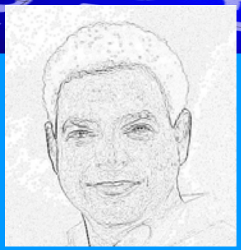
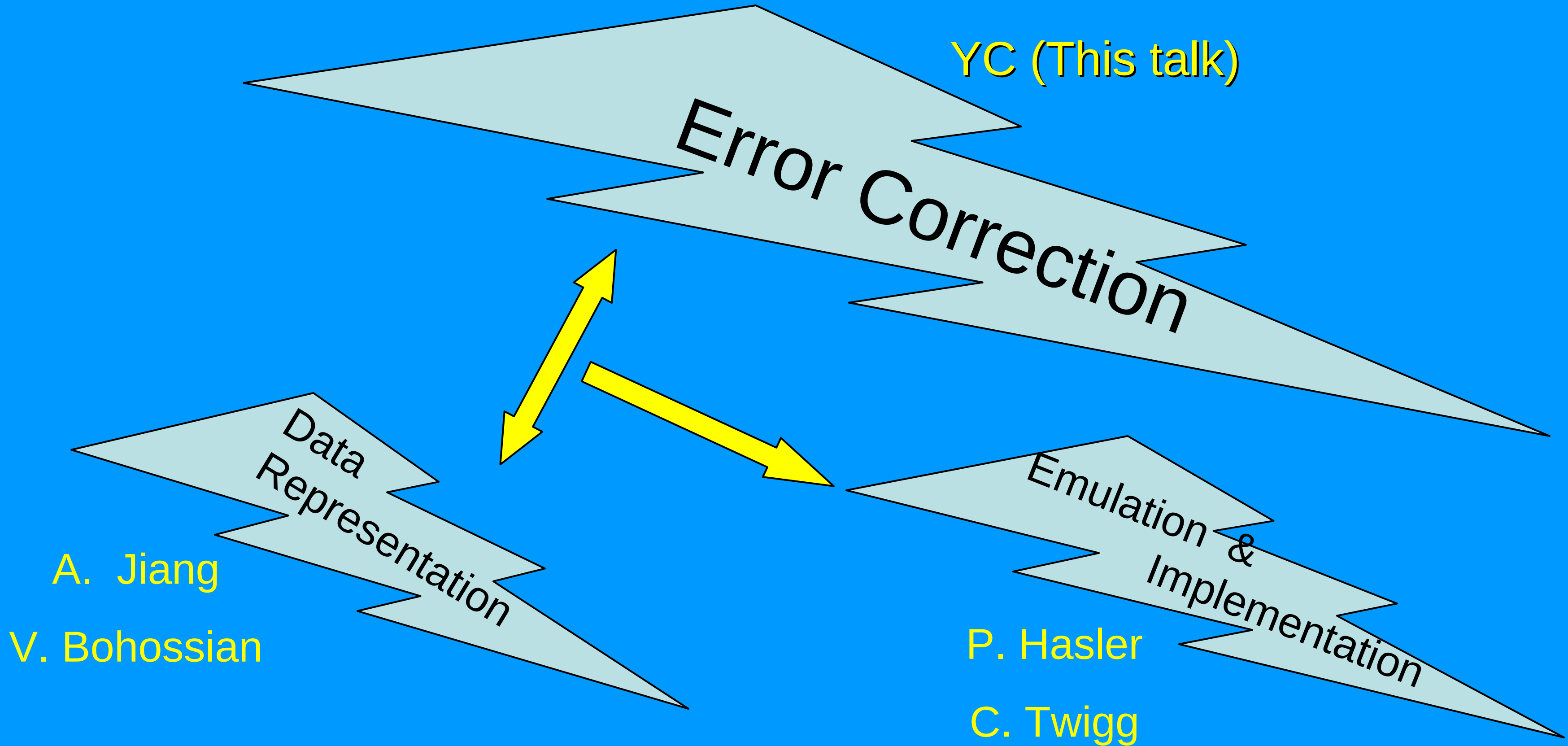
# Coding for Denser and Faster Flash Memories

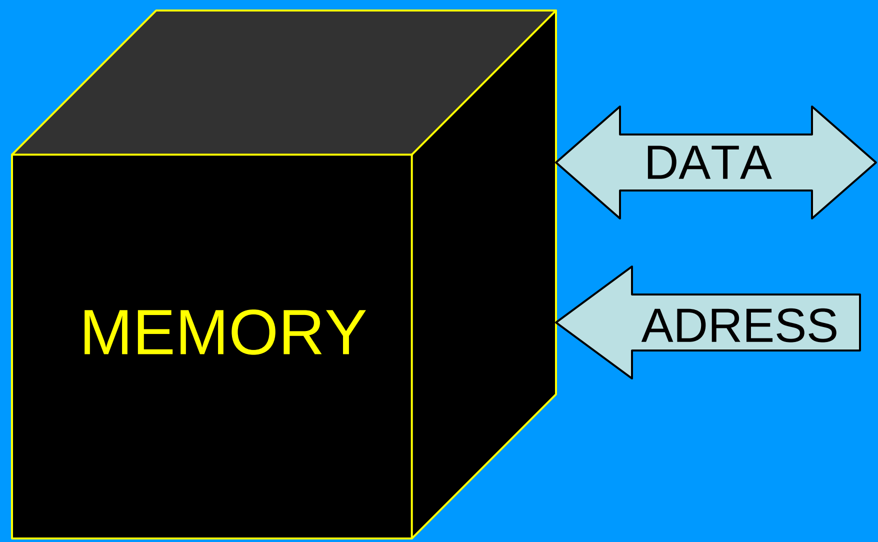
Yuval Cassuto
Paradise Laboratory
EE Department
Caltech

# The Caltech Flash Project





# Memory Devices




**Functions**

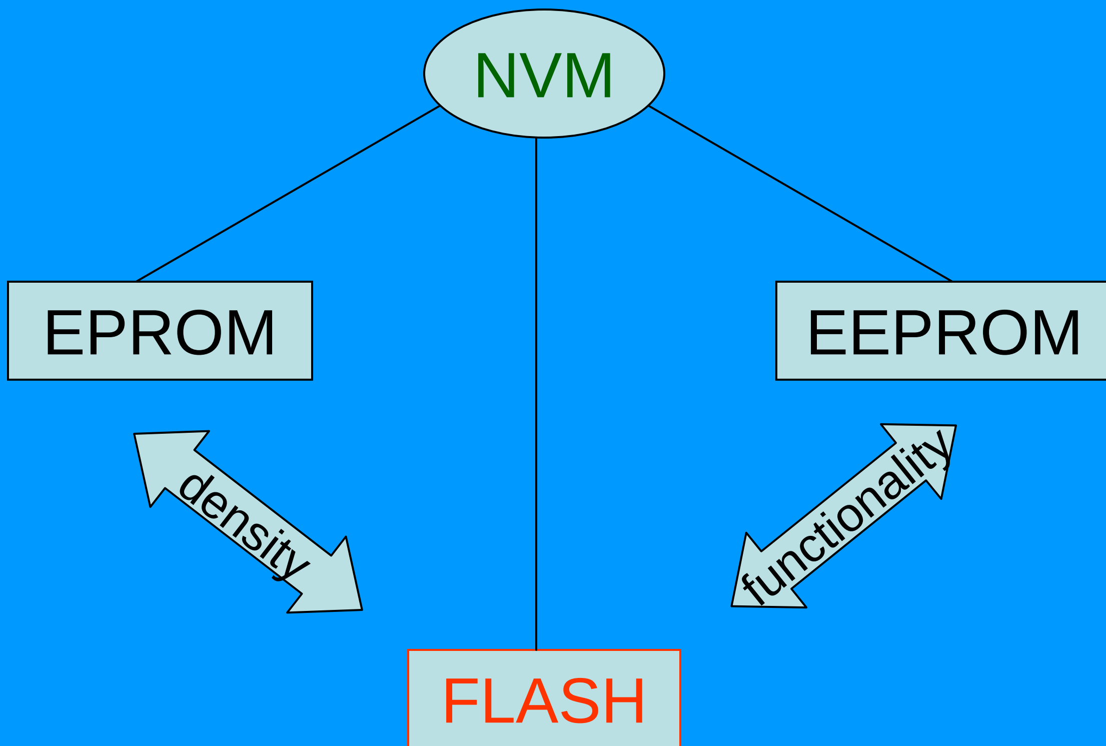
- Write DATA to ADRESS
- Read DATA from ADRESS

**Characterizations**

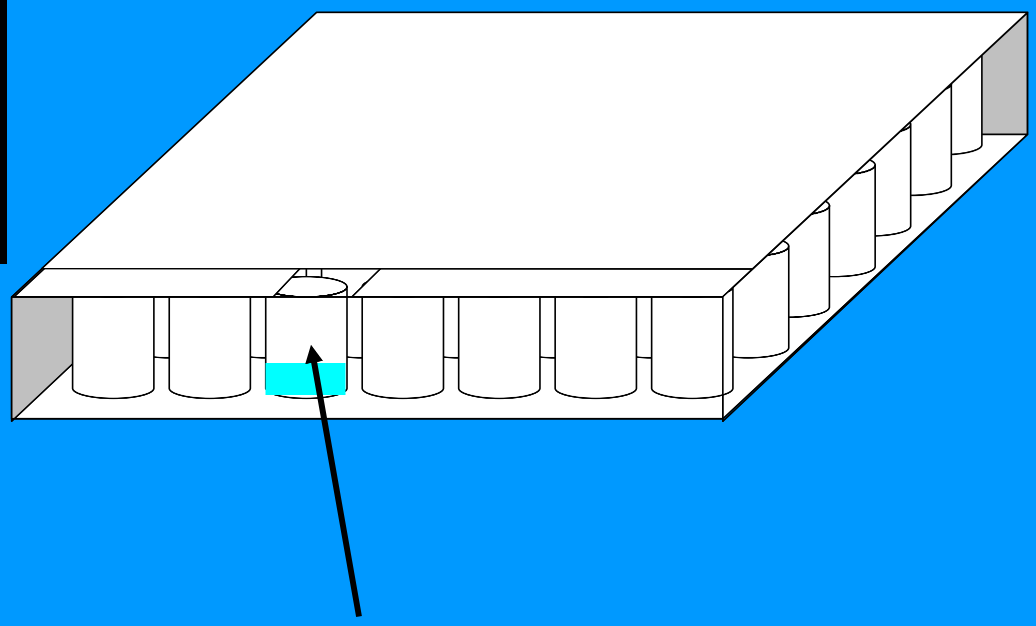
- Density (Capacity/Volume)
- Speed
- Reliability
- Functionality

# Flash Memory

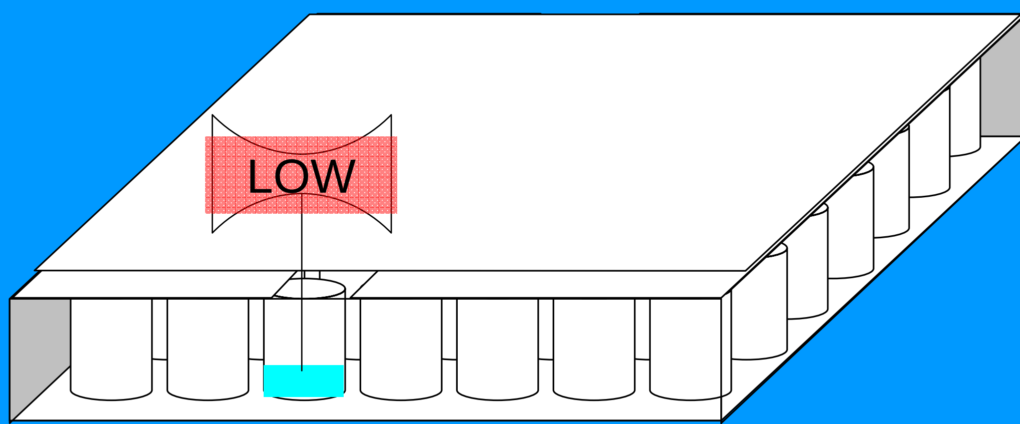
Flash is a Non-Volatile Memory Technology (1979)



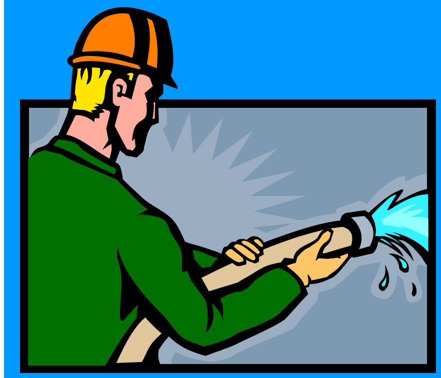
# Flash Program Principles

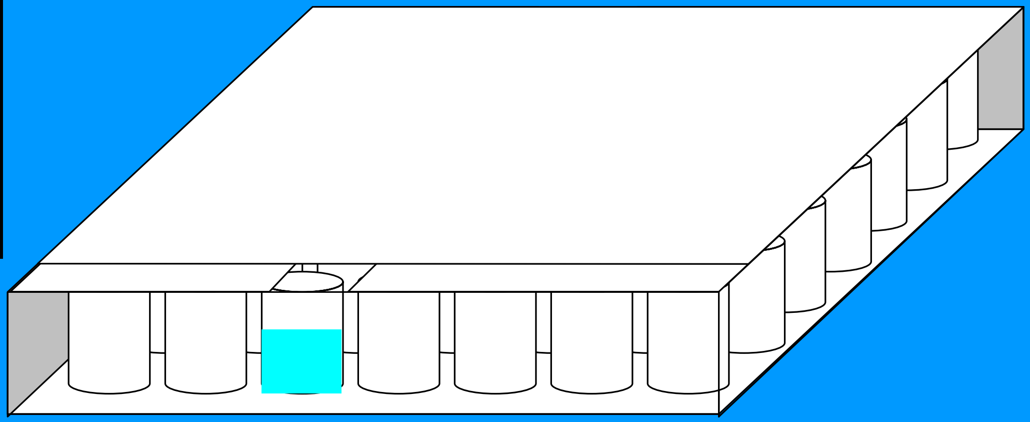


# Flash Iterative Programming

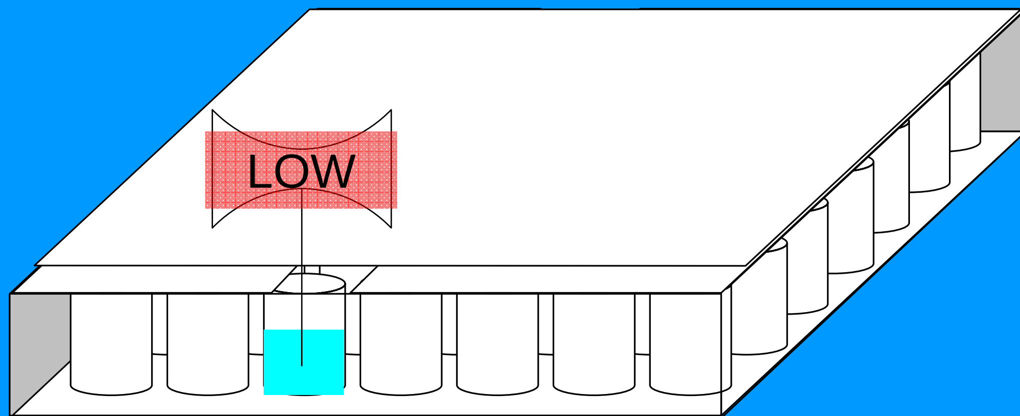


# Flash Iterative Programming

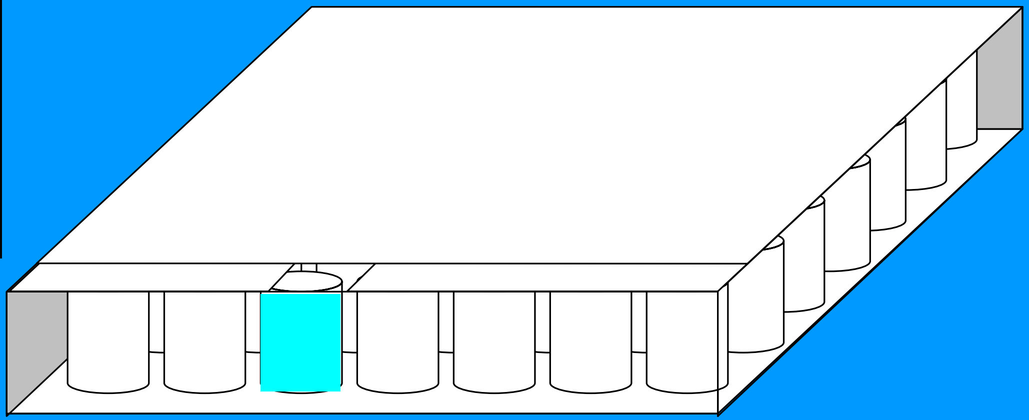




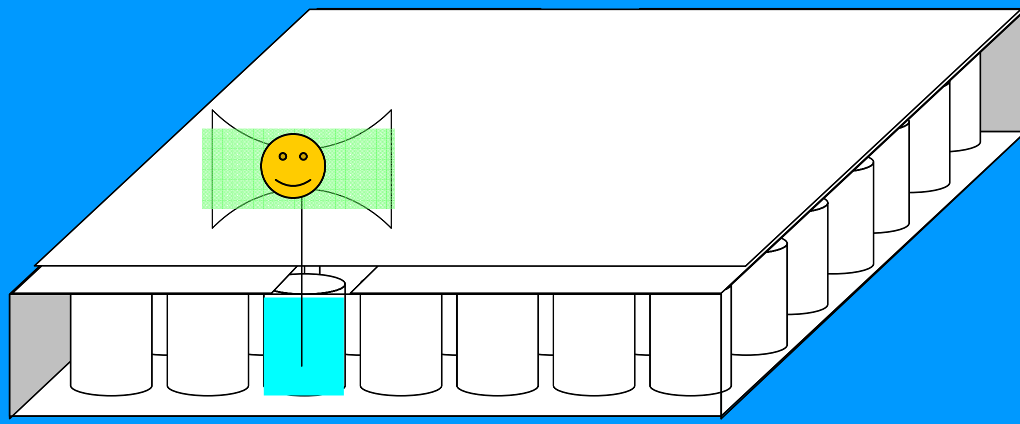
# Flash Iterative Programming



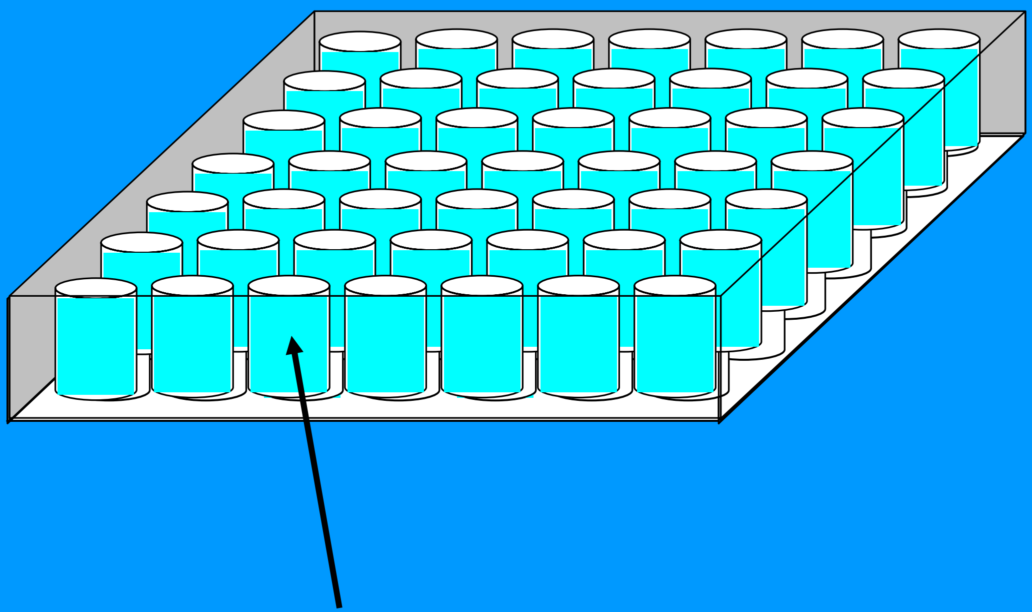
# Flash Iterative Programming



# Flash Iterative Programming



# Flash Block Erase



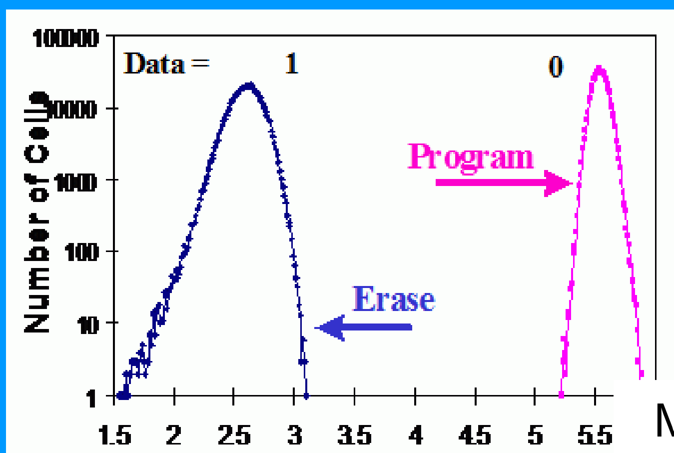
# Flash Essentials

Method : Charge placement on Floating Gates

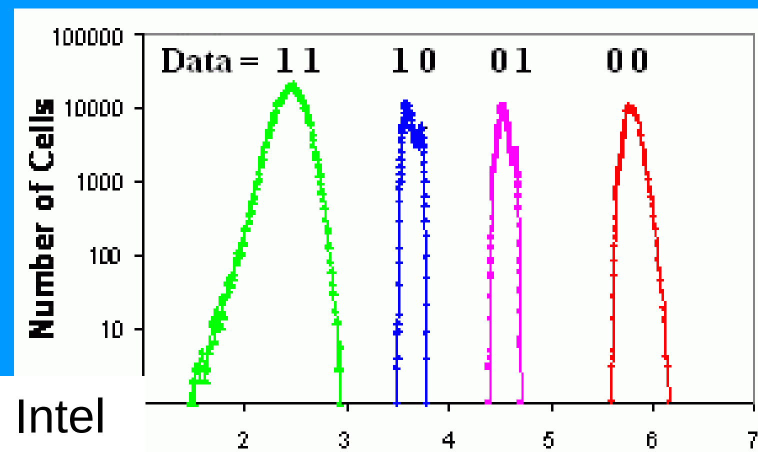
- Iterative programming
- Reliability issues
  - Disturbs
  - Retention
  - Endurance
- Block erase

# Multi-Level Flash Cell

## 2-Level cell




## Multi-Level cell




M. Bower, Intel

- Iterative programming
- Reliability issues
  - Disturbs
  - Retention
  - Endurance


- Iterative programming
- Reliability issues
  - Disturbs
  - Retention
  - Endurance

# Flash and ECC

intel Leap ahead™ Flash File Systems Overview

2006

## 4.2 Error Code Correction (ECC)

Error Code Correction (ECC) is a way to identify and correct errors during read or write operations to flash. ECC is very common in NAND flash due to NAND reliability issues. Most ECC for flash can detect and correct single bit errors. However as NAND, and some NOR, devices trend toward Multi-Level (MLC) architecture and smaller lithographies, there is a need to perform multiple bit correction as error rates increase.

ECC is generally performed within a memory controller although software ECC is also possible. Several ECC algorithms are available, including Hamming Code, BCH (Bose, Chaudhuri, Hocquenghem), and Reed-Solomon - three that are among the most popular. ECC algorithms vary in complexity and their impact on design cost.

# An Information Theoretic View of MLC Flash

q-level cells

Alphab q-ary 1,…,q-1}

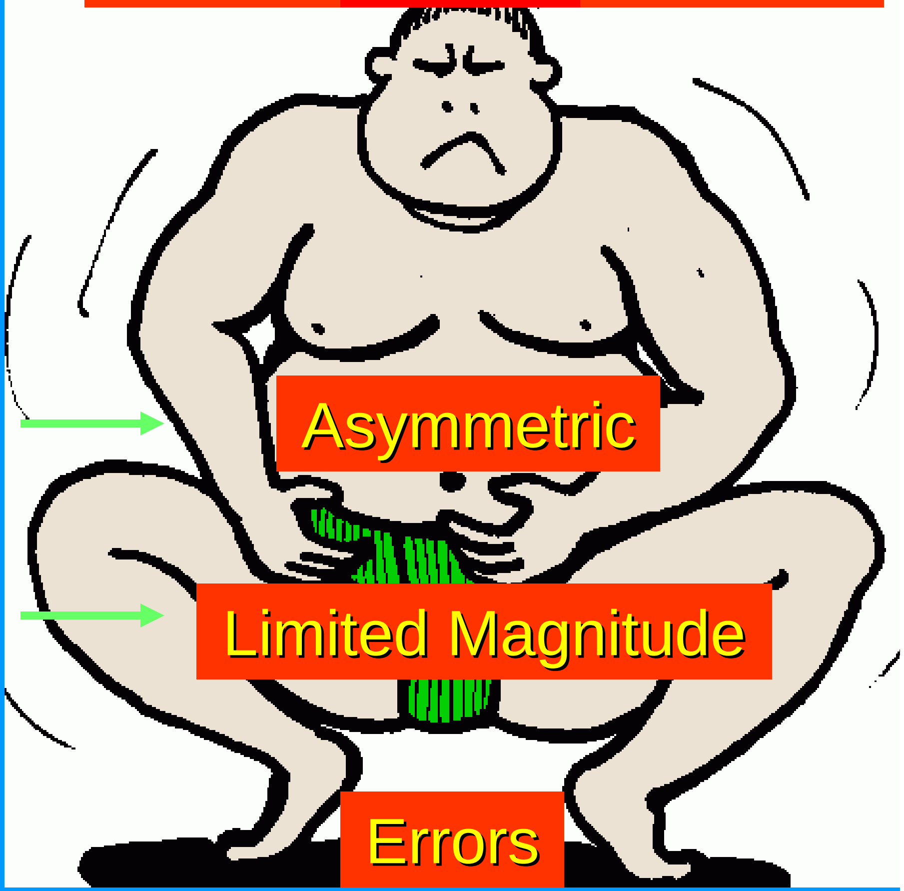
– Disturbs
– Retention
– Endurance

Move cells in a

Dominant direction

To

Proximate levels



# Asymmetric l Limited-Magnitude Errors

$$Q = \{0, 1, \ldots, q-1\}$$


Asymmetric l-limited-magnitude error: $0 \le y - x \le \ell$

Asymmetric l-limited-magnitude error with wrap around: $(y - x) \in \{0, 1, \ldots, \ell\} \bmod q$

# Codes for AlM Errors

Research Problem:

Code $C \subset Q^n$ that corrects $t$ $A\ell M$ errors

For each $(n, \ell, t)$

Known Result: [Ahlswede et.al '06]

Optimal $(n, \ell, n)$ codes

Trivial construction : code has all the words of $L^n$ , where $L \subset Q$

# Analyzing Codes for AlM Errors

AlM "distance" for code analysis:

For a pair of vectors $\boldsymbol{x} \in Q^n, \ \boldsymbol{z} \in Q^n$

$$d_\ell(\boldsymbol{x}, \boldsymbol{z}) = \begin{cases} n+1 & \text{if } \max_i(|x_i - z_i|) > \ell \\ \max(N(\boldsymbol{x}, \boldsymbol{z}), N(\boldsymbol{z}, \boldsymbol{x})) & \text{otherwise} \end{cases}$$

$$N(\boldsymbol{x}, \boldsymbol{z}) = |\{i : x_i > z_i\}| \qquad N(\boldsymbol{z}, \boldsymbol{x}) = |\{i : z_i > x_i\}|$$

*Theorem*

$C \subset Q^n$ corrects $t$ A$\ell$M errors $\Leftrightarrow$

$d_\ell(\boldsymbol{x}, \boldsymbol{z}) \geq t+1$ for all distinct $\boldsymbol{x}, \boldsymbol{z}$ in $C$.

# Example: Correcting A1M Errors

| | | | | |
|---|---|---|---|---|
| 8 | 2 | 4 | 2 | 0 |

A1M A1M

| | | | | |
|---|---|---|---|---|
| 8 | 3 | 4 | 2 | 1 |

| | | | | |
|---|---|---|---|---|
| 7 | 3 | 3 | 1 | 1 |

A1M A1M

| | | | | |
|---|---|---|---|---|
| 8 | 3 | 4 | 1 | 1 |

$$n = 5, \; t = 2, \; \ell = 1$$

Majority Even/Odd

| | | | | |
|---|---|---|---|---|
| 8 | ③ | 4 | 2 | ① |

| | | | | |
|---|---|---|---|---|
| ⑧ | 3 | ④ | 1 | 1 |

-1 for all minority symbols

| | | | | |
|---|---|---|---|---|
| 8 | **2** | 4 | 2 | **0** |

| | | | | |
|---|---|---|---|---|
| **7** | 3 | **3** | 1 | 1 |

# Construction 1

Given $\Sigma \subset Q'^n$ $\quad Q' = \{0, \ldots, q'-1\},\ \ell < q' < q$

$$C = \{\boldsymbol{x} \in Q^n : \boldsymbol{x} \bmod q' \in \Sigma\}$$

*Theorem*

$\Sigma$ corrects $t$ A$\ell$M errors with wrap-around

$\Longrightarrow$ $\quad C$ corrects $t$ A$\ell$M errors

$\Longleftarrow$

$q \geq q' + \ell$

# Construction 1': $q' = \ell + 1$

Given $\Sigma \subset Q'^n$ $\quad Q' = \{0, \ldots, \ell\}, \ \ell < q$

$$C = \{\boldsymbol{x} \in Q^n : \boldsymbol{x} \bmod (\ell + 1) \in \Sigma\}$$

*Theorem*

$\Sigma$ corrects $t$ symmetric errors

$\Longrightarrow$ $\quad C$ corrects $t$ A$\ell$M errors

$\Longleftarrow$
$q > 2\ell$

# Construction 1: Proof

If $\boldsymbol{x} \in C,\ \boldsymbol{z} \in C$ come from the same codeword of $\Sigma$

$$d_\ell(\boldsymbol{x},\boldsymbol{z}) = \begin{cases} n+1 & \text{if } \max_i(|x_i - z_i|) > \ell \\ \max(N(\boldsymbol{x},\boldsymbol{z}), N(\boldsymbol{z},\boldsymbol{x})) & \text{otherwise} \end{cases}$$

If $\boldsymbol{x} \in C,\ \boldsymbol{z} \in C$ come from different codewords of $\Sigma$

$$d_\ell(\boldsymbol{x},\boldsymbol{z}) = \begin{cases} n+1 & \text{if } \max_i(|x_i - z_i|) > \ell \\ \max(N(\boldsymbol{x},\boldsymbol{z}), N(\boldsymbol{z},\boldsymbol{x})) & \text{otherwise} \end{cases}$$

# Construction 1: Perfect Codes

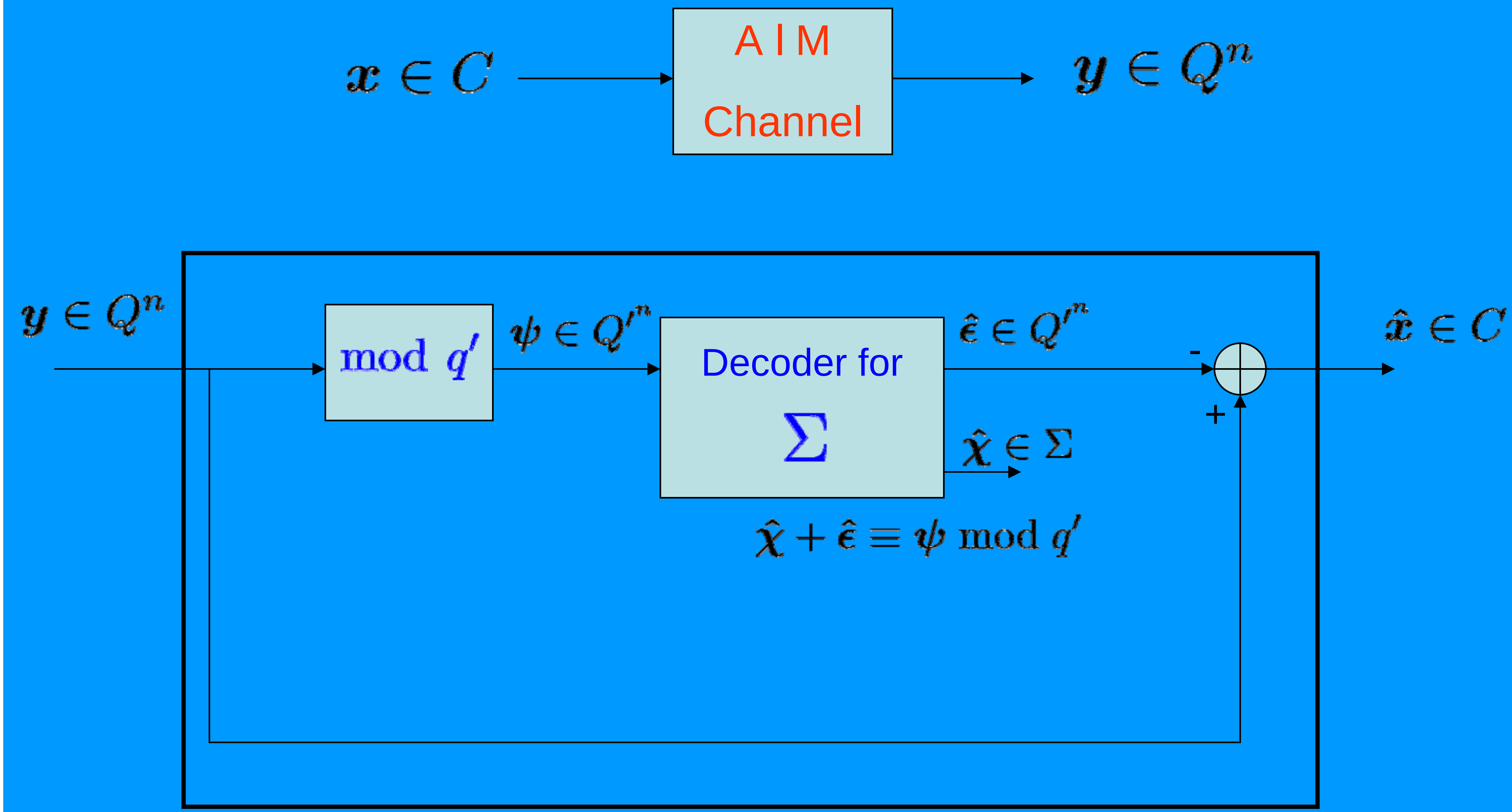
If $\Sigma$ is a "perfect" code for symmetric errors, then

$C$ is a "perfect" code for A$\ell$M errors

AlM Sphere packing bound is met with =

$$|C| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i \leq q^n$$

# Construction 1: Decoding

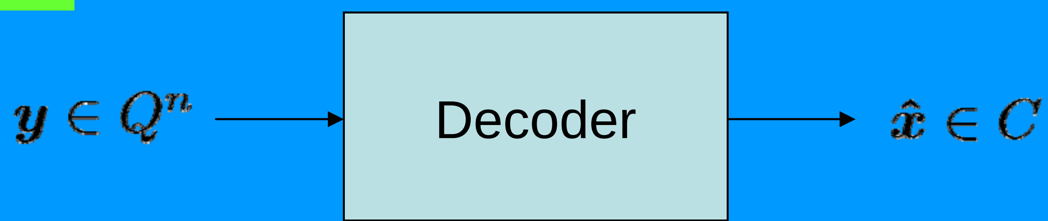


# Something is Missing

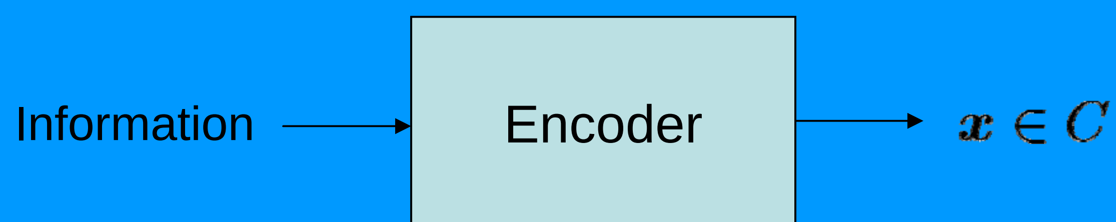
We know

$$C \subset Q^n$$

We have




But still need



# Encoding Example

18 information bits → $q = 8$ Flash cells with $t = 1, \ell = 1$

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | | | |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | $\in \Sigma_H$ |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 | |

lsb

msb

$\rightarrow x_i \in Q = \{0, 1, \ldots, 7\}$

$x =$ | 4 | 5 | 3 | 6 | 2 | 5 | 7 | $\in C$

# Symbol Mappings

| | | | | | | |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 | 1 | 1 |

$\in \Sigma_H$

lsb

msb

$x_i \in Q = \{0, 1, \ldots, 7\}$

| | | |
|---|---|---|
| 0 | 000 | 0 |
| 1 | 001 | 1 |
| 2 | 010 | 2 |
| 3 | 011 | 3 |
| 6 | 100 | 4 |
| 7 | 101 | 5 |
| 4 | 110 | 6 |
| 5 | 111 | 7 |

$b_0 + 2 \cdot Gray(b_1, b_2)$

$\sum_{j=0}^{2} b_j 2^j$

$x =$

| | | | | | | |
|---|---|---|---|---|---|---|
| 6 | 7 | 3 | 4 | 2 | 7 | 5 |

$\in C$

$x =$

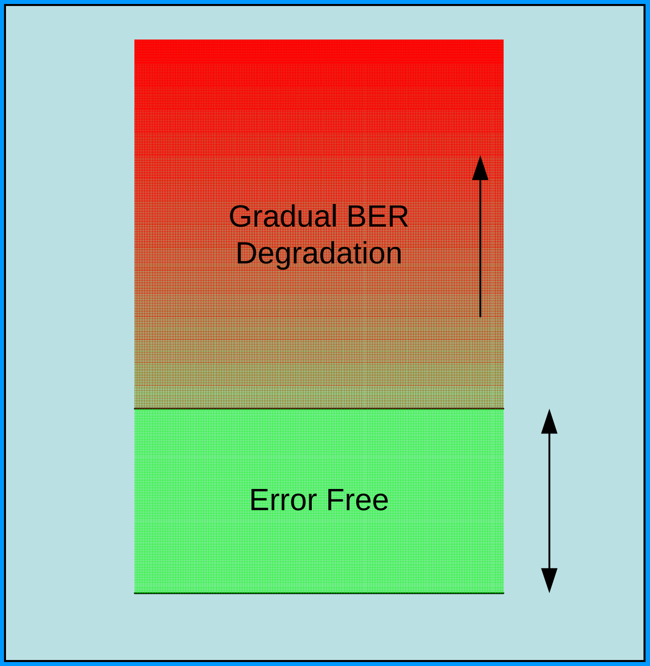
| | | | | | | |
|---|---|---|---|---|---|---|
| 4 | 5 | 3 | 6 | 2 | 5 | 7 |

$\in C$

# Symbol Mappings

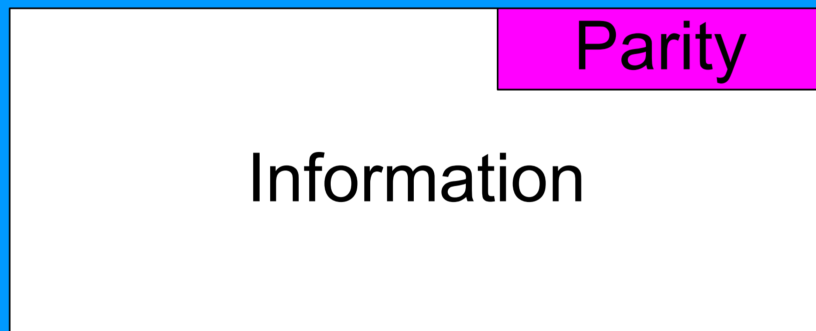
- The same code can be obtained by different symbol mappings
- Clever mappings can be used to reduce the Bit-Error Probability

Coding + Mapping = 2-phase Error Protection:



# AIM Codes in Memory Devices

1) Encoding of $\Sigma$ :

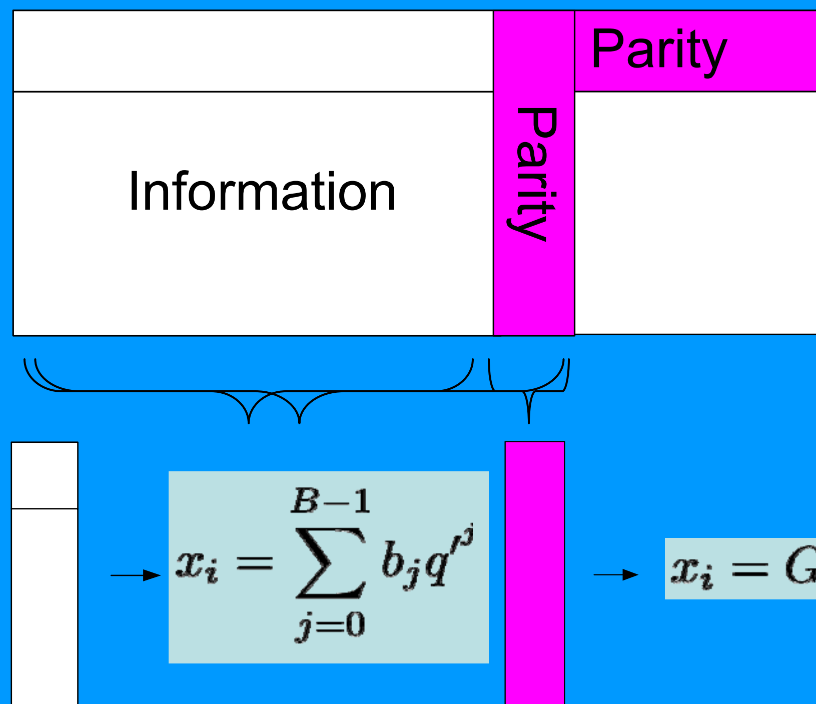



2) Mapping to $Q^n$ :


In Memory devices


Provides faster reads and updates

# Systematic AIM Codes




$$x_i = \sum_{j=0}^{B-1} b_j q^{r^j}$$

$$x_i = Gray(b_0, \ldots, b_{B-1})$$

Works for $l = 1$

An A1M error in a parity symbol $\overset{Gray}{\Longrightarrow}$ One bit error for $\Sigma$

# Mappings for Systematic AlM Codes

For $l > 1$

An AlM error in a parity symbol $\not\Rightarrow$ One symbol error for $\Sigma$

Example: $q' = 3, \ \ell = 2$

| | |
|---|---|
| 0 | 000 |
| 1 | 001 |
| 2 | 002 |
| 3 | 012 |
| 4 | ... |

+1

+2

2 errors

# Mappings for Systematic AlM Codes

For general $q'$ and $\ell$ , how many worst-case $Q'$ symbol errors per an AlM error?

At least 2 (for any mapping)

At most 2 (for some mappings):

*Lemma*

For any $q', \ell$, the $q'$-ary Generalized Reflected Gray Code induces at most two $Q'$ symbol errors for any A$\ell$M error.

- One symbol error is always at a fixed location
- The other error is always $\pm 1$ modulo $q'$

# Mappings for Systematic AIM Codes

<u>Research Problem:</u>

Find high rate (<1) mappings that induce only a single $Q'$ error

$$|\tilde{Q}| = \frac{1}{2}|Q| = \frac{1}{2}q$$

$$R \triangleq \log_q |\tilde{Q}|$$

$$\Downarrow$$

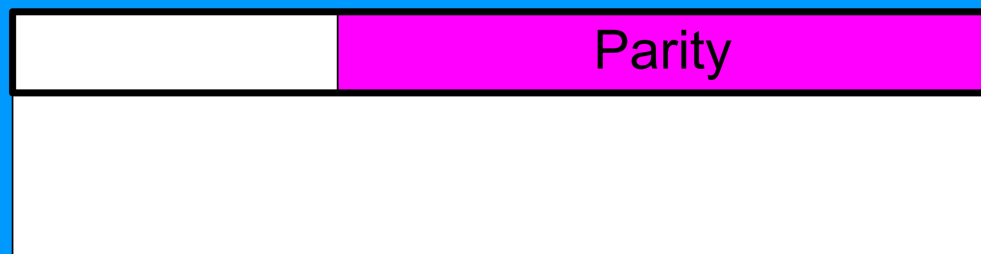
$$R = 1 - \frac{1}{\log_2 q}$$

| | $\leftarrow B \rightarrow$ |
|---|---|
| 24 | 110 |
| 25 | 111 |
| 26 | 112 |
| 27 | 113 |
| ~~28~~ | ~~103~~ |
| ~~29~~ | ~~102~~ |
| ~~30~~ | ~~101~~ |
| ~~31~~ | ~~100~~ |
| 32 | 200 |
| 33 | 201 |
| 34 | 202 |
| 35 | 203 |
| ~~36~~ | ~~213~~ |
| ~~37~~ | ~~212~~ |
| ~~38~~ | ~~211~~ |
| ~~39~~ | ~~210~~ |

$$q = (q')^B$$

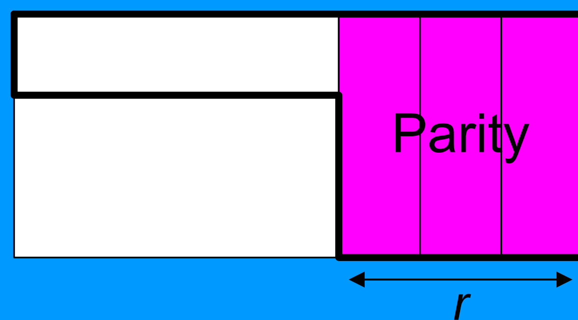
# Construction 2

Construction 1




Folding parity symbols

(with *r* <u>bits</u> extra redundancy)




Lengthening code

(when $\Sigma$ is linear)

# Coding for Denser and Faster Flash Memories

Yuval Cassuto
Paradise Laboratory
EE Department
Caltech

# Programming with Controlled Errors

– Disturbs
– Retention
– Endurance

Uncontrolled AIM Errors (Read Errors)

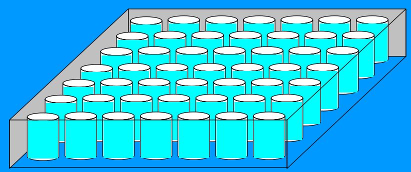
– Iterative Programming

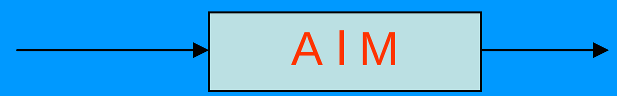
Controlled AIM Errors (Write Errors)

- Programmed level is a random function of the program pulse
- Programming to exact levels requires many write/read iterations
- Allowing AIM errors speeds up programming or avoids costly erases
- Optimization problem: Budget AIM errors to minimize program time

(experimental study on Floating Gate Arrays with Georgia Tech.)

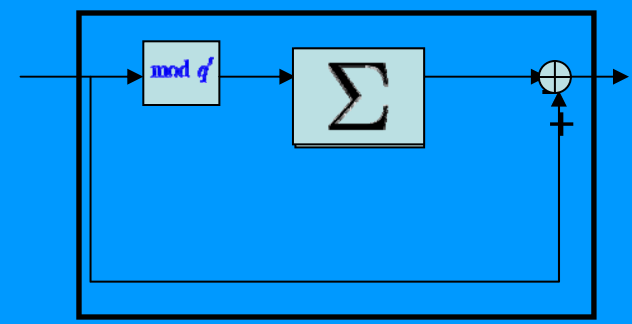
# Conclusion






$$|C| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i = q^n$$

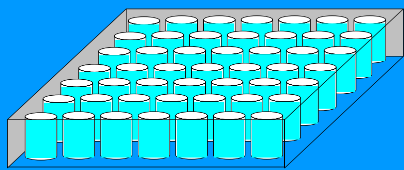
$$C = \{x \in Q^n : x \bmod (\ell + 1) \in \Sigma\}$$




| | |
|---|---|
| 0 | 000 |
| 1 | 001 |
| 2 | 002 |
| 3 | 012 |
| 4 | ... |

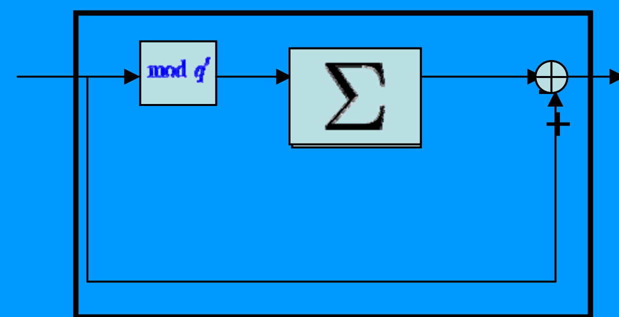
Information | P

# Questions?




$$|C| \cdot \sum_{i=0}^{t} \binom{n}{i} \ell^i = q^n$$

$$C = \{x \in Q^n : x \bmod (\ell + 1) \in \Sigma\}$$




| | |
|---|---|
| 0 | 000 |
| 1 | 001 |
| 2 | 002 |
| 3 | 012 |
| 4 | ... |

Information | P