FlexECC: Partially Relaxing ECC of MLC SSD for Better Cache Performance

Ping Huang, Pradeep Subedi, Xubin He, Shuang He and Ke Zhou

Department of Electrical and Computer Engineering, Virginia Commonwealth University
Richmond, VA, USA

Wuhan National Lab for Optoelectronics, Huazhong University of Science and Technology
Wuhan, Hubei, China

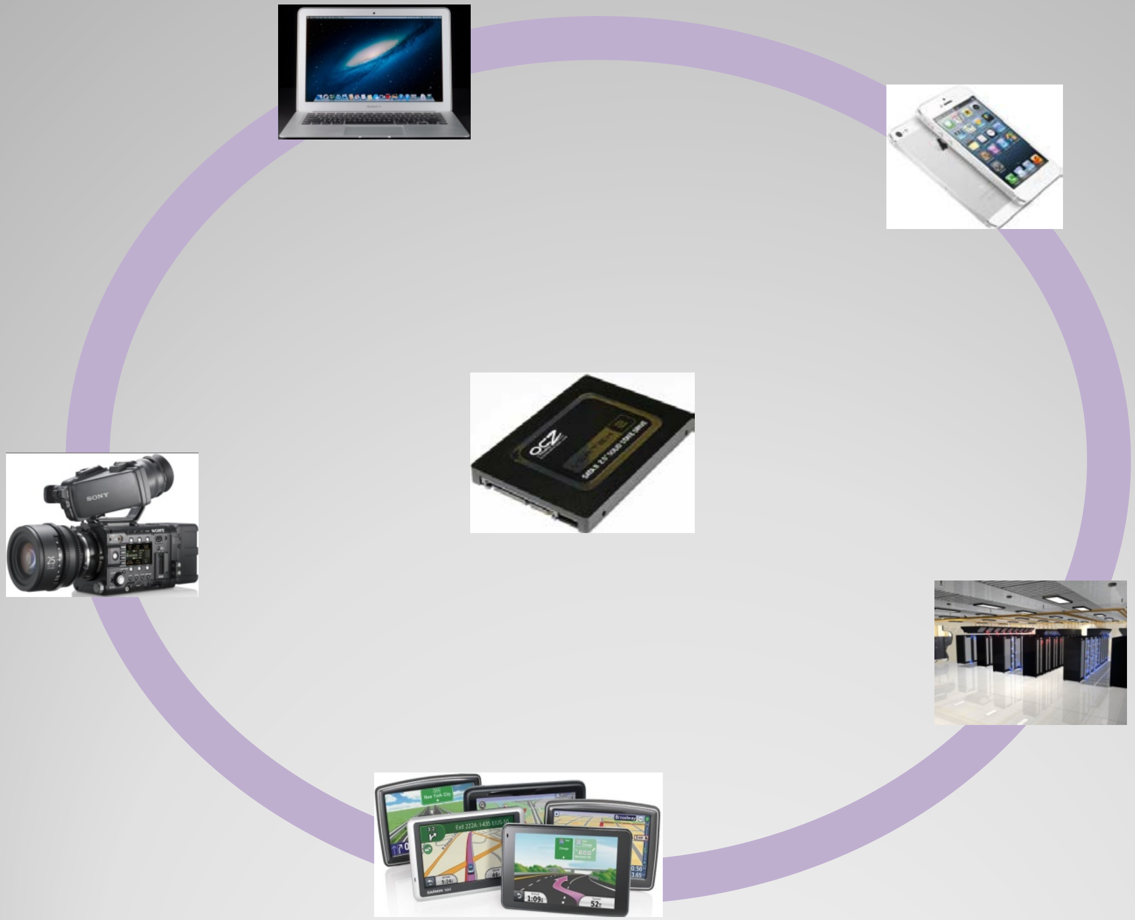
Presentation at the 2014 USENIX ATC

6/30/2014

Outline

- Background and Motivation
- Cache-oriented SSD Design
- Experimental Results
- Conclusion

SSDs are intruding:



features:

performance

energy

feature size

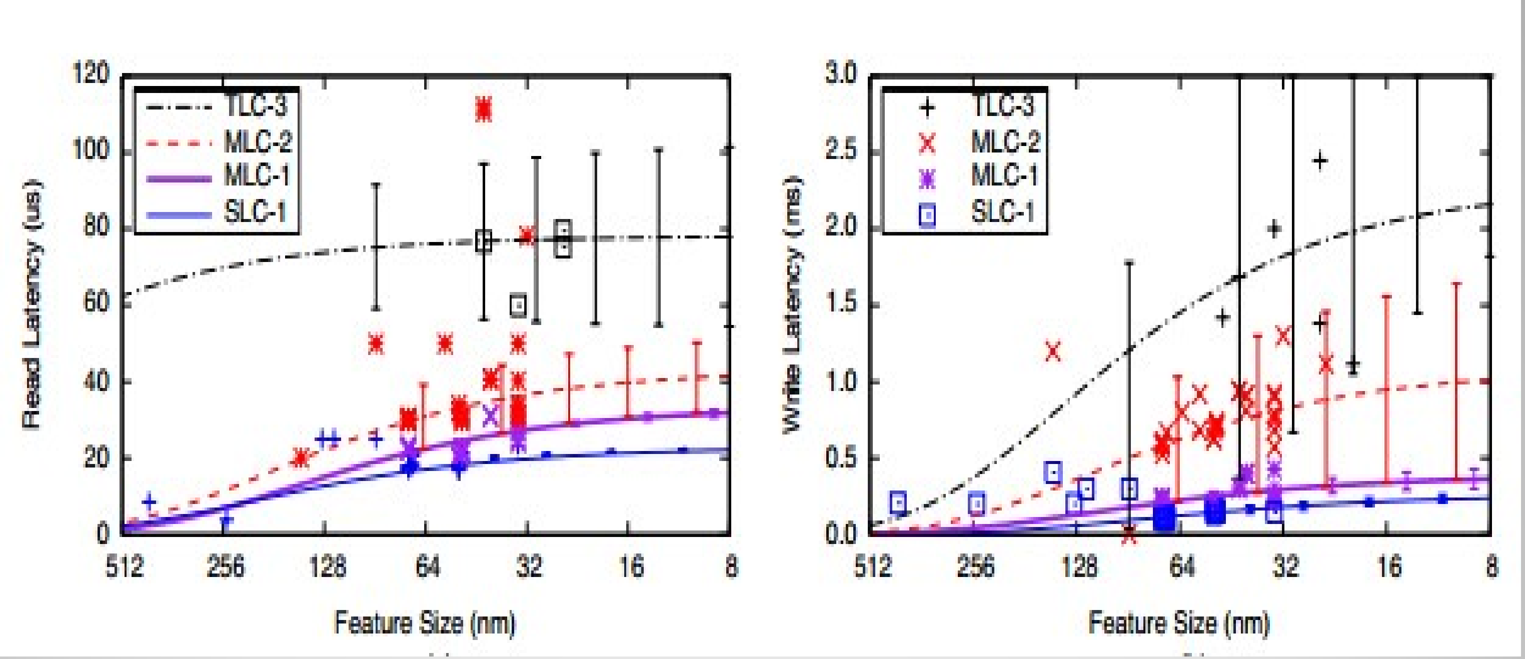
- The Main Enabling Factors:
  - ❑ Technology Scaling: flash cells are becoming smaller, coming to 20nm
  - ❑ Multi-bit technique: each cell is designed to hold multi bits information.

    SLC→MLC→TLC→ (6 bits per cell)
- We have witnessed the Moore’s law in the per-die capacity so far [Chien, Computer’2013]

The Bleak Future

- More difficult to read and write small cells
- Unpredictable Performance
- Worse cell-interference
- Reduced lifetime
- Decreased cell retention time

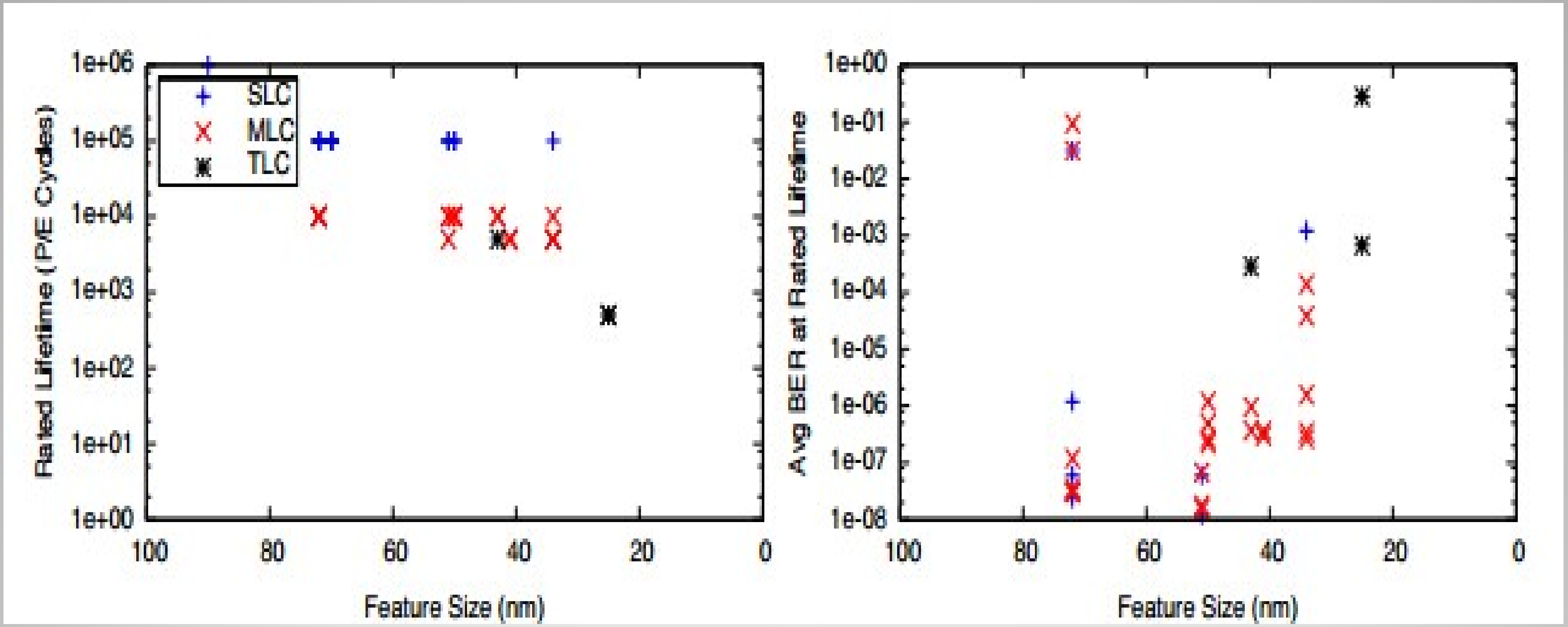
Increased Latencies



2.3x latency increase for an additional bit

Grupp et al. The Bleak Future of NAND Flash Memory. FAST'2012

Decreased Lifetime and Reliability

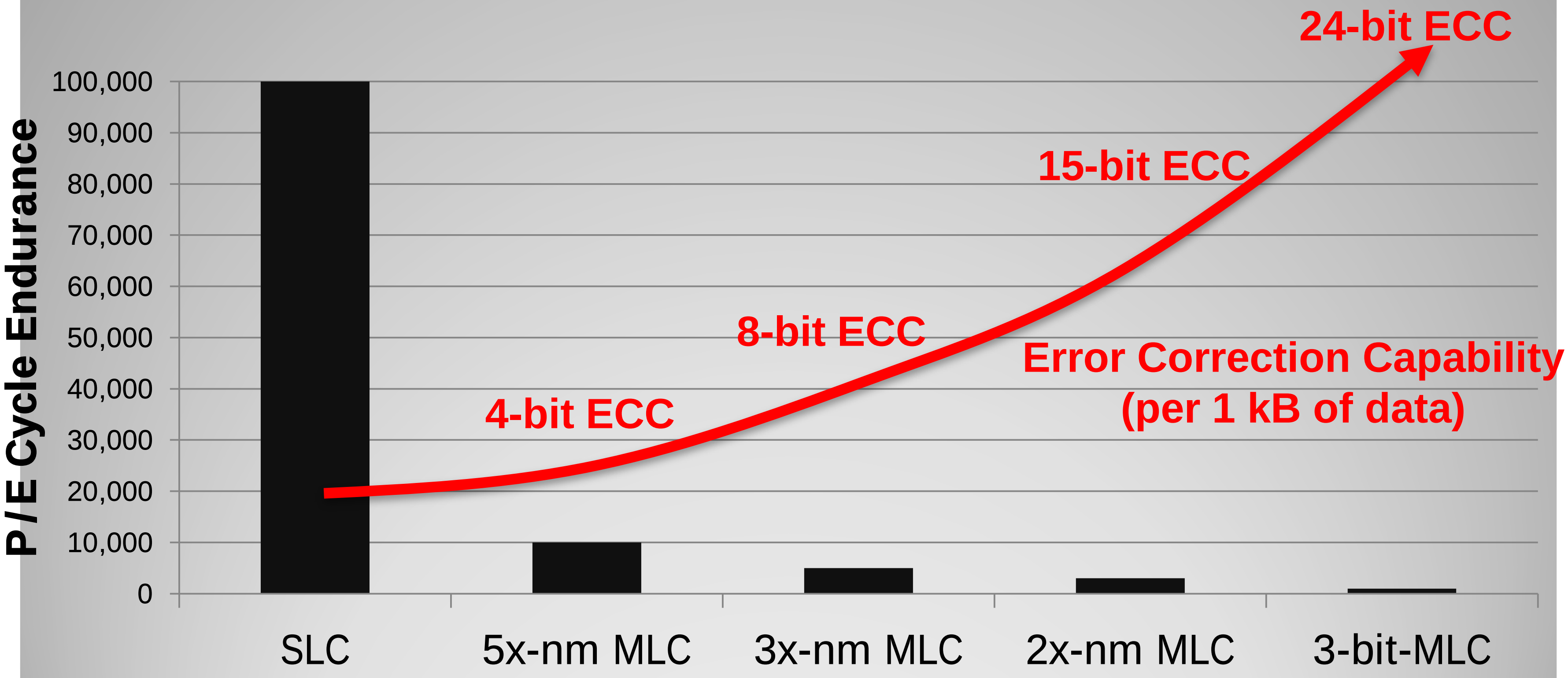


Grupp et al. The Bleak Future of NAND Flash Memory. FAST’2012

Error-tolerant Schemes

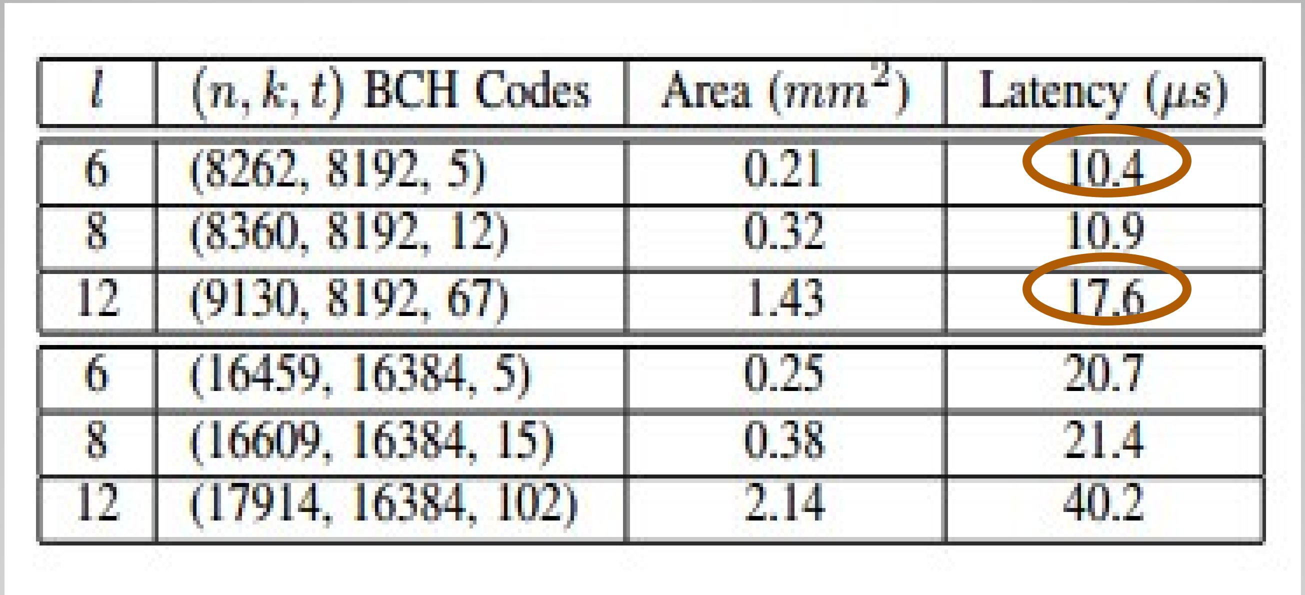
- Hamming Code
  - ❑Detect 2 bit errors and correct 1 bit error
- Bose-Chaudhuri-Hocquengham (BCH)
  - ❑Widely deployed in commercial SSDs
  - ❑Correction capability increases across generations
  - ❑Tens of micro second decoding overhead
- Low-density parity-check (LDPC)[Zhao, FAST’2013]
  - ❑Tolerate more errors at low cost
  - ❑Emerging to accommodate increasing error rates
  - ❑High decoding overhead (in the order of ms)
- More advanced codes are desired...

The Effects of Flash Scaling



Cai et al. Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime. ICCD'2012

Increasing Overhead

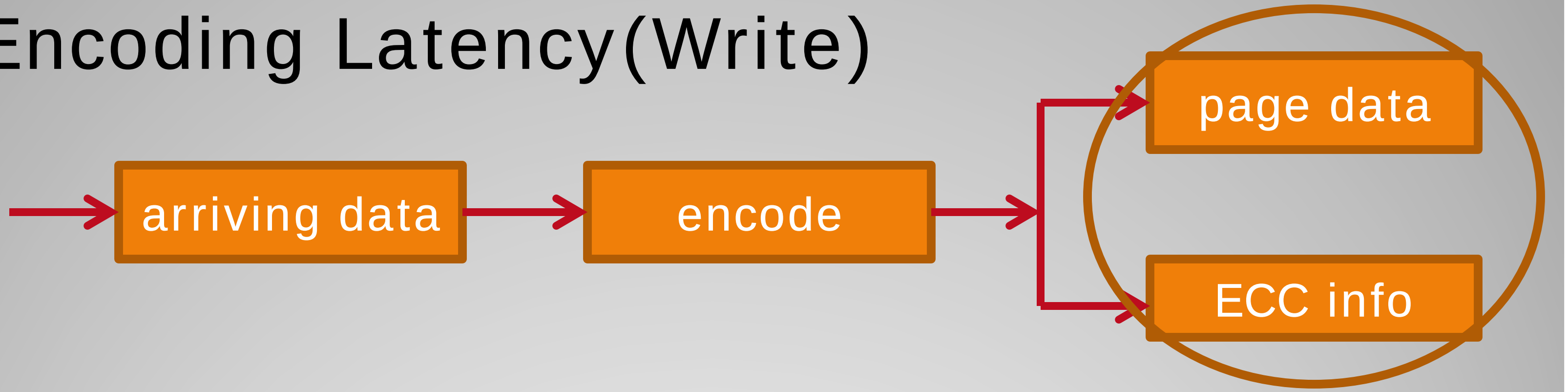


l	(n, k, t) BCH Codes	Area (mm^2)	Latency (μs)
6	(8262, 8192, 5)	0.21	10.4
8	(8360, 8192, 12)	0.32	10.9
12	(9130, 8192, 67)	1.43	17.6
6	(16459, 16384, 5)	0.25	20.7
8	(16609, 16384, 15)	0.38	21.4
12	(17914, 16384, 102)	2.14	40.2

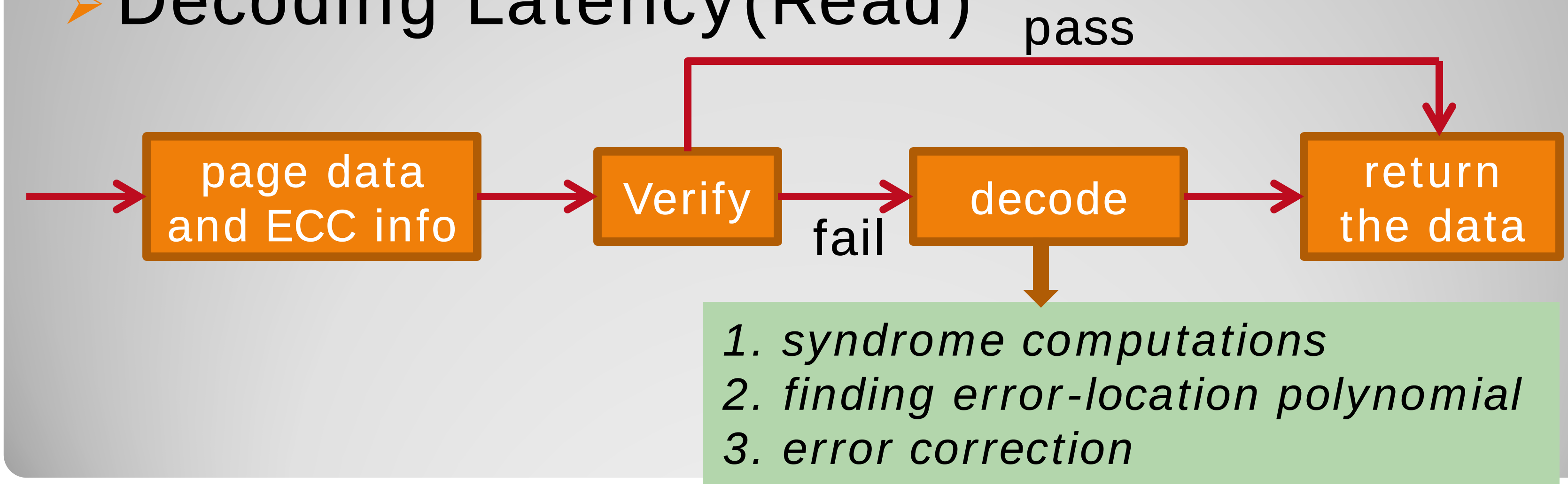
Tolerating 5 bit error out of 1KB data imposes 10 micro seconds, and the overhead becomes 17.6 us when 67 bits are tolerable [Sun, SiPS 2006].

Effects on Flash Operations

- Encoding Latency(Write)



- Decoding Latency(Read)



1. *syndrome computations*
2. *finding error-location polynomial*
3. *error correction*

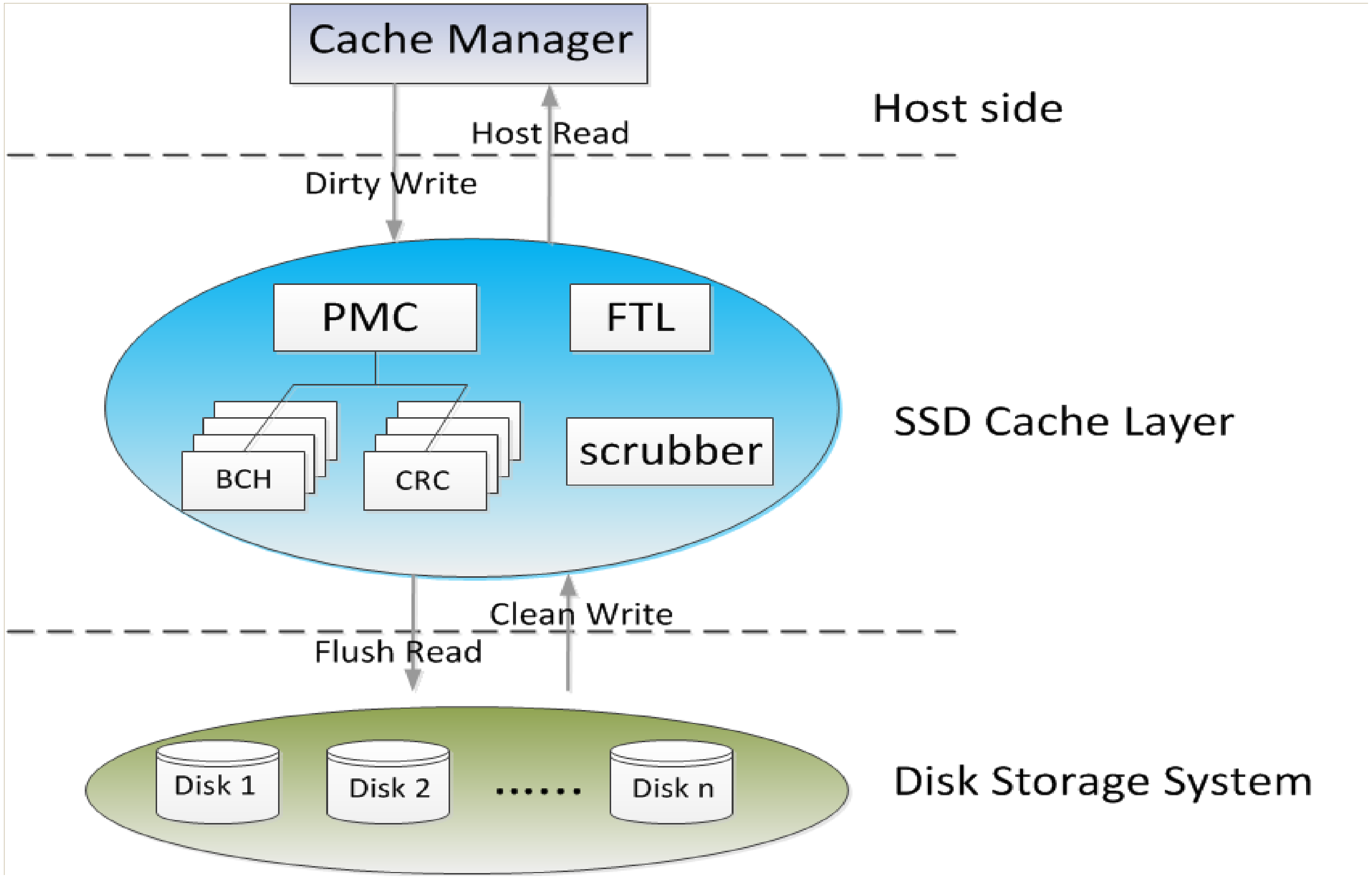
Avoid the overhead

- Flash-based SSDs are mainly used as cache
- Cache-Oriented SSD design
  - idea: for cache usage, we can avoid ECC overhead to improve performance by selectively using EDC, without losing reliability
- Techniques Employed
  - identify the different requirements of cache blocks
  - provide interfaces to help hint passing
  - partition SSD cache into EDC(CRC) and ECC(BCH) regions
  - utilize a programmable memory controller
  - perform scrubbing to prefetch corrupted blocks

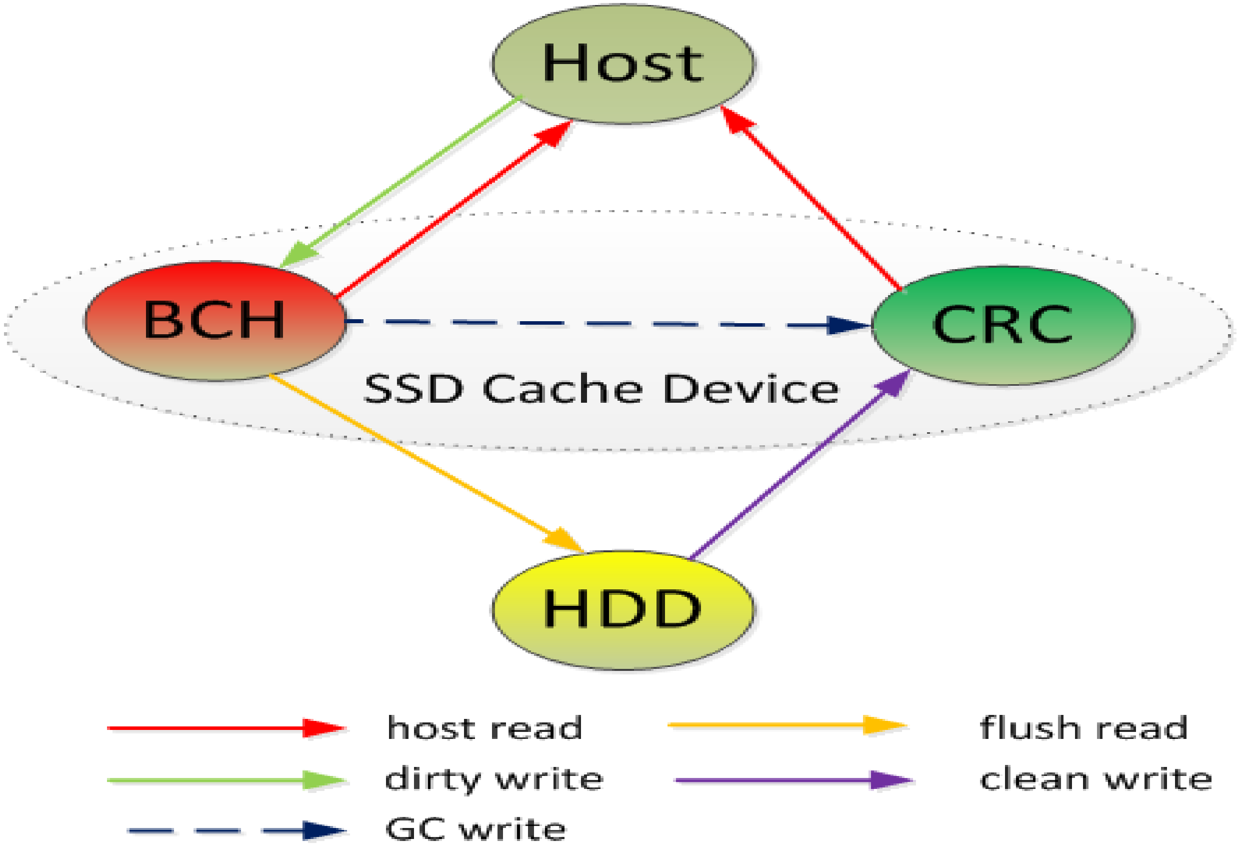
Main Interfaces

- Dirty Write
  - ✓ new data from upper-layer application
  - ✓ ECC applied
- Clean Write
  - ✓ populating cache write (e.g., cache misses)
  - ✓ EDC applied
- Host Read
  - ✓ read requests from application
  - ✓ EDC or ECC applied
- Flush Read
  - ✓ caused by cache write back
  - ✓ make ECC-encoded as EDC-eligible for GC

System Architecture



Page state transition

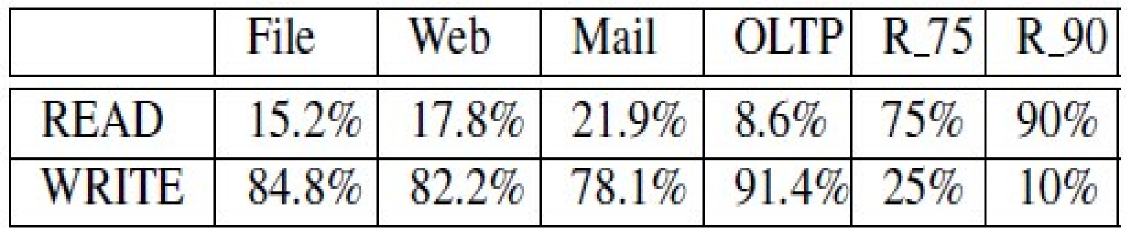


The page's state changes in response to the operations

Evaluation

- Experimental setup
  - □ Use flashcache to collect traces
  - □ SSD simulator to run the traces

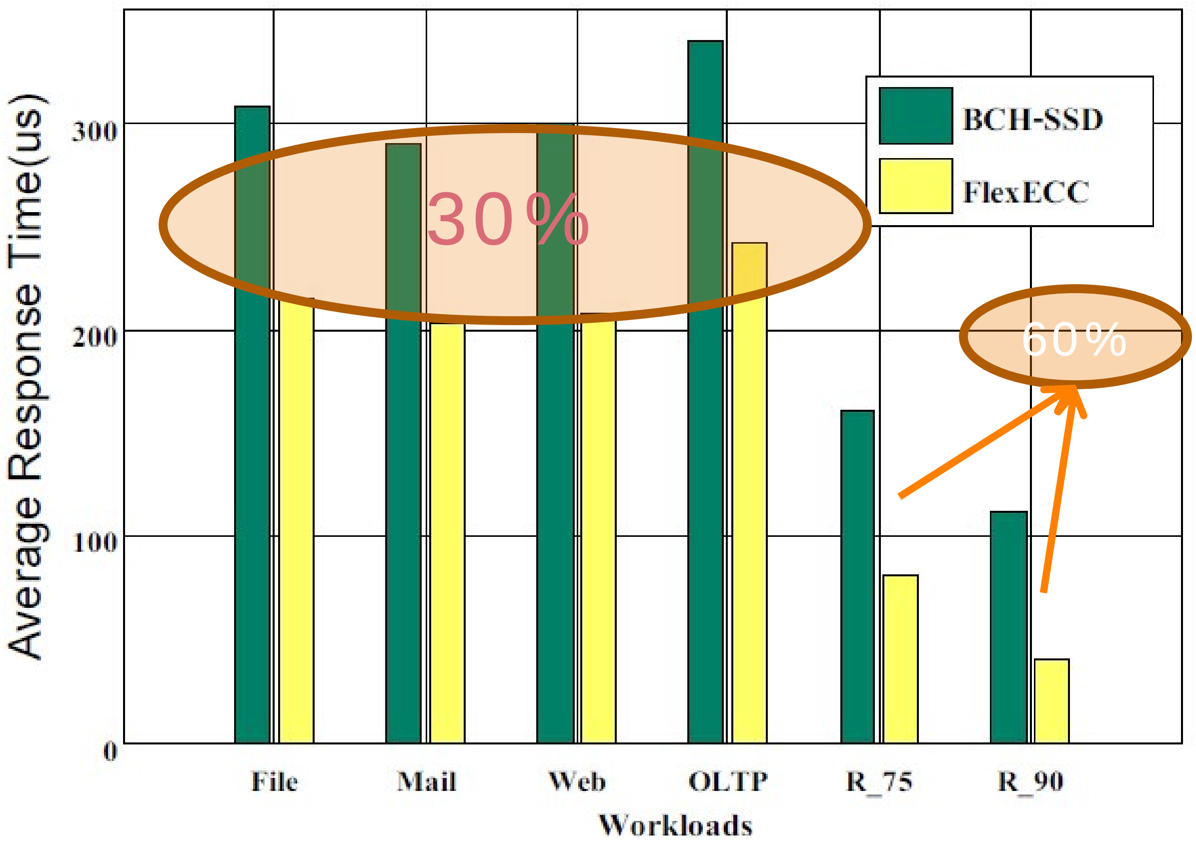
- Workloads
  - □ Use filebench to generate workloads



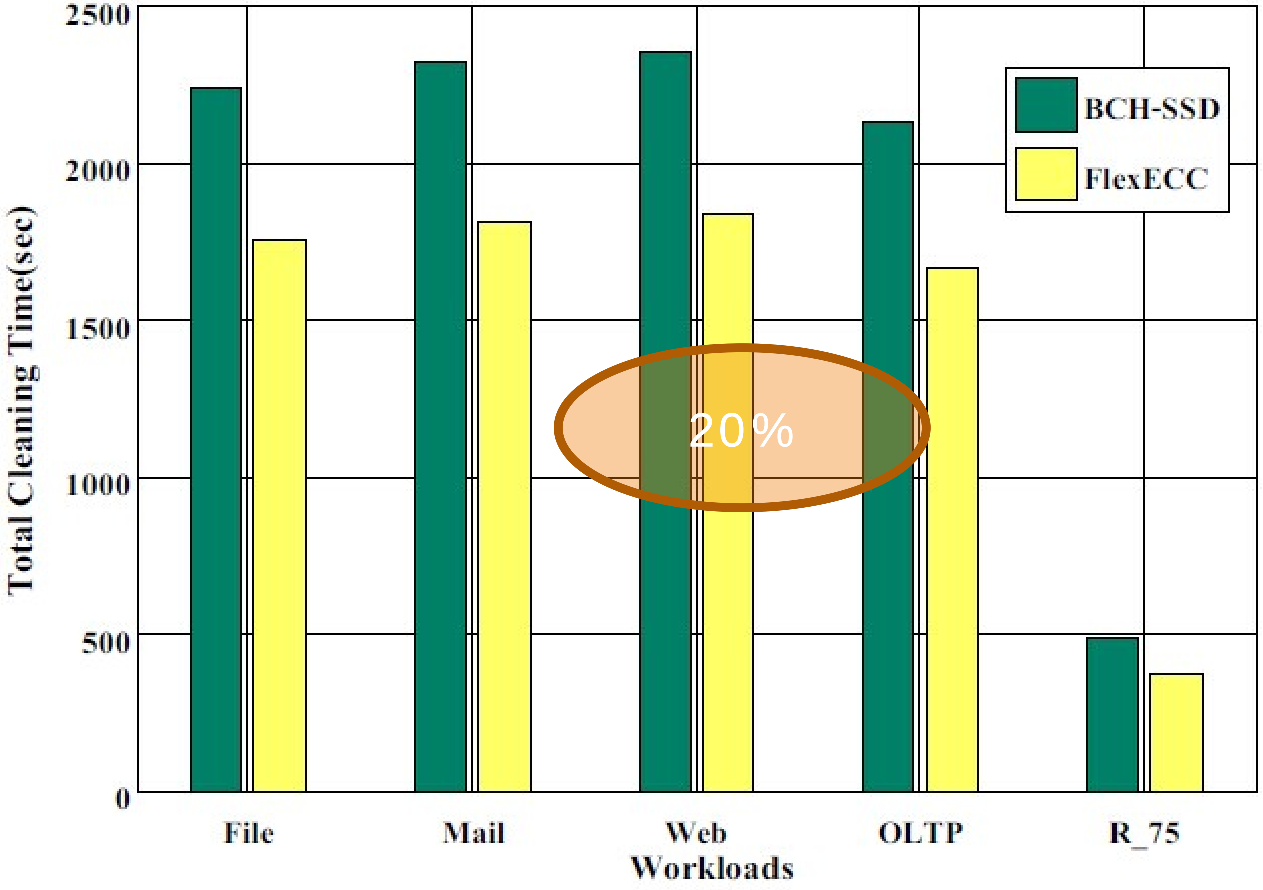
	File	Web	Mail	OLTP	R_75	R_90
READ	15.2%	17.8%	21.9%	8.6%	75%	90%
WRITE	84.8%	82.2%	78.1%	91.4%	25%	10%

- Compare with BCH-SSD

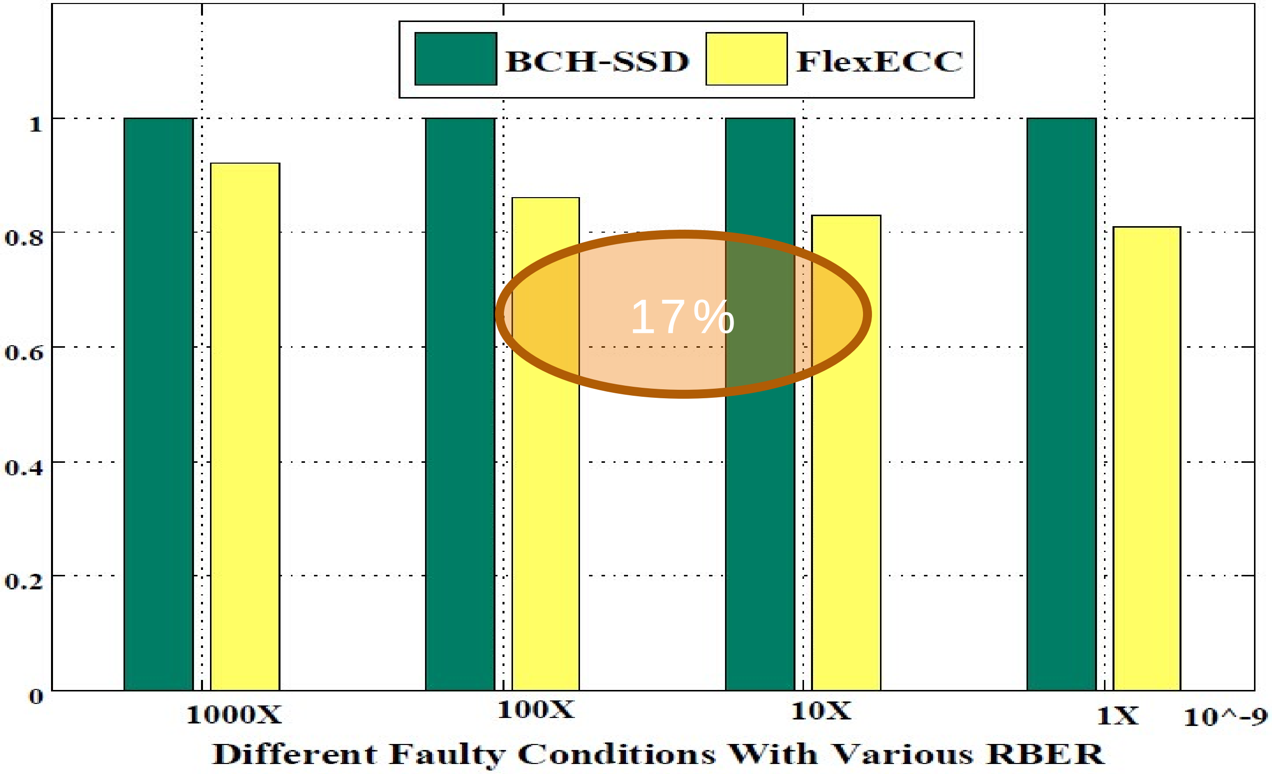
Average response time



Cleaning time



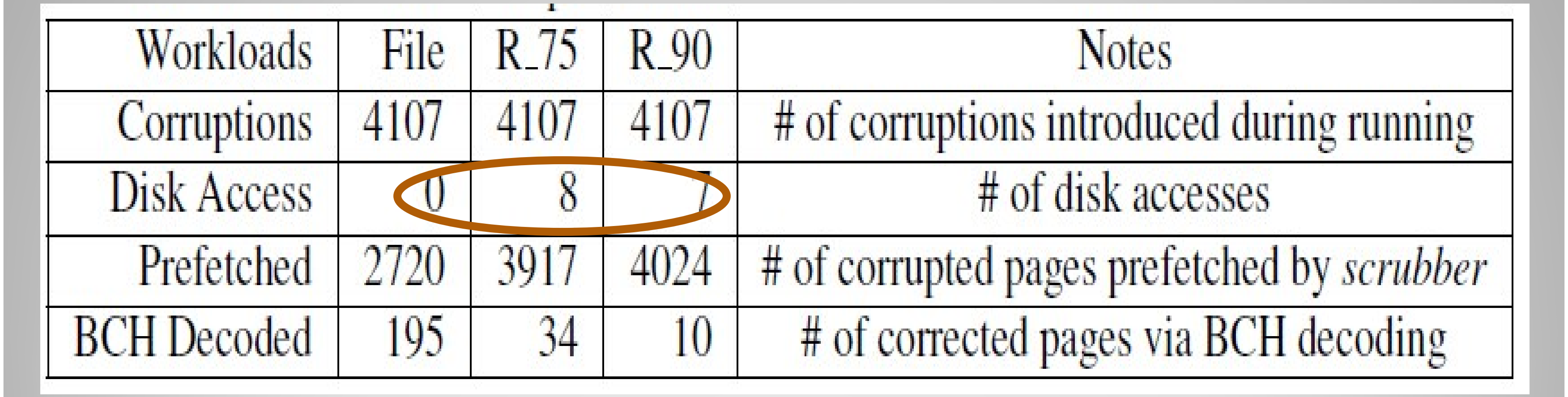
Performance in faulty conditions



The x-axis represents the various RBER the flash exhibits, and the y-axis represents the normalized average response time of fileserver workload relative to the BCH-SSD.

The efficacy of scrubber

Corruption Related Statistics When RBER=10^{-7}



Workloads	File	R_75	R_90	Notes
Corruptions	4107	4107	4107	# of corruptions introduced during running
Disk Access	0	8	7	# of disk accesses
Prefetched	2720	3917	4024	# of corrupted pages prefetched by *scrubber*
BCH Decoded	195	34	10	# of corrected pages via BCH decoding

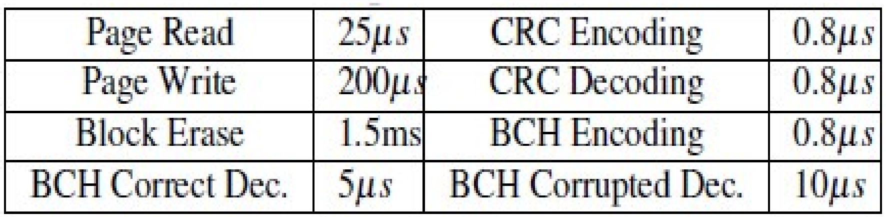
Conclusion

- Propose a cache-oriented SSD design
  - identify cache block reliability requirements
  - apply either EDC(CRC) or ECC(BCH) to blocks
  - schedule a scrubber to ensure data integrity
- Demonstrate its performance advantages
  - improve the performance by 30% on average
  - up to 60% for read-intensive workloads
  - even under various faulty conditions

Thank you!
Q & A

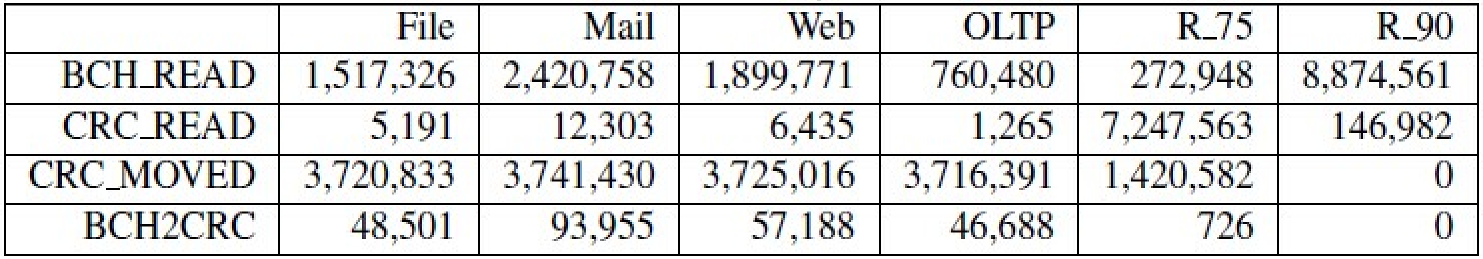
Latencies & Statistics

Operational Latencies



Page Read	25μs	CRC Encoding	0.8μs
Page Write	200μs	CRC Decoding	0.8μs
Block Erase	1.5ms	BCH Encoding	0.8μs
BCH Correct Dec.	5μs	BCH Corrupted Dec.	10μs

Page transition statistics



	File	Mail	Web	OLTP	R_75	R_90
BCH_READ	1,517,326	2,420,758	1,899,771	760,480	272,948	8,874,561
CRC_READ	5,191	12,303	6,435	1,265	7,247,563	146,982
CRC_MOVED	3,720,833	3,741,430	3,725,016	3,716,391	1,420,582	0
BCH2CRC	48,501	93,955	57,188	46,688	726	0