and thus we could generate boot or normal traces by adjusting the LBAs on the traces obtained from the host system.

Details of the boot and normal traces are shown in TABLE I, II, and III. As shown in TABLE I, the boot trace consists of a large number of sequential accesses and a small number of random accesses. The sequentiality threshold is 32 sectors. This means that if an LBA of a currently accessed block is continued to LBAs of prior accessed blocks by the number of 32 without intermission, the block is labeled as sequential. Otherwise, it is labeled random. In addition, every I/O is requested with a start LBA and an LBA count, and thus if the count is lager than 32, all the LBAs belonging to this request are labeled sequential.

We selected several Windows applications (which we call the generic-user benchmark), as shown in TABLE II, and ran them to generate a generic-user trace. TABLE III describes the trace collection environment and provides statistical information about a trace, which we see contains a large number of write requests and fewer read requests.

The sequentiality information in the boot trace is important because the pinning policy for the boot data in the host simulator is based on a sequentiality threshold. This causes only random data to be stored to the NVC, in order to exploit the random access performance of flash memory. The I/O type information in the generic-user trace is also important because the cache manager of SimHybrid uses the sequentiality of a block to decide whether to write data to the disk or to the NVC, as described in Section III. Furthermore, during the simulation of application launching, we depend on the sequentiality of data to determine which data to pin.

VI. Simulation Results

As a reference, we implemented a legacy hard disk simulator, by using the hybrid hard disk simulator, combined with an LRU caching algorithm for the DRAM cache. To evaluate our performance optimization techniques as accurately as possible, we used parameters that corresponded as closely as possible to those of a real hybrid hard disk system, using data from both public and private sources. Nevertheless, we had to interpolate some parameters, such as the times and power requirements for state transition, from the existing research.

There are thee sets of results, for booting time; application launching time; spin-down time and energy consumption.

A. System boot time

The booting time is estimated as a sum of the I/O response times of the disk, the flash memory, and the DRAM for the boot trace. TABLE IV and Fig. 6 show the total I/O response times of a hybrid hard disk and a legacy disk for the boot trace when the sequentiality threshold varies. As described above, all boot data which has a sequentiality below the threshold should be pinned into the NVC by the host simulator before the system is shut down. When the system boots, the data to be required is written to and read from the pinned region or the disk, based on its sequentiality. The symbol of ∞ in TABLE IV means that all boot data is pinned into the NVC.

The response time of the hybrid hard disk is always lower than that of a legacy disk, regardless of the sequentiality threshold, except when the sequentiality threshold is 0 and no boot data is pinned into the NVC. Although the hybrid hard disk and the legacy hard disk use different algorithms to manage the DRAM cache, there was little temporal locality in the boot trace, indicating that the DRAM cache had a negligible effect, and the performance of both disks were virtually the same.

As we would expect, increasing the sequentiality threshold reduces the I/O response time of the hybrid hard disk (even

TABLE I
Information of a Windows XP boot trace

Pattern	Request count	Data size	Percentage of trace
Random	747	11,148 sectors (5.4 MB)	4%
Sequential	1604	280,722 sectors (137.1 MB)	96%
Total	2351	291,870 sectors (142.5 MB)	100%

TABLE II
Generic-user benchmark

Application	Description
Powerpoint	Edit PPT files
Internet Explorer	Search Web
Outlook Express	Send and receive emails
Explorer	Search and find files
MSN Messenger	Send and receive messages
Photoshop	View and Edit picture files

TABLE III
Information of a Windows XP normal trace collected by using Generic-user benchmark

Collection environment	OS	Microsoft Windows XP
	RAM	512 MB
	Disk	8 GB
Trace information	Total execution time	3 hr 30 min
	Request count (r/w)	72822 (31208/41614)
	Request size (r/w)	1722 MB (484/1288)
	Working-set size	955 MB

[2] We used VMWare, which is a virtual machine platform which provides each user with an isolated and dedicated physical machine [15].

[3] We used Diskmon, which logs and displays all hard disk activity on a Windows system, using a built-in subsystem that collects logs [16].

TABLE IV

I/O RESPONSE TIMES OF A HYBRID AND LEGACY DISK SYSTEMS FOR A BOOT TRACE WITH A VARYING SEUQNEUNTIALITY THRESHOLD

Device	Hybrid hard disk (Sequentiality threshold)								Legacy hard disk
	0	32	64	128	256	512	1024	∞	
DRAM I/O time (s)	1.13	0.97	0.92	0.83	0.68	0.65	0.62	0.58	1.13
Flash I/O time (s)	0	0.37	0.50	1.02	1.62	2.57	3.66	4.33	N/A
Disk I/O time (s)	16.41	10.04	8.72	6.35	5.14	3.48	1.82	0	16.41
Total I/O response time (s)	17.54	11.37	10.15	8.19	7.67	6.69	6.10	4.91	17.54

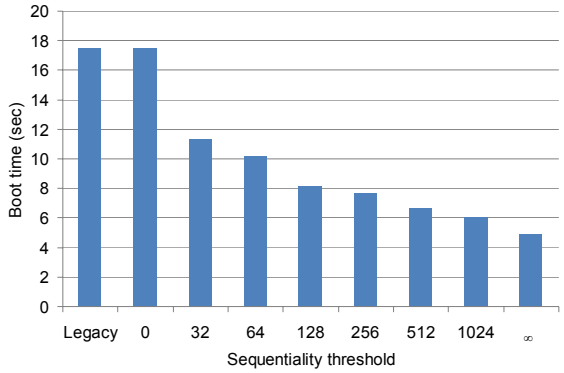

Fig. 6. I/O response times of a hybrid hard disk and a legacy disk for the boot trace with a varying sequentiality threshold.

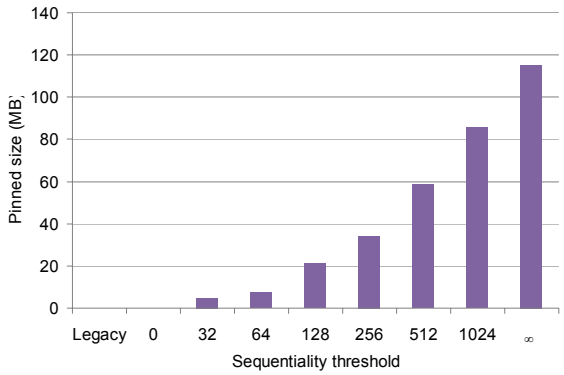

Fig. 7. Pinned data sizes for the boot trace with a varying sequentiality threshold.

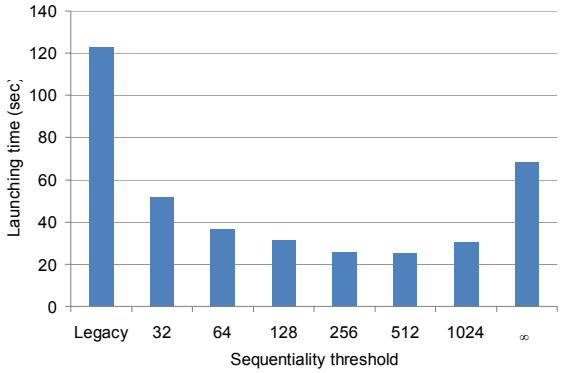

Fig. 8. Launching times for Photoshop with a hybrid hard disk and a legacy disk with a varying sequentiality threshold.

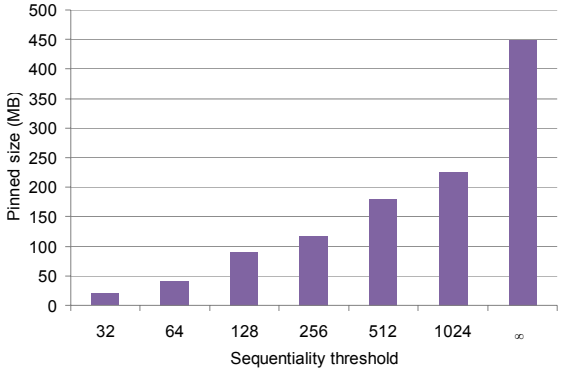

Fig. 9. Amount of pinned data for Photoshop with a varying sequentiality threshold.

the default sequentiality threshold reduced the system boot time by 35%). Fig. 7 shows pinned data sizes for the boot trace as the sequentiality threshold varies. As the sequentiality threshold increases, the amount of pinned data in the NVC increases, as shown in Fig. 7, and thus flash access represents a higher proportion of the overall I/O response time, resulting in a significant performance improvement.

From Figs. 6 and 7, we see that the amount of pinned data roughly doubles, while the total I/O response time decreases almost linearly as the sequentiality threshold increases, and more boot data is identified as random. This is because the pinned data is still a small proportion of the total amount of boot data (142.5 MB from TABLE I) and the disk shows good I/O performance on the sequential requests for the remainders of the data.

These results might be taken to suggest that pinning all the boot data into the NVC will give the best boot performance. But this would require a large NVC. In addition, the boot data remaining in the NVC must be removed after booting to make space for frequently used data. Putting more boot data in the NVC would increase this overhead. Furthermore, the amount of pinned boot data is related to the pinning policy for OEM programs or widely-used but long-loading applications to be pinned. In order to balance the system performance between booting and normal processes, more elaborate management of the data in the NVC is probably necessary, and this is a topic for future work.

B. Application launching time

The application launching time is the aggregate loading times of all the necessary executables, such as dynamic link libraries. To evaluate this time, we used the Adobe Photoshop 9.0 CS2, which is a widely-used graphic tool, which takes quite a long time to load. Since our host simulator is trace-driven and mimics Windows XP in a simple way, we assume that we know when Photoshop will be used by a user in the host simulator. The pinning data for Photoshop can be pinned

TABLE V
SPIN-DOWN TIMES OF HYBRID AND LEGACY DISK SYSTEMS FOR THE GENERIC-USER TRACE WITH A DEFAULT SEUQUENTIALITY THRESHOLD

	Total elapsed time	Total I/O response time	% of spin-down time	Power state transitions			
				A → I	I → A	I → S	S → I
Hybrid hard disk	3 hr 42 min	41 min	60	735	445	290	290
Legacy hard disk	4 hr 12 min	1hr 37min	19	1769	1171	598	598

TABLE VI
DRAM AND NVC ACCESS INFORMATION OF HYBRID AND LEGACY DISK SYSTEMS FOR THE GENERIC-USER TRACE

	DRAM			Flash			
	Access #	Hit	Miss	Pinned region access #	Unpinned region access #	Hit	Miss
Hybrid hard disk	5490208	23%	77%	714107	223276	32%	68%
Legacy hard disk	8244896	37%	63%	N/A	N/A	N/A	N/A

in one operation or in a distributed fashion. We tried both methods, basing pinning on sequentiality as before.

Fig. 8 shows the launching times for Photoshop on a hybrid hard disk and on a legacy disk, as the sequentiality threshold varies. With the default sequentiality threshold value (i.e., 32), the hybrid hard disk takes 58% less time than the legacy hard disk. This indicates that the launch involves a mix of random and sequential data accesses that can be split effectively between the NVC and the disk.

The launching time appears to be a quadratic function of the sequentiality threshold, with a global minimum at 256. This is because the size of the pinned region is limited, and as the sequentiality threshold increases less random data is likely to be pinned to the pinned region. We observed that the larger the sequentiality threshold value the less random data was pinned into the NVC (thus more random data stayed in the disk). Therefore, with a larger sequentiality threshold, the disk suffered from larger seek times and rotational delays.

Fig. 9 shows that more Photoshop data is pinned as the sequentiality threshold increases, but the amount of pinned data does not rise as steeply as in Fig. 7. We conjecture that this is because the random data is skewed in terms of request time, whereas the boot trace contains a small number of random blocks which are uniformly distributed.

C. Spin-down time and energy consumption

Our metric for energy efficiency is the aggregate spin-down time as a percentage of the total elapsed time, and is obtained using SimHybrid with a generic-user trace. We also compared the energy consumptions of a hybrid hard disk and a legacy disk. TABLE V shows the total elapsed times and the I/O response times of a hybrid hard disk and a legacy disk for the generic-user trace. The total elapsed time is 12% less for the hybrid hard disk and the I/O response time is reduced by 58%. These reductions can largely be ascribed to improved write buffering and read caching by the NVC.

TABLE V also shows that the hybrid hard disk spends about 3 times longer spun down than the legacy disk for the generic-user trace. By staying in standby for longer, the hybrid hard disk uses 23% less energy than the legacy disk: the figures are 15,354J and 19,917J, respectively. TABLE V itemizes the active to idle, idle to active, idle to standby, and standby to idle transitions. The legacy disk makes more transitions but stays in the lower power states (i.e., idle or standby) for less time than the hybrid disk.

To look more closely at the reasons why a hybrid hard disk spends longer spun down, we examined the behaviors of the DRAM cache and the NVC. TABLE VI shows that the NVC absorbs a lot of I/O accesses (about 17% of the total), achieving a 32% hit rate. The hit rate in the DRAM cache of the legacy disk was higher than this, but the number of missed accesses was also higher. In the hybrid hard disk, the read-ahead mechanism tries to read sequential blocks from the disk, but in the legacy disk it reads all types of blocks. This reduces the numbers of I/O operations to be handled by the NVC, of which 32% are absorbed, providing the disk with longer idle periods.

VII. RELATED WORK

There has been a lot of research on combining hard disks with flash memory in mobile storage systems. March et al. [3], Bisson *et al.* [4], Chen *et al.* [5], and Kgil *et al.* [6], have all proposed using flash memory as an NVC, to store blocks which are likely to be accessed in the near future, and thus allowing a hard disk to spin down for longer. Bisson *et al*. [4] focused on the redirection of write requests to a flash memory instead of a hard disk, while Chen *et al*. recently studied partitioning a flash memory into a cache, a prefetch buffer, and a write buffer to save energy. Kgil *et al.* focused on reducing main memory power consumption by using a flash memory as a second-level buffer cache. Douglis *et al.* [17] examined three alternatives for mobile storage in terms of energy consumption and I/O performance: a hard disk, a flash disk, and a flash memory card.

Recently, Bisson *et al.* [18] studied spin-down algorithms for hybrid hard disks with I/O subsystem enhancements, partitioning a flash memory into separate read and write caches, and redirecting I/Os to them while the disk is spun down. Their work largely addresses energy consumption and spin-up latency, and does not use a DRAM cache, nor do they examine the interaction between a hybrid hard disk and its host system.

Samsung and Microsoft have commercially developed a hybrid hard disk drive technology, which combines a hard disk and a NAND flash memory device as an NVC, to boost the performance, reduce the power consumption, and increase the reliability of mobile computers [8, 9, 10]. Intel's Robson technology [19], also uses flash memory as an NVC to increase system responsiveness, speed up multi-tasking, and extend battery life time. While the hybrid hard disk drive technology exploits the proximity of the disk and the flash memory, and uses a SATA interface, the Robson technology is more concerned with the relationship between the main board and the flash memory, and uses a PCI-like interface.

VIII. CONCLUSION

We have presented I/O optimization techniques for hybrid hard disks, which include an intelligent data pinning policy for the NVC, and a caching technique which is sensitive to access patterns. Our proposed techniques are efficient in reducing the system boot time and application launching time of mobile consumer devices. By making a power-hungry hard disk to stay longer in the low-power mode, the proposed techniques are also efficient in reducing the energy consumption of the mobile devices. In order to evaluate our approach, we built SimHybrid, an extensive trace-driven hybrid hard disk evaluation environment. Experimental results showed that the boot time and the application launching time are reduced by more than 35% and 58%, respectively, over using a legacy disk system while saving energy by 23%.

ACKNOWLEDGMENT

We thank Mr. Kyung-Il Bang, Dr. Dongkun Shin, Dr. Junghwan Kim, and Dr. SeongJun Ahn of Samsung Electronics CO., LTD., for their technical discussions which significantly improved the quality of the proposed techniques.

REFERENCES

[1] G. Lawton, "Improved flash memory grows in popularity," *IEEE Computer*, vol. 39, no. 1, pp. 16-18, January. 2006.
[2] Apple Inc., iPod Nano. http://www.apple.com/ipodnano/
[3] B. Marsh, F. Douglis, and P. Krishnan, "Flash memory file caching for mobile computers," in *Proc. of the 27th Hawaii International Conference on System Sciences*, Hawaii, USA, pp. 451-460, January. 1994.
[4] T. Bisson and S. Brandt, "Reducing energy consumption with a non-volatile storage cache," in *Proc. of InternationalWork-shop on Software Support for Portable Storage (IWSSPS)*, held in conjunction with *the IEEE Real-Time and Embedded Systems and Applications Symposium (RTAS 2005)*, San Francisco, California, USA, March, 2005.
[5] F. Chen, S. Jiang, and X. Zhang, "SmartSaver: turning flash drive into a disk energy saver for mobile computers," in *Proc. of 11th ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED'06)*, Tegernsee, Germany, October 4-6, 2006.
[6] T. Kgil and T. Mudge, "FlashCache: A NAND flash memory file cache for low power web servers", in *Proc. of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems (CASES '06)*, Seoul, Korea, October 22-25, 2006.
[7] Y.-J. Kim, K.-T. Kwon, and J. Kim, "Energy-efficient file placement techniques for heterogeneous mobile storage systems," in *Proc. of the 6th ACM & IEEE Conference on Embedded Software (EMSOFT)*, Seoul, Korea, October 22-25 2006.
[8] Microsoft, ReadyDrive and Hybrid Disk. http://www.microsoft.com/whdc/device/storage/hybrid.mspx.
[9] http://www.samsung.com/Products/HardDiskDrive/news/HardDiskDrive_20050425_0000117556.htm.
[10] R. Panabaker, "Hybrid hard disk & ReadyDrive™ technology: improving performance and power for Windows Vista mobile PCs," in *Proc. of Microsoft WinHEC 2006*. http://www.microsoft.com/whdc/winhec/pres06.mspx.
[11] http://www.samsung.com/global/business/hdd/productmodel.do?group=&type=62&subtype=67&model_cd=308&ppmi=1159#.
[12] ATA8-ACS Command Set Specification. http://www.t13.org/.
[13] T.-S. Chung, S. Park, M.-J. Jung, and B. Kim, "STAFF: state transition applied fast flash translation layer," *Lecture Notes in Computer Science (LNCS) 2981*, Springer-Verlag, 2004.
[14] http://www.samsung.com/products/semiconductor/NANDFlash/SLC_LargeBlock/8Gbit/K9K8G08U0M/K9K8G08U0M.htm.
[15] EMC, VMWare. http://www.vmware.com/
[16] Diskmon. http://www.microsoft.com/technet/sysinternals/default.mspx.
[17] F. Douglis, F. Kaashoek, K. Li, R. Caceres, B.Marsh, and J. A. Tauber, "Storage alternatives for mobile computers," in *Proc. of the 1st Symposium on Operating Systems Design and Implementation (OSDI)*, 1994.
[18] T. Bisson, S. Brandt, and D. Long, "A hybrid disk-aware spin-down algorithm with I/O subsystem support," in *Proc. of the 26th IEEE International Performance Computing and Communications Conference (IPCCC)*, New Orleans, Louisiana, USA, April 11-13, 2007.
[19] M. Trainor, "Overcoming disk drive access bottlenecks with Intel Robson technology," *Technology®Intel Magazine*, December, 2006. http://www.intel.com/technology/magazine/computing/robson-1206.htm.

Young-Jin Kim (S'05) received the B.E. degree and the M.E. degree in electrical engineering from Seoul National University, Seoul, Korea, in 1997 and 1999, respectively. From 1999 to 2003, he was with Electronics and Telecommunications Research Institute (ETRI), Daejoen, Korea. He is currently working toward the Ph.D. degree at Seoul National University. His research interests include low-power embedded systems, low-power storage systems, and power measurement and analysis.

Sung-Jin Lee received the B.E. degree in electrical engineering from the Korea University, Seoul, Korea, in 2005, and the M.E. degree in computer science and engineering from Seoul National University, Korea, in 2007. He is currently working toward the Ph.D. degree at Seoul National University. From 1999 to 2002, he was a software engineer at Bridgetec Co., Seoul, Korea. His research interests include storage systems, low-power embedded systems, operating system, and power measurement and analysis.

Kangwon Zhang received the B.E. degree in the School of Computer Science and Engineering from Seoul National University, Korea in 2006. He was a game developer at Nexon Co., Korea, from 2002 to 2004. He is currently a M.E. graduate student in Seoul National University. His research interests include embedded storage systems and file system internals.

Jihong Kim (M'00) received the B.S. degree in computer science and statistics from Seoul National University, Seoul, Korea, in 1986, and the M.S. and Ph.D. degrees in computer science and engineering from the University of Washington, Seattle, WA, in 1988 and 1995, respectively. Before joining SNU in 1997, he was a Member of Technical Staff in the DSPS R&D Center of Texas Instruments in Dallas, Texas. He is currently a Professor in the School of Computer Science and Engineering, Seoul National University. His research interests include embedded software, low-power systems, computer architecture, and multimedia and real-time systems.