

# Data Distribution Algorithms for Reliable Parallel Storage on Flash Memories

Kathrin Peter

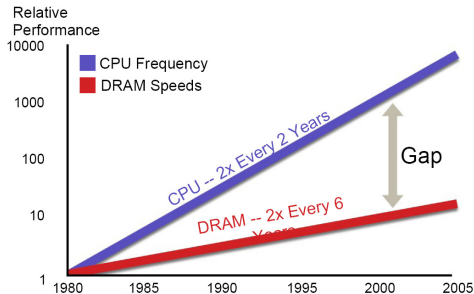
Zuse Institute Berlin

November 2008, MEMICS Workshop

# Motivation

- Nonvolatile storage
- Flash memory -  
Invented by Dr. Fujio Masuoka 1984
- Type of EEPROM
- Usage in a RAID-like configuration instead of hard disk drives

## Memory Bottleneck



Flash memory based storage: **Alternative to hard disk drives?**

# Outline

- 1 Flash based memory
- 2 Parallel and distributed storage based on Flash memories
- 3 Discussion and Summary

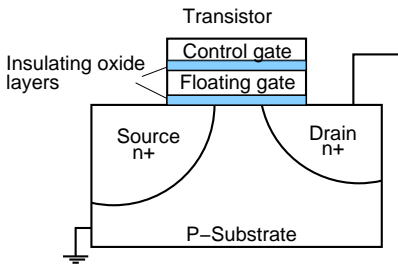


# Pros and Cons

- Less power consumption
- Higher access rates (in some cases)
- Uniform access time for random access - no seeks
- Robustness (extreme temperatures, vibration, shock)
- Price
- Limited erase cycles
- Flash management

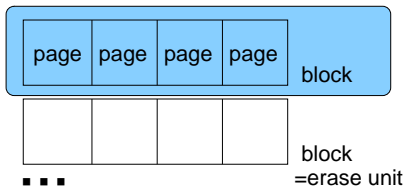
Model	2.5" SATA 3.0 Gbps SSD	2.5" SATA 3.0Gbps HDD
Mechanism type	Solid NAND flash based	Magnetic rotating platters
Density	64 GByte	80 GByte
Weight	73 g	365g
Active Power consumption	1 W	3.86 W
Operating temperature	0°C-70°C	5°C-55°C
Acoustic Noise	None	0.3 dB
Endurance	MTBF > 2M hours	MTBF <0.7M hours
Av. access time	0.1 msec	17 msec
Read performance	100 MB/s	34 MB/s
Write performance	80 MB/s	34 MB/s

# Limited erase cycles and flash management



- Memory cell: floating gate transistor
- Retention and Endurance

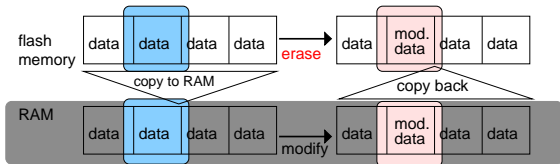
NAND – flash



- Typical page size: 16896 B (=2 KB+64 B spare)
- Typical block size:  
64 pages = 128 KB

# Mapping

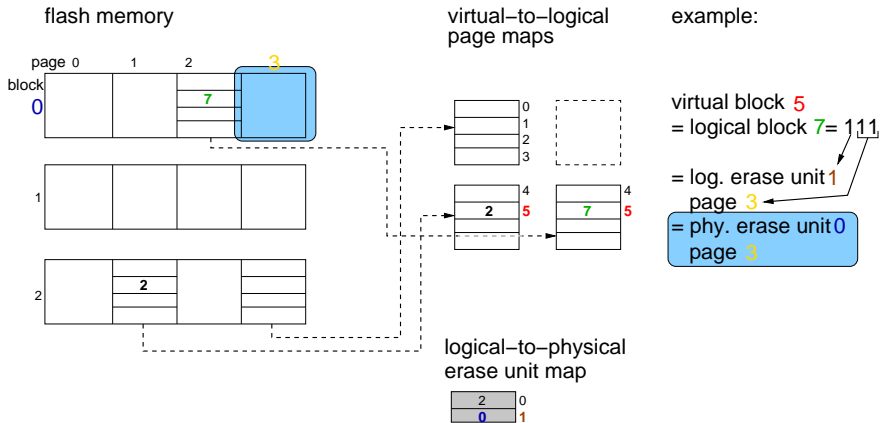
- Problem: **How to map logical blocks to flash addresses?**
- Disadvantage of linear mapping (one-to-one mapping):
  - Frequently-used **erase units wear out**
  - Identity mapping requires **lots of copying to and from RAM** (fixed block size). Example:



- Solution: **Sophisticated block-to-flash mapping** and moving around blocks: **wear leveling, garbage collection**

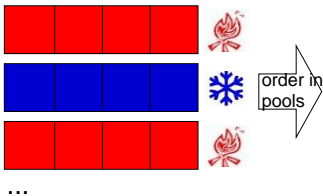
# Mapping - example

- Mapping of virtual block 5 to physical block 0, page 3



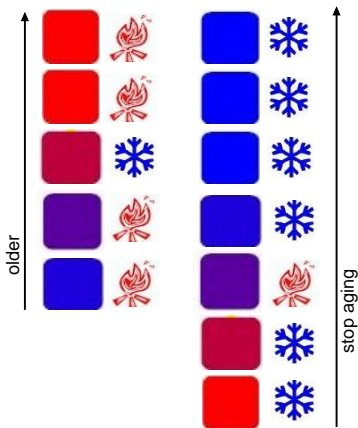
# Wear leveling - example

flash memory



hot pool

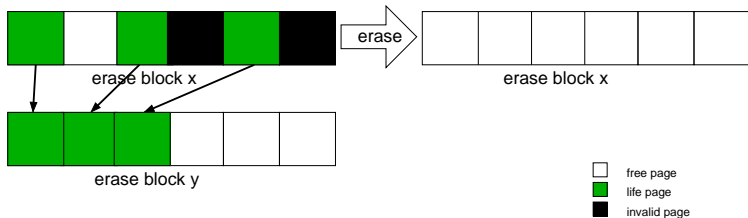
cold pool



*On Efficient Wear Leveling for Large-Scale Flash-Memory Storage Systems, Li-Pin Chang, 2007*

# Garbage collection - example

- Free space drops below a threshold
- Select blocks for reclamation
  - Copy all live pages to free pages somewhere else
  - Change mapping entry, update to new position
  - Erase block and allocate pages as free

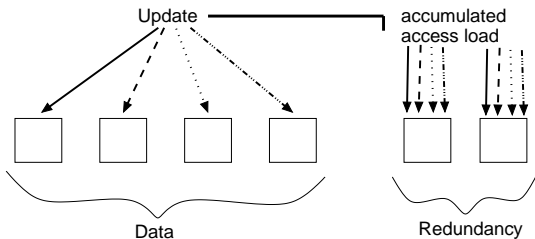


# Facts (local) wear leveling

- Performance: Erase operation is **slow**
- No in-place update
- Controller: **Efficient mapping** and erase distribution

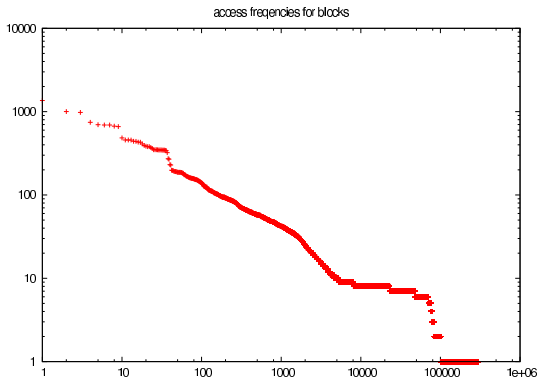
# Flash memory in a RAID-like system

- Aggregate **higher bandwidth**
- **Reliability** (redundancy for fault tolerance)
- **Problem:** uneven usage of flash memories, more writes to redundant blocks when data becomes updated



- **flash-RAIM:** Redundant Array of Independent flash Memories

# Uneven distribution of writes

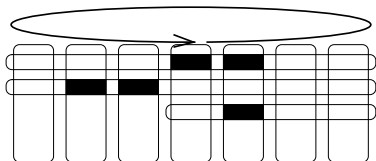


# flash-RAIM and wear leveling

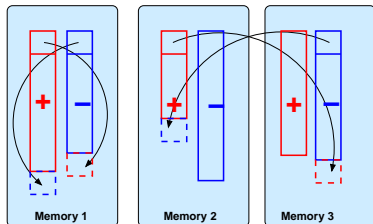
- There exist data distribution algorithms to place data evenly on a single memory
- Goal: long lifetime of a single memory
- We work on even distribution of writes across all memories in a memory array
- Goal: even usage of cells on all flash memories, reliable data storage, high throughput
- Method: global wear leveling

# Data Distribution Algorithms

## Staggered Striping



## Hierarchical dual-pool Algorithm



- Different starting points of algorithms
  - Explicit placing of data
  - Data movement after storing and local wear leveling

# Discussion of Future work

- Extension of the simulator to evaluate the global wear leveling algorithms
- Parameter study (trade-off)
- Define metrics to compare algorithms
  - Lifetime of the flash-RAIM
  - Speed
  - Overhead
- Usage of traces

# Summary

- Use aggregated **bandwidth and fault-tolerance**
- **flash-RAIM**
- Simulator for evaluation
- Next steps:  
Implementation and evaluation of the global wear leveling algorithm

# Erase cycles data/ redundancy memory

