J Sign Process Syst (2010) 60:149–168
DOI 10.1007/s11265-008-0308-1

Energy-Efficient Streaming Using Non-volatile Memory

Mohammed G. Khatib · Pieter H. Hartel · Hylke W. van Dijk

Received: 14 April 2008 / Revised: 19 September 2008 / Accepted: 22 October 2008 / Published online: 26 November 2008

Abstract The disk and the DRAM in a typical mobile system consume a significant fraction (up to 30%) of the total system energy. To save on storage energy, the DRAM should be small and the disk should be spun down for long periods of time. We show that this can be achieved for predominantly streaming workloads by connecting the disk to the DRAM via a large non-volatile memory (NVM). We refer to this as the NVM-based architecture (NVMBA); the conventional architecture with only a DRAM and a disk is referred to as DRAMBA. The NVM in the NVMBA acts as a traffic reshaper from the disk to the DRAM. The total system costs are balanced, since the cost increase due to adding the NVM is compensated by the decrease in DRAM cost. We analyze the energy saving of NVMBA, with NAND flash memory serving as NVM, relative to DRAMBA with respect to (1) the streaming demand, (2) the disk form factor, (3) the best-effort provision, and (4) the stream location on the disk. We present a worst-case analysis of the reliability of the disk drive and the flash memory, and show that a small flash capacity is sufficient to operate the system over a year at negligible cost. Disk lifetime is superior to flash, so that is of no concern.

M. G. Khatib (✉) · P. H. Hartel · H. W. van Dijk
Department of Computer Science, University of Twente, Enschede, The Netherlands
e-mail: m.g.khatib@utwente.nl

P. H. Hartel
e-mail: p.h.hartel@utwente.nl

H. W. van Dijk
e-mail: h.w.vandijk@utwente.nl

Keywords Energy efficiency · Streaming architecture · Soft real-time system · Disk drive · Non-volatile memory · DRAM · NAND flash

1 Introduction

Energy efficiency is an important design issue in mobile computer systems. Energy efficiency extends battery lifetime of mobile systems and thus the time period during which a system is operational.

The conventional storage hierarchy, composed of a disk and a DRAM (henceforth referred to as DRAMBA), accounts for a large proportion of the energy consumed by a computer system. For example, a spinning disk drive may account for as much as 30% of the total energy consumed [19]. In streaming experiments in our laboratory, we found that a Microdrive plugged into an HP iPAQ H2215 PDA consumes as much as 23% of the total energy (Fig. 1).

Many different solutions have been proposed to save on disk energy consumption. The basic idea, when operating the disk, is to spin off the disk drive for as long and as often as possible. Adaptive power management policies have been proposed to spin off the disk, while controlling the response time [9, 20]. Flash memory is used to cache write requests, leaving the disk spun off to save on energy [11].

If the system workload is predictable, which is typically the case for predominantly streaming workloads, the disk can be spun off for long time periods, thus saving significantly on disk energy. In the conventional architecture (i.e., a DRAMBA), Mesut et al. [22] prefetch streaming data into DRAM to maximize the spin-off period. However, DRAM dissipates a few milliwatts

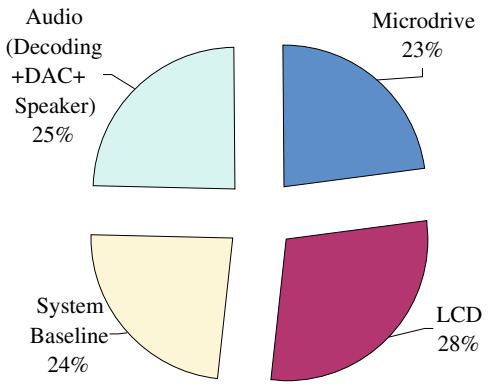


Figure 1 Energy breakdown of a PDA when streaming from a Microdrive.

per megabyte to refresh (and thus retain) its contents. Hence, the maximum energy saving is attained when the sum of the disk energy and the DRAM retention energy is minimal. In other words, the energy saving has an upper bound in DRAMBA.

Two factors influence the energy saving of DRAMBA. Firstly, if the streaming demand, represented by the number of concurrent streams and their bit rates, increases, the amount of DRAM needed for pre-fetching increases, and so does its refresh energy. As a result, leaving the disk spinning all the time may even result in less total energy than pre-fetching. Secondly, if the form factor of the disk drive increases, the necessary amount of DRAM also increases to account for the larger latency due to the larger mechanical inertia. Both factors influence the DRAM capacity, and thus its retention energy, effectively lowering the energy saving bound to a point that renders pre-fetching impractical in DRAMBA.

Our contribution is to decouple DRAM from disk, thereby raising the upper bound on the energy saving of DRAMBA. We interpose a non-volatile memory (NVM) between the disk and the DRAM, where the NVM acts as a traffic reshaper; this architecture will be referred to as NVMBA. NVM does not require refresh cycles, and the NVM size is limited only by its installation cost. NVM latency is lower than disk latency, but higher than DRAM latency, so that a small DRAM buffer is sufficient to sustain the required streaming rates. The combination (disk, NVM, and DRAM) makes optimal use of the disk (by extending its spin-off period), and it makes optimal use of the DRAM (by using little of it). The cost increase of the system through the addition of NVM can be balanced through a reduced DRAM capacity.

We propose to use NAND flash, because of its suitable characteristics (see next section). Our work analyzes the energy-saving merit of NVMBA over DRAMBA for predominantly streaming workloads. Hence, in both architectures all intermediate storage is organized as a FIFO (First-In, First-Out) buffer. We quantify the energy saving merit of NVMBA relative to DRAMBA with respect to four setup and workload parameters: (1) the streaming demand, (2) the disk form factor, (3) the best-effort provision, and (4) the stream location. We include the lifetime of the disk drive and the flash in our analysis.

The rest of the paper is organized as follows. In the next section, we motivate the design of NVMBA, and explain the functionality of its components. Sections 3 and 4 lay out the analytical foundation of the analysis method and tools. Since in reality streaming architectures must also support some best-effort traffic, we extend our analysis to deal with a small percentage of best-effort service in Section 5. We study in detail the saving of each architecture and compare the results in Section 6. Section 7 follows up with detailed studies to quantify the energy difference with respect to four setup and workload parameters. A worst-case study is carried out to estimate the lifetime of the disk drive and the flash memory in Section 8. We provide an experimental validation of our analytical results in Section 9. Section 10 discusses the related work, and the final section concludes.

2 The Nonvolatile-memory–based Architecture

We propose NVMBA, *nonvolatile-memory–based architecture*, as an extension to the conventional DRAMBA, *DRAM-based architecture*. We show that NVMBA is a feasible and economical alternative to DRAMBA. Before delving into its functionality, we first describe how to realize NVMBA starting from DRAMBA.

2.1 Realization

There are architectural and performance criteria that should be satisfied by the NVM for energy-efficient streaming in NVMBA. The architectural criteria are the number of ports, and the interfaces across which the buffer is exposed to the system. The performance criteria are the buffer throughput, and latency.

Secondary Buffer The NVM communicates with the host system via a shared I/O subsystem. As a shared

resource, the I/O subsystem should not be monopolized by the NVM. To guarantee smooth streaming, we deploy a secondary buffer close to the CPU that buffers data during the unavailability periods of NVM.

Dual Ported I/O At times, the NVM refills from the disk and flushes into the secondary buffer concurrently. To allow such concurrent transfer of data, the NVM should have two independent I/O ports.

Throughput Interposing the NVM between the secondary buffer and the disk drive should not make it a performance bottleneck, since throttling the throughput results directly in an energy loss. In case of a throttling NVM, the disk refills the buffer at a lower rate, which causes idle disk rotations. We show that implementing an NVM with a throughput greater or equal than the disk throughput is a sufficient condition to prevent the disk from idling.

Latency The NVM should have a shorter latency than the disk. As we will see later, latency directly influences the capacity of the secondary buffer, and thus its energy consumption.

Technologies The NVM can be realized by NAND flash memory or future technologies such as Phase-Change memory, and MEMS-based storage devices (Micro-Electro-Mechanical Systems) [2, 18]. In this work, we consider NAND flash because it has (1) better performance at large writes than small ones [10], (2) no moving parts, (3) short response time, unlike disk drives, (4) no retention energy, unlike DRAM, (5) lower cost than NOR flash, and (6) wide availability.

As we will see later, the secondary buffer capacity is in the order of a few kilobits that could be realized by an on-chip embedded memory. In this study, however, we realize it by DRAM for two reasons. Firstly, for a fair comparison with the conventional architecture, we fix the disk and memory technologies that we use in both architectures to evaluate the exact saving due to the NVM. Secondly, DRAM has lower cost than on-chip embedded memory.

2.2 Two Architectures

The diagrams of Fig. 2 show our two architectures: DRAMBA and NVMBA. The top diagram shows a block diagram of the conventional streaming DRAMBA architecture, where DRAM is a primary buffer of the disk drive. The bottom diagram of Fig. 2 shows our proposed architecture, NVMBA, where an NVM serves as the primary buffer, and DRAM as a secondary buffer. The NVM has orders of magnitude shorter latency than the disk drive. As a result, the DRAM in NVMBA can be smaller than the DRAM in DRAMBA. The disk fills the NVM at rate r_d, i.e., the throughput of the disk, while the NVM fills the DRAM at rate r_{NVM}. The DRAM data are consumed at rate r_s, the streaming bit rate. For the sake of brevity, we consider throughout our study streaming from the disk drive. Nonetheless, our analysis applies equally

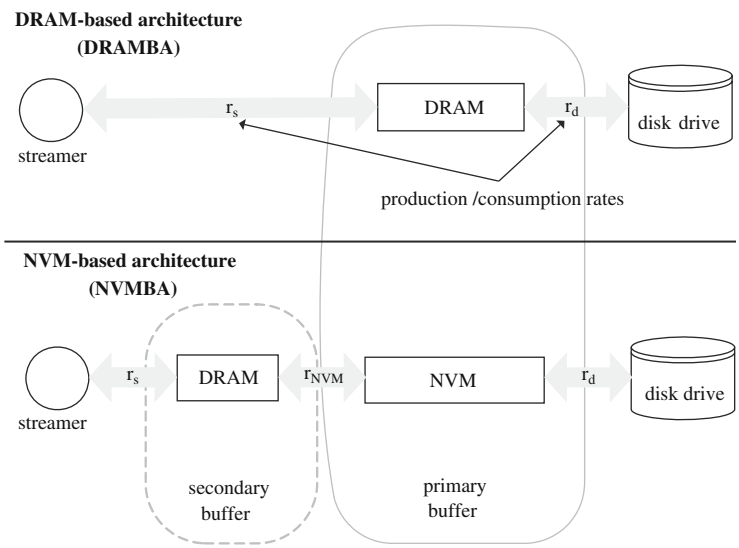


Figure 2 Block diagram of DRAMBA (*top*) and NVMBA (*bottom*).

to streaming to the disk drive or a mix of read/write streaming.

2.3 Traffic Reshaping

Figure 3 shows the activity for two consecutive *cycles* of the disk, NVM, and DRAM in NVMBA (A cycle is the time that the system diverts from one specific state until it returns to that same state). To stream from the disk, the disk starts every cycle of T_d time units filling the NVM with a relatively large amount of data at a net rate $r_d - r_s$. In fact, the disk fills at r_d and the buffer empties at r_s at the same time. At each cycle, the disk spins up and seeks before the actual refill. After the refill state the disk spins down immediately and remains in standby (i.e., spin-off state) to save energy. The NVM repeatedly refills the DRAM at rate $r_{NVM} - r_s$ with a relatively small amount of data, so that DRAM can be kept small to reduce its retention energy. When the NVM is nearly empty, the disk spins up, to prepare for the next cycle.

2.4 Minimizing Energy

The objective of our study is to build an energy-efficient streaming architecture, thus our focus is to minimize energy consumption. Although storage technologies vary in the ways they consume energy, we develop a generic template to model the energy components for each particular technology.

A storage device (or a memory) consumes two types of energy: *static* and *dynamic* energy. Static energy is consumed to retain the content of a store, such as the refresh energy of a DRAM. Static energy is consumed as long as the device is powered on and contains useful data. The amount of consumed energy is device dependent, but predominantly determined by its capacity. Dynamic energy, on the other hand, represents the energy consumed by each activity of the device associated with accessing data. Figure 4 shows a generic power-state machine with five states: active (read/write), idle, shutdown, standby, and startup. The dynamic energy is the sum of the energy consumed in each state.

For each state, Fig. 4 specifies the power dissipation (P_x) and the period t_x, i.e., the duration of the respective state. The period for the active and standby states, t_{RW} and t_{sb}, respectively, are determined by the workload. The period for the idle and shutdown states have predetermined fixed lengths after which a state transfer occurs. Finally, the period of the startup state, t_{su}, specifies the time involved for the device to become

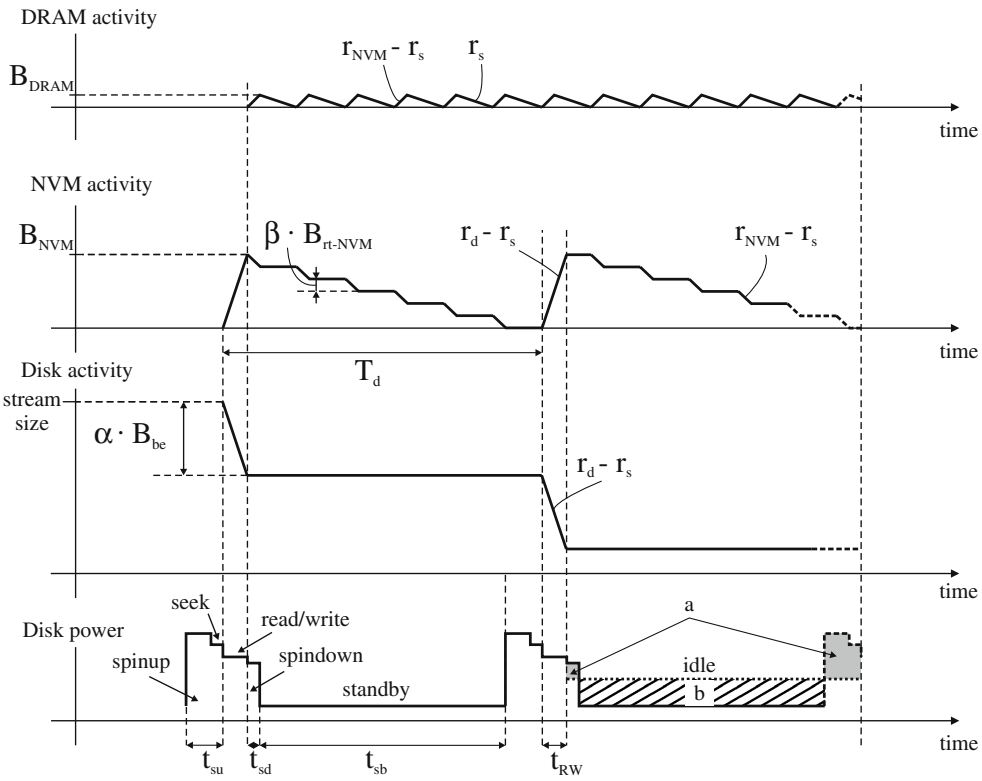


Figure 3 Activities of the disk drive, the NVM, and the DRAM in NVMBA during a refill cycle (T_d). Table 1 describes each parameter.

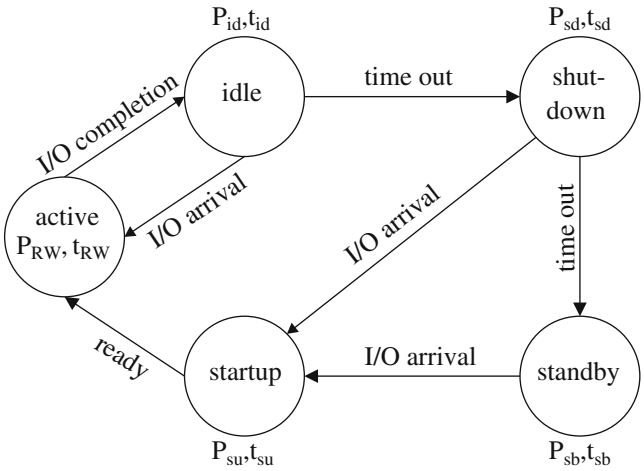


Figure 4 A power-state machine of a storage device or memory. It shows per state the dynamically consumed energy. P_{RW} and t_{RW} refer to the read/write active power and time respectively.

ready, which has a fixed device-dependent part and a dynamic workload dependent part. Note that an I/O arrival triggers a shortcut to the active state.

To reduce energy, we should reduce the static and/or the dynamic energy. Static energy is reduced by decreasing the device capacity. Reducing the dynamic energy requires: (1) limitation of the number of state transitions from standby to active, (2) avoidance of the idle state, and (3) extension of the standby periods.

Suppose an example storage device that implements the power-state machine of Fig. 4. Deploying a buffer for such a device helps to reduce the dynamic energy of the device. For instance, let us stream 10 MB of data from the storage device. If we deploy a buffer with a capacity of 2 MB, then the power-state machine triggers 5 active-to-standby transitions, whereas if we deploy a 5 MB buffer, just 2 of these transitions are triggered. Further, fewer transitions yield less transition time, which prolongs the period that the device is in standby state. Thus, an appropriately sized buffer saves energy.

Minimizing the energy of DRAMBA requires a reduction of the disk dynamic energy and the DRAM static energy. To reduce the disk dynamic energy, we should increase its buffer, namely the DRAM. On the contrary, to reduce the DRAM dynamic energy its capacity should be reduced. Thus, we must find a balanced DRAM capacity that minimizes the total energy consumption.

Minimizing the energy of NVMBA requires reducing the disk dynamic energy, the NVM dynamic energy, and the DRAM static energy. Reducing the disk energy does not conflict with reducing the NVM energy, since the latter has no static energy. However, reducing the NVM dynamic energy requires increasing the DRAM capacity, which increases DRAM energy. Because we choose an NVM such that its transitions are inexpensive (or even do not exist as in flash), the DRAM capacity is small and so is its energy, mitigating the conflict.

In summary, the objective is to minimize the total energy by (a) maximizing the capacity of the primary buffer in either architecture to reduce the disk energy, and (b) minimizing the capacity of the secondary buffer in NVMBA to save on the DRAM retention energy. This optimization problem has three constraints: (1) the capacity of the buffers should be larger than or equal to the capacity of the real-time buffer to guarantee smooth streaming; (2) the throughput rate of any buffer should be larger than that of the component right below it in the hierarchy, so that it does become a bottleneck; and (3) the buffer capacities are limited by the budget the designer is willing to spend.

To satisfy the first constraint, we derive the real-time buffer in the following section to ensure that the buffer capacities are sufficient. For the second constraint, we deploy as many physical modules as necessary to achieve the demanded throughput and operate them in parallel; so that bottlenecks are avoided. The third constraint, however, is relaxed in order to investigate the full potential of the energy saving as a function of the buffer capacity. In fact, this relaxation leads us to two key findings. Firstly, increasing the buffer capacity beyond a certain point leads to a larger increase in the system lifetime than in energy saving (Section 8). Secondly, a small buffer size is sufficient to achieve large energy savings for a reasonably long lifetime at negligible cost.

3 Buffer Capacities

In this section, we analytically derive the capacities of the primary and secondary buffers. To guarantee quality streaming, however, both buffers should be larger than a minimum capacity, called the real-time buffer. We derive the real-time buffer capacity first. We list in Table 1 the key parameters of the models devised in Sections 3, 4, and 5.

3.1 The Real-time Buffer Capacity

In streaming environments, we deal with (soft) real-time applications where throughput should be guaranteed and deadlines should be met to prevent

Table 1 Key parameters of our analytical models.

Parameter	Description
Input	
r_{NVM}	NVM throughput
r_{d}	Disk throughput
r_{s}	Streaming bit rate
Intermediate	
t_{bs}	Best-effort time slack
T_{d}	Disk refill cycle without best-effort service
T'_{d}	Refill cycle with best-effort (non-compensating approach)
T''_{d}	Refill cycle with best-effort (compensating approach)
B_{be}	Capacity of the break-even buffer of the disk
$B_{\text{rt-NVM}}$	Capacity of the real-time buffer of the NVM
Output	
B_{pm}	Capacity of the primary buffer
B_{sc}	Capacity of the secondary buffer
Model parameters	
α	Sizing factor of the primary buffer ($B_{\text{pm}} = \alpha \cdot B_{\text{be}}$)
β	Sizing factor of the secondary buffer ($B_{\text{sc}} = \beta \cdot B_{\text{rt-NVM}}$)
γ	Best-effort percentage ($t_{\text{bs}} = \gamma \cdot T_{\text{d}}$)

degradation in quality (due to frame dropping) and/or loss of streaming data (due to buffer overflow). The throughput requirement is guaranteed by building the system out of components that have sufficient processing and communication bandwidth. The deadline requirement, however, arises when dealing with resources that incur latency to satisfy requests; like waiting to position the head arm over the right data track before data can be transferred from the disk. This requirement can be met by deploying a buffer that can hold the data consumed during the waiting time. The minimum buffer capacity to guarantee no underrun of consumed data is called the *real-time buffer* capacity B_{rt}.

Assume a given storage device capable of reading data at a sustained throughput of r, see Fig. 5. Every request exhibits a latency l incurred by the storage device as well as other resources such as the I/O bus. To sustain real-time streaming at a throughput r_{s} from the storage device, a real-time buffer of capacity B_{rt} is needed. The real-time buffer is a leaky bucket, which is filled at a rate $r - r_{\text{s}}$ (Fig. 5), and should sustain data during disk unavailability for l time units. Thus, its minimum capacity (B_{rt}) is:

$$B_{\text{rt}} = l \cdot r_{\text{s}} \tag{1}$$

This buffer should be refilled periodically. The cycle period, T_{rt}, is the sum of the incurred latency and the actual refill time:

$$T_{\text{rt}} = \frac{B_{\text{rt}}}{r_{\text{s}}} + \frac{B_{\text{rt}}}{r - r_{\text{s}}} = \frac{l \cdot r}{r - r_{\text{s}}}$$

Here, the throughput requirement is $r > r_{\text{s}}$.

3.2 The Primary Buffer Capacity

The buffer that communicates directly with the disk is referred to as the primary buffer. Since streaming workloads have predictable data access patterns, their data can be fetched ahead into the primary buffer (See Fig. 3), so that the disk can go into standby state to save energy. However, if the disk goes standby (still dissipating P_{sb} J/s), additional energy is consumed every cycle in the shutdown state (dissipating P_{sd} J/s), and the startup state (dissipating P_{su} J/s), which includes seek. To save energy using standby, this overhead must be at least compensated for, or the disk could be left spinning in the idle mode (dissipating P_{id} J/s). Compensation of shutdown and startup overheads boils down to requiring a minimum standby period, t_{sb}, according to:

$$P_{\text{id}} \cdot \hat{t}_{\text{id}} \geq t_{\text{sb}} \cdot P_{\text{sb}} + E_{\text{oh}}, \tag{2}$$

where

$$\hat{t}_{\text{id}} = t_{\text{sb}} + t_{\text{oh}}$$

$$t_{\text{oh}} = t_{\text{su}} + t_{\text{sd}}$$

$$E_{\text{oh}} = t_{\text{su}} \cdot P_{\text{su}} + t_{\text{sd}} \cdot P_{\text{sd}}$$

Figure 3 shows graphically that the standby period should be sufficiently long to make area "b" larger than area "a". The idle period ($\hat{t}_{\text{id}}$) that balances Inequality (2) is called the break-even period, t_{be}:

$$t_{\text{be}} = \frac{E_{\text{oh}} - t_{\text{oh}} \cdot P_{\text{sb}}}{P_{\text{id}} - P_{\text{sb}}},$$

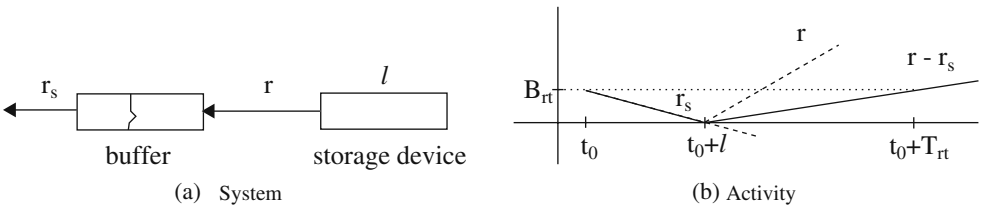


Figure 5 Leaky bucket streaming system and its activity (**a**, **b**).

and the corresponding break-even buffer capacity (B_{be}) is:

$$B_{be} = t_{be} \cdot r_s .$$

The break-even buffer capacity is that capacity at which putting the disk standby saves no energy compared to leaving it idle for the entire t_{be} period. Nonetheless, different pre-fetching levels (and thus different levels of energy savings) can be achieved by deploying a buffer capacity larger than the break-even buffer. We express this by a sizing parameter called $\alpha \geq 1$ that scales up the primary buffer capacity based on the break-even buffer capacity as the designer chooses depending on his budget. We express the capacity of the primary buffer (B_{pm}) as follows:

$$B_{pm} = \alpha \cdot B_{be} \tag{3}$$

To prevent underrun, we must guarantee $B_{pm} \geq B_{rt\text{-}d}$, where $B_{rt\text{-}d}$ refers to the real-time buffer capacity of the disk according to Eq. 1. Hence

$$\alpha \geq \max(1, \frac{B_{rt\text{-}d}}{B_{be}})$$

3.3 The Secondary Buffer Capacity

We envision an NVM located close to the disk drive from an architecture perspective. That is, it will be interfaced via the I/O subsystem, which is shared across several I/O devices (Section 2.1). As a consequence, additional buffering (DRAM) is needed to free the resources of the I/O subsystem, so that shared resources on the I/O bus get their fair share. We call this buffer the secondary buffer. The NVM exhibits orders of magnitude shorter latency than the disk. As a result, the secondary buffer is small.

The smallest capacity of the secondary buffer is equal to the capacity of the real-time buffer of the NVM, $B_{rt\text{-}NVM}$. In order to relax the load on the I/O system, we use a scaling factor, β ($\beta \geq 1$), to tune the capacity of the secondary buffer:

$$B_{sc} = \beta \cdot B_{rt\text{-}NVM} . \tag{4}$$

The refresh power of DRAM scales proportionally with its capacity. Therefore, we tune β such that the DRAM energy consumption is traded off for the I/O bus availability. Recall that, unlike in NVMBA, in DRAMBA the secondary buffer does not exist, since the main buffer is located close to the CPU.

4 Energy Consumption

We calculate the energy consumption of the system for both architectures by summing the consumed energy of its constituent components. Since the system periodically refills from the disk (see Fig. 3), this section derives the average power dissipated by the disk, the NVM, and the DRAM, i.e., the energy per refill cycle (T_d).

4.1 Disk Power Dissipation

The average power dissipated by the disk drive (P_d) is equal to E_d / T_d, the energy consumption during a cycle period T_d, where E_d is:

$$E_d = E_{oh} + P_{RW} \cdot t_{RW} + P_{sb} \cdot t_{sb}, \tag{5}$$

with:

$$t_{RW} = \frac{B_{pm}}{r_d - r_s}, \quad t_{sb} = \frac{B_{pm}}{r_s} - t_{oh}, \text{ and}$$
$$T_d = t_{RW} + t_{sb} + t_{oh}$$

Rewriting Eq. 5, substituting $P_{oh} \cdot t_{oh} = E_{oh}$ yields:

$$E_d = t_{oh} \cdot (P_{oh} - P_{sb}) + t_{RW} \cdot (P_{RW} - P_{sb}) + T_d \cdot P_{sb} \tag{6}$$

$$T_d = \frac{B_{pm}}{r_d - r_s} \cdot \frac{r_d}{r_s} \tag{7}$$

with $r_d > r_s$. The disk energy E_d of Eq. 6 is determined by three factors. The first two terms favor small overhead energy ($t_{oh} \cdot P_{oh}$) and small active energy ($t_{RW} \cdot P_{RW}$). The third term scales the actual standby power by the cycle period. Observe that the T_d scaling of Eq. 7 is a non-linear bathtub function, see Fig. 6.

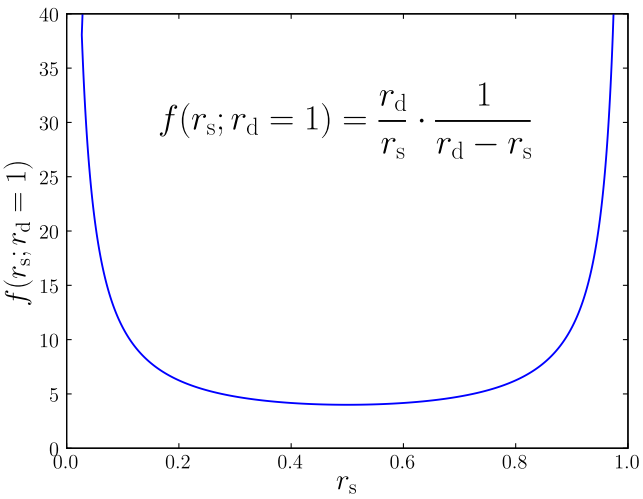


Figure 6 Non-linear, bathtub, T_d scaling of the standby power.

4.2 NVM Power Dissipation

For the NVM, we consider similar states as for the disk: active, standby, and overhead. In the active state data are read/written from/to the NVM (dissipating $P_{\text{NVM-RW}}$ J/s). In the standby, state the interface awaits requests (dissipating $P_{\text{NVM-sb}}$ J/s). The overhead state represents overhead associated with any transition from and to the standby state. The average power dissipated by the NVM (P_{NVM}) is computed from the energy per refill cycle (T_{d}).

$$E_{\text{NVM}} = E_{\text{NVM-RW}} + E_{\text{NVM-sb}} + E_{\text{NVM-oh}}$$

where $E_{\text{NVM-RW}}$ is the energy consumed to write in and then read out an amount of data of size B_{pm} in $t_{\text{NVM-RW}}$ time units. $E_{\text{NVM-oh}}$ is the total energy to transition between the two states and any other overheads associated with that transition such as seeking in mechanical devices. Assuming the NVM has equal read and write bandwidth, r_{NVM}, as well as read and write power, $P_{\text{NVM-RW}}$, we calculate E_{NVM} as follows:

$$E_{\text{NVM}} = P_{\text{NVM-RW}} \cdot t_{\text{NVM-RW}} + P_{\text{NVM-sb}} \cdot t_{\text{NVM-sb}} + E_{\text{NVM-oh}} \quad (8)$$

where

$$t_{\text{NVM-RW}} = \frac{B_{\text{pm}}}{r_{\text{d}} - r_{\text{s}}} + \frac{B_{\text{pm}}}{r_{\text{NVM}}}, \quad t_{\text{NVM-sb}} = T_{\text{d}} - t_{\text{NVM-RW}} - t_{\text{NVM-oh}}$$

and the throughput constraints are:

$$r_{\text{NVM}} > r_{\text{d}} > r_{\text{s}}$$

Since solid-state memories like flash do not incur overhead energy, so that $E_{\text{NVM-oh}} = 0$ and $t_{\text{NVM-oh}} = 0$, Eq. 8 reduces to:

$$E_{\text{NVM}} = t_{\text{NVM-RW}} \cdot (P_{\text{NVM-RW}} - P_{\text{sb}}) + T_{\text{d}} \cdot P_{\text{sb}} \quad (9)$$

Here again, small active energy is favorable and the standby energy scales with the bathtub behavior of T_{d} (Eq. 7 and Fig. 6).

4.3 DRAM Power Dissipation

DRAM consumes energy to retain data and to access (i.e., read/write) data. The retention energy of the DRAM scales proportionally with its capacity, whereas the access energy depends on the access pattern. We refer the reader to a technical report by Micron [16] that details the calculation of the DRAM energy. We implemented the Micron power calculator in our evaluation tool to calculate the energy consumption for different DRAM capacities, and access patterns in both architectures.

This section presented complete models for two streaming architectures. The next section, extends the models to account for a fraction of best-effort traffic.

5 Servicing Best-effort Requests

Streaming architectures accommodate servicing a small amount of best-effort traffic in addition to the prominent streaming traffic. Best-effort activities include, but are not limited to, loading the Operating System libraries, running application binaries, and executing file system operations. In such predominantly streaming architectures, the disk drive, as a backing store, has to provide access to best-effort data as well as streaming data.

This section extends the analytical study provided in the previous two sections to account for best-effort data access. We express the amount of time the disk drive spends accessing best-effort data t_{bs} as a percentage of the refill period T_{d}, represented by a parameter $\gamma \geq 0$, which yields $t_{\text{bs}} = \gamma \cdot T_{\text{d}}$. Consequently $T'_{\text{d}} = T_{\text{d}} \cdot (1 + \gamma) + \epsilon$, where ϵ represents the amount of time needed to buffer more data to account for the disk unavailability due to best-effort service. Since streaming is the prominent activity, we assume that γ would be relatively small, say $0 \leq \gamma \leq 0.1$. That said, this has no implication on our analytical study, but just on its orientation toward streaming architectures.

5.1 Design Issues

We scale the best-effort service period based on the available primary buffer capacity. The reason is that the longer the disk is in standby due to a larger primary buffer, the more outstanding best-effort requests can be serviced.

In the diagram of Fig. 7, best-effort requests to the disk drive are serviced after streaming requests and before spinning the disk down. The order is irrelevant for the buffer capacities and energy consumption. Best-effort data are routed from the disk drive directly to a best-effort store, typically a DRAM. Thus, best-effort data bypasses the primary and the secondary streaming buffers. Although in practice the best-effort store and the primary/secondary buffer can be realized by just one physical module, we logically separate them in order to model accurately the energy consumption of the streaming primary/secondary buffer.

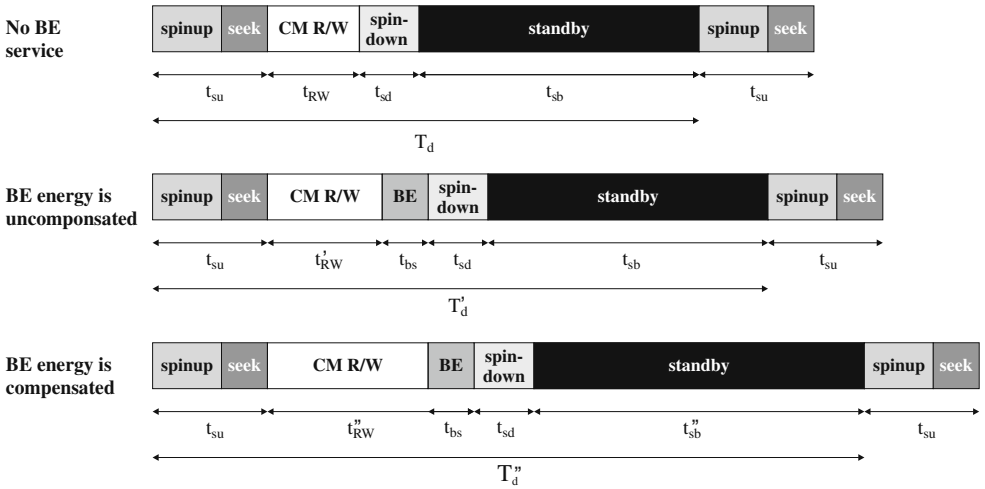


Figure 7 Two possible approaches to account for best-effort (*BE*) service; a non-compensating and a compensating approach.

We do not save substantial energy on best-effort data by caching it in NVM, given its relatively small fraction ($\gamma \leq 0.1$). Hence, best-effort data are stored elsewhere, consuming a marginal amount of energy. That said, in other mixed-media environments of larger amounts of best-effort data, our buffering technique can be combined with existing best-effort techniques, like Bisson et al's technique [11] (see Section 10), to reduce the energy consumption for both types of media.

5.2 Implications on the Streaming Service

Reserving a service slack of t_{bs} time units in T_d for best-effort service shortens the standby time of the disk and thus decreases energy saving. To compensate for the energy loss, we present two design approaches the designer can implement. The first approach extends T_d by enlarging the primary buffer capacity by $t_{bs} \cdot r_s$, resulting in a standby period length as if no best-effort service exists. The second approach, on the other hand, turns the NVM into a compensator for the energy consumed by the disk to service best-effort requests. The latter approach is referred to as *the compensating approach*, whereas the former is referred to as *the non-compensating approach*.

Since the best-effort period is a percentage of the streaming period ($t_{bs} = \gamma \cdot T_d$), we calculate T_d first as shown in Section 4.1, assuming no best-effort traffic. Then, we calculate the new capacity of the primary buffer and thus the new energy consumption of every component. In the following, we detail the models of both approaches.

5.2.1 The Non-compensating Approach

In this approach, the primary buffer capacity calculated in Section 3.2 is enlarged by $t_{bs} \cdot r_s$, the amount of data that the buffer has to supply while the disk is servicing best-effort requests. The primary buffer capacity becomes:

$$\begin{aligned} B'_{pm} &= B_{pm} + t_{bs} \cdot r_s \\ &= \alpha \cdot B_{be} + \gamma \cdot T_d \cdot r_s \end{aligned} \tag{10}$$

Here, T_d corresponds to the previously calculated refill period, without any best-effort services (see Eq. 7 of Section 4.1). From Fig. 7 we observe that only the read/write time changes, whereas the standby time does not change, because only the overhead energy is compensated for as in the best-effort–free case.

The active time period is updated as follows:

$$\begin{aligned} t'_{RW} &= \frac{B'_{pm}}{r_d - r_s} \\ &= t_{RW} + t_{bs} \cdot \frac{r_s}{r_d - r_s}, \end{aligned}$$

which yields an average disk power dissipation per period T'_d, after appropriate substitution of $E'_d = P'_d \cdot T'_d$, as:

$$\begin{aligned} T'_d &= T_d + (t'_{RW} - t_{RW}) + t_{bs} \\ &= T_d + t_{bs} \cdot \frac{r_d}{r_d - r_s} \end{aligned} \tag{11}$$

$$\begin{aligned} E'_d &= E_{oh} + P_{RW} \cdot t'_{RW} + P_{RW} \cdot t_{bs} + P_{sb} \cdot t_{sb} \\ &= E_d + P_{RW} \cdot t_{bs} \cdot \frac{r_d}{r_d - r_s} \end{aligned} \tag{12}$$

The last term of the disk energy consumption involves a scaled version of the minimum required energy to service best-effort requests: $P_{\text{RW}} \cdot t_{\text{bs}}$. The disk energy to service best-effort requests has been included for the sake of a fair comparison, although it is, strictly speaking, not related to the streaming energy. In the next section, we introduce an alternative compensating approach, after which we compare the disk energy of both approaches. Similarly, the energy of the NVM and DRAM in NVMBA respectively DRAMBA increases with the new capacity.

5.2.2 The Compensating Approach

The compensating approach prevents underflow of the primary buffer, and compensates for the energy consumed by the disk to service best-effort requests. As a result, putting the disk standby should not only compensate for the overhead energy, but also for the best-effort energy. Thus, Inequality (2) is rewritten:

$$P_{\text{id}} \cdot \hat{t_{\text{id}}} \geq t_{\text{sb}} \cdot P_{\text{sb}} + E_{\text{oh}} + t_{\text{bs}} \cdot P_{\text{RW}} \tag{13}$$

The break-even period is:

$$t''_{\text{be}} = \frac{E_{\text{oh}} + t_{\text{bs}} \cdot P_{\text{RW}} - t_{\text{oh}} \cdot P_{\text{sb}}}{P_{\text{id}} - P_{\text{sb}}} \tag{14}$$

$$= t_{\text{be}} + t_{\text{bs}} \cdot \frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}} \tag{15}$$

and its corresponding break-even buffer capacity ($B''_{\text{be}} = t''_{\text{be}} \cdot r_{\text{s}}$) is:

$$B''_{\text{pm}} = \alpha \cdot B''_{\text{be}}$$

$$= B_{\text{pm}} + t_{\text{bs}} \cdot r_{\text{s}} \cdot \frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}}. \tag{16}$$

Note that with respect to the non-compensating approach (Eq. 10), the second term in Eq. 16 is scaled with a factor larger than 1. We scale the primary buffer based on the break-even buffer to increase the energy saving. Unlike the previous approach, the compensating approach updates the read/write time as well as the standby time as illustrated in Fig. 7. The power dissipation is calculated as detailed in Section 4.1 after substituting the new capacity of the primary buffer in Eq. 6. The active period time and standby period are updated as follows:

$$t''_{\text{RW}} = \frac{B''_{\text{pm}}}{r_{\text{d}} - r_{\text{s}}}$$

$$= t_{\text{RW}} + t_{\text{bs}} \cdot \frac{r_{\text{s}}}{r_{\text{d}} - r_{\text{s}}} \cdot \frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}},$$

and

$$t''_{\text{sb}} = \frac{B''_{\text{pm}}}{r_{\text{d}} - r_{\text{s}}} - t_{\text{bs}} - t_{\text{oh}}$$

$$= t_{\text{sb}} + t_{\text{bs}} \cdot \frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}} - t_{\text{bs}},$$

which yields an average disk power dissipation per period T''_{d}, after appropriate substitution of $E''_{\text{d}} = P''_{\text{d}} \cdot T''_{\text{d}}$, as:

$$T''_{\text{d}} = T_{\text{d}} + (t''_{\text{RW}} - t_{\text{RW}}) + (t''_{\text{sb}} - t_{\text{sb}}) + t_{\text{bs}}$$

$$= T_{\text{d}} + t_{\text{bs}} \cdot \frac{r_{\text{d}}}{r_{\text{d}} - r_{\text{s}}} \cdot \frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}}, \tag{17}$$

$$E''_{\text{d}} = E_{\text{oh}} + P_{\text{RW}} \cdot t''_{\text{RW}} + P_{\text{RW}} \cdot t_{\text{bs}} + P_{\text{sb}} \cdot t''_{\text{sb}}$$

$$= E_{\text{d}} + P_{\text{RW}} \cdot t_{\text{bs}} \cdot \left(\frac{r_{\text{s}}}{r_{\text{d}} - r_{\text{s}}} \cdot \frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}} + 1 \right)$$

$$+ P_{\text{sb}} \cdot t_{\text{bs}} \cdot \left(\frac{P_{\text{RW}}}{P_{\text{id}} - P_{\text{sb}}} - 1 \right) \tag{18}$$

Similarly, the energy of the NVM and DRAM in NVMBA respectively DRAMBA increases with the new capacity.

6 Evaluation

We have implemented the analytical models presented in the previous sections in Matlab to evaluate all relevant architectural decisions. This section discusses the energy consumption of DRAMBA and NVMBA and compares them. A series of studies follows in the next section that evaluates the energy difference with respect to workload and setup parameters. Throughout all evaluation studies, flash memory serves as the NVM in NVMBA for the reasons mentioned in Section 2.

6.1 Detailed Comparison

We compare the average power dissipation of each architecture in one cycle T_{d} as a function of the capacity of the primary buffer ($B_{\text{pm}} = \alpha \cdot B_{\text{be}}$). The capacity of the break-even buffer is the same in both architectures,

since it depends on the disk and the streaming demand, which we fix for both architectures. Therefore, we always use α to express the amount of pre-fetching.

Setup In this study, we consider a mobile streaming device on which a user streams a video of a typical bit rate of 2048 Kbps from the disk drive. A Hitachi 1.8-inch Travelstar C4K40 hard disk drive (HDD) [4] serves as the backing store in both architectures. The NVM is represented by three SanDisk CompactFlash Extreme-III cards. These cards operate in parallel to achieve a higher aggregate throughput than the disk, so that flash is not a bottleneck. Finally, Micron's DDR SDRAM [16] serves as the primary and secondary buffer in DRAMBA and NVMBA, respectively. Table 2 lists the settings of the disk drive and the flash memory.

Assumptions Unless stated otherwise, we assume that streams of video or audio are stored on the middle track of the disk drive, thus reporting on the average case. Further we assume that the read throughput of the flash is equal to the write throughput. We take into account the erasing overhead of flash by assuming a low absolute throughput. Recently, Intel and Micron developed a technology for NAND flash, whereby a flash chip can reach throughput of up to 200 MB/s for reading data and 100 MB/s for writing data [1]. This technology will cut on costs to realize our proposed NVMBA with flash memory. Also, we assume no best-effort service ($\gamma = 0$). We fix $\beta = 10$ due to the minor contribution of its energy. As we see shortly, even having a DRAM capacity that is one order of magnitude larger than the real-time buffer (i.e., $\beta = 10$), DRAM energy is four orders of magnitude smaller than the total energy of the system, thanks to flash.

Table 2 Characteristics of the 1.8-inch disk and CF Extreme-III card.

Parameter		1.8″ HDD	Flash
Throughput	[Mbps]	187.2	240
Spinup power	[W]	1.485	–
Seek power	[W]	1.122	–
Access power	[W]	1.155	0.2 × 3
Spindown power	[W]	0.330	–
Idle power	[W]	0.330	–
Standby power	[W]	0.099	0.005
Spinup time	[s]	3	–
Seek time	[s]	0.015	–
Spindown time	[s]	0.5	–
Overhead time	[s]	–	0.002

6.1.1 Buffer Capacities

The capacity of the primary buffer (Flash in NVMBA and DRAM in DRAMBA) is approximately 39 Mb (mega*bit*). The capacity of the secondary buffer (i.e., DRAM in NVMBA) is approximately 4 Kb. These capacities are for $\alpha = 1$ and $\beta = 1$, respectively. Due to the use of a flash primary buffer, the DRAM capacity drops by four orders of magnitude in NVMBA compared to DRAMBA.

6.1.2 Power Dissipation

The primary buffer pre-fetches large amounts of data from the disk drive. Here, we compare the influence of pre-fetching for $\alpha > 1$ on the average power dissipated in one refill cycle (T_d). We let α range over $[1 - 10)$ and study the resulting power dissipation for each architecture. This range suffices to show the trends for both architectures (see Fig. 8).

NVMBA—Fig. 8a shows that by using a flash capacity that is just twice as large as the break-even buffer (i.e., $\alpha = 2$), the average power drops about 100 mW (approximately 35%). An additional decrease of about 80 mW is possible at $\alpha = 5$ (approximately 55% relative to $\alpha = 1$). As α increases, the number of spinups, seeks, and spindowns decreases, decreasing the amount of overhead that can be saved on. This corresponds to Amdahl's law, for which the power curve flattens for large values of α.

Increasing β results in larger DRAM capacities, and thus more power dissipation. Although the DRAM capacity is scaled up ten times (i.e., $\beta = 10$) in Fig. 8, its capacity remains in the order of tens of kilobits and its power dissipation is in the order of a few micro Watts. In contrast, the total power dissipation is in the order of hundreds of milliwatts. This explains why the DRAM power contribution disappears in Fig. 8a.

Figure 8a also shows a constant contribution of flash to the total power distribution. This is unsurprising, since increasing the mount of pre-fetching into flash capacity increases proportionally to the amount of time during which it sustains video streaming, keeping the ratio constant.

DRAMBA—Unlike NVMBA, DRAMBA has a maximum energy saving at $\alpha = 4$, where the total energy consumption is minimal. DRAM energy consumption worsens for values of α larger than 4, because DRAM retention energy increases proportionally with its increasing capacity. DRAMBA has a minimum power dissipation of approximately 0.22 W, whereas NVMBA can drop below 0.15 W.

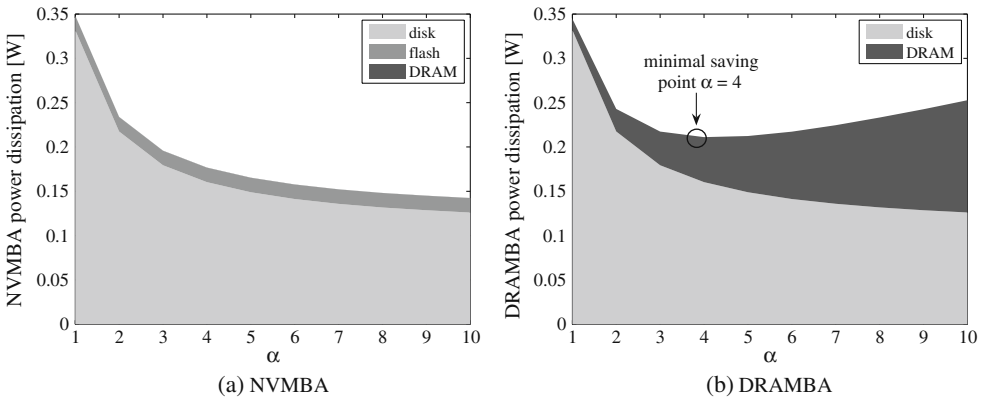


Figure 8 Average power dissipation in one cycle (T_d) of the respective architectures, differentiated to components (**a**, **b**).

6.1.3 Difference in Energy Consumption

Figure 9 plots the difference in energy consumption for the 2048 Kbps between NVMBA and DRAMBA calculated as $\frac{E_{\text{DRAMBA}}-E_{\text{NVMBA}}}{E_{\text{DRAMBA}}}$. NVMBA consumes more energy for $\alpha = 1$ (i.e., break-even capacity). The break-even capacity (1) does not achieve energy saving in either architecture and NVMBA (2) incurs additional energy to read and write data to/from flash. For these two reasons, NVMBA consumes more energy. On the other hand, for all $\alpha > 1$ NVMBA consumes less energy, because NVMBA saves on the DRAM retention energy. The saving trend for large values of α is, in fact, steeper than the actual saving of NVMBA (compare Fig. 8a to Fig. 9). This is because the energy consumption of DRAMBA worsens after $\alpha = 4$, making the trend look similar to that for small values of α. The saving at $\alpha = 4$ (the maximum saving point of DRAMBA) is 17%. Further increase in energy saving in NVMBA is possible but at larger setup cost.

Bit rate [Kbps]	128	2048	2×2048	10×4096
Buffer capacity [Mb]	2.33	37.14	74.27	742.66

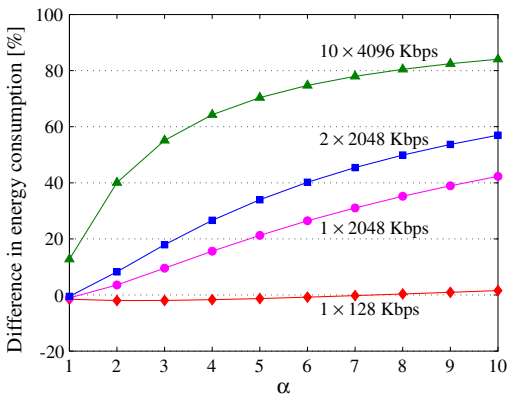


Figure 9 Relative difference in energy consumption between NVMBA and DRAMBA ($\frac{E_{\text{DRAMBA}}-E_{\text{NVMBA}}}{E_{\text{DRAMBA}}}$) for various streaming demands. The *table* above lists the primary buffer capacity at $\alpha = 1$ for each streaming demand. The larger the difference, the better NVMBA.

7 Energy-saving Merit of NVMBA

This section studies the influence of the workload and setup parameters on the energy saving of NVMBA relative to DRAMBA with respect to (1) the streaming demand, (2) the disk form factor, (3) the best-effort service provision, and (4) the stream location on the disk. Thus, we can determine the energy-saving merit of NVMBA.

7.1 Streaming Demand

The streaming demand is the number of concurrent streams that are played from/to the disk drive, and the bit rate of each of these streams. Streams can be audio and video of various qualities, with corresponding different bit rates. We evaluate the energy saving that NVMBA can achieve compared to DRAMBA for different streaming demands. We choose 128 Kbps and 2048 Kbps bit rates as single audio and video streaming, respectively. For multiple concurrent streaming, we choose 2 × 2048 Kbps and 10 × 4096 Kbps as aggregate bit rates. We take the extreme case of 10 × 4096 Kbps to project the energy saving of NVMBA versus the worst-case flash lifetime in Section 8.

Figure 9 shows the difference in energy consumption between the two architectures for the four streaming demands. We use the same hardware setup as in Section 6.1. For audio streaming at 128 Kbps, Fig. 9 shows that NVMBA consumes more energy than DRAMBA for two reasons: firstly, audio playback requires a small buffer capacity (B_{pm}), so that DRAM retention energy is not an issue, secondly, NVMBA incurs additional energy to read and write to the additional buffering level, increasing its energy consumption.

Noticeably, for the first three bit rates, NVMBA also consumes more energy at $\alpha = 1$ because of the additional buffering level. On the contrary, because of the huge streaming demand of 10×4096 Kbps, NVMBA consumes 17% less energy at $\alpha = 1$ already. This is because of the large amount of DRAM (approximately 90 MB dissipating 0.3 W) in DRAMBA that is required for buffering. In fact, $\alpha = 1$ is the minimum energy consumption point for DRAMBA, rendering larger values of α non-optimal. That means for this particular workload the disk drive should be left spinning the whole time, since pre-fetching achieves no saving and is thus infeasible. In contrast, NVMBA saves more energy as α increases. Figure 9 confirms the fact that the higher the streaming demand, the more energy is saved by NVMBA relative to DRAMBA.

7.2 Disk form Factor

Another parameter that influences the energy saving, is the form factor of the disk drive. Recall from Section 3.2 that in order to save energy the primary buffer of the disk drive has to be larger than the break-even buffer to compensate for the overhead energy. When the disk form factor increases, the overhead increases, since the disk mechanical inertia increases. As a consequence, a larger primary buffer is required for compensation.

The disk form factor also indirectly influences the data rate of the primary buffer. That is, as the form factor increases, the disk sustained data rate (r_d) increases, demanding a buffer that supports a higher data rate, so that it does not become a performance bottleneck. This subsection studies the influence of four commodity disk form factors on the energy consumption of both architectures when streaming at 2048 Kbps. These are 1.0-, 1.8-, 2.5-, and 3.5-inch disk drives. Table 3 summarizes the characteristics of the four disk drives with their respective flash configurations.

The disk drives in Table 3 span over three years (2003–2006), which suggests that they differ in more than one respect. Any improvement in their electronics have negligible influence on the overall energy consumption we target in this study, while mechanical differences are accounted for. The energy consumption due to electronics is one order of magnitude smaller than the energy due to mechanics; compare the standby power to the access power of all disks in Table 3.

Figure 10a shows the absolute difference in power dissipation of a range of disk drives when applied in either architecture. It shows that the larger the disk, the larger the buffer capacity. As a consequence, DRAM energy in DRAMBA increases when the difference in the average power dissipation increases. Only the 1.0-inch disk DRAMBA dissipates less power than NVMBA, because of the small DRAM capacity required, rendering additional buffering level—as in NVMBA—not worthwhile.

Figure 10b plots the relative difference in energy consumption between the two architectures as $\frac{E_{\text{DRAMBA}} - E_{\text{NVMBA}}}{E_{\text{DRAMBA}}}$. This graph shows that when the primary buffer increases, the energy saving increases, but the 1.0-inch trend changes after $\alpha = 6$ where DRAM energy becomes substantial. As a result, DRAMBA energy increases relative to NVMBA. Further, the relative saving with the 3.5-inch disk is less than with the 1.8- and 2.5-inch disks although it differs in absolute sense. Because of the inherently large amount of energy consumed by the 3.5-inch disk, all values of $\alpha \in [1, 10]$ are feasible for DRAMBA. As a result, the energy

Table 3 Characteristics of the Hitachi disk drives with their respective flash modules to prevent performance bottlenecks.

Parameter		1.0″ HDD MD [7]	1.8″ HDD C4K40 [4]	2.5″ HDD E7K100 [5]	3.5″ HDD 7K500 [6]
Model		2005	2003	2005	2006
Throughput	[Mbps]	96.0	187.2	318.50	383.2
Spinup power	[W]	1.023	1.485	5.5	29.5
Seek power	[W]	0.660	1.122	2.3	10.0
Access power	[W]	0.990	1.155	2.0	11.0
Spindown power	[W]	0.215	0.330	1.8	8.0
Idle power	[W]	0.215	0.330	0.85	8.0
Standby power	[W]	0.043	0.099	0.2	1.0
Spinup time	[s]	0.5	3.0	4.0	15.0
Seek time	[s]	0.012	0.015	0.016	0.020
Spindown time	[s]	0.5	0.5	1.0	5
Flash configurations					
Throughput	[Mbps]	160	240	320	400
Access power	[W]	0.4	0.6	0.8	1.0
Standby power	[W]	0.005	0.005	0.005	0.005
Overhead time	[s]	0.002	0.002	0.002	0.002

difference between the two architectures boils down to saving on the DRAM energy only, which explains the relatively small energy saving for the 3.5-inch disk.

7.3 Best-effort Provision

This section studies the influence of best-effort service on the difference in energy saving between both architectures. Recall from Section 5 that we assume the best-effort period to be relatively short and not exceeding 10% of the total refill cycle, $\gamma < 0.1$.

To study the energy difference, we use our default hardware and workload setup explained in Section 6.1. The difference is studied using the compensating and non-compensating approaches. Two different primary-buffer capacities are chosen, corresponding to $\alpha = 2$ and $\alpha = 4$, both are feasible for DRAMBA.

Figure 11a plots the relative difference in energy consumption, calculated as $\frac{E_{\text{DRAMBA}} - E_{\text{NVMBA}}}{E_{\text{DRAMBA}}}$, when best-effort energy is uncompensated for. In this case, the disk energy is increased, since the disk accesses best-effort data and additional streaming data to prevent underrun during best-effort periods. Also, the primary-buffer capacity increases to hold the additional streaming data, resulting in additional retention energy if this buffer is a DRAM. The disk energy increases proportionally to γ in both architectures, whereas the DRAM energy does but just in DRAMBA. The actual energy difference boils down to the amount of energy the DRAM consumes to retain the additional streaming data, whereas both architectures are equal at disk energy, the prominent one (see Fig. 8). As a result, the absolute difference has a slow increasing trend (due to DRAM energy) with respect to γ. However, the total energy has a faster increasing trend (due to disk and DRAM energy) with respect to γ, which explains the decreasing trend of the relative difference in Fig. 11a.

The figure also shows the superior saving of NVMBA relative to DRAMBA as the pre-fetching level increases; compare 4% to 13% saving when pre-fetching is doubled. The decaying trend suggests the lesser importance of the saving on DRAM energy by NVMBA, since disk energy becomes more prominent due to the increasing best-effort demand.

Unlike the non-compensating approach, the compensating approach shows an increasing trend of the relative energy difference with respect to γ as Fig. 11b shows. This is because the disk energy is held constant with respect to γ, since it is compensated for. This, however, moves the energy problem from the disk to the primary buffer, since its capacity increases to compensate for the best-effort energy. In NVMBA, this is not a problem since retention has no cost, but it is a problem in DRAMBA. In fact, for (relatively) large values of γ, compensating best-effort energy is infeasible (like for large values of α). This explains the huge relative difference depicted in Fig. 11b for $\alpha = 4$.

In summary, due to its NVM, NVMBA is capable of sustaining streams for some period of time while the disk is servicing best-effort. Further, it is capable of compensating for the disk energy consumed to service best-effort requests. DRAMBA, however, falls short on both accounts due to the DRAM retention energy.

7.4 Data Placement on Disk

Data tracks are laid out on disk platters concentrically. In our model we assume that (1) the further the track from the center, the larger its storage capacity. Platters rotate at a constant angular velocity, dissipating a fixed amount of power. Hence, (2) the further the track from the center, the higher the linear velocity and thus its sustained data rate. Given (1) and (2) it follows

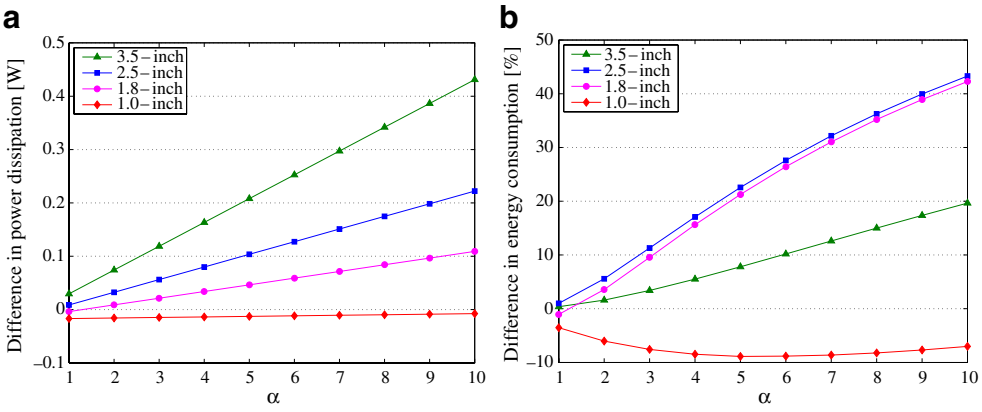


Figure 10 Absolute and relative difference in power dissipation between DRAMBA and NVMBA in one refill cycle for four different disk drives (**a**, **b**). The *table* above lists the primary buffer capacity at $\alpha = 1$ for each disk form factor.

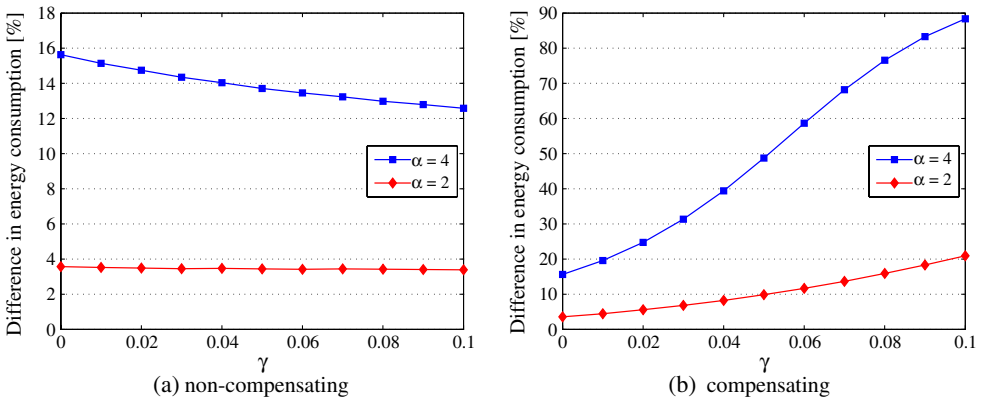


Figure 11 Difference in energy consumption between DRAMBA and NVMBA for different best-effort service fraction of the refill cycle when streaming at 2048 Kbps (**a**, **b**).

that the further the track from the center, the lower the energy consumed to access data (i.e., joule-per-bit cost). A body of work exists already that exploits this difference in energy cost by intelligently placing frequently demanded streams and/or streams of high bit rate on the outer tracks of disk drives [12].

As explained in Section 4.1, with the increase of α (the primary-buffer capacity), the influence of the overhead energy (such as spinup energy) on the total energy decreases, and the access energy becomes more prominent. Therefore, cutting down on the prominent access energy becomes more important. Put otherwise, NVMBA makes stream placement techniques more profitable, since it minimizes all overheads associated with data access. In contrast, these techniques are not always profitable with DRAMBA, because not all values of α are feasible as shown in the previous subsections.

Parameter	innermost track	outermost track
seek time [s]	0.026	0.004
data rate [Mbps]	134.4	240.0

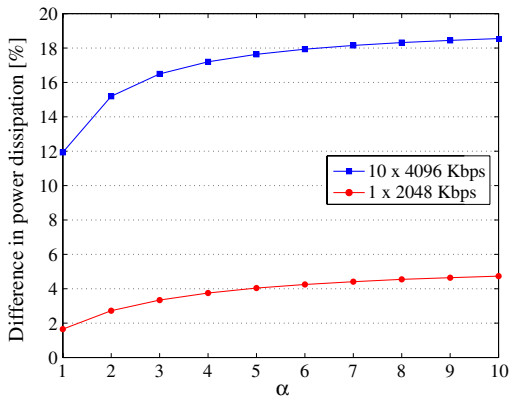


Figure 12 Difference in power dissipation when streaming from the innermost and outermost tracks, calculated as $\frac{E_{\text{inner}}-E_{\text{outer}}}{E_{\text{inner}}}$. The *table* above lists the seek time and data rate for each track.

We carry out an experiment to compare the power dissipation of NVMBA when streaming from the innermost track and the outermost track of the 1.8-inch disk at 2048 Kbps and 10 × 4096 Kbps. Figure 12 plots the relative difference in power dissipation (calculated as $\frac{E_{\text{inner}}-E_{\text{outer}}}{E_{\text{inner}}}$) between both tracks for the two streaming demands. This confirms that as α increases the significance increases, since the access energy becomes dominant. It also shows the importance of placement as the streaming demand increases; compare 5% energy saving to 18% for 2048 Kbps and 10 × 4096 Kbps, respectively.

8 Reliability of NVMBA

Saving energy requires frequent cycles, which includes spinning the disk drive on and off repeatedly and overwriting the flash memory extensively. Compared to DRAMBA, NVMBA can maintain longer cycle periods by exploiting larger buffers efficiently. Nonetheless, the disk and the flash lifetime remain important issues to guarantee a reliable system for the expected lifetime of the system. This section studies the lifetime of these two critical components. We give some background on disk and flash reliability first.

8.1 Background

Disk Drive There are three methods to express disk drive reliability: (1) Mean Time To Failure (MTTF), (2) Mean Time Between Failures (MTBF), and (3) duty cycle rating, i.e., the number of times the drive can be spun down before the probability of failure on spin up becomes larger than 50%. The duty cycle ratio is more relevant than MTTF or MTBF in energy-conservative streaming architectures. Although aggressively spun down disk drives save energy, it results in an accelerated consumption of the duty cycles of the drive.

Disk manufacturers report on the duty cycle rating of their disk drives. The rating depends on the head parking mechanism implemented in the drive. There are two parking mechanisms, namely Contact Start/Stop (CSS) and the Ramp Load/Unload [15]. The Contact Start/Stop parks the head in the central zone of the platter, whereas the Ramp Load/Unload parks the head completely outside the platter area on a plastic ramp. The former mechanism is typically deployed in 3.5-inch disk drives, whereas the latter is typical in 1.8-inch and 2.5-inch drives, because of its better shock resistance. Duty cycles are in the range of 50,000 cycles for 3.5-inch disks and, 500,000 cycles for 1.8- and 2.5-inch disks. The large difference is mainly due to stiction effects.

Flash Memory Flash memory has an endurance problem; a flash cell can be rewritten for a fixed number of cycles (100,000–1,000,000 [8]). After that, its reliability to retain data drops dramatically. To extend its lifetime, wear-levelling algorithms map writes to flash, so that all cells are rewritten the same number of cycles. When used as a FIFO buffer, wear levelling of flash is straightforward by the inherent circular nature of the buffer refill.

8.2 Evaluation

We estimate the lifetime of the components of NVMBA by measuring their duty cycle consumption. Every time the disk is spun up, one duty cycle is consumed from its lifetime and so from flash lifetime, since every spin up is associated with a complete write to the flash. We carry out a worst-case study to compare the energy merit with the flash lifetime. We consider two workloads: (1) streaming two videos at 2048 Kbps for 4 hours a day, and (2) streaming ten videos at 4096 Kbps for 12 hours a day to project the potential energy saving versus flash lifetime. We use the same hardware setup as described in Section 6.1.

Figure 13 plots the flash lifetime versus the system-level energy saving of NVMBA for the two workloads. Despite the extensive usage of flash in these workloads, a 100 MB and 1 GB flash suffices for 3.5 and 1.2 years, for the 2×2048 Kbps and 10×4096 Kbps workloads, respectively. A 1.8-inch disk drive will live three times as long as the flash, since its duty-cycle rating is 300,000 cycles (compared to 100,000 of flash). Note that the relative energy saving is taken to the maximum energy savings of DRAMBA for each workload, which are at $\alpha = 3$ and $\alpha = 1$, respectively. We present the saving in terms of the overall system saving to give the overall extension in the battery lifetime. To calculate the system-level saving, we assume that the memory hierarchy in DRAMBA consumes 33% respectively 50% of the total energy consumed by the system. This explains the larger saving for the second workload compared to the first one.

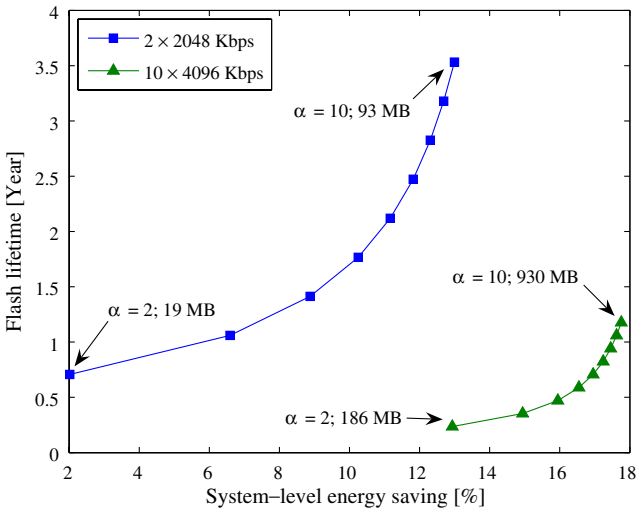


Figure 13 Flash lifetime versus system-level energy saving for two worst-case streaming demands in PC-like environments.

Figure 13 also shows that enlarging the flash capacity to achieve more energy saving increases its lifetime. In fact, we can see that after a certain flash capacity the energy saving starts saturating (i.e., little improvement) whereas a clear extension of flash lifetime is still possible. This increase is particularly important, since the flash lifetime can be shorter (e.g., 0.5–3.5 years) than the lifetime of a mobile system, assuming it is about 6 years. Taking this factor into account, the designer should mount enough flash capacity into the system to guarantee an energy-efficient system for a desired lifetime. For example, by just mounting a 200 MB flash, the system lives for 7 years with a 13% overall reduction in energy. A 10 GB flash is needed in the extreme case for a 12-year lifetime at an energy saving of 18%. Summarizing, a 13% and 18% overall reduction in direct energy costs is feasible with reasonably priced flash memory.

9 Experimental Validation

In this section, we validate the reported energy savings of DRAMBA and NVMBA that resulted from the numerical evaluation. Instead of implementing a full system, we reuse an existing setup in our laboratory and measure dedicated design points. In addition, we examine the influence of the overhead due to excessive

number of refills (between NVM and DRAM) on the energy.

Our experimental platform is an HP iPAQ H2215 PDA that runs Linux. A Hitachi 4 GB Microdrive (approximately 7 MB/s) and a SanDisk 2 GB CompactFlash Extreme-III card (approximately 20/10 MB/s read/write throughput) are chosen as representatives of the disk and the flash. The PDA has a single CompactFlash interface, into which we can plug either the Microdrive or the flash card. We measure the energy consumption of the storage device (across the CF interface) and the rest of the system separately.

9.1 Methodology

For DRAMBA, we measure with different DRAM capacities (i.e., different pre-fetching levels). Given that we use a standard PDA, the amount of physical DRAM cannot be changed. Changing the DRAM capacity is achieved in software by allocating a portion of the total DRAM. Henceforth we can measure the influence of the DRAM capacity only on the Microdrive energy, whereas the DRAM energy is bounded by the energy consumption of the whole physical 64 MB DRAM.

We also adopt this method to evaluate the Flash–DRAM part of NVMBA for different DRAM capacities. Here, the Microdrive is replaced by the CF card. The Disk–Flash part, however, cannot be directly measured in our setup. Therefore, we measure for one point where the disk spins up just once to fill the flash with the whole stream and then spins off (i.e., maximizing pre-fetching into the flash). Thus, the Disk–Flash energy measurement boils down to measuring (1) the energy of one read from the disk and (2) one write to the flash of the stream size. We can actually obtain both measurements from the previous Disk–DRAM and Flash–DRAM measurements, respectively. Adding up the Flash–DRAM energy, disk read energy, and flash write energy, we can obtain an upper bound of the total energy consumed by NVMBA.

9.2 Results

We implemented a streaming emulator that reads data from the storage device at a predefined rate into an allocated buffer of a predefined capacity. We experimented with a typical rate range for PDA-like devices of 32–512 Kbps.

For DRAMBA, we measure the energy consumed by the disk as well as the energy consumed by the rest of the system for different DRAM capacities, ranging from 32 KB to 256 KB. As the physical DRAM capacity cannot be changed, the energy consumed by the rest of the system virtually does not change (11.9 to 11.8 joules for 32 KB to 256 KB). The slight difference is due to the difference in the number of refills and thus the incurred transfer overhead. Since the whole DRAM is always on, the energy measured for the rest of the system is in fact the worst case. However, the energy measured for the disk varies from 11.6 to 6.3 joules for 32 KB to 256 KB: it increases as the DRAM capacity decreases, because the disk is started and stopped more often.

We measure the energy consumption for NVMBA for the same range of DRAM capacities. The energy consumed by the rest of the system is virtually the same (12.2 to 12.1 joules for 32 and 256 KB, respectively) as explained before. The energy consumed by the Disk–Flash part is 7.1 joules for 32 KB through 256 KB, because, unlike the disk, flash has no spinup energy. Therefore, varying the DRAM capacity has no influence on the energy. In fact, the main contributor to the energy consumed by the Disk–Flash part is the energy to spinup, seek, read from, and spindown the disk.

The previous discussion is summarized by the following findings:

1. Deploying large buffers for pre-fetching saves significantly on disk energy. This is in agreement with the related work.
2. NVMBA consumes 17% less total energy than DRAMBA when deploying 32 KB DRAMs in both architectures.
3. NVMBA consumes 5% less total energy than DRAMBA when deploying 256 KB DRAM in DRAMBA and 32 KB DRAM in NVMBA.
4. The overhead due to large number of refills between the flash and the DRAM has a marginal influence on the total energy consumption.

10 Related Work

A body of work is available on enhancing the energy-efficiency of the storage hierarchy. Hardware and software techniques reduce disk energy and DRAM energy in the conventional storage hierarchy.

With regard to hardware techniques, disk manufacturers reduce disk energy in several ways including reducing the weight of the platters and producing multi-speed disks. DRAM technology has also advanced. Manufactures reduce the operating power of DRAM significantly by improving their process, as a result from, among others, smaller capacitance, smaller voltage swings, and smaller leakage currents. Also,

manufacturers produce components that exhibit multiple power states, so that energy consumption can be controlled by either the host system or the firmware of the component itself, such as the Adaptive Battery Lifetime Extender (ABLE) from Hitachi [14]. Modern DRAM modules offer power modes to reduce the retention power [16].

Beside the hardware techniques, many software techniques exist that reduce the energy consumption of the disk, the DRAM, and even both at the same time. Exploiting the multi-speed feature of disk drives, Gurumurthi [13] lowers the disk speed in short idle periods to increase the energy saving. Weissel et al. [25] propose and experiment with energy-aware interfaces, through which the operating system obtains information about the actual state of each component in the storage system and urgency/seriousness of I/O requests. Based on this high-level information, the operating system can defer requests till the appropriate moment when the disk drive should be spun up, and thus avoid non-urgent request from spoiling energy. Son et al. [24] restructure application code, inserting prefetching hints to drive the disk into spin-off mode or a low-speed mode, so that energy is reduced while performance is not impacted. For streaming workloads, which have a high locality of reference, future data can be predicted and thus pre-fetched. In the disk–DRAM hierarchy, Mesut el al. [22] pre-fetch as much streaming data as possible into DRAM to increase the spin-off period of the disk, reducing the total energy due to disk and DRAM combined.

The advent of flash memory with the non-volatility and solid-state features has opened a new landscape to enhance energy efficiency. Flash has been proposed as an additional caching level between the disk and the DRAM in the conventional memory hierarchy. Two different hardware solutions realize the caching vision: (1) Samsung hybrid hard disk drive (HHD) [3], and (2) Intel Turbo Memory technology [21]. The essential difference between the two solutions is that flash is unexposed to the host in HHDs, unlike Turbo Memory, which mounts flash as a separate module on the motherboard. Nonetheless, the main question, which has seen a lot of research, is finding the proper policies that drive the data in and out of the flash in either technology. Proper policies should populate flash with hot data and redirect write traffic, so that the disk drive can stay in spin-off for long periods to save energy. Simulating HHDs, Bisson et al. [11] show savings up to 40% on the disk energy for workloads from personal computers. Shimpi [23] studies the energy saving of Intel Turbo Memory against several workloads and show that savings up to 16% on the system energy are possible. In web servers, Kgil et al. [17] use flash as a cache for DRAM to offload infrequently accessed files (approximately 80% of the total accessed files). The authors show a significant reduction in the DRAM capacity, saving orders of magnitude on its retention energy.

Our research evaluates the merit of using flash memory in an environment that is different from web servers and personal computers. In a mobile multimedia environment, we use flash as a streaming buffer that is written repeatedly in a circular fashion. Our approach is an extension to that by Mesut et al. [22] and a complement to that by Bisson et al. [11] for mixed-media environments. NAND Flash has (1) better performance at large writes than small ones [10], (2) no moving parts, (3) short response time, unlike disk drives, (4) no retention energy, unlike DRAM, (5) lower cost than NOR flash, and (6) wide availability. Exploiting these promising characteristics, while taking into account its erasing overhead, we show that savings of 13% over the optimized conventional architecture are achievable for reasonably priced flash.

11 Conclusions

In this work, we study the energy merit to shift from the conventional storage hierarchy, composed of a disk drive and a DRAM (DRAMBA), to a new hierarchy, where an NVM serves as a buffer between the disk and the DRAM (NVMBA). The study targets popular streaming applications with a small best-effort provision (less than 10%). To evaluate our ideas, we build analytical models of the two architectures. We use NAND as the NVM buffer, because of its energy efficiency and widespread use. We show that NVMBA provides design points that reduce the energy consumption beyond the capability of DRAMBA, since it targets the combined disk energy and DRAM energy. We carry out several dedicated analytical studies to investigate the energy saving trend of NVMBA with respect to (1) the streaming demand, (2) the disk form factor, (3) the best-effort provision, and (4) the stream location on the disk.

When the streaming demand increases, the energy saving of NVMBA increases. For audio streaming DRAMBA outperforms NVMBA, because the audio workload renders additional buffering level unnecessary. If the disk form factor decreases, then the energy saving of NVMBA decreases. In case of the 1.0-inch disk NVMBA consumes more energy than DRAMBA, because of the small overhead energy of this drive. One key finding is that the relative energy savings achieved

with the 1.8- and 2.5-inch disk drives are larger than that with the 3.5-inch disk drive, despite of its larger overhead. The reason is that for the 3.5-inch disk drive the energy difference between the two architectures is due to the saving on the DRAM energy and not the disk energy.

Servicing a small percentage of best-effort traffic (less than or equal to 10%) has no influence on the energy consumption of NVMBA, but increases DRAMBA energy, since more DRAM capacity is needed. Further, in NVMBA we can turn the NVM into a compensator for the energy consumed by the disk to service best-effort requests.

In NVMBA, the energy saving is sensitive to the placement of the stream on the disk. The reason is that NVMBA reduces the overhead energy of the disk more than DRAMBA. As a result of this reduction, the access energy becomes dominant, so that reducing it becomes more substantial.

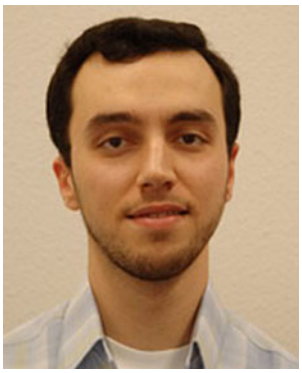
We carried out a worst-case reliability study of NVMBA when flash memory is used as NVM. For streaming of 10 concurrent videos each of bit rate 4096 Kbps for 12 hours a day, a flash memory of size approximately 950 MB lives for 1.2 years before it wears out, assuming a 100,000 erase cycles. A 1.8-inch disk drive, which typically has 300,000 duty cycles, lives three times as long as the flash memory. In this particular scenario, NVMBA saves as much as 18% on the system energy relative to DRAMBA.

Acknowledgements The authors thank the anonymous reviewers for their useful comments. Also, we thank Philips Research Laboratory in Eindhoven for providing us with the measurement setup. This research is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Ministry of Economic Affairs under project number TES.06369.

References

1. InformationWeek (2008). Intel and Micron claim to boost solid-state drive speeds by 500%. http://www.informationweek.com/news/showArticle.jhtml?articleID=206102135&subSection=News. Accessed February 2008.
2. Nanochip (2006). Nanochip develops MEMS-based storage. http://nanochipinc.com/tech.htm. Accessed November 2007.
3. Samsung (2008). Samsung's Hybrid HDD, FlashON. http://www.samsung.com/Products/HardDiskDrive/HybridHDD_FlashON/index.asp. Accessed June 2007.
4. Hitachi Global Storage Technologies (2003). OEM Hard Disk specifications for HTC424040F9AT00. Hitachi Travelstar C4K40 1.8-inch drive, November. Revision (7).
5. Hitachi Global Storage Technologies (2005). OEM Hard Disk specifications for HTE721010G9AT00. Hitachi Travelstar E7K100 2.5-inch drive, July. Version (1.0).
6. Hitachi Global Storage Technologies (2006). OEM Hard Disk specifications for HDS725050KLAT80. Hitachi Deskstar 7K500 3.5-inch drive, September. Version (1.5).
7. Hitachi Global Storage Technologies (2006). OEM Hard Disk specifications for HMS360606D5CF00. Hitachi Microdrive 1.0-inch drive, April. Revision (1.11).
8. International Technology Roadmap for Semiconductors (2006). International Technology Roadmap for Semiconductors (ITRS) 2006 update. Technical report, 2006. http://www.itrs.net/Links/2006Update/2006UpdateFinal.htm.
9. Benini, L., Bogliolo, A., & De Micheli, G. (2000). A survey of design techniques for system-level dynamic power management. *IEEE Transactions on Very Large Scale Integration Systems, 8*(3), 299–316, June.
10. Birrell, A., Isard, M., Thacker, C., & Wobber, T. (2007). A design for high-performance flash disks. *SIGOPS Operating Systems Review, 41*(2), 88–93.
11. Bisson, T., Brandt, S., & Long, D. (2007). A hybrid disk-aware spin-down algorithm with I/O subsystem support. In *Performance, computing, and communications conference, 2007. IPCCC 2007. IEEE Internationa*, 236–245, April.
12. Ghandeharizadeh, S., Kim, S. H., Shahabi, C., & Zimmermann, R. (1996). *Placement of continuous media in multi-zone disks* (chapter 2, pp. 23–57). Dordrecht: Kluwer Academic Publisher.
13. Gurumurthi, S. (2005). *Power management of enterprise storage systems*. PhD thesis, The Pennsylvania State University, University Park, Pennsylvania.
14. Heybruck, W. (2005). Enhanced adaptive battery life extender (ABLE).
15. Jacob, B., Ng, S. W., & Wang, D. T. (2008). *Memory systems (cache, DRAM, disk)* (chapter 17, pp. 636–637). San Francisco: Morgan Kaufmann.
16. Janzen, J. W. (2003). *TN-46-03 - Calculating memory system power for DDR*. Boise: Micron Technology.
17. Kgil, T., & Mudge, T. (2006). FlashCache: A NAND flash memory file cache for low power web servers. In *CASES'06: Proceedings of the 2006 international conference on compilers, architecture and synthesis for embedded systems* (pp. 103–112). New York: ACM.
18. Lantz, M. A., Rothuizen, H. E., Drechsler, U., Häberle, W., & Despont, M. (2007). A vibration resistant nonopositioner for mobile parallel-probe storage applications. *Journal of Microelectromechanical Systems, 16*(1), 130–139.
19. Lorch, J., & Smith, A. (1998). Software strategies for portable computer energy management. *IEEE Personal Communications, 5*(3), 60–73.
20. Lu, Y.-H., Chung, E.-Y., Šimunić, T., Benini, L., & De Micheli, G. (2000). Quantitative comparison of power management algorithms. In *Proceedings of the conference on design, automation and test in Europe (DATE)* (pp. 20–26). New York: ACM.
21. Matthews, J., Trika, S., Hensgen, D., Coulson, R., & Grimsrud, K. (2008). Intel turbo memory: Nonvolatile disk caches in the storage hierarchy of mainstream computer systems. *ACM Transaction on Storage, 4*(2), 1–24.
22. Mesut, Ö., van den Brink, B., Blijlevens, J., Bos, E., & de Nijs, G. (2005). Hard disk drive power management for multi-stream applications. In: *International workshop on software support for portable storage (IWSSPS)*, March.

23. Shimpi, A. L. (2007). Investigating Intel's Turbo Memory: Does it really work? www.anandtech.com/mobile/showdoc.aspx?i=3009. Accessed 5 August 2008.
24. Son, S. W., & Kandemir, M. (2006). Energy-aware data prefetching for multi-speed disks. In: *CF '06: Proceedings of the 3rd conference on computing frontiers* (pp. 105–114). New York: ACM.
25. Weissel, A., Beutel, B., & Bellosa, F. (2002). Cooperative I/O: A novel I/O semantics for energy-aware applications. In: *OSDI'02: Proceedings of the 5th symposium on operating systems design and implementation* (pp. 117–129). New York: ACM.



Mohammed G. Khatib received his Master degree in Computer Science from the Technical University of Braunschweig in 2004. He has worked at the Technical University of Braunschweig and the University of California at Santa Cruz. He is currently a PhD candidate at the University of Twente in the Netherlands.

His research interest is energy-efficient architectures and storage systems.

Pieter H. Hartel received his Master degree in Mathematics and Computer Science from the Free University of Amsterdam in 1978 and his PhD degree in Computer Science from the University of Amsterdam in 1989. He has worked at CERN in Geneva, and the Universities of Nijmegen, Amsterdam, and Southampton. He is currently a full Professor of Computer Science at the University of Twente. His research interest is distributed and embedded systems security.

Hylke W. van Dijk received a M.Sc. degree in 1993 in electrical engineering and a Ph.D. degree in 2004 from Delft University of Technology (TU Delft). After some years in industry he rejoined TU Delft in 1998 as a system architect in the Ubicom research programme. His research interests concentrate on systematic development of distributed embedded systems, in particular on the coordination among system components and system processes in a multidisciplinary environment. Hylke has been involved in projects ranging from augmented reality, software engineering, service discovery, and most recently, probe-based storage architectures. Currently he holds a post-doc position at the University of Twente.