Optimizing NAND Flash-Based SSDs via Retention Relaxation

Ren-Shuo Liu*, Chia-Lin Yang*, Wei Wu†
*National Taiwan University and †Intel Corporation

2/15/2012 USENIX FAST '12
NAND Flash in Reliable Storage

- Two main reliability specifications
  - Bit error rate (BER): $10^{-13} \sim 10^{-16}$
  - Data retention: 10 years (cycled to 10% of the max. endurance)
    1 year (cycled to 100% of the max. endurance)

- As NAND Flash’s density increases, its raw reliability degrades
  - Need to slow down writes to mitigate the worsening BER
  - Need stronger ECCs
    - When the BER $\geq 10^{-3}$, advanced ECCs such as LDPC (low-density parity-check) are required*

* S. Li and T. Zhang. Approaching the information theoretical bound of multi-level NAND Flash memory storage efficiency. IMW '09
Actual Retention Requirements

- Retention requirements in real-world applications are usually much shorter than a year

Retention breakdown of a TPC-C workload:
- Unknown (1%)
- $\leq$ 1 min (53%)
- 1 min ~ 1 hr (43%)
- 1 hr ~ 1 day (3%)
Our Contribution

• Retention Relaxation
  – Exploit the gap between retention specification vs. actual retention requirements to improve write speed or ECC cost/performance in Flash-based SSDs

| Industrial standards: 1 to 10 years | vs. | Actual requirements: days or shorter |
Outline

• Motivation
• **NAND Flash background**
• Main idea
• Methodology
• Evaluation
• Conclusions
NAND Flash Background

- NAND Flash memories
  - Composed of floating gate (FG) transistors
  - Injecting charge on the FG can adjust a transistor’s threshold voltage ($V_{th}$)
  - Different $V_{th}$ levels are used to represent different data

Flash cell structure

Example $V_{th}$ levels for 2-bit cells

<table>
<thead>
<tr>
<th>2-bit data</th>
<th>‘11’</th>
<th>‘10’</th>
<th>‘00’</th>
<th>‘01’</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{th}$</td>
<td>0</td>
<td>1.2</td>
<td>2.4</td>
<td></td>
</tr>
</tbody>
</table>
Programming NAND Flash

• Incremental step pulse programming (ISPP)*
  – Increase $V_{th}$ step-by-step with step increment = $\Delta V_p$

• Tradeoffs in ISPP
  – $\Delta V_p \uparrow \rightarrow$ fewer steps (faster)
  – $\Delta V_p \downarrow \rightarrow$ more precise control on $V_{th}$ (fewer write errors)

* Suh et al.. A 3.3 V 32 Mb NAND Flash memory with incremental step pulse programming scheme. JSSC '95
Probability density function of cells’ $V_{th}$

- Modeled using bell-shaped functions in the previous work*

$$P_k(v) = \alpha_0 \cdot e^{-\frac{(v-\mu_0)^2}{2\sigma_0^2}}, \quad k = 0$$

$$P_k(v) = \begin{cases} 
\alpha \cdot e^{-\frac{(v-\mu_k+0.5\Delta V_P)^2}{2\sigma^2}}, & \mu_k - \frac{\Delta V_P}{2} \leq v \leq \mu_k + \frac{\Delta V_P}{2} \\
\alpha \cdot e^{-\frac{(v-\mu_k-0.5\Delta V_P)^2}{2\sigma^2}}, & v > \mu_k + \frac{\Delta V_P}{2}
\end{cases}$$
Bit Error Rate vs. $V_{th}$ Distribution

- RBER is the integral of the $V_{th}$ distributions that fall into wrong states

\[
BER = \sum_{k=0}^{3} \left( \int_{-\infty}^{V_{R,k}} P_k(v) \, dv \right) + \int_{V_{th,k+1}}^{\infty} P_k(v) \, dv
\]

Error probability

- $00$' and $01$'
Outline

• Motivation
• NAND Flash background

• **Main idea**
  – Retention time vs. Write speed
  – Retention time vs. ECC

• Methodology
• Evaluation
• Conclusions
Retention Relaxation vs. Write Speed

- Shorter retention guarantee
  → Can tolerate more write errors
  → Allow larger $\Delta V_p$ in the ISPP
  → Faster write

![Graph showing BER of NAND Flash vs. Time]

- Required BER
- BER of NAND Flash

- Time (1 year)
- BER
  - $10^{-3}$
  - $10^{-4}$
  - $10^{-5}$
  - $10^{-6}$
  - $10^{-16}$
Retention Relaxation vs. ECC

- Shorter retention guarantee
  → Need to tolerate fewer retention errors
  → Allow less capability of the ECC
  → Simpler ECC design
Outline

• Motivation
• NAND Flash background
• Main idea
• **Methodology**
  – NAND Flash model
  – Retention analysis on real-world applications
  – Retention-aware system architecture
• Evaluation
• Conclusions
Model Extension

- Base NAND Flash model is not able to capture the charge-loss effect which causes the low-$V_{th}$ tail to widen over time*

* Arai et al.. Extended data retention process technology for highly reliable Flash EEPROMs of $10^6$ to $10^7$ W/E cycles, IPRS '98
Model Extension

- Model the standard deviation of the low-$V_{th}$ tail as a time-increasing function, $\sigma_{\text{low}}(t)$

<table>
<thead>
<tr>
<th>Low-$V_{th}$ tail</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base model</strong></td>
</tr>
<tr>
<td>$P_k(v) = \alpha \cdot e^{-\frac{(v-\mu_k+0.5\Delta V_P)^2}{2\sigma^2}}$</td>
</tr>
<tr>
<td><strong>Extended model</strong></td>
</tr>
<tr>
<td>$P_k(v, t) = \alpha(t) \cdot e^{-\frac{(v-\mu_k+0.5\Delta V_P)^2}{2\sigma_{\text{low}}(t)^2}}$</td>
</tr>
</tbody>
</table>

![Diagram showing distributions and parameters](image)
Modeling Results

$V_{th}$ Distribution vs. Time

- **Time = 0** (gray)
- **Time = 1 year** (blue)

- **Probability**
  - 0.6
  - 0.4
  - 0.2
  - 0

- **$V_{th}$**
  - -5 to 4
• Relax retention from 1 year to 2 weeks
  – 2.3x write speedup is achievable
• Relax retention from 1 year to 2 weeks
  – We can use the BCH (24 corrections per 1080B) to replace an LDPC whose strength is $2.2 \times 10^{-2}$
Outline

• Motivation
• NAND Flash background
• Main idea
• Methodology
  – NAND Flash model
  – Retention analysis on real-world applications
  – Retention-aware system architecture
• Evaluation
• Conclusions
Retention Analysis

- Retention requirement is defined as
  - The time period from writing the sector until the sector is overwritten

- We analyze 3 sets of real-world applications

<table>
<thead>
<tr>
<th>Category</th>
<th>Name</th>
<th>Description</th>
<th>Span</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSRC</td>
<td>prn_0, proj_0, proj_2</td>
<td>Print server, Project directories</td>
<td>1 week</td>
</tr>
<tr>
<td></td>
<td>prxy_0, prxy_1</td>
<td>Web proxy, Web proxy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>src1_0, src1_2</td>
<td>Source control</td>
<td></td>
</tr>
<tr>
<td></td>
<td>src2_2</td>
<td>Source control</td>
<td></td>
</tr>
<tr>
<td></td>
<td>usr_1, usr_2</td>
<td>User home directories</td>
<td></td>
</tr>
<tr>
<td>Hadoop</td>
<td>hd1, hd2</td>
<td>WordCount benchmark</td>
<td>1 day</td>
</tr>
<tr>
<td>TPC-C</td>
<td>tpcc1, tpcc2</td>
<td>OLTP benchmark</td>
<td>1 day</td>
</tr>
</tbody>
</table>
Retention Analysis

- MSRC

The graph shows the cumulative percentage over different retention time requirements. The lines represent different data sets, each marked with a distinct symbol.

- The red circle highlights that for some data sets, the cumulative percentage exceeds 86%.
- The blue circle indicates that for another set of data, the percentage is greater than 49%.

The x-axis represents the retention time requirement, ranging from seconds (sec.) to weeks (week), while the y-axis shows the cumulative percentage (%) ranging from 0% to 100%.
Retention Analysis

• Hadoop and TPC-C

Retention Time Requirement

Cumulative Percentage (%) vs. Retention Time Requirement

- >96%
- >57%
Outline

• Motivation
• NAND Flash background
• Main idea
• Methodology
  – NAND Flash model
  – Retention analysis on real-world applications
    – Retention-aware system architecture
• Evaluation
• Conclusions
# Two-Level Retention Guarantee

<table>
<thead>
<tr>
<th></th>
<th>Host writes</th>
<th>Background writes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description</strong></td>
<td>Writes from the host</td>
<td>E.g., cleaning, wear-leveling</td>
</tr>
<tr>
<td><strong>Importance of</strong></td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td><strong>performance</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Retention</strong></td>
<td>Low</td>
<td>High (cold data)</td>
</tr>
<tr>
<td><strong>requirements</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Write</strong></td>
<td>Short retention guarantee</td>
<td>Normal retention guarantee</td>
</tr>
<tr>
<td><strong>operation</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Fast write</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Less-strong ECCs</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Retention-Aware FTL Design

- FTL (Flash Translation Layer)
  - Software layer emulating a block device over NAND Flash memories in SSDs

![Diagram of FTL and SSD components]

FTL (Flash Translation Layer) – Software layer emulating a block device over NAND Flash memories in SSDs
Retention-Aware FTL

- Two new components employed in the FTL
  - Mode Selector
    - Invoke different NAND Flash write commands or different ECC engines for blocks with different retention guarantees
  - Retention Tracker
    - Monitor the remaining retention time of blocks
    - Reprogramming a block when the block is about to run out of retention
Retention Relaxation for Write Speedup

- Write stream
  - a, b, b, a, c, a ...

Diagram of SSD with various components including:
- CPU
- OS
- FTL
- Address Translation
- Mode Selector
- Retention Tracker
- Background Cleaning
- NAND Flash
## Retention Relaxation for ECC Optimization

### Issue:
Since all host writes go through the LDPC encoder, a high-throughput LDPC encoder is required, otherwise it will become the bottleneck.

### Advantages:
- Time-consuming LDPC is kept out of the critical performance path
- LDPC encodes only data with retention longer than what the BCH guarantees
- LDPC encoding can be scheduled over a period of time in the background
  - Reduce the throughput requirements of the LDPC

### Diagrams:
- **Concatenated BCH-LDPC**
  - Host Writes
    - BCH Encoder
    - LDPC Encoder
    - NAND Flash
    - NAND Flash
    - NAND Flash
    - NAND Flash
  - Background Writes

- **Retention-Aware Architecture**
  - Host Writes
    - BCH Encoder
    - LDPC Encoder
    - NAND Flash
    - NAND Flash
    - NAND Flash
    - NAND Flash
  - Reprogramming & Background Writes
    - Already LDPC Encoded?
    - Y
    - N
      - Already LDPC Encoded?
      - Y
      - N

---

**Table:**

<table>
<thead>
<tr>
<th>Concatenated BCH-LDPC</th>
<th>Retention-Aware Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Host Writes</td>
<td>Host Writes</td>
</tr>
<tr>
<td>BCH Encoder</td>
<td>BCH Encoder</td>
</tr>
<tr>
<td>LDPC Encoder</td>
<td>LDPC Encoder</td>
</tr>
<tr>
<td>NAND Flash</td>
<td>NAND Flash</td>
</tr>
<tr>
<td></td>
<td>NAND Flash</td>
</tr>
<tr>
<td></td>
<td>NAND Flash</td>
</tr>
<tr>
<td></td>
<td>NAND Flash</td>
</tr>
</tbody>
</table>
Outline

• Motivation
• NAND Flash background
• Main idea
• Methodology
  • Evaluation
• Conclusions
**Experimental Setup**

- Simulations using Disksim 4.0 & SSDsim
- Workloads
  - 11 traces from MSRC, Hadoop, and TPC-C
- Two configurations are evaluated
  - Baseline: SSDs with 1-year retention for all writes
  - Proposed retention-relaxation design: RR-2week
    - 2-week retention guarantee for host writes
    - Blocks not overwritten in one week are reprogrammed with full retention guarantees

<table>
<thead>
<tr>
<th>Trace Name</th>
<th>Dies per Disk</th>
<th>Exported Capacity (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>prn_0, proj_0, prxy_0, src1_2</td>
<td>16</td>
<td>106</td>
</tr>
<tr>
<td>src2_2</td>
<td>32</td>
<td>212</td>
</tr>
<tr>
<td>src1_0</td>
<td>64</td>
<td>423</td>
</tr>
<tr>
<td>proj_2, hd1, hd2, tpcc1, tpcc2, tpcc1_n, tpcc2_n</td>
<td>128</td>
<td>847</td>
</tr>
</tbody>
</table>
Improving Write Speed

- Typical 2x to 2.5x speedup
- 3.9x to 5.7x for hd1 and hd2
  - Due to long queuing time
  - Retention relaxation reduces queuing time by 5.4 to 6.1x
**Improving Cost & Performance of ECCs**

- SSD performance vs. various LDPC throughput
  - Under the same LDPC throughput (HW cost)
    - RR-2week outperforms the baseline
  - RR-2week approaches the ideal performance with 20MB/s LDPC

*The curve of the baseline presents a zigzag appearance because several traces cause the I/O queue saturation in the SSD simulator.*
Conclusions

• First work to exploit retention relaxation for optimizing NAND Flash-based SSDs
  – Improving write speed
  – Improving the cost & performance of ECCs

• Quantitative analysis on the retention requirements of datacenter workloads
  – In most cases, 49% to 99% of writes have retention less than a week.

• Retention-aware FTL design
  – Enabling retention relaxation without hampering reliability
  – Achieving 2x to 5.7x write speedup or 8x less LDPC throughput requirements for SSDs
Thank You