Wear Estimation for Devices with eMMC Flash Memory
WITH YOU TODAY...

- Joined Toradex 2011
- Spearheaded Embedded Linux Adoption
- Introduced Upstream First Policy
- Top 10 U-Boot Contributor
- Top 10 Linux Kernel ARM SoC Contributor
- Industrial Embedded Linux Platform Torizon Fully Based on Mainline Technology
  - Mainline U-Boot with Distroboot
  - KMS/DRM Graphics with Etnaviv & Nouveau
  - OTA with OSTree
  - Docker

Marcel Ziswiler
Platform Manager Embedded Linux
marcel.ziswiler@toradex.com
Toradex AG

LOCATED IN HORW BY LUCERNE, SWITZERLAND
WHAT WE’LL COVER TODAY

• A Technology Overview
• eMMC
• Flash Health
• I/O Tracking
• Lifespan Estimation
• Flash Analytics Tool
• Conclusion
Flash – Non-Volatile Memory of Choice

In Embedded Systems

- Decreased Size
- Increased Robustness
- No Moving Parts
- Reduced Power Consumption
- Keep Redundant Data On-Site
- For Intermittent Connectivity Reasons
NOR vs. NAND

- Difference at Transistor Level How to Store Bits
- NOR and NAND Logic Gates

- Simpler Principle of Operation
- Higher Reliability
- Higher Pin-Count
- Lower Density in Silicon
- Bigger Size
- More Expensive
- Only for Specific Applications
- Highly Critical Industrial-Grade
NAND Structure

Cell
- Smallest Entity
- Storing Data at Bit-Level

Page
- Smallest Array of Cells
- Addressable for Read/Write Operations
- Flipping Bits from 1 to 0
- Page Size: Range of Kilobytes e.g. 4 kB

(Erase-)Block
- Smallest Array of Pages
- Addressable for Erase Operation
- Return Logic State of Bits from 0 Back to 1
- Block Size: Range of Megabytes e.g. 4 MB
- Erase Operation is Slow
- Wears out Flash over Time
- Develops Bad Blocks
- Block Erase Count
NAND: SLC vs. MLC

Cell
• How Many Bits Stored
• Depends on Voltage Level Thresholds

SLC
• Single-Level Cell
• Stores 1 Bit per Cell

pSLC
• Pseudo-SLC
• MLC Operating in SLC Mode
• Stores 1 Bit per Cell

MLC
• Multi-Level Cell
• Stores 2 Bits per Cell

TLC, QLC, ...
• You Get the Idea...
Trade-Off Between Density and Cost vs. Reliability and Lifespan
ECC and Bad Blocks

Error Correction Code Algorithms
- Adding Redundancy
- Allow Correcting resp. Detecting Certain Bit Errors
- Random Bit Flips Even in Healthy Blocks

Bad Blocks
- Over Time Probability of Bit-Flips Increases
- Blocks Wear out Becoming Bad
- Factory Bad Blocks
- Spare Blocks
Wear-Leveling and Garbage Collection

Wear-Leveling
- Same Physical Pages/Blocks Used for e.g. File Update
- Increased Wear out Causing Premature Bad Blocks
- Using Blocks Evenly
- Moving Data Around
- Dynamic vs. Static

Garbage Collection
- Slow Erase Operation
- Avoid Immediate Erasure
- Just Marking Blocks Dirty
- Erase Later e.g. Idle Time
Write Amplification Factor (WAF)

- Actual Data Written to NAND Flash Cells vs.
- Data Sent from Host to Memory
- Difference Between Program and Erase Size
- Data Needs Erasing Before (Re-)Writing
- Memory Management Features:
  - Wear-Leveling
  - Garbage Collection
- Typical WAF in eMMC: Good Average is 4
- Depends on Usage Scenario
- Select Optimal Data Size Related to Page Size
Embedded MultiMediaCard (eMMC)

Managed NAND
- Raw NAND Die & Accompanying NAND Controller
- Abstracting Large Part of Management SW-Stack
- Latest JEDEC Standard 5.1
- Allows for Regular Block Device Operations
- Using Regular File Systems e.g. EXT4
- Example eMMC
  - Micron MTFC4GACAJCN-1M-WT
  - 4 GB MLC
  - 1024 Blocks of 4 KB Size
  - Lifespan 3000 Write/Erase Cycles
  - 15 nm Process
MMC Protocol

- Bus: Command, Clock and 7 Data Lines
- CMD: Serial Command/Response Channel
- DAT0-7: Parallel Read/Write Data plus CRC
- Single or Multiple Block Read/Write Operations
## MMC Registers

<table>
<thead>
<tr>
<th>Name</th>
<th>Width (bytes)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CID</td>
<td>16</td>
<td>Unique Card/Device Identifier.</td>
</tr>
<tr>
<td>RCA</td>
<td>2</td>
<td>Relative Card/Device Address: device’s system address, dynamically assigned by the host during initialization.</td>
</tr>
<tr>
<td>DSR</td>
<td>2</td>
<td>Driver Stage Register: to configure the device’s output drivers.</td>
</tr>
<tr>
<td>CSD</td>
<td>16</td>
<td>Card/Device Specific Data: information about the device’s operation conditions.</td>
</tr>
<tr>
<td>OCR</td>
<td>4</td>
<td>Operation Conditions Register: used by a special broadcast command to identify the voltage type of the device.</td>
</tr>
<tr>
<td>EXT_CSD</td>
<td>512</td>
<td>Extended Card/Device Specific Data: contains information about the device’s capabilities and selected modes. Introduced in standard v4.0.</td>
</tr>
</tbody>
</table>
JEDEC Standard Health Reporting

• Device Life Time Estimation Type A:
  • Health Status in Increments of 10 %
  • Refers to pSLC Blocks in our eMMC

• Device Life Time Estimation Type B:
  • Health Status in Increments of 10 %
  • Refers to MLC Blocks in our eMMC

• Pre-EOL Information:
  • Normal: Up to 80 % of Reserved Blocks Consumed
  • Warning: More than 80 % Consumed
  • Urgent: More than 90 % Consumed

• Introduced with Standard v5.0
• Low Resolution Requiring Very Long Benchmark Runs
Micron Proprietary Health Report

- TN-FC-32: e.MMC Device Health Report
- Bad Block Counters and Information:
  - Factory Bad Block Count
  - Run-Time Bad Block Count
  - Remaining Spare Block Count
  - Per Block Failed Erase vs. Program Operations with Page Addresses
- Block Erase Counters:
  - Minimum, Maximum and Average Among all Blocks
  - Per Block Erase Count
- Block Configuration:
  - Physical Address of Each Block
  - pSLC vs. MLC Configuration
  - Accessed by General Command (GEN_CMD) aka CMD56
Flash Health

- Percentage of Capacity Already Worn Out

\[
\text{endurance} = \text{number of blocks} \cdot \text{average block lifespan}
\]

\[
\text{endurance} = 1024 \cdot 3000 = 3,072,000 \text{ block erases}
\]

or

\[
\text{endurance} = \text{block size} \cdot \text{blocks} \cdot \text{average block lifespan}
\]

\[
\text{endurance} = 4 \text{ MB} \cdot 1024 \cdot 3000 = 12 \text{ TB written}
\]
Monitoring Flash Health in Linux

**mmc-utils**

- Software to Extracts Meaningful Information From eMMC Devices
- Reading Data From Extended Card/Device Specific Data (EXT_CSD)
- Includes Device Lifespan Defined by JEDEC eMMC 5.0 Standard

```
root@colibri-imx6-05097264:/app# mmc extcsd read /dev/mmcblk1

Device life time estimation type B [DEVICE_LIFE_TIME_EST_TYP_B: 0x01]
Device life time estimation type A [DEVICE_LIFE_TIME_EST_TYP_A: 0x01]
eMMC Life Time Estimation A [EXT_CSD_DEVICE_LIFE_TIME_EST_TYP_A]: 0x01
eMMC Life Time Estimation B [EXT_CSD_DEVICE_LIFE_TIME_EST_TYP_B]: 0x01

root@colibri-imx6-05097264:/app# mmc extcsd read /dev/mmcblk1 | grep LIFE
```

```
root@colibri-imx6-05097264:/app# mmc extcsd read /dev/mmcblk1 | grep EOL
Pre EOL information [PRE_EOL_INFO: 0x01]
eMMC Pre EOL information [EXT_CSD_PRE_EOL_INFO: 0x01]
```
Vendor Proprietary Health Report

```c
/ Retrieve the erase count for each block
// A two-step approach is needed (read number of tables and then read tables)
int do_block_erase_info(int nargs, char **argv)
{
    ret = CMD56_data_in(fd, cmd56_how_many_tables, data_in);
    printf("Block erase count\n");
    printf("Block\tErase\n");
    for(table_idx = 0; table_idx < how_many_tables; table_idx++){
        ret = CMD56_data_in(fd, (table_idx * 256) + cmd56_retrieve_base, data_in);
        for(physical_block = 0; physical_block < 128; physical_block++){
            printf("%d\t%d\n",
                    (256*data_in[0+2*physical_block]) + data_in[1+2*physical_block],
                    (256*data_in[256+2*physical_block]) + data_in[257+2*physical_block]);
        }
    }
}
```

```c
int do_bad_block_count(int nargs, char **argv);
int do_bad_block_info(int nargs, char **argv);
int do_block_erase_count(int nargs, char **argv);
int do_block_erase_info(int nargs, char **argv);
int do_block_addr_type_info(int nargs, char **argv);
```
Vendor Proprietary Health Report 2nd

- Vendor-Specific Tool
- Micron's emmcparm
- Provides Consolidated Lifespan Report
- More Granular Parameters

```
1  root@colibri-imx6:~# emmcparm_arm
2  --spare_block
3  --bad_block
4  --erase_count
5  --sect_count
```
I/O Tracking

- Useful Indicator that Flash Wears out Quickly
- Debug Indicator Showing What Applications Write too Much Data
- Generates Input Data for Wear Estimation Model
- Independent of JEDEC Standards or eMMC Vendor Health Reports
- Applicable to any NAND Flash Based Storage Technology
Linux I/O Stack for eMMC and Raw NAND

- Userspace File Operations at Application-Level
- System Calls into Kernelspace
- Ends up in Linux I/O Stack
- Finally Sending Data to Low-Level Device Driver
Block Device I/O Stack

- VFS Abstracting Userspace API
- FS File Concept
- Gen Block Layer Handling Block IO
- IO Scheduler Queuing IO Requests
- Max. Block IO Performance
- Why Not Monitor Userspace?
- Not Very Accurate
- Layers of Caches
Caches, Buffers, Queues and Syncs

Userspace
- Page cache
- Write-back cache
- Write-buffers
- I/O merges
- I/O queues

Flash memory

Sync data
- Files x global
- Data x metadata
- Only when you need to
- Power-cut tolerance
- Atomic file update
Measuring I/O Writes

Linux Performance Observability Tools

- Monitoring Writes That Actually Hit the Flash
- Where Exactly in the Linux I/O Stack to Measure?
- How to Measure (e.g. What Tool to Use)?
iótop

- Tracking Userspace Operations
- Easy to Use

```
root@colibri-imx6:-# iotop -help

Options:
- -o, --only only show processes or threads actually doing I/O
- -b, --batch non-interactive mode
- -a, --accumulated show accumulated I/O instead of bandwidth
- -k, --kilobytes use kilobytes instead of a human friendly unit
- -t, --time add a timestamp on each line (implies --batch)
- -q, --quiet suppress some lines of header (implies --batch)
```

```
root@colibri-imx6:-# dd if=/dev/urandom bs=4k count=100000 | pv -L 25k > testfile

root@colibri-imx6:-# iotop --only --batch --accumulated --kilobytes --time --quiet

<table>
<thead>
<tr>
<th>TIME</th>
<th>TID</th>
<th>PRIO</th>
<th>USER</th>
<th>DISK</th>
<th>READ</th>
<th>DISK</th>
<th>WRITE</th>
<th>SWAPIN</th>
<th>IO COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>2019-08-02 03:11:19</td>
<td>50 be/4 root</td>
<td>0.00 K</td>
<td>24.00 K</td>
<td>-0.00 %</td>
<td>-0.00 %</td>
<td>pv -L 25k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2019-08-02 03:11:20</td>
<td>50 be/4 root</td>
<td>0.00 K</td>
<td>52.00 K</td>
<td>-0.00 %</td>
<td>-0.00 %</td>
<td>pv -L 25k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2019-08-02 03:11:21</td>
<td>50 be/4 root</td>
<td>0.00 K</td>
<td>80.00 K</td>
<td>-0.00 %</td>
<td>-0.00 %</td>
<td>pv -L 25k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2019-08-02 03:11:22</td>
<td>50 be/4 root</td>
<td>0.00 K</td>
<td>104.00 K</td>
<td>-0.00 %</td>
<td>-0.00 %</td>
<td>pv -L 25k</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2019-08-02 03:11:23</td>
<td>50 be/4 root</td>
<td>0.00 K</td>
<td>128.00 K</td>
<td>-0.00 %</td>
<td>-0.00 %</td>
<td>pv -L 25k</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
blktrace/blkparse

- Overwhelming Amount of Output
- Make use of Filters
- Goal: Tracking Userspace PID Once Write to Flash is Confirmed
- C (Complete): Request Completed (Details Sector, Request Size and Success/Failure)
- I (Inserted): Request Sent to I/O Scheduler for Addition to Internal Queue

```
root@colibri-imx6:~# blktrace -o /dev/mmcblk1 blkparse
```

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>179,0</td>
<td>0</td>
<td>26</td>
<td>0.000114661</td>
</tr>
<tr>
<td>2</td>
<td>179,0</td>
<td>0</td>
<td>27</td>
<td>0.000117328</td>
</tr>
<tr>
<td>3</td>
<td>179,0</td>
<td>0</td>
<td>28</td>
<td>0.000119661</td>
</tr>
<tr>
<td>4</td>
<td>179,0</td>
<td>0</td>
<td>29</td>
<td>0.000127328</td>
</tr>
<tr>
<td>5</td>
<td>179,0</td>
<td>0</td>
<td>30</td>
<td>0.000131661</td>
</tr>
<tr>
<td>6</td>
<td>179,0</td>
<td>0</td>
<td>31</td>
<td>0.000860277</td>
</tr>
<tr>
<td>7</td>
<td>179,0</td>
<td>0</td>
<td>32</td>
<td>0.012586780</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>barrier</th>
<th>barrier attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>complete</td>
<td>completed by driver</td>
</tr>
<tr>
<td>fs</td>
<td>FS requests</td>
</tr>
<tr>
<td>issue</td>
<td>issued to driver</td>
</tr>
<tr>
<td>pc</td>
<td>packet command events</td>
</tr>
<tr>
<td>queue</td>
<td>queue operations</td>
</tr>
<tr>
<td>read</td>
<td>read traces</td>
</tr>
<tr>
<td>requeue</td>
<td>requeue operations</td>
</tr>
<tr>
<td>sync</td>
<td>synchronous attribute</td>
</tr>
<tr>
<td>write</td>
<td>write traces</td>
</tr>
<tr>
<td>notify</td>
<td>notify trace messages</td>
</tr>
</tbody>
</table>
Lifespan Estimation

- Logging Flash Health and I/O Tracking
- Storing in Local Database
- Correlations:
  - Flash Health Over Time

\[
s_{\text{lifespan}} = \frac{\text{endurance}}{\text{average global block erase count}}
\]

- Flash Health Dependent on Write Rate

\[
s_{\text{lifespan}} = \frac{\text{endurance}}{\text{adjusted average write rate}}
\]
Remark on Wear Estimation

- Temperature Strongly Affects Flash Lifespan!
Flash Analytics Tool

- Under Development at Toradex Labs
- Abstracting Away Complexity of Wear Estimation
- Targeting Application Developers
- Current Prediction Model Implemented Using Linear Regression
Live Demo

Flash End-of-Life Estimate
Based on Flash Activity Since Tool Initialization

May 28, 2030
(in almost 11 years)
Please note that this value will become more accurate over time.
Conclusion

Estimation

$\text{lifespan} \approx \frac{\text{blocks} \cdot \text{capacity} \cdot \text{erasecycles}}{\text{write rate}}$

Challenges
- Write rate of application
- OS contribution
- What actually hits flash
- Write merges
- Health Status
- Partitioning accounting
- PSLC mode
- Write amplification
- Manufacturer info

Production / In-field
- Models for precise EOL
- Live monitors and alarms

Measurement and Modeling
- Accuracy
- Validation

Product Development
- Benchmark tests
- Estimation from models
Questions
References

- Toradex Labs - https://labs.toradex.com
- Toradex blog – What you should know about Flash storage - https://www.toradex.com/pt-br/blog/what-you-should-know-about-flash-storage
- Micron NOR | NAND Flash Guide
- Micron TN-29-42 – Wear-Leveling Techniques in NAND Flash Devices
- Micron TN-2960: Garbage Collection in SLC NAND Flash Memory
- Embedded Multi-Media Card (e.MMC) Electrical Standard (5.0) - https://www.jedec.org/sites/default/files/docs/JESD84-B50.pdf
- Micron TN-FC-32: e.MMC Device Health Report
References Continued

- Budget Fair Queueing (BFQ) Storage/I/O Scheduler - http://algo.unimi.it/people/paolo/disk_sched/
- The Linux Kernel/Storage - Wikibooks - http://algo.unimi.it/people/paolo/disk_sched/
- Don’t fear the fsync - http://thunk.org/tytso/blog/2009/03/15/dont-fear-the-fsync/
- The future of the page cache - https://lwn.net/Articles/712467/
- Micron TN-FC-25: Understanding Linux Driver Support for e.MMC - https://lwn.net/Articles/712467/
THANK YOU FOR YOUR INTEREST.