A Self Learning Algorithm for NAND Flash Controllers

Hao Zhi, Lee

Firmware Manager
Core Storage Electronics Corp./Phison Electronics Corp.
haozhi_lee@phison.com
Outline

- Basic FW Architecture
- Challenges & Problems
- Self-learning & Adaptive Algorithm
- Conclusion
Basic FW Architecture
Brief Overview

Host Interface Layer (HIL)
- Command Task Management
- Buffer Management

Flash Translation Layer (FTL)
- Logical-To-Physical Mapping
- Garbage Collection
- Wear Leveling Management
- Read Disturb Management
- Bad Block Management

Flash Interface Layer (FIL)
- Flash Operation Management
- Buffer Management
- Flash Error Recovery
Challenges & Problems
Performance

- **Read/Write Latency**
  - Different hosts have different I/O patterns
  - Inefficient host latency management results in unstable performance

- **Idle Time Operations**
  - Controller performs background operations or enters power saving mode during system idle
  - Redundant operations will not only consume more power, but also impact WAF and latency time of next command
Endurance

- **Data Integrity**
  - Sudden power loss, data retention and read disturbance will lead to data corruption in NAND flash

- **Device Lifetime**
  - NAND flash has limited P/E cycle
  - Unbalance SLC/TLC block usage will induce higher WAF and cause NAND flash to wear out quickly
Cost

- **Over Provision**
  - WAF can be greatly reduced by allocating more spare blocks
  - However, higher OP means lesser logical space for user
Flexibility

- Firmware Update
  - Different host platforms have dissimilar IO patterns
  - Several FW versions optimized to serve each platform can be costly to maintain
Self Learning & Adaptive Algorithm
Objective

- FW with configurable parameters is simply not enough
- The following to be considered
  a. Data/pattern aware
  b. Self adaptive
  c. Self learning
Key Parameters

**HOST**
1. Read/Write IO Pattern
2. System Idle Behavior

**FTL**
1. Block Usage
2. Block Density
3. Block Age
4. Table Hit Pattern

**NAND FLASH**
1. Erase Count
2. Read Count
3. Flash Error Condition

---

![Diagram](diagram.png)

**FW Algorithm**
- Feed parameters & pattern

**Generate adaptive strategies**

![Diagram](diagram.png)

**HOST**

**FTL**

**NAND Flash**
Dynamic Read Recovery Strategy

- Due to the reliability issues of 3D NAND, the read recovery flow has been more sophisticated and time consuming than before.
- Such as, the expansion of read retry tables in NAND flash.
- The device performance can be severely impacted especially during end of life.
Controller can monitor the status of NAND flash and learn from previous recovery statistics to optimize the recovery flow.
Dynamic Read Recovery Strategy

Adaptive recovery strategy can rearrange the priority of recovery flow

- Controller will eventually learn along the iterative process and determine the best approach
Idle Time Optimization

- Controller usually waits for a period of idle time (pre-configured by user) and then performs background operations or enters power saving mode.
- During these operations, some data will be programmed into NAND flash.
Idle Time Optimization

Host idle time

Previous host command

Idle wait time
pre-configured

Enter power saving
mode or perform
background
operations

Next host command

Program to NAND flash
(i.e., flush cache data,
save table, etc.)

Flash Memory Summit 2017
Santa Clara, CA
However, different hosts may have different idle time behavior.
Idle Time Optimization

- Redundant NAND flash program operations will increase erase count
What if the controller can learn from the host behavior and predict for an optimized strategy?

- **Idle time prediction**
- **Decision for idle time strategy**
- **Idle time detection** (when next host CMD arrives)

- If predicted time is short, wait for a time period
- If next host command is still not coming, proceed to power saving mode or background operation
Idle Time Optimization

- An adaptive idle time strategy can greatly reduce redundant programs to NAND flash
Simulation results based on host usage model

Saved 45% of NAND flash program during idle time
The number of SLC/TLC blocks are usually allocated during device initialization.

We expect both SLC/TLC pools to wear out equally during device end of life.
Mixed Pool Dynamic Wear Leveling

- However, different host I/O behavior might have different impact on SLC/TLC block usage

<table>
<thead>
<tr>
<th></th>
<th>SLC Max Erase Count: 40K</th>
</tr>
</thead>
<tbody>
<tr>
<td>TLC Max Erase Count: 1.5K</td>
<td></td>
</tr>
</tbody>
</table>

Device will fail early when either one of the pool has achieved its max erase count

Increment of erase count

<table>
<thead>
<tr>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Erase Count</td>
</tr>
</tbody>
</table>

- SLC max erase count
- TLC max erase count
Mixed Pool Dynamic Wear Leveling

- Controller can constantly monitor the erase count ratio of both pools and dynamically configure the block selection algorithm to keep within reasonable TLC/SLC ratio.
Simulation results based on host usage model
- This will prevent device to fail early due to wear out

**Mixed Pool Dynamic Wear Leveling**

**Increment of erase count**

- SLC max erase count
- TLC max erase count

<table>
<thead>
<tr>
<th>Erase Count</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SLC</td>
<td></td>
</tr>
<tr>
<td>SLC”</td>
<td></td>
</tr>
<tr>
<td>TLC</td>
<td></td>
</tr>
<tr>
<td>TLC”</td>
<td></td>
</tr>
</tbody>
</table>

SLC Max Erase Count: 40K
TLC Max Erase Count: 1.5K

Lifetime extended by 10%
Conclusion

- By exploring the key parameters of host behavior, system data structures, and NAND flash condition, a self learning FTL with simple adaptive design can benefit storage device in terms of performance, endurance, cost and flexibility.
THANK YOU