Coding for Flash Memories

Eitan Yaakobi, Jing Ma, Adrian Caulfield, Laura Grupp
Steven Swanson, Paul H. Siegel, Jack K. Wolf

Center for Magnetic Recording Research
University of California San Diego

Non-Volatile Memories Workshop, April 2010

CMRR
Center for Magnetic Recording Research

UCSD

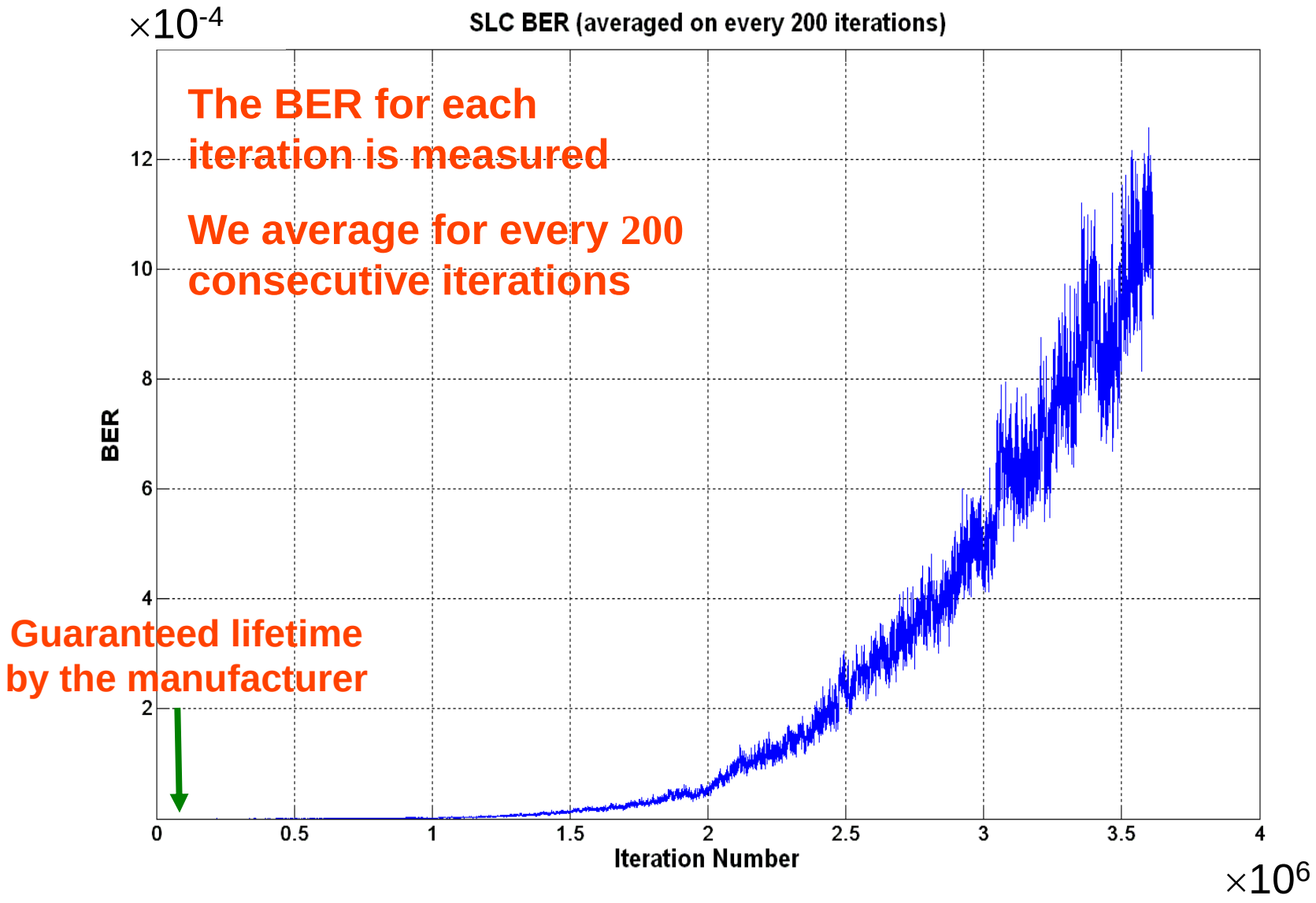
Outline

- Error Characterization of Flash Memories
- Implementing WOM-codes in SLC Flash
- WOM-codes with Better Rates

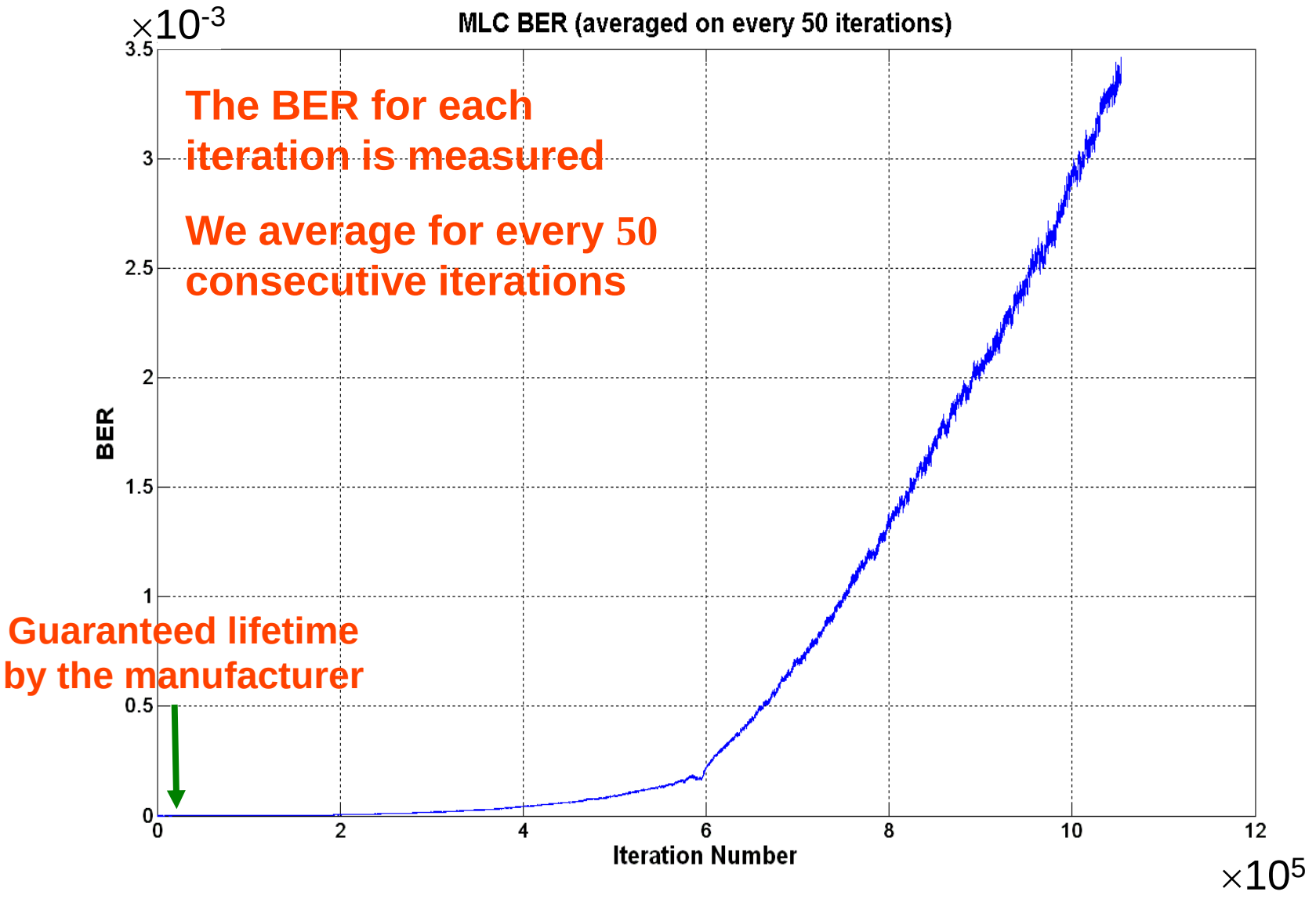
Experiment Description

- We checked several flash memory blocks - **SLC** and **MLC**
- For each block the following steps are repeated
  - The block is **erased**
  - A pseudo-random data is **written** to the block
  - The data is **read** and **compared** to find errors
- Two runs are chosen as representatives for SLC and MLC
  - **SLC** - Collected Iterations: **3,615,224**
  - **MLC** - Collected Iterations: **1,054,031**
- **Remarks**:
  - We measured many more iterations than the manufacturer's guaranteed number of erasures
  - The experiment was done in a lab conditions and related factors such as temperature change, intervals between erasure or multiple readings before erasures were not considered

Raw BER for SLC block



Raw BER for MLC block



Basic Error Characteristic

- **Directional Errors** - Number of Plus errors Vs. Minus errors
  - **Plus error**: Cell in state ‘0’ changes to state ‘1’
  - **Minus error**: Cell in state ‘1’ changes to state ‘0’
  - We check the distribution of the errors between plus and minus errors and the numbers are the same even for individual bits
  - **Conclusion**: There is no preference to correct more plus than minus errors (or vice versa)
- **Burst behavior**
  - Check how close the errors are to each other
  - Explore the potential (dis)advantage of using a RS code VS a BCH code
  - **Conclusion**: The errors are random and do not have burst pattern behavior

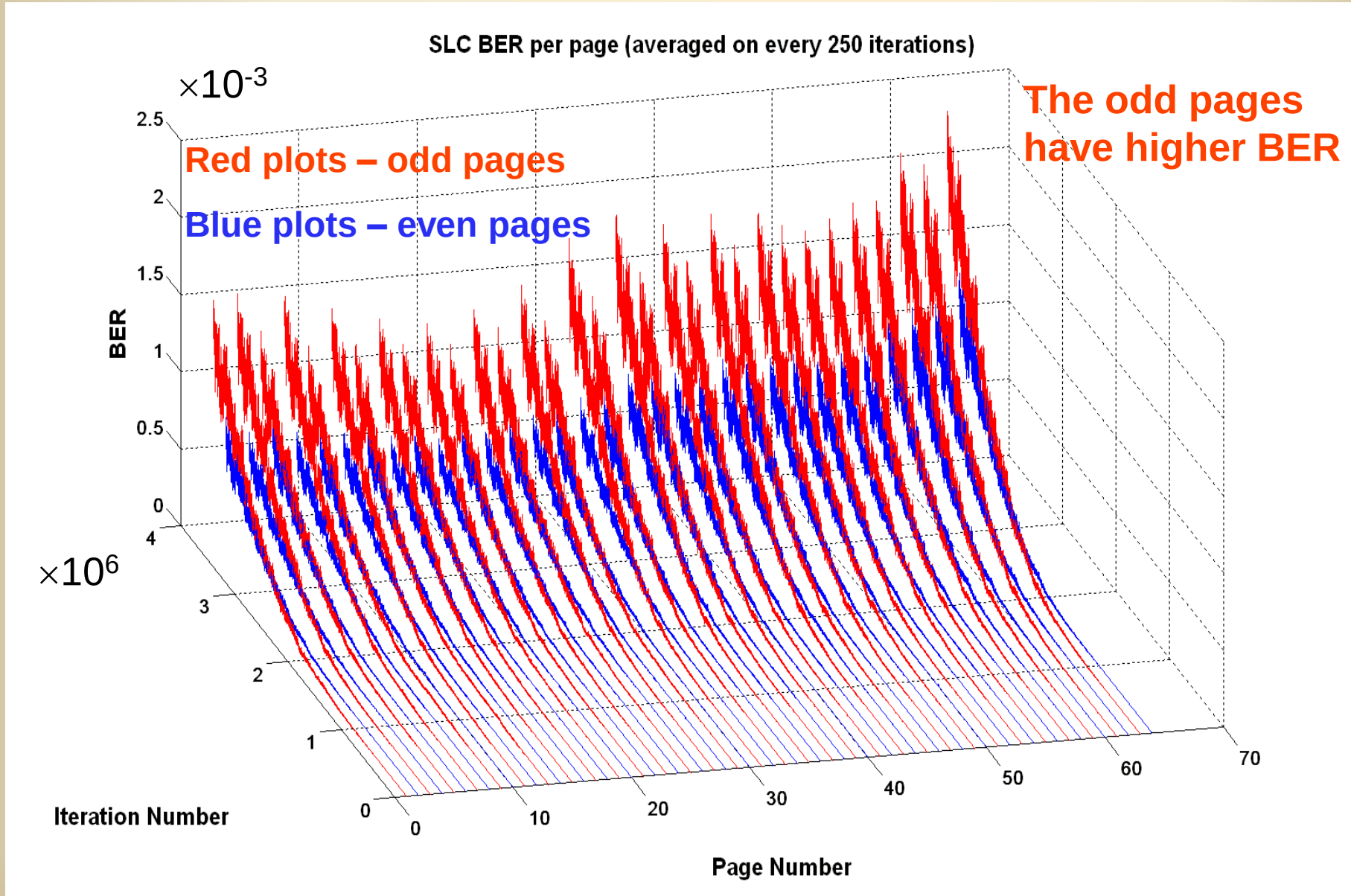
BER Page Dependency

- We checked the BER for each page independently
- SLC block cells layout:

  $32x2^{15}$ cells; each page has 2^{14} cells

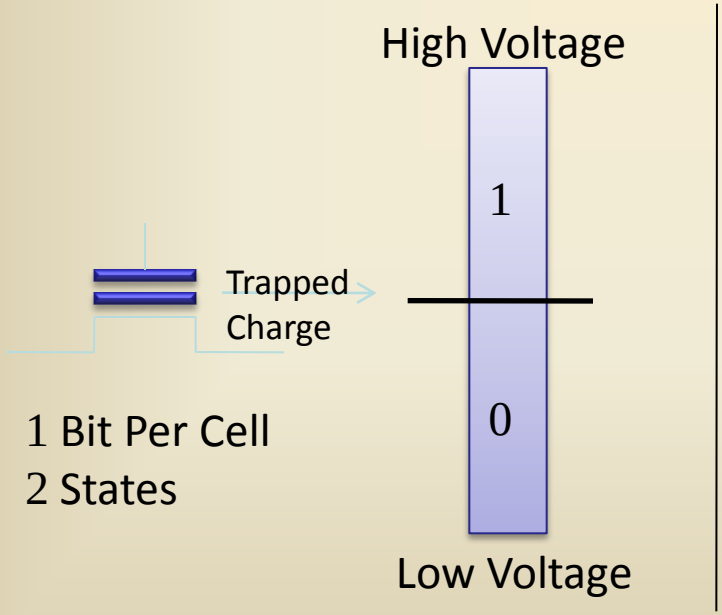
page 1	page 2
page 3	page 4
page 5	page 6
.	.
.	.
.	.
page 63	page 64

BER per page for SLC block

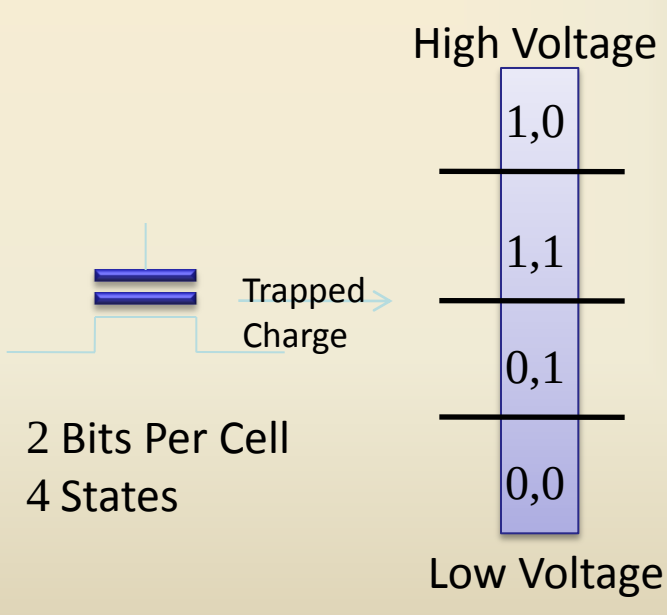


BER Page Dependency for MLC

- The Most Significant Bit (**MSB**) and Least Significant Bit (**LSB**) represented by one cell belong to two different pages
- Pages storing the LSB are less protected from errors



Single Level Cell (SLC)



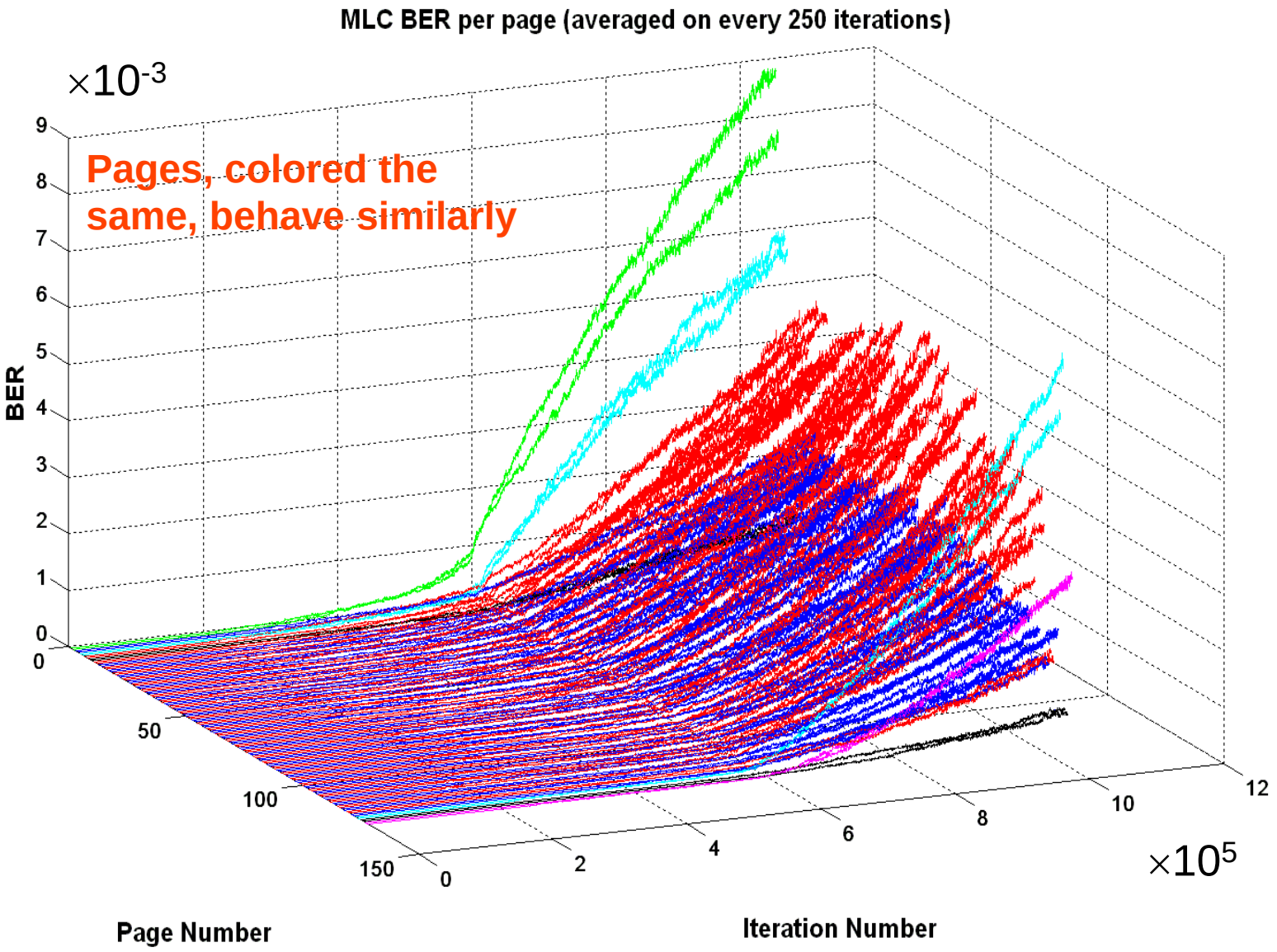
Multi Level Cell (MLC)

BER Page Dependency for MLC

A possible MLC block cell layout:

Row index	**LSB** of **first** 2^{14} cells	**MSB** of **first** 2^{14} cells	**LSB** of **last** 2^{14} cells	**MSB** of **last** 2^{14} cells
1	page 1	page 5	page 2	page 6
2	page 3	page 9	page 4	page 10
3	page 7	page 13	page 8	page 14
4	page 11	page 17	page 12	page 18
5	page 15	page 21	page 16	page 22
M	M	M	M	M
31	page 119	page 125	page 120	page 126
32	page 123	page 127	page 124	page 128

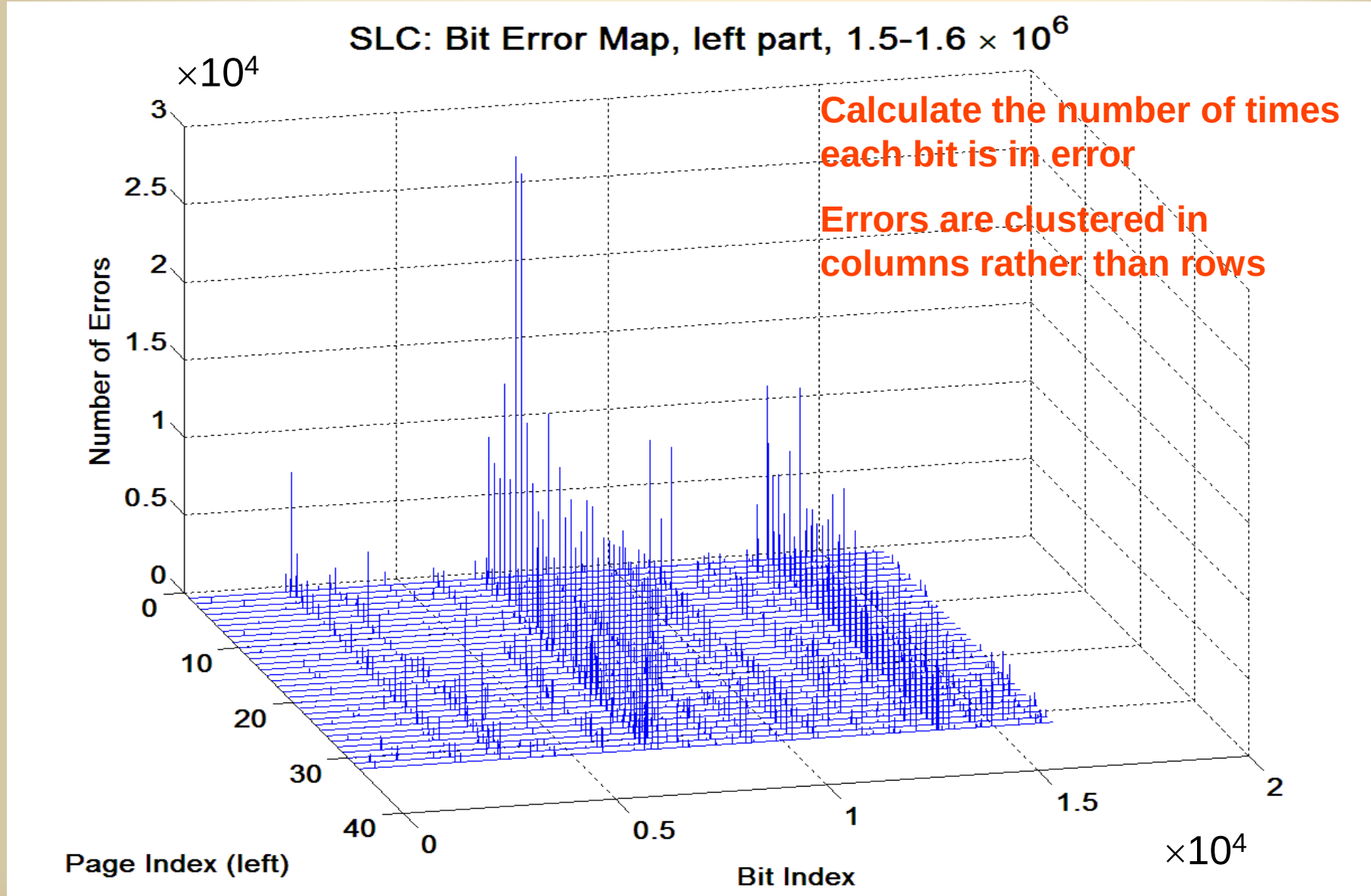
BER per page for MLC block



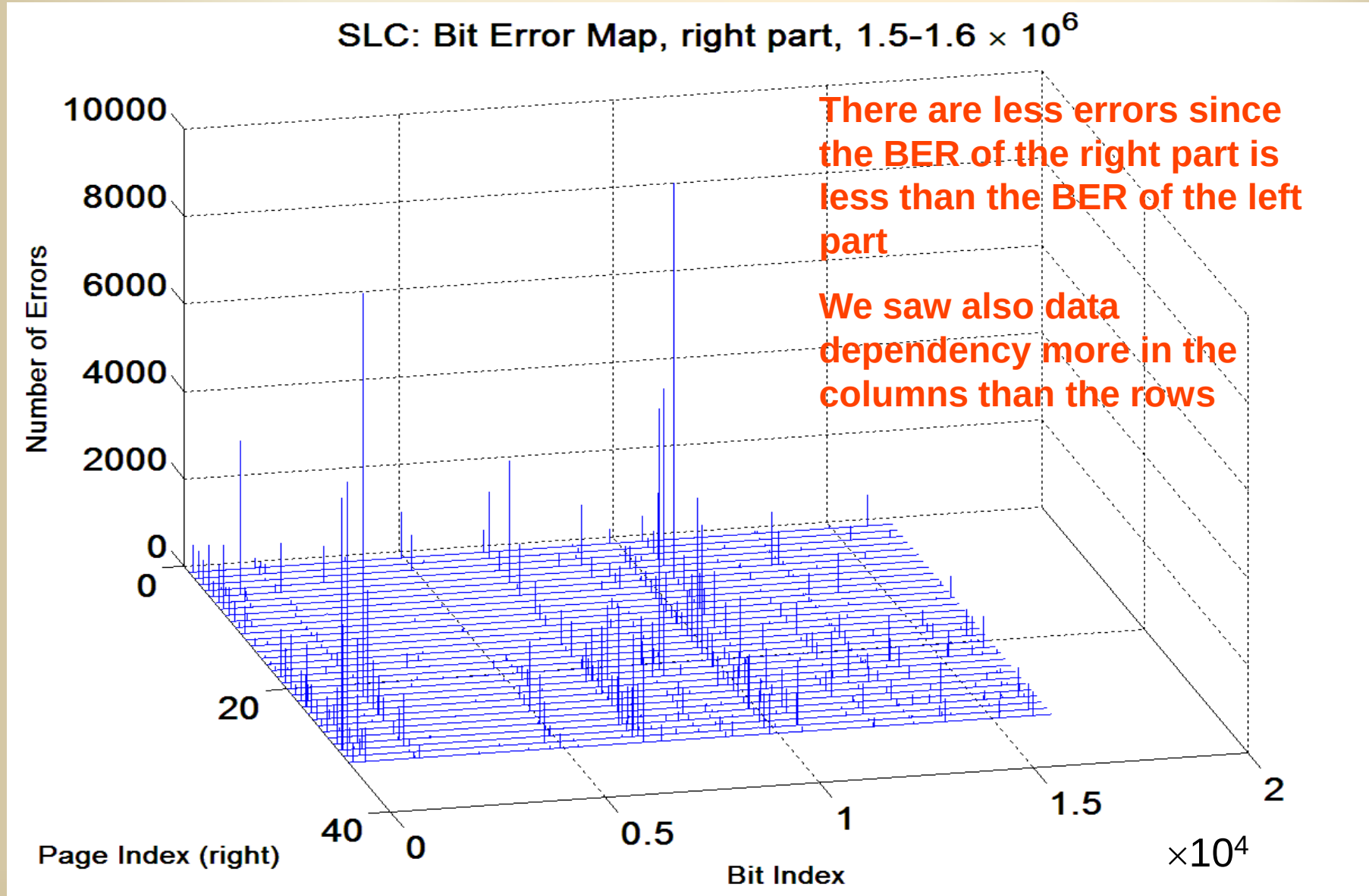
Bit Error Map in SLC

- We checked how the errors behave for every bit
- For a small window of iterations, $1.5\text{-}1.6\times10^{6}$ iterations - the BER is roughly fixed, we checked for each bit the number of times it was in error

Bit Error Map for Odd Pages in SLC



Bit Error Map for Odd Pages in MLC



More Work

- We compared between BCH codes, RS codes, and burst correcting codes
- We evaluated another ECC scheme that works in two levels:
  - First level: ECC for each page
  - Second level: ECC for the entire block

Write Once Memories (WOM)

- Introduced by **Rivest and Shamir**, "*How to reuse a write-once memory*", 1982
- The memory elements represent bits (2 levels) and can be irreversibly programmed from the **'0'** state to the **'1'** state
- **The problem:**

  What is the required minimum number of cells, w, to write k bits t times?
  - In order to store k bits twice ($t = 2$) at least $\sim 1.29k$ cells are required
  - A simple construction achieves $1.5k$ cells
- Alternatively, it is possible to ask: what is the **maximum total number of bits** that can be written using n cells and t writes?

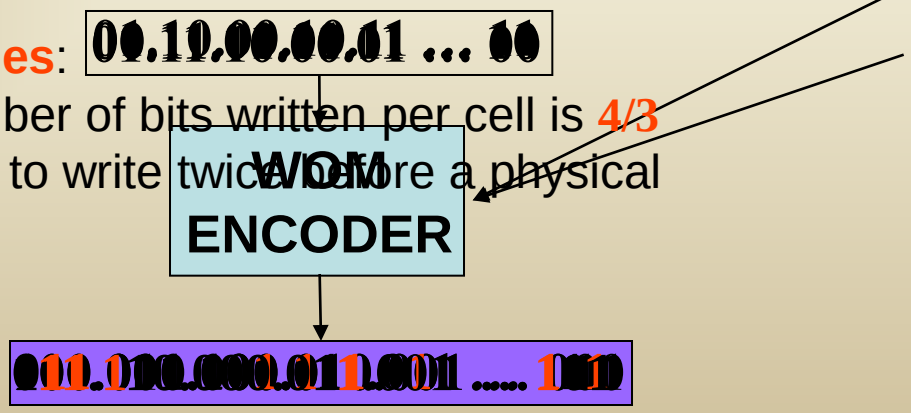
Write Once Memory (WOM) Codes for SLC

- A scheme for storing two bits twice using only three cells **before erasing the cells**
- The cells only **increase** their level
- How to implement? (in **SLC** block)
  - Each page stores **2KB/1.5 = 4/3KB** per write
  - A page can be written twice before erasing
  - Pages are **encoded** using the **WOM code**
  - When the block has to be rewritten, mark its pages as **invalid**
  - Again write pages using the WOM code **without erasing**
  - **Read before write** at the second write

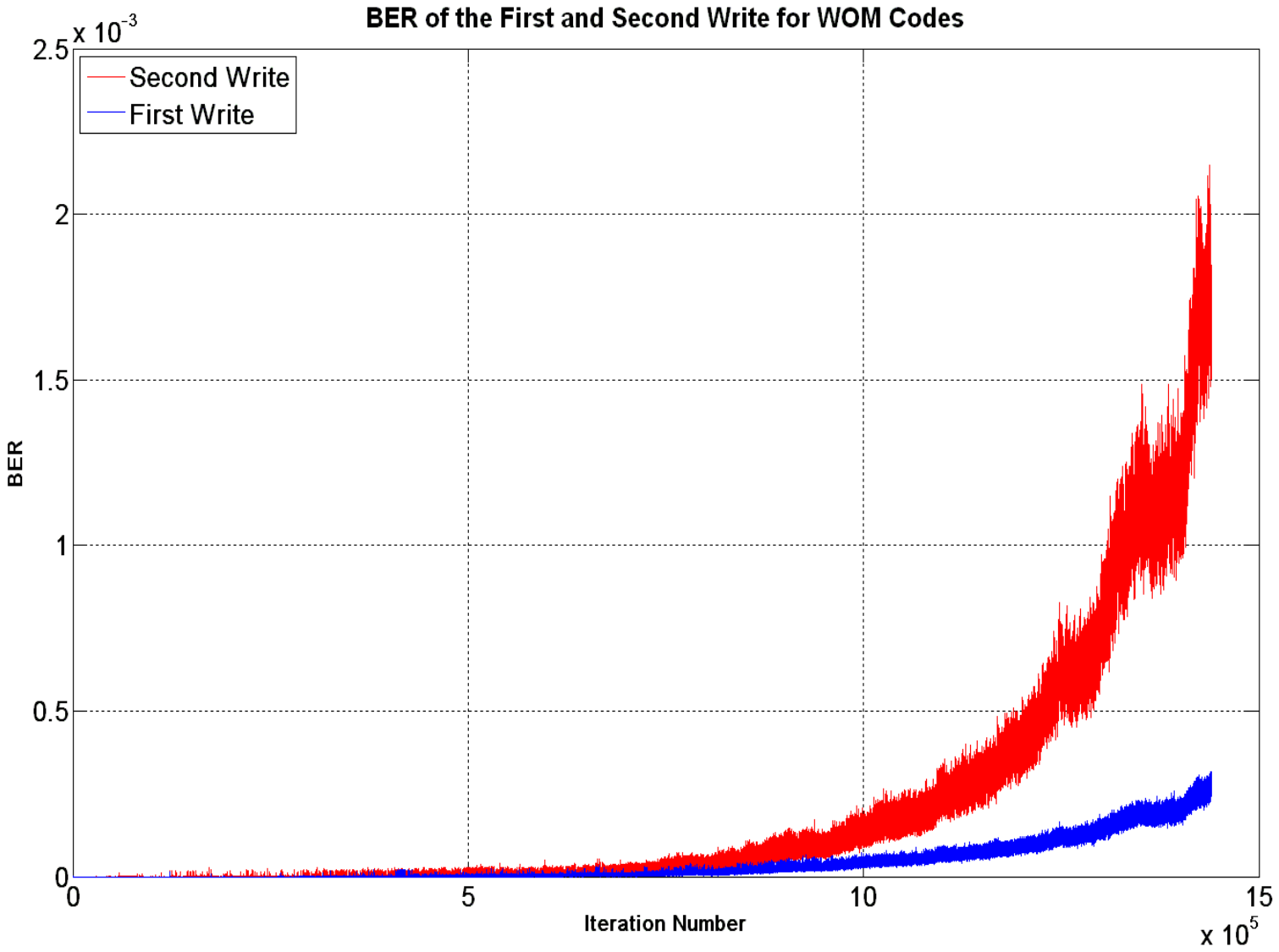
data	1st write	2nd write
00	000	111
01	100	011
10	010	101
11	001	110

Cells state

Advantages:
- The number of bits written per cell is **4/3**
- Possible to write twice before a physical erasure



BER for the First and Second Writes



WOM-Codes with two writes

- Joint work with Scott Kayser
- Assume there are n cells and two writes, $t = 2$
  - First write: k_1 bits, $R_1 = k_1/n$, second write: k_2 bits, $R_2 = k_2/n$
  - Capacity region (**Heegard** 1986, **Fu and Han Vinck** 1999)

    $C = \{ (R_1, R_2) \mid \exists p \in [0, 0.5], R_1 \leq h(p), R_2 \leq 1 - p \}$

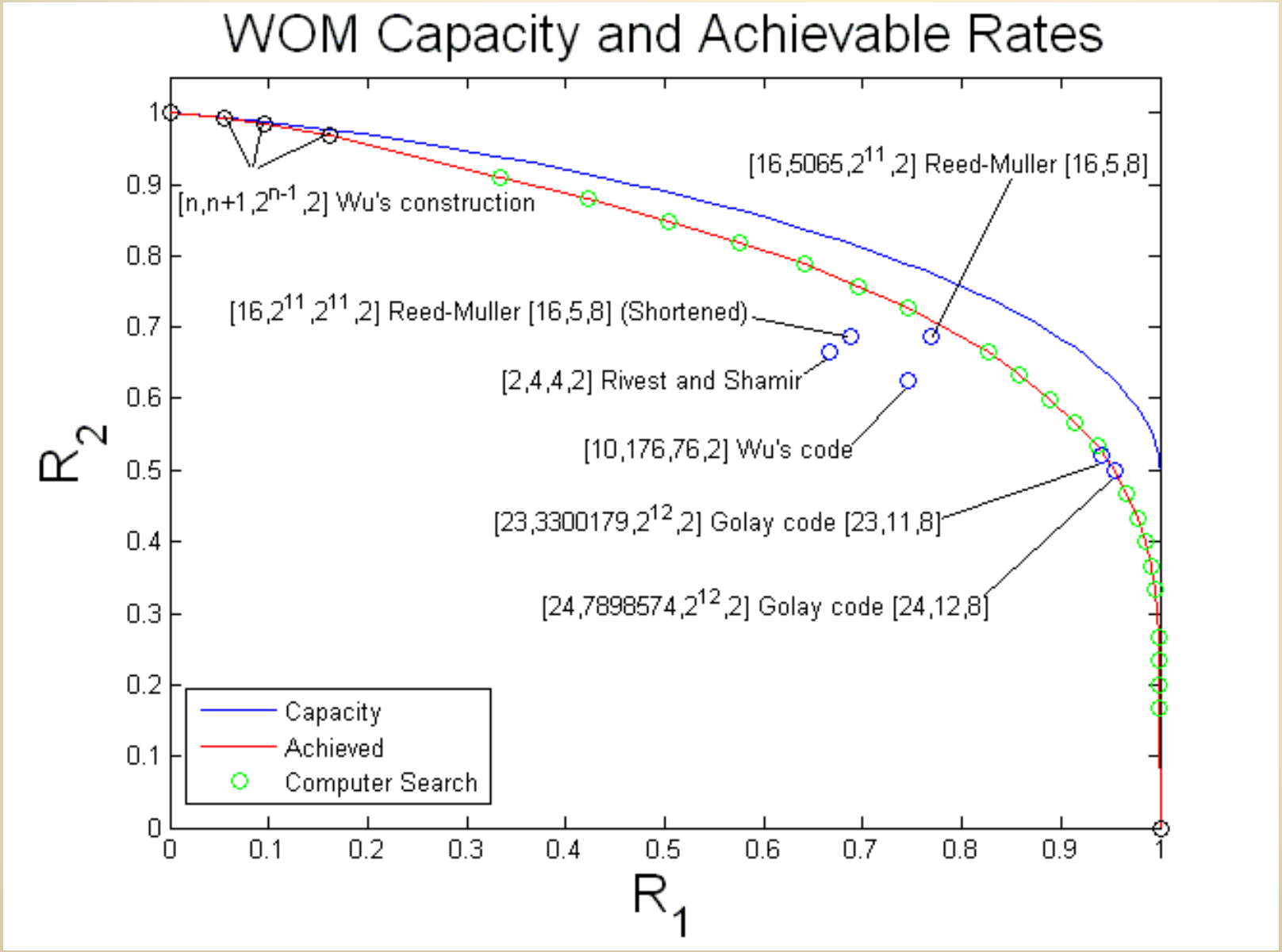
    The WOM-rate $R = R_1 + R_2 \leq \log_2(3) \approx 1.58$, achieved for $p = 1/3$
- **Rivest and Shamir** constructed WOM-codes of rates $(2/3, 2/3)$ and $(0.67, 0.67)$, $R = 1.34$
- Recently, **Wu** found a WOM-code of rate $(0.746, 0.625)$, $R = 1.371$

WOM-Codes with two writes

- Capacity region (**Heegard** 1986, **Fu and Han Vinck** 1999)

  $C = \{ (R_1, R_2) \mid \exists p \in [0, 0.5], R_1 \le h(p), R_2 \le 1 - p \}$

  The WOM-rate $R = R_1 + R_2 \le \log_2(3) \approx 1.58$, achieved for $p = 1/3$
- **Rivest and Shamir** constructed WOM-codes of rates (**2/3**, **2/3**) and (**0.67**, **0.67**)
- Recently, **Wu** found a WOM-code of rate (**0.746**, **0.625**), $R = 1.371$
- We construct WOM-codes from any linear code:
  - Given a linear code $C[n, k]$, if on the first write the vector of programmed cells **does not cover a codeword from the code**, then it is possible to write k bits on the second write
  - $V_C = \{ v \in \{0,1\}^n \mid v$ **does not cover a codeword from** $C \}$
  - $R_1 = \log|V_C|/n$, $R_2 = k/n$, $R = (\log|V_C| + k)/n$
  - The [**16**,**11**,**4**] extended Hamming code (**0.769**, **0.6875**), $R = 1.4566$
  - The [**23**,**12**,**7**] Golay code: (**0.9458**, **0.5217**) $R = 1.4632$
- We also show that by choosing randomly the generator matrix of the linear code C we can achieve the capacity


WOM Capacity and Achievable Rates
[n,n+1,2^{n-1},2] Wu's construction
[16,5065,2^{11},2] Reed-Muller [16,5,8]
[16,2^{11},2^{11},2] Reed-Muller [16,5,8] (Shortened)
[2,4,4,2] Rivest and Shamir
[10,176,76,2] Wu's code
[23,3300179,2^{12},2] Golay code [23,11,8]
[24,7898574,2^{12},2] Golay code [24,12,8]
Capacity
Achieved
Computer Search
R_2
R_1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Future Work

- Implementing WOM-codes in MLC Flash
- ECC for WOM-codes

Summary

- Error characterization in flash memories
- Write Once Memory (WOM) codes
  - Implementation of a simple WOM-codes scheme in SLC flash
  - WOM-codes with better rates are found