NAND Flash Basics & Error Characteristics

Why Do We Need Smart Controllers?

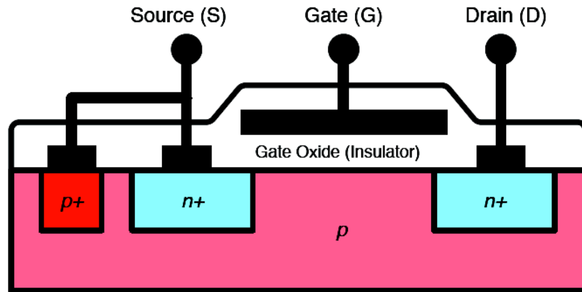
Thomas Parnell, Roman Pletka
IBM Research - Zurich

Agenda

- Part I. **NAND Flash Basics**
  - Device Architecture (2D + 3D)
  - SLC, MLC & TLC
  - Program/Read/Erase Procedure
- Part II. **Error Characteristics**
  - Program/erase cycling stress
  - Cell-to-cell Interference
  - Data Retention / Read Disturb
  - Programming Errors
  - 2D vs. 3D Reliability Comparison

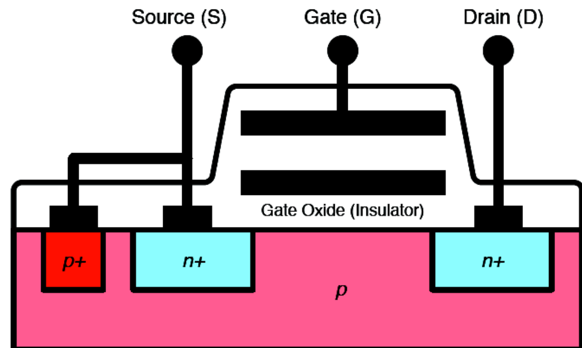
Part I: NAND Flash Basics

Flash Fundamentals



N-Channel MOSFET transistor

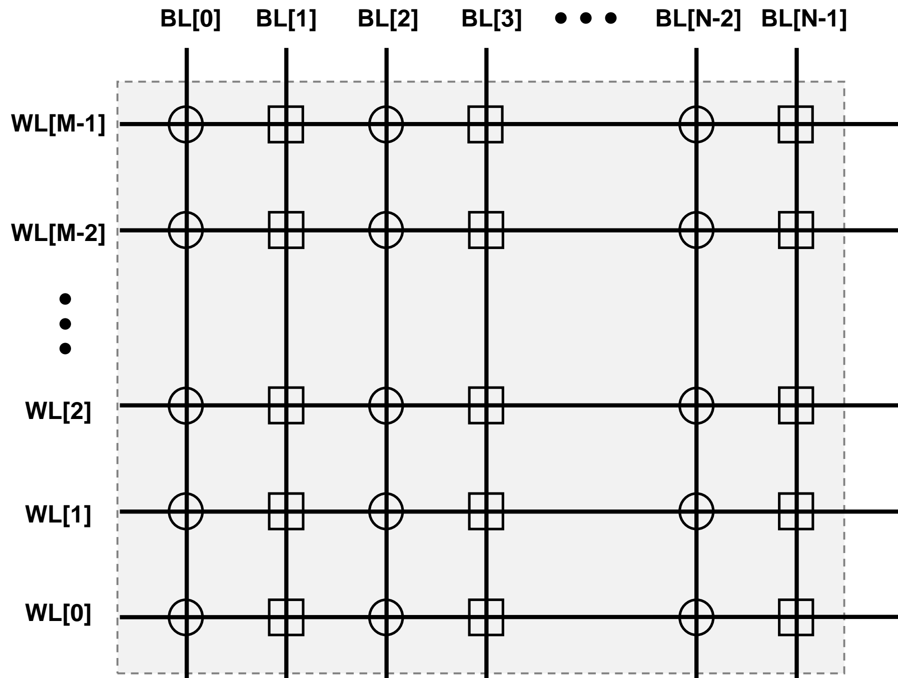
- Applying a gate-to-source voltage generates an electric field through insulator and creates a conduction channel through which current can pass from drain to source.



Floating Gate N-Channel MOSFET

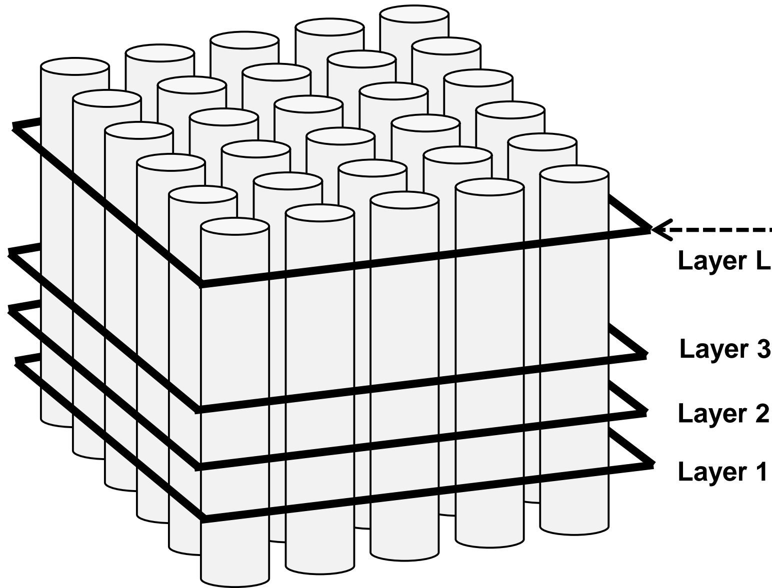
- The fundamental storage cell for Flash memory.
- Electrons can be stored onto and removed from the isolated floating gate (tunneling effect).
- Electrons residing on the floating gate remain there when power is removed
- The tunneling effect is destructive (e- get stuck in the insulator), hence limiting the number of program erase cycles.
- Electrons may “fall off” the floating gate over time, especially with increased temperature.

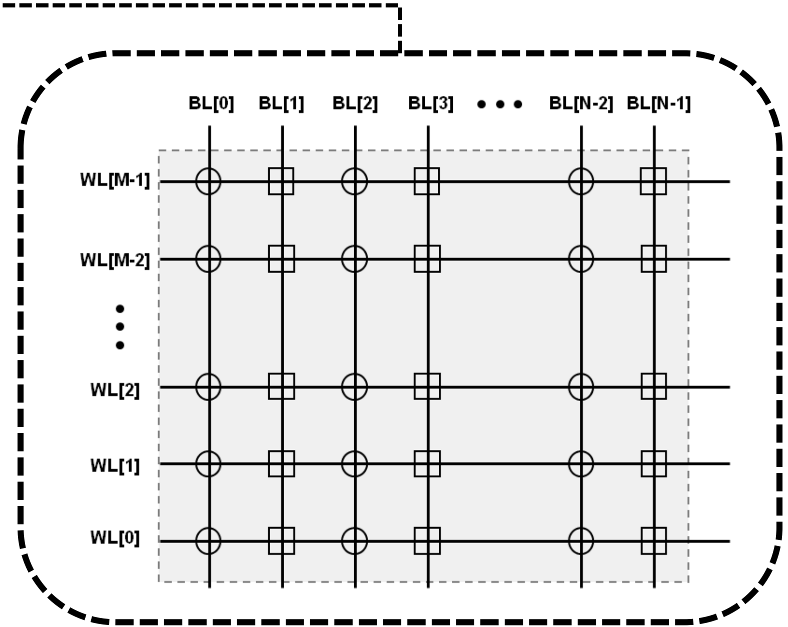
NAND Flash Architecture (2D)



- A block of planar NAND Flash consists of a grid of cells connected by word lines (WLs) and bit lines (BLs)
- Data is programmed/read from the device page-by-page (~16KB)
- Every WL in the block contains:
  - 1 page (SLC)
  - 2 pages (MLC)
  - 3 pages (TLC)
- Within a WL, pages can be further interleaved so that each WL contains 2/4/6 pages ("*Even-Odd BL Architecture*")

NAND Flash Architecture (3D)





A block consists of vertically-stacked layers of NAND Flash cells

Each layer consists of a grid of cells connected by WLs and BLS

Flash Memory Organization

NAND Flash cell

Source (S)
Gate (G)
Drain (D)
Gate Oxide (Insulator)
p+
n+
p
n+

Erase operation

Block
512 – 1.5k
Pages

Multi-plane
operations

Die
2 – 4
Planes

3D Architecture

Source:
Applied
Materials
2015

32 – 96 Layers

Page
16k+ Bytes

Plane
512 – 1024
Blocks

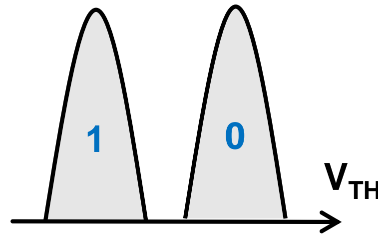
Target
1/2/4/8 dies

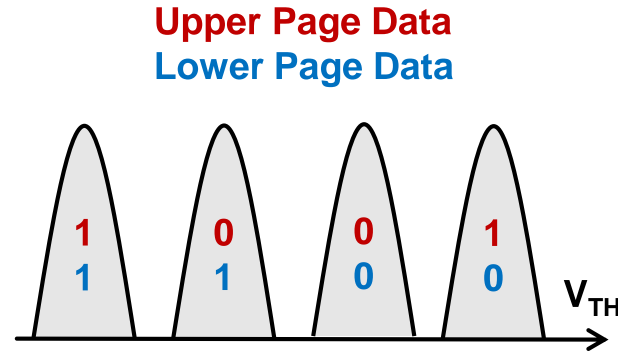
Read/write
operations

Data & cache
registers

Separate
I/O Interfaces

SLC vs. MLC

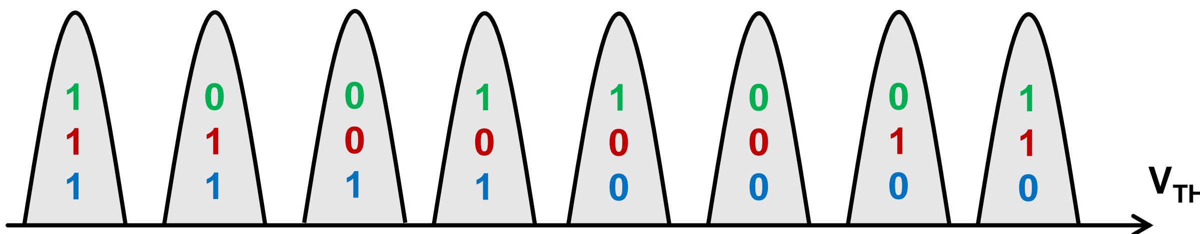




Single **L**evel **C**ell (**SLC**)
2 States (1 Erase + 1 Pgm)
= 1 bit of information per cell

Multi **L**evel **C**ell (**MLC**)
4 States (1 Erase + 3 Pgm)
= 2 bits of information per cell
= 2x capacity of SLC!

TLC



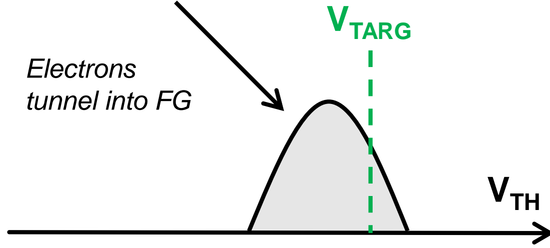
Triple **L**evel **C**ell (**TLC**)
8 States (1 Erase + 7 Pgm)
= 3 bits of information per cell
= **1.5x capacity of MLC = 3.0x capacity of SLC**

Incremental Programming

IBM

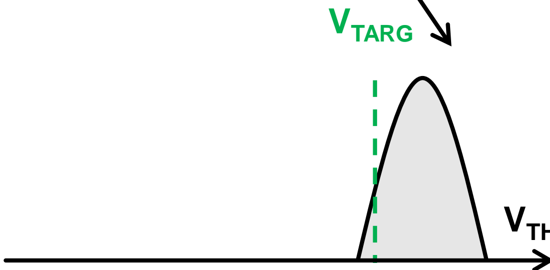
(a) Erased State

(b) First programming pulse

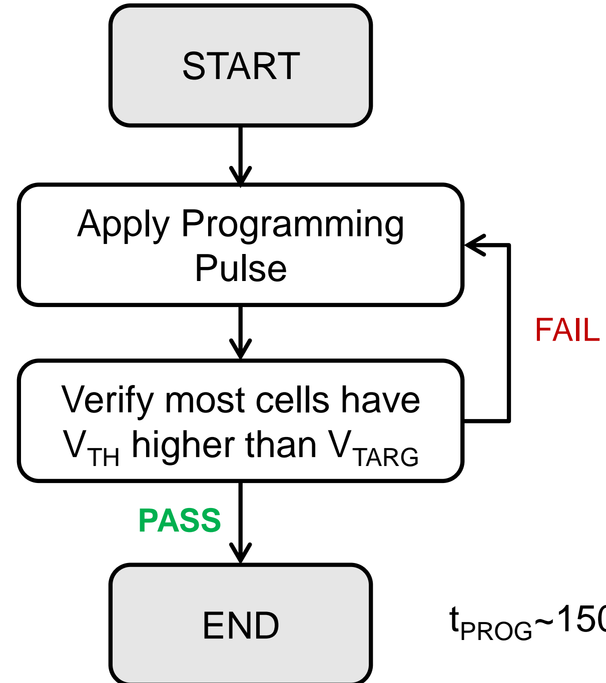


⋮

(c) N programming pulses



ISPP Procedure

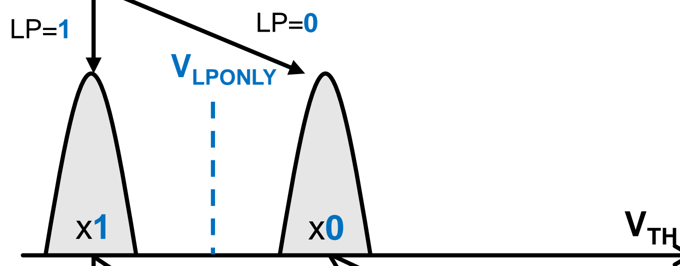


t_{PROG}~1500us

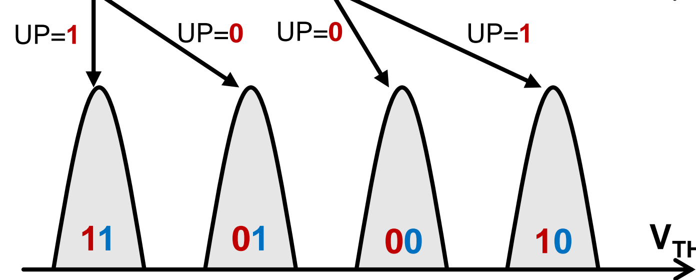
MLC Two-Pass Programming

(a) Erased State

(b) Program **Lower Page**



(c) Program **Upper Page**

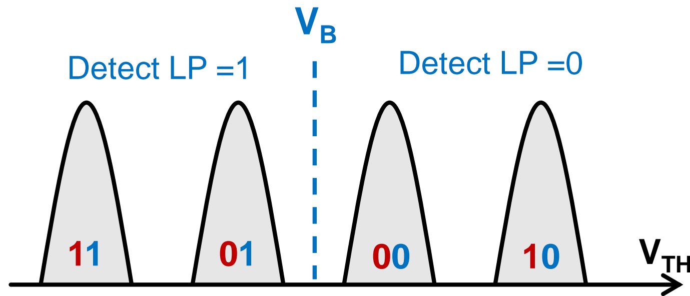


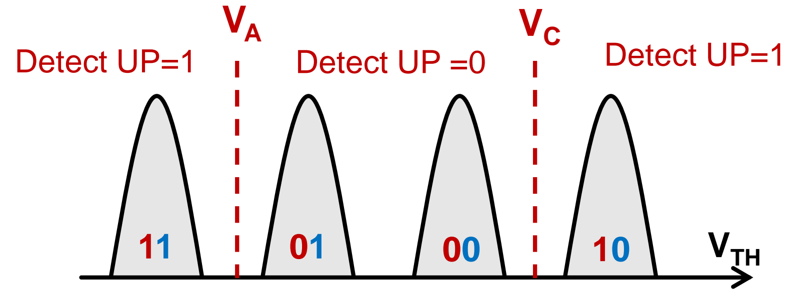
Data is programmed to the device one page at a time

The cells are either left in the erased state of programmed to an intermediate state depending on the lower page data.

An intermediate read determines the previously programmed lower page data and the cell distribution for the WL is "finalized" using the upper page data

Reading Data Back (MLC)



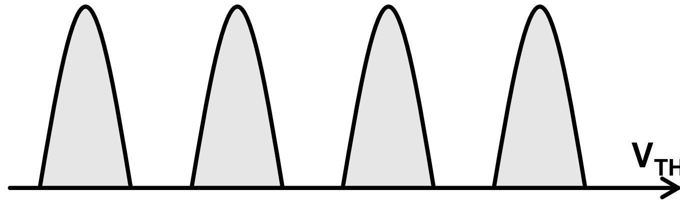


- Lower page can be read using a single read voltage (V_B)
- Upper page can be read using a pair of read voltages (V_A,V_C)
- A page read typically takes up to 100us

Erasing

Data is erased one block at a time.
An individual page cannot be erased.

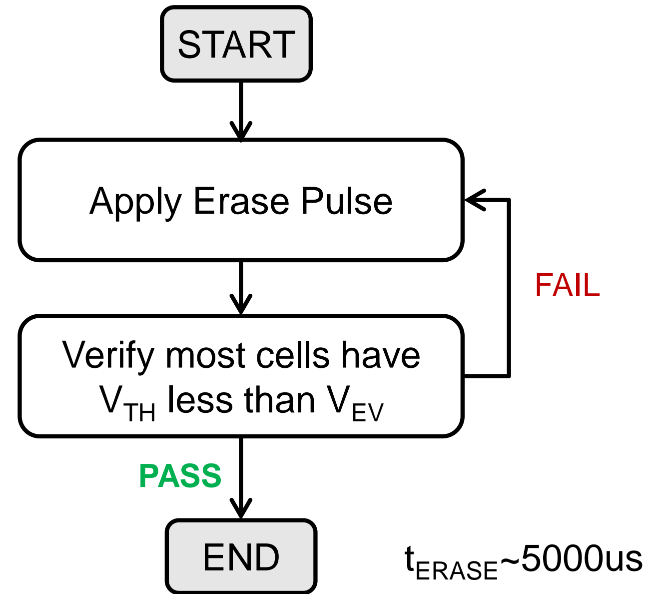
(a) Fully prog. State



(b) First erase pulse

⋮

(c) N erase pulses

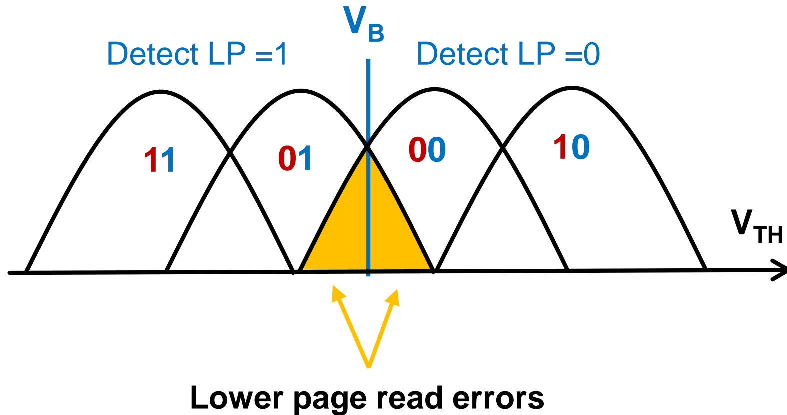


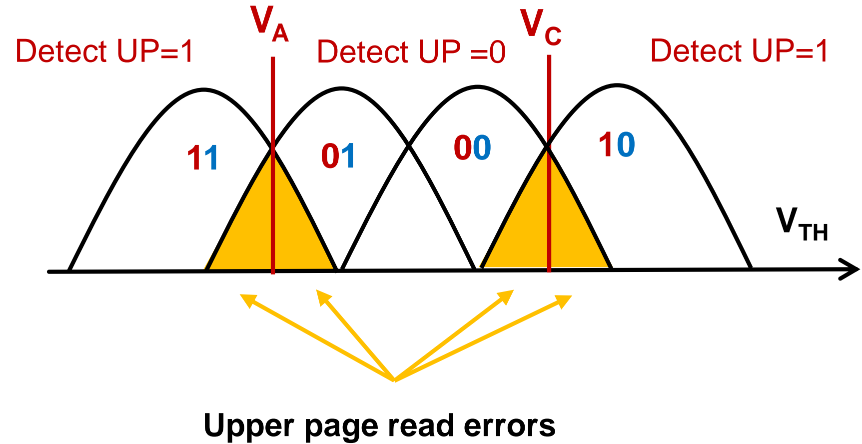
t_{ERASE}~5000us

Part II: Error Characteristics

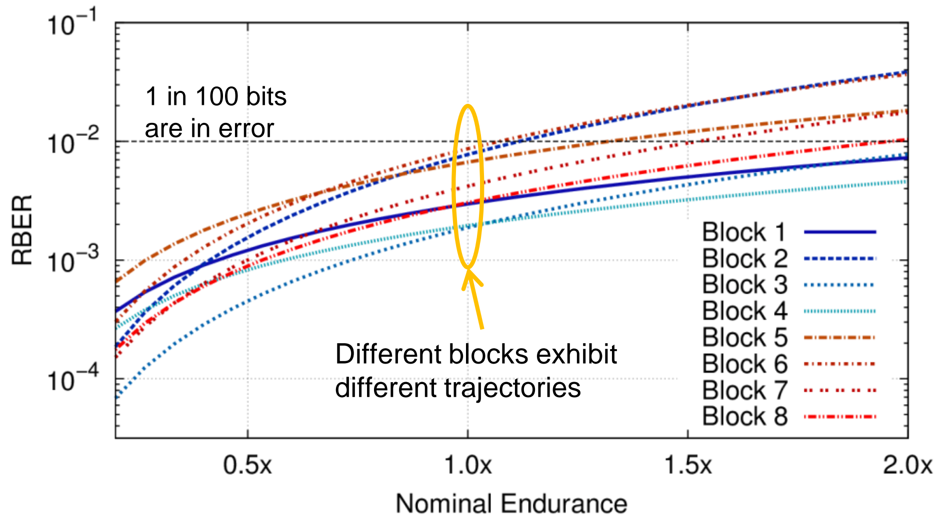
Read Errors

- Broadening of V_{TH} distributions due to noise can lead to read errors
- What are the main sources of noise?





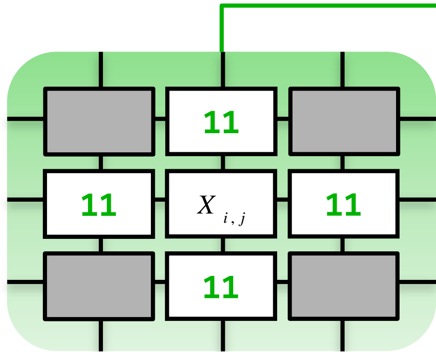
Program/Erase Cycling Stress

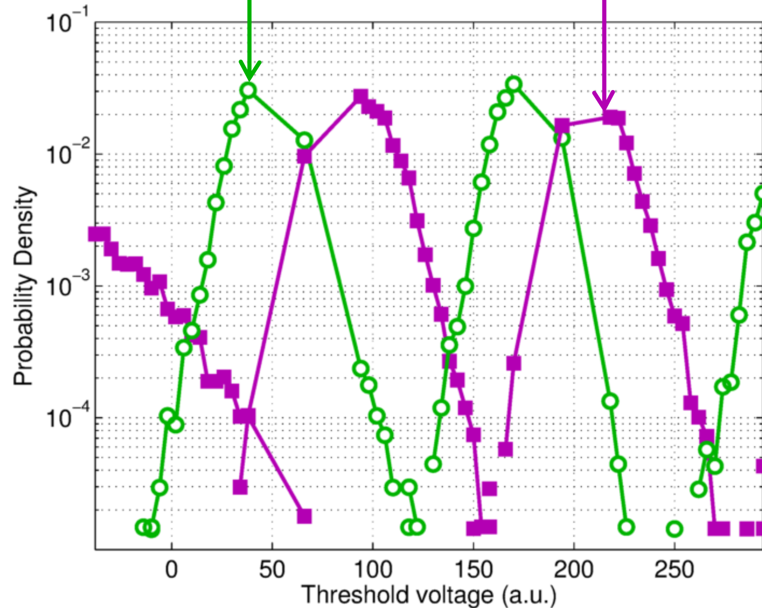


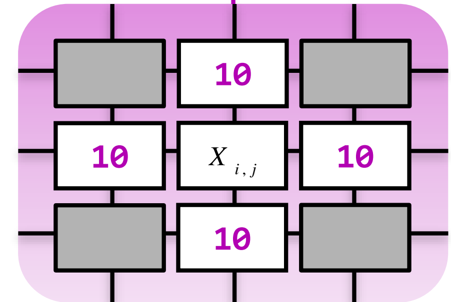
- Repeated application of program/erase (P/E) pulses leads to degraded reliability of the underlying NAND flash cells
- The measured raw bit error rate (RBER) increases as a function of P/E cycles
- Low RBER at early life does not indicate a good block, and an early high RBER not a weak one!
- Strong error-correction codes must be implemented on the controller to be able to deal with increased RBER

Cell-to-Cell Interference

- Threshold voltage of "victim" cell is strongly affected by programming of neighboring "aggressor" cells → **can the controller compensate?**

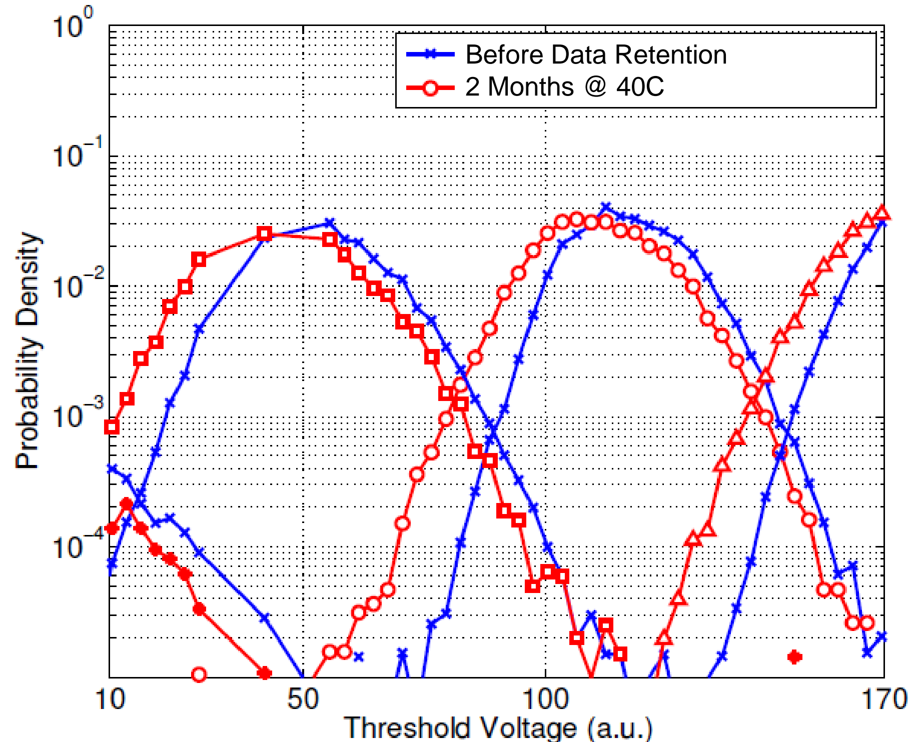




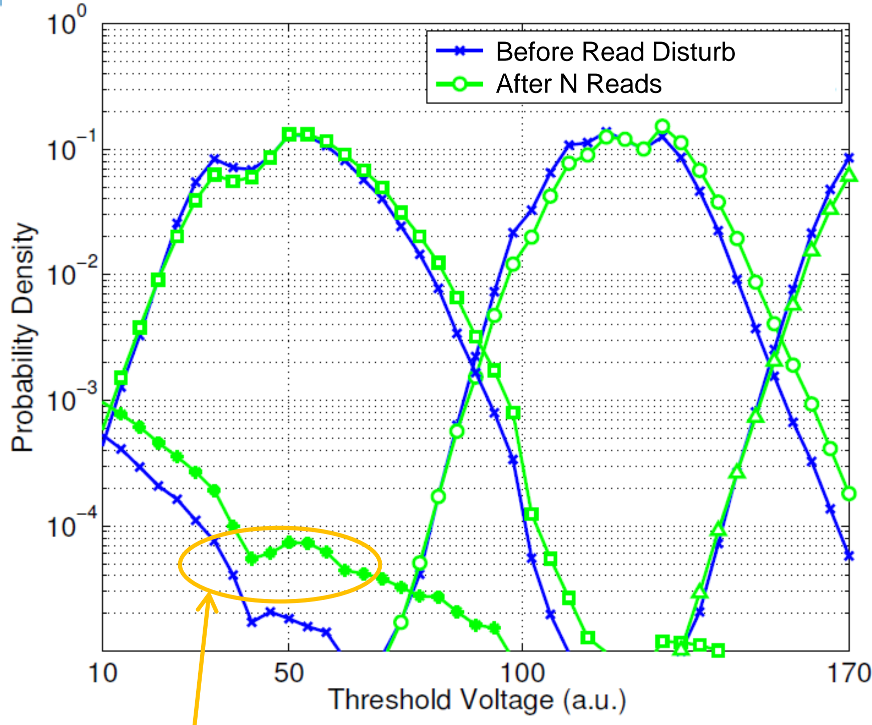


Data Retention

- Over time electrons can escape from the programmed flash cells, causing a loss of threshold voltage
- This can cause a large increase in RBER unless the controller can shift the read voltage to compensate for charge loss
- The data retention effect is temperature dependent (charge escapes faster at higher temperature)



Read Disturb

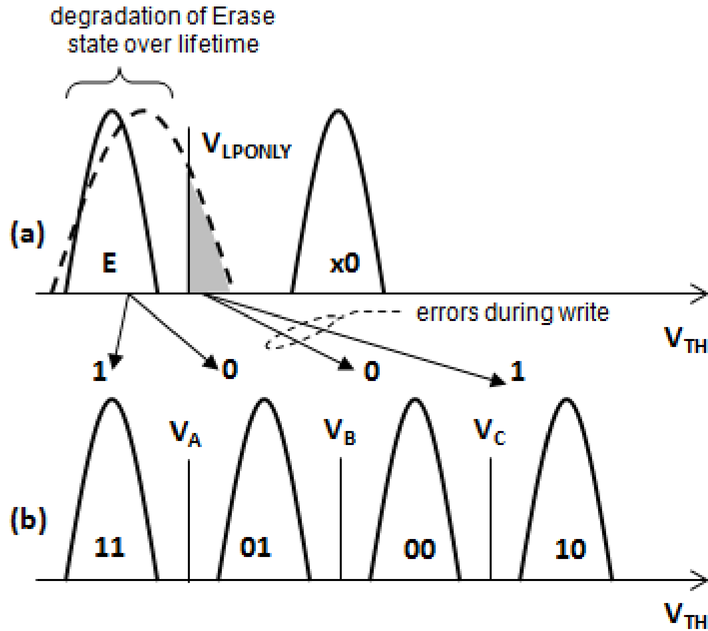


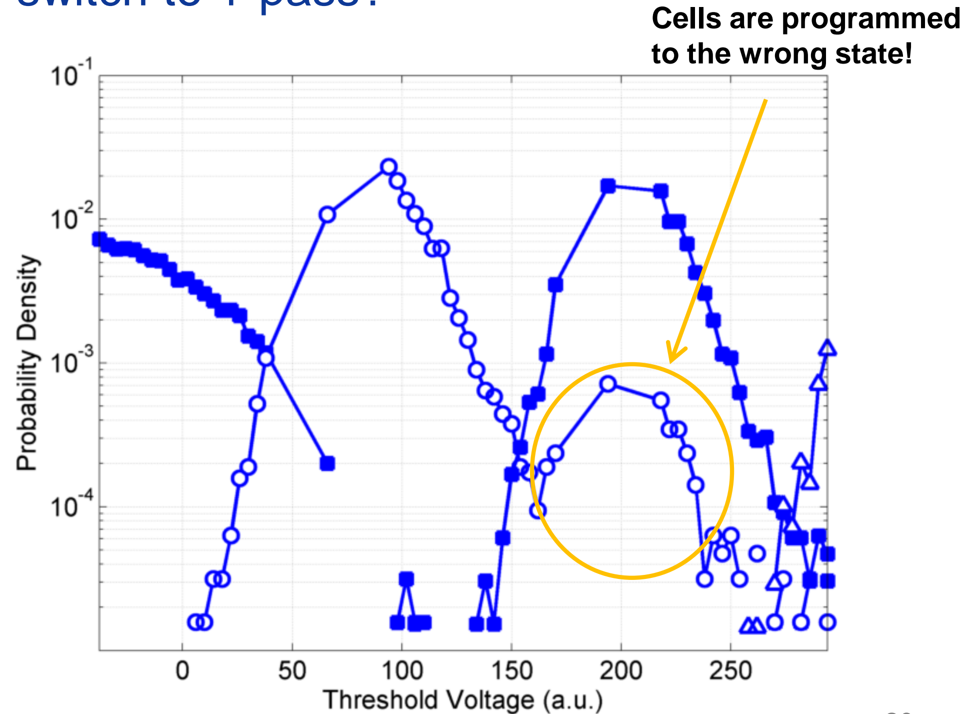
Dominant effect of read disturb is seen on Erase state

- When reading a particular page in a block of NAND Flash, a voltage is applied to all other WL in order to “deselect” them
- This applied voltage can affect the V_{TH} distributed of the unselected WLs
- If a block is read from too many times, the RBER will increase to a point that the ECC is no longer able to correct
- The controller must be able to manage such effects

Programming Errors

- Degradation of erase state can cause error propagation during the two-pass programming procedure → switch to 1-pass?





2D vs. 3D Reliability Scorecard

Reliability Issue	2D	3D	Comment
Program/Erase Cycling	TLC endurance: ~100 cycles	TLC endurance: >1000 cycles	Increased cell dimensions enable new applications for TLC Flash
Cell-to-cell Interference	X/Y-direction	Z-direction	**Controller management required**
Data Retention	Years (consumer) Months (enterprise)	Fast Initial Charge Loss	**Controller management required**
Read Disturb	Affects both		**Controller management required**
Programming Errors	2-pass programming	1-pass programming	Improved algorithm can remove programming errors entirely

Conclusions

- NAND Flash is currently unrivalled technology in terms of the performance/cost trade-off
- However, it is inherently unreliable and cannot be used without a controller providing additional functionality
- What do we require of a controller?
  - Media management / signal processing
  - Powerful error-correction
  - Data placement and garbage collection algorithms
  - Wear-leveling algorithms
  - Efficient FPGA/ASIC/firmware implementations