Rewriting Codes for Flash Memory

Ryan Gabrys, Lara Dolecek

Department of Electrical Engineering
University of California-Los Angeles

1. Background
   - Introduction to Flash Memory
   - Modeling the Memory Device
   - Recent Work
   - The Problem
2. Constructions
   - Binary Expansion
   - Non-Binary Concatenation
   - Spider Codes
3. Conclusion and future work

Technical Constraint

- Information is stored by modulating the number of electrons stored within each of the floating gate cell.

Technical Constraint

- Information is stored by modulating the number of electrons stored within each of the floating gate cell.
- This information is then compared against some threshold and a discrete value is read.

Technical Constraint

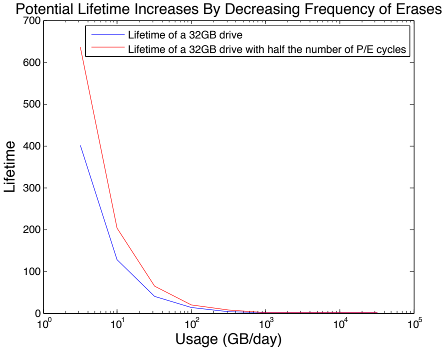
- Information is stored by modulating the number of electrons stored within each of the floating gate cell.
- This information is then compared against some threshold and a discrete value is read.
- Programming is performed at the granularity of an individual cell while erasing is performed on blocks of cells ($\approx 10^6$ cells).

Technical Constraint

- Information is stored by modulating the number of electrons stored within each of the floating gate cell.
- This information is then compared against some threshold and a discrete value is read.
- Programming is performed at the granularity of an individual cell while erasing is performed on blocks of cells ($\approx 10^6$ cells).
- This need for frequent erases speeds up the aging process drastically [1].

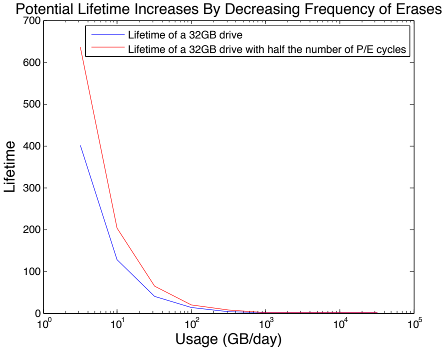
[1] A.R. Olson and D.J. Langlois, "Solid State Drives Data Reliability and Lifetime," *Imation White Paper* April 2008.

Impact of program and erase cycles on lifetime



[1] A.R. Olson and D.J. Langlois, "Solid State Drives Data Reliability and Lifetime," *Imation White Paper* April 2008.

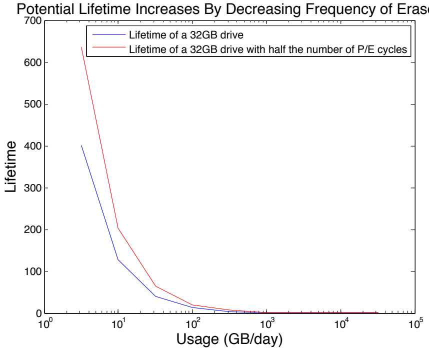
Impact of program and erase cycles on lifetime



- Lifetime is a function of number of program and erases

[1] A.R. Olson and D.J. Langlois, "Solid State Drives Data Reliability and Lifetime," *Imation White Paper* April 2008.

Impact of program and erase cycles on lifetime



- Lifetime is a function of number of program and erases
- Goal is to prolong the lifetime of the memory by reducing the number of erases.

[1] A.R. Olson and D.J. Langlois, "Solid State Drives Data Reliability and Lifetime," *Imation White Paper* April 2008.

Model

Memory Model

The memory state is modeled as a vector $\mathbf{y}^j$ of length n where j is the current write (or generation). Each element y_i^j, $1 \leq i \leq n$, takes values in the set $\{0, 1, 2, \ldots, q-1\}$. On write j, the encoder writes one of M_j messages to the memory by updating $\mathbf{y}^{j-1}$ to $\mathbf{y}^j$ while satisfying the WOM-constraint $\mathbf{y}^j \geq \mathbf{y}^{j-1}$. Decoder sees $\mathbf{y}^j$.

Sum-Rate

Definition

If M_j codewords can be represented at generation j, then generation j has rate $\frac{1}{n}\log(M_j)$. Sum rate is the sum of rates across generations.

Rivest and Shamir two write code [1]

Information	First Generation	Second Generation
00	000	111
01	001	110
10	010	101
11	100	011

[1] R. L. Rivest, A. Shamir, "How to reuse a write-once memory" *Information and Control*, vol. 55, no. 1-3, pp. 119, 1982.

Previous work

- Early capacity results
  - Rivest and Shamir, "How to Reuse a "Write-Once" Memory," 1982.
  - Wolf *et al.*, "Coding for Write-Once Memory," 1984.
  - Heegard, "On the Capacity of Permanent Memory," 1985.
  - Fu and Han Vinck, "On the Capacity of Generalized Write-Once Memory with State Transitions Described by an Arbitrary Directed Acyclic Graph," 1999.

Previous work

- Early capacity results
  - Rivest and Shamir, "How to Reuse a "Write-Once" Memory," 1982.
  - Wolf *et al.*, "Coding for Write-Once Memory," 1984.
  - Heegard, "On the Capacity of Permanent Memory," 1985.
  - Fu and Han Vinck, "On the Capacity of Generalized Write-Once Memory with State Transitions Described by an Arbitrary Directed Acyclic Graph," 1999.
- Flash Memory Codes
  - Yaakobi *et al.*, "Codes for Write-Once Memories", 2012.
  - Huang *et al.*, "Error-Correcting Codes for Flash Coding", 2011.
  - Shpilka, "Capacity achieving multiwrite WOM codes", 2012.

Recent WOM-code Constructions

- Yaakobi *et al.*, “Codes for Write-Once Memories,” 2012.
  - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.

Recent WOM-code Constructions

- Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.
  - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.
  - Constructions were limited to the binary case.

Recent WOM-code Constructions

- Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.
  - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.
  - Constructions were limited to the binary case.
  - Constructions had relatively small block lengths.

Recent WOM-code Constructions

- Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.
  - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.
  - Constructions were limited to the binary case.
  - Constructions had relatively small block lengths.
- Shpilka, "Capacity achieving multiwrite WOM codes," 2012.

Recent WOM-code Constructions

- Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.
  - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.
  - Constructions were limited to the binary case.
  - Constructions had relatively small block lengths.
- Shpilka, "Capacity achieving multiwrite WOM codes," 2012.
  - Achieved capacity for the multiple write case when the block length goes to infinity using collections of random matrices for mappings messages to second generation codewords.

Recent WOM-code Constructions

- Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.
    - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.
    - Constructions were limited to the binary case.
    - Constructions had relatively small block lengths.
- Shpilka, "Capacity achieving multiwrite WOM codes," 2012.
    - Achieved capacity for the multiple write case when the block length goes to infinity using collections of random matrices for mappings messages to second generation codewords.
    - Also achieved capacity for non-binary WOM-codes (any number of writes) .

Recent WOM-code Constructions

- Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.
  - Achieved capacity for the two write case when the block length goes to infinity using random matrices for mappings messages to second generation codewords.
  - Constructions were limited to the binary case.
  - Constructions had relatively small block lengths.
- Shpilka, "Capacity achieving multiwrite WOM codes," 2012.
  - Achieved capacity for the multiple write case when the block length goes to infinity using collections of random matrices for mappings messages to second generation codewords.
  - Also achieved capacity for non-binary WOM-codes (any number of writes) .
  - Codes have block lengths that grow exponentially as the gap to capacity becomes smaller.

Problem Statement and Our Contribution

- The goal is to provide flexible high-rate *non-binary* strategies for *multilevel* WOM.

Problem Statement and Our Contribution

- The goal is to provide flexible high-rate *non-binary* strategies for *multilevel* WOM.
- The contribution of this work is to provide the best known non-binary WOM-codes with block lengths that need not grow exponentially with the block length of the code.

Our Approach

- Begin with good t-write codes over smaller alphabets.

Our Approach

- Begin with good t-write codes over smaller alphabets.
- Define a mapping into a code with a larger alphabet.

Our Approach

- Begin with good t-write codes over smaller alphabets.
- Define a mapping into a code with a larger alphabet.
- Construct a new non-binary code based on these constituents.

Construction 1

Construction 1 - Binary Expansion Construction

Binary Expansion Construction

- Consider $q = (q^{'})^k$.

Binary Expansion Construction

- Consider $q = (q^{'})^{k}$.
- Let $C^{1}_{t,n,q^{'}}, C^{2}_{t,n,q^{'}}, \ldots, C^{k}_{t,n,q^{'}}$ be k independent t-write WOM-Codes over the alphabet $q^{'}$ and of length n.

Binary Expansion Construction

- Consider $q = (q^{'})^k$.
- Let $C^1_{t,n,q^{'}}, C^2_{t,n,q^{'}}, \ldots, C^k_{t,n,q^{'}}$ be k independent t-write WOM-Codes over the alphabet $q^{'}$ and of length n.
- For any collection of codewords $\mathbf{c_1} \in C^1_{t,n,q^{'}}, \mathbf{c_2} \in C^2_{t,n,q^{'}}, \ldots, \mathbf{c_k} \in C^k_{t,n,q^{'}}$, a codeword $\mathbf{c} \in C_{t,n,q}$ is formed through the operation $\mathbf{c} = \sum_{j=0}^{k-1} (q^{'})^j \mathbf{c_j}$.

Binary Expansion Construction

- Consider $q = (q^{'})^k$.
- Let $C^1_{t,n,q^{'}}, C^2_{t,n,q^{'}}, \ldots, C^k_{t,n,q^{'}}$ be k independent t-write WOM-Codes over the alphabet $q^{'}$ and of length n.
- For any collection of codewords $\mathbf{c_1} \in C^1_{t,n,q^{'}}, \mathbf{c_2} \in C^2_{t,n,q^{'}}, \ldots, \mathbf{c_k} \in C^k_{t,n,q^{'}}$, a codeword $\mathbf{c} \in C_{t,n,q}$ is formed through the operation $\mathbf{c} = \sum_{j=0}^{k-1} (q^{'})^j \mathbf{c_j}$.
- Resultant maximum sum-rate is $\log_2(\binom{q^{'}+t-1}{t}) \log_{q^{'}}(q)$.

An example of Construction 1 using Rivest and Shamir write-twice code

Write number	*Data bits*	*Encoding by the base-code $\mathcal{C}_2$*	*Encoded values in the 8-ary cell*
1	*(11,01,10)*	*(100,001,010)*	*(4,1,2)*
2	*(00,11,01)*	*(101,111,110)*	*(5,7,6)*

Information	First Generation	Second Generation
00	000	111
01	001	110
10	010	101
11	100	011

[1] R. L. Rivest, A. Shamir, "How to reuse a write-once memory" *Information and Control*, vol. 55, no. 1-3, pp. 119, 1982.

Construction 2

Construction 2 - Non-Binary Concatenation Construction

Non-Binary Concatenation Construction

- Let $q^{'}$ be such that $q^{'} + t - 1 \mid q$ where t is the number of writes.

Non-Binary Concatenation Construction

- Let $q^{'}$ be such that $q^{'} + t - 1 \mid q$ where t is the number of writes.
- Let $C_{t,n,q^{'}}$ be a base t-write WOM-code over the alphabet $q^{'}$ of length n.

Non-Binary Concatenation Construction

- Let $q^{'}$ be such that $q^{'} + t - 1 \mid q$ where t is the number of writes.
- Let $C_{t,n,q^{'}}$ be a base t-write WOM-code over the alphabet $q^{'}$ of length n.
- Represent information sequences as the ordered pair $(\mathbf{c_{j,q^{'}}}, \mathbf{v})$ where $\mathbf{c_{j,q^{'}}}$ can be any codeword from $C_{t,n,q^{'}}$ and $\mathbf{v}$ is a vector of length n with elements $\{0, 1, 2, \ldots, \frac{q}{q^{'}+t-1} - 1\}$.

Non-Binary Concatenation Construction

- Let q' be such that $q' + t - 1 \mid q$ where t is the number of writes.
- Let $C_{t,n,q'}$ be a base t-write WOM-code over the alphabet q' of length n.
- Represent information sequences as the ordered pair $(\mathbf{c_{j,q'}}, \mathbf{v})$ where $\mathbf{c_{j,q'}}$ can be any codeword from $C_{t,n,q'}$ and $\mathbf{v}$ is a vector of length n with elements $\{0, 1, 2, \ldots, \frac{q}{q'+t-1} - 1\}$.
- Codewords are $\mathbf{c} = \frac{q}{q'+t-1}(\mathbf{c_{j,q'}} + j\mathbf{1_n}) + \mathbf{v}$ where $\mathbf{1_n}$ is the all ones vector of length n.

Non-Binary Concatenation Construction

- Let $q^{'}$ be such that $q^{'} + t - 1 \mid q$ where t is the number of writes.
- Let $C_{t,n,q^{'}}$ be a base t-write WOM-code over the alphabet $q^{'}$ of length n.
- Represent information sequences as the ordered pair $(\mathbf{c_{j,q^{'}}}, \mathbf{v})$ where $\mathbf{c_{j,q^{'}}}$ can be any codeword from $C_{t,n,q^{'}}$ and $\mathbf{v}$ is a vector of length n with elements $\{0, 1, 2, \ldots, \frac{q}{q^{'}+t-1} - 1\}$.
- Codewords are $\mathbf{c} = \frac{q}{q^{'}+t-1}(\mathbf{c_{j,q^{'}}} + j\mathbf{1_n}) + \mathbf{v}$ where $\mathbf{1_n}$ is the all ones vector of length n.
- Resultant maximum sum-rate $t\log_2(\frac{q}{q^{'}+t-1}) + \log_2(\binom{q^{'}+t-1}{t})$.

A simple example

Construct a 2-write code using 9 levels from a $t = 2$ write code of $q' = 2$ levels.

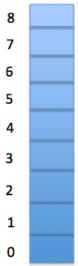


Figure: Cell with $q = 9$ levels.

First write

Encode $(\mathbf{c}_{0,2}, \mathbf{v})$ where $\mathbf{c}_{0,2}$ is first generation codeword from a write-twice binary codebook and $\mathbf{v}$ is a vector with elements $\{0, 1, 2\}$.

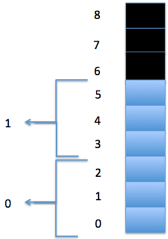


Figure: Cell with $q = 9$ levels after 1st generation.

Second write

Encode $(\mathbf{c}_{1,2}, \mathbf{v})$ where $\mathbf{c}_{1,2}$ is a second generation codeword from a write-twice binary codebook and $\mathbf{v}$ is a vector with elements $\{0, 1, 2\}$.

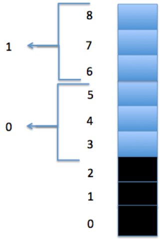


Figure: Cell with $q = 9$ levels after 2nd generation.

Construction 3

Two-Write Spider Codes

Two-Write Spider Codes

- Idea is to use very long binary WOM-codes that, for a parameter p the following is true.

[1] E. Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.

Two-Write Spider Codes

- Idea is to use very long binary WOM-codes that, for a parameter p the following is true.
  - Encode with rate $H(p)$ on the first generation.

[1] E. Yaakobi *et al.*, “Codes for Write-Once Memories,” 2012.

Two-Write Spider Codes

- Idea is to use very long binary WOM-codes that, for a parameter p the following is true.
  - Encode with rate $H(p)$ on the first generation.
  - A rate $1-p$ is written the second generation.

[1] E. Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.

Two-Write Spider Codes

- Idea is to use very long binary WOM-codes that, for a parameter p the following is true.
  - Encode with rate $H(p)$ on the first generation.
  - A rate $1-p$ is written the second generation.
  - Such codes were shown to exist in [1].

[1] E. Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.

Two-Write Spider Codes

- Idea is to use very long binary WOM-codes that, for a parameter p the following is true.
  - Encode with rate $H(p)$ on the first generation.
  - A rate $1-p$ is written the second generation.
  - Such codes were shown to exist in [1].
- In the first generation an equal weight sequence is sent.

[1] E. Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.

Two-Write Spider Codes

- Idea is to use very long binary WOM-codes that, for a parameter p the following is true.
  - Encode with rate $H(p)$ on the first generation.
  - A rate $1 - p$ is written the second generation.
  - Such codes were shown to exist in [1].
- In the first generation an equal weight sequence is sent.
- In the second generation, the second generation of a collection of binary codebooks will be used to encode more information.

[1] E. Yaakobi *et al.*, "Codes for Write-Once Memories," 2012.

The idea when $q = 4$

- For the first generation codewords, equal weight sequences are written.

The idea when $q = 4$

- For the first generation codewords, equal weight sequences are written.
- For the second generation, 3 auxiliary binary codebooks are used. There are 3 stages:

The idea when $q = 4$

- For the first generation codewords, equal weight sequences are written.
- For the second generation, 3 auxiliary binary codebooks are used. There are 3 stages:
  - Stage 1: Increment the current state of the memory using a second generation codebook in such a way that every codeword element is incremented by at most two levels.

The idea when $q = 4$

- For the first generation codewords, equal weight sequences are written.
- For the second generation, 3 auxiliary binary codebooks are used. There are 3 stages:
  - Stage 1: Increment the current state of the memory using a second generation codebook in such a way that every codeword element is incremented by at most two levels.
  - Stage 2: Update the memory further by using a codebook that only increments the elements in the codeword vector with value 0.

The idea when $q = 4$

- For the first generation codewords, equal weight sequences are written.
- For the second generation, 3 auxiliary binary codebooks are used. There are 3 stages:
  - Stage 1: Increment the current state of the memory using a second generation codebook in such a way that every codeword element is incremented by at most two levels.
  - Stage 2: Update the memory further by using a codebook that only increments the elements in the codeword vector with value 0.
  - Stage 3: Update the memory further by using a codebook that only increments the elements in the codeword vector with value 1.

Summary - Rates achieved for 2 writes

q	B.E.	N.B.C.	S.C.	Capacity
4	2.9856	-	3.2970	3.3219
8	4.4784	4.3299	5.0506	5.1699
16	5.9712	6.3299	6.8875	7.0875
32	7.4640	8.3299	8.8059	9.0444
64	8.9568	10.3299	10.733	11.0224
128	10.4496	12.3299	12.7010	13.0112

Conclusion and future work

- Conclusion
  - It is possible to use leverage high-rate binary codebooks to produce high-rate non-binary codebooks.
  - The complexity of the resulting non-binary constructions are on the same order of magnitude as their binary constituents.

Conclusion and future work

- Conclusion
  - It is possible to use leverage high-rate binary codebooks to produce high-rate non-binary codebooks.
  - The complexity of the resulting non-binary constructions are on the same order of magnitude as their binary constituents.
- Future Work
  - Enhancement with error correction/detection capabilities.
  - Constructing better codes for the expected case.

Thank You

Thank you for your attention