

# Memory Diagnosis

Yi-Shing Chang

Principal Engineer, GEMS, Intel Corp

Adjunct Prof, GSAT, NTU

# Agenda

Why Memory Diagnosis

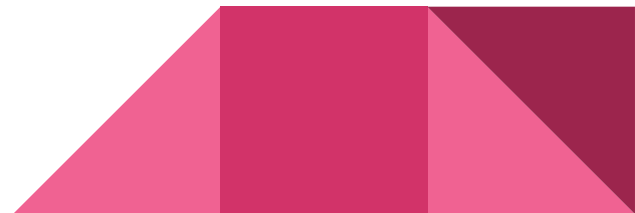
Memory Built-In Self Test (BIST)

Built-In Repair Analysis (BIRA)

Built-In Self Repair (BISR)

Memory Diagnosis

Read/Write Assist

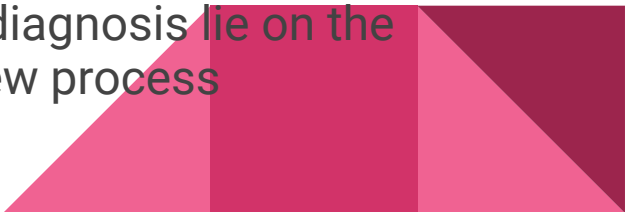


# Why Memory Diagnosis

Memory arrays are key components in CPUs, SOC's, GPUs, and etc.

Memory array numbers and area have been steadily increasing in last decades -> Moore's Law

Memory plays a strategic role as technology "driver" for Si process development

- The finest line widths and highest transistor densities of each new process are pioneered by memory designs
  - Each new logic process is first interpreted and mastered through their effects on memory yield
  - Effective memory testing and prompt memory fault diagnosis lie on the critical path in the race to max-production of each new process
- 

# Why Memory Repair/Replace

To avoid yield loss, redundant elements or spare elements (i.e., spare rows and columns of storage cells) are often added so most faulty cells can be repaired (i.e., replaced by spare cells)

Redundancy adds to cost in another form

- Analysis of redundancies to maximize yield (after repair) and minimize cost is an important process during manufacturing
- Foundries have redundancy requirements e.g. > 1 million bits must have x% of redundancy to guarantee the memory yield



# Array Test

The big picture: Higher order algorithmic patterns are designed to screen very subtle faults and defect mechanisms

- Gate Leakage, cell imbalances,  $V_t$  mismatch/variance, coupling, linked,...
- The bitcells affected by these faults can exhibit themselves at any corner of test - Min or Max  $V_{cc}$ , Freq and Temperature
- Need to screen these fault types to meet DPM goals



# Memory Addressing

Physical address: the physical location of memory cells in an array

- The location of a corner (or the center) of a memory cell relative to some specified origin on a die. It could also be specified relative to an origin point for an array such as the lower left corner of the array
- A count of bit cells to the left or right and up or down from a specified origin of an array
- A count of the number of rows and the columns from a specified origin

Logical address: the address would be used functionally usually as those in RTL or design hierarchy. E.g.

- The logical address for a four way set associative cache with 1K sets would include the set (0x000 to 0x3FF) and the way (0x0 to 0x3)

# Logical Address to Physical Location Mapping

The mapping of logical addresses to physical locations in an array is determined by the layout of the array and by the address decoding circuitry. In most cases, there is a simple correspondence of logical address bits to physical address bits. However,...

- Memory data scrambling: Memory controller turns user data written to the memory into pseudo-random patterns

Need to add de-scrambling in the mapping to find the right physical location for defects

# Why Memory Data Scrambling

Memory design and layout practices, which are the main causes for scrambling e.g.

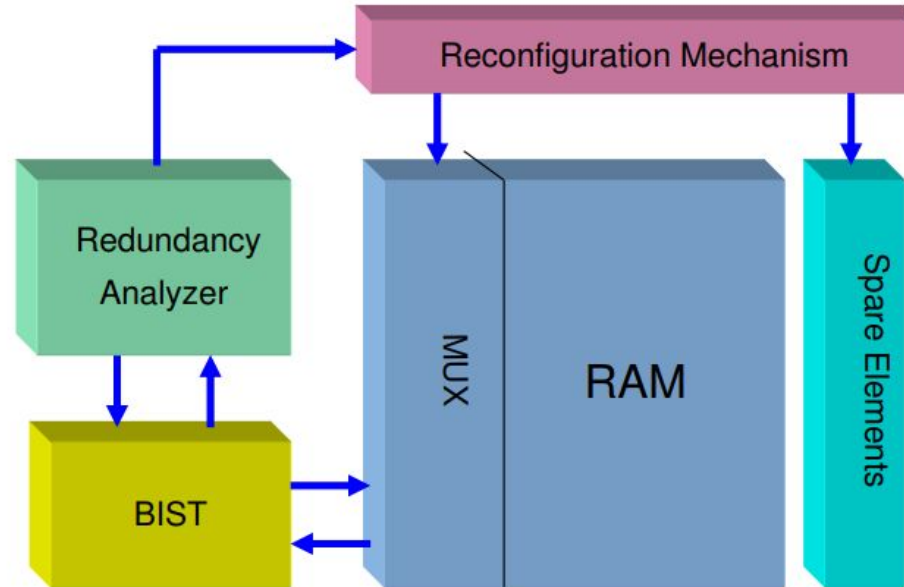
- Geometry optimization introducing folding
- Address decoder optimization
- Cell area optimization by sharing contacts and well areas
- Speed and robustness optimization based on bitline twisting
- Redundancy for yield
- Achieving I/O pin compatibility utilising address or data line swap

Minimize the impact of excessive  $di/dt$  due to successive 1s and 0s on the data bus. Past experience has demonstrated that traffic on the data bus is not random (locality) and can have energy concentrated at specific spectral harmonics creating high  $di/dt$

Security: Memory scrambling prevents forensic and reverse-engineering analysis based on DRAM data remanence



## *RAM Built-In Self-Repair (BISR)*



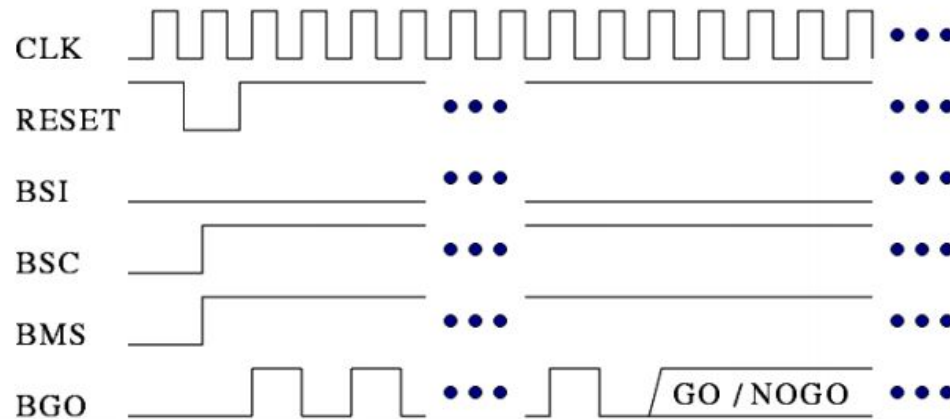
## *From BIST to BISR*



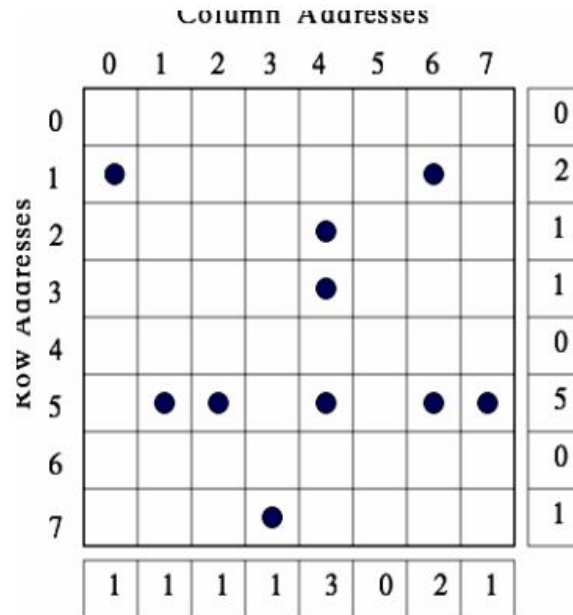
- BIST: built-in self-test
- BIECA: built-in error catch & analysis
  - BISD: built-in self diagnosis
  - BIRA: built-in redundancy analysis
- BISR: built-in self-repair

## Test Mode

- In Test Mode it runs a fixed algorithm for production test and repair.
  - Only a few pins need to be controlled, and *BGO* reports the result (Go/No-Go).



## Repair-Most (RM)



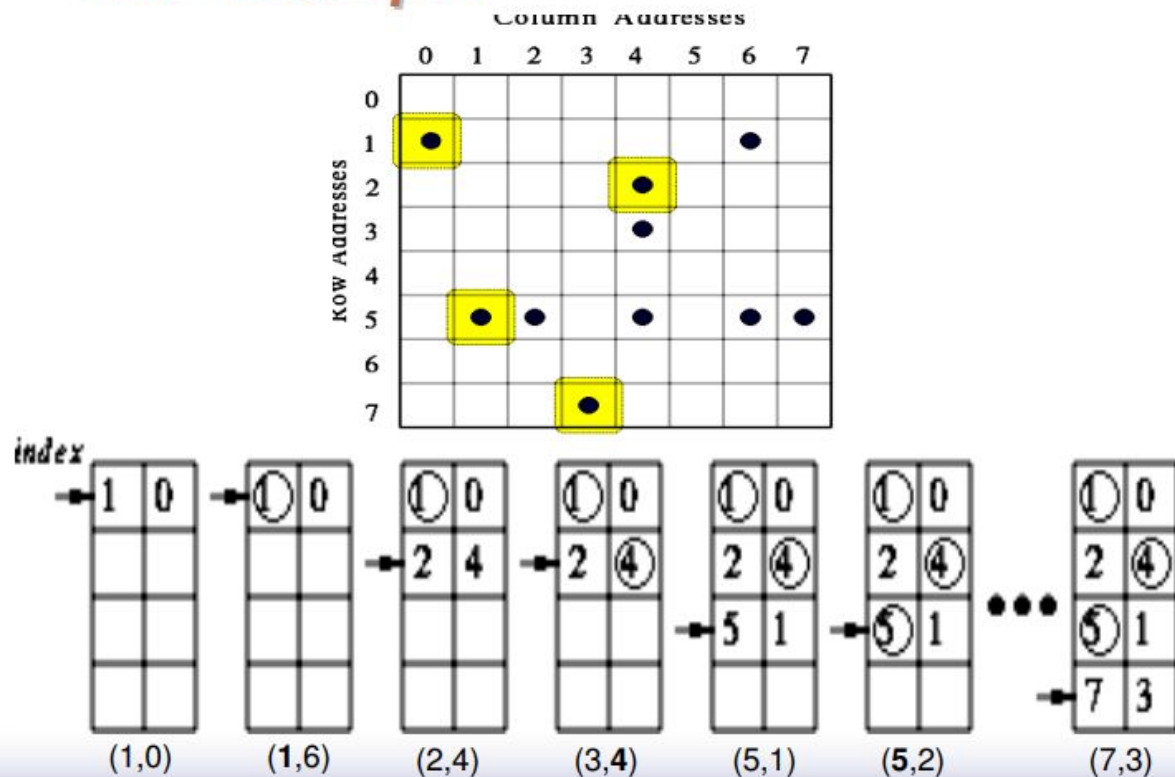
1. Run BIST and construct bitmap
2. Construct row and column error counters
3. Run Must-Repair algorithm
4. Run **greedy** final-repair algorithm

# Essential Spare Pivoting (ESP)

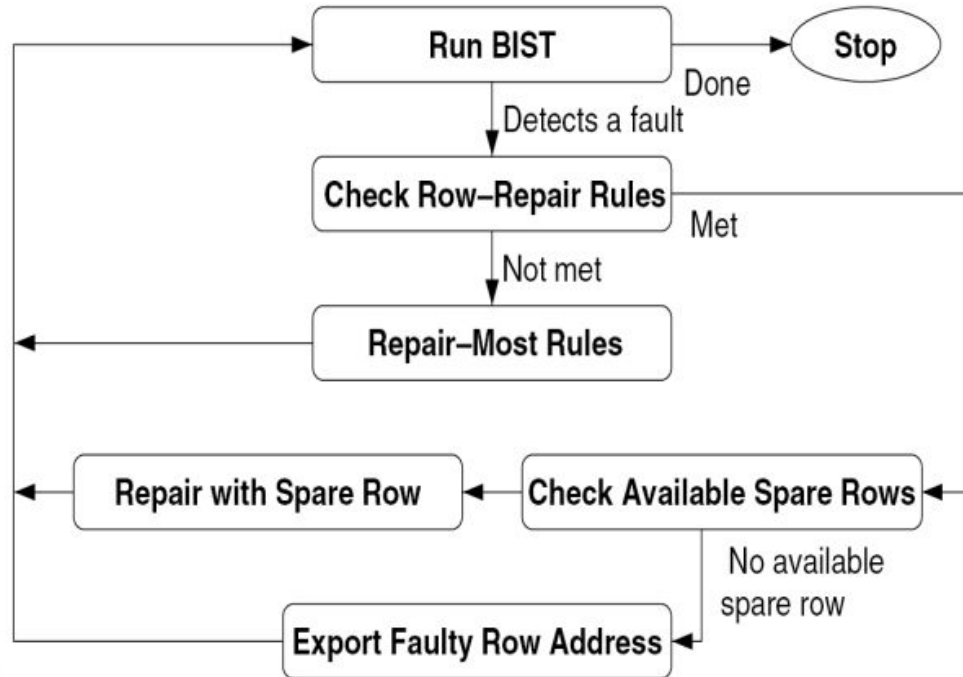
- Maintain high repair rate without using a bitmap
  - Small area overhead
- Fault Collection (FC)
  - Collect and store faulty-cell address using row-pivot and column-pivot registers
    - If there is a match for row (col) pivot, the pivot is an essential pivot
    - If there is no match, store the row/col addresses in the pivot registers
  - If  $F > r+c$ , the RAM is irreparable
- Spare Allocation (SA)
  - Use row and column pivots for spare allocation
    - Spare rows (cols) for essential row (col) pivots
  - SA for orthogonal faults

Ref: Huang *et al.*, IEEE TR, 11/03

# ESP Example



## ***BIRA Procedure***




# Hard Repair or Soft Repair

Hard Repair: repair instructions are stored permanently within the die through the programming of fuses. Electrical fuses (eFuses) are typically one-time programmable or flash memory elements and are programmed using an elevated voltage level

Soft Repair: repair instructions are stored in volatile memory, typically in scan registers, at each power up of the device

Soft repair has the advantage of being able to address defects that may arise over time as new repair instructions can be created and stored in the system

For this reason, soft repair is almost exclusively associated with a BIRA mechanism to calculate repair instructions on-chip at power up





# Memory Design/Test Challenges

How much redundancy?

- Foundries provide minimum requirements based on memory defect density of each process
- Design many need to add extra for multiple types of memory e.g. multi-port or customized memory

Row Repair or Column Repair?

When to repair? At Sort or FT? At Cold temp or Hot temp

Vmin repair or Spec with margin Repair



# Memory Diagnosis

The BIST engine can define one or more memory test algorithms that will be programmed into the memory BIST controller

Run the memory BIST controller for all the steps with the specific algorithms assigned at the generation time

Determine the memory repair info or fail as unrepairable

Run the memory BIST controller in diagnostic mode where you can freeze on a specific BIST step, specific memory test port, or specific error count



# Memory Diagnosis

In diagnosis mode, BIST engine could stop on the fail cell and shift out the failing cell address, BIST cmd (step), and error info

The BIST engine would scan each memory cell using specific algorithms and dump out the failing cell addresses up to a specific error count (stop on N fail) -> to construct error bitmap of a memory instance

From the failing memory test algorithm, error map, fault dictionary, and physical locations of failing cells, fault bitmaps would be created



## Fault Model Subtypes

Name	Agr	Vtm	Addr
SAF <sub>0</sub>	-	1/0	-
SAF <sub>1</sub>	-	0/1	-
TF <sub>0</sub>	-	↓/1	-
TF <sub>1</sub>	-	↑/0	-
CFin <sub>0</sub>	↓	↕	A < V
CFin <sub>1</sub>	↓	↕	A > V
CFin <sub>2</sub>	↑	↕	A < V
CFin <sub>3</sub>	↑	↕	A > V
CFst <sub>0</sub>	0	1/0	A < V
CFst <sub>1</sub>	0	1/0	A > V
CFst <sub>2</sub>	0	0/1	A < V
CFst <sub>3</sub>	0	0/1	A > V
CFst <sub>4</sub>	1	1/0	A < V
CFst <sub>5</sub>	1	1/0	A > V

Name	Agr	Vtm	Addr
CFst <sub>6</sub>	1	0/1	A < V
CFst <sub>7</sub>	1	0/1	A > V
CFid <sub>0</sub>	↓	1/0	A < V
CFid <sub>1</sub>	↓	1/0	A > V
CFid <sub>2</sub>	↓	0/1	A < V
CFid <sub>3</sub>	↓	0/1	A > V
CFid <sub>4</sub>	↑	1/0	A < V
CFid <sub>5</sub>	↑	1/0	A > V
CFid <sub>6</sub>	↑	0/1	A < V
CFid <sub>7</sub>	↑	0/1	A > V
AF <sub>0</sub>	-	-	A < V
AF <sub>1</sub>	-	-	A > V
SOF	-	-	-

**Inversion Coupling Fault**  
(CFin): A transition in one

cell (Agr) inverts the  
content of another (Vtm)

**Idempotent coupling fault**

(CFid): A transition in one  
cell forces a constant  
value (1 or 0) into another

**State coupling fault:**

(CFst): A Victim cell is  
forced to a certain value if  
Agr cell is in a given state

# March Signature & Dictionary

March 11N

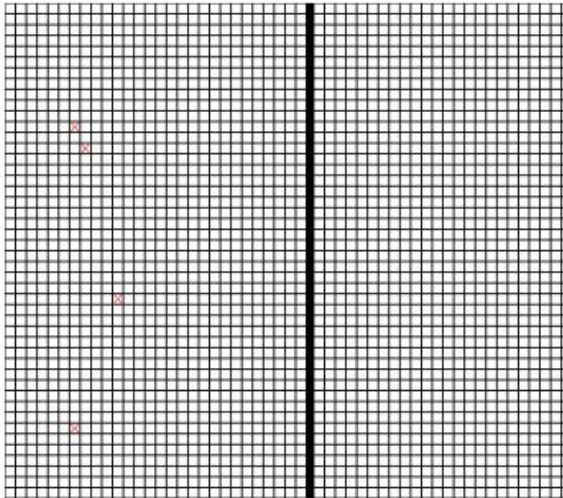
$\Downarrow(w0) \Uparrow(r0, w1) \Downarrow(r1) \Uparrow(r1, w0) \Downarrow(r0, w1) \Downarrow(r1, w0) \Downarrow(r0)$

$E_0 \quad E_1 E_2 \quad E_3 \quad E_4 E_5 \quad E_6 E_7 \quad E_8 E_9 \quad E_{10}$

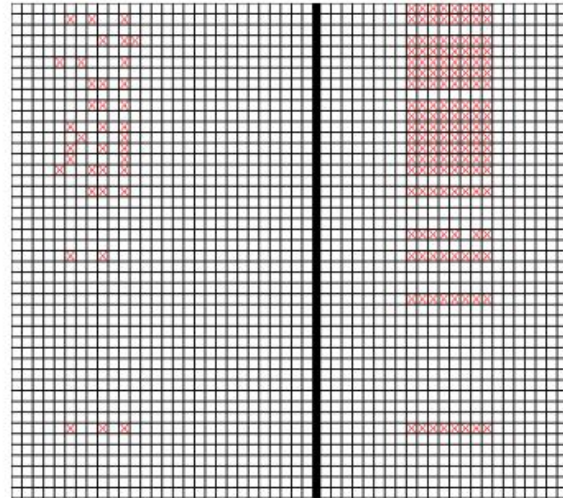
Fault/Error Bitmap	$E_0$	$E_1$	$E_2$	$E_3$	$E_4$	$E_5$	$E_6$	$E_7$	$E_8$	$E_9$	$E_{10}$
SAF <sub>0</sub>	0	0	0	1	1	0	0	0	1	0	0
SAF <sub>1</sub>	0	1	0	0	0	0	1	0	0	0	1
CFin <sub>0</sub>	0	0	0	0	1	0	0	0	0	0	1
CFin <sub>1</sub>	0	0	0	0	0	0	1	0	1	0	0
CFin <sub>2</sub>	0	1	0	0	0	0	0	0	1	0	0
CFin <sub>3</sub>	0	0	0	1	1	0	1	0	0	0	0
CFst <sub>0</sub>	0	0	0	0	1	0	0	0	0	0	0
CFst <sub>1</sub>	0	0	0	0	0	0	0	0	1	0	0
CFst <sub>2</sub>	0	0	0	0	0	0	1	0	0	0	1
CFst <sub>3</sub>	0	1	0	0	0	0	0	0	0	0	1
CFst <sub>4</sub>	0	0	0	1	0	0	0	0	1	0	0
CFst <sub>5</sub>	0	0	0	1	1	0	0	0	0	0	0
CFst <sub>6</sub>	0	1	0	0	0	0	0	0	0	0	0
CFst <sub>7</sub>	0	0	0	0	0	0	1	0	0	0	0

# Fault Bitmap Examples

Idempotent Coupling Fault



Stuck-at 0




# Memory Diagnosis Summary

To save test time, many memory test algorithms would be executed during production test to make sure the memory work properly within spec

Need volume diagnosis to be able to determine the priority of failing signatures and the pFA candidates

Multiple very high order of memory test algorithms, such as 59N or 61N, may require to understand the failing mechanism for advanced processes and modern memory cell design with very high density

Highly resource intensive, but required especially for pFA and worth the efforts if yield improvement can be realized

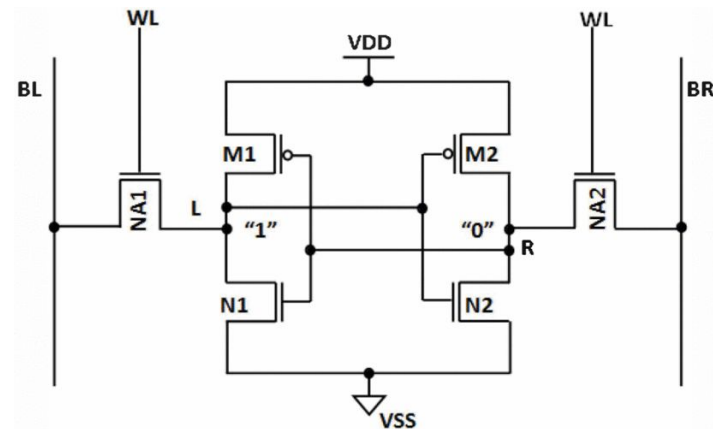


# Memory Read/Write Assist

What: Techniques to improve Read-/Write- ability and Read stability under  $\sim V_{min}$  condition

Why? Reducing SRAM operational voltage ( $V_{min}$ ) can greatly improve energy efficiency. However, (1) process variations (2) In advanced process node,

- $V_t$  &  $V_{dd}$  scaling down  $\rightarrow$  Noise margin going down
- only basic cells with fixed fin count are allowed to be used for high density memory
  - Beta-ratio =  $(W_{pd}/L_{pd})/(W_{tr}/L_{tr}) \sim 1$  for minimum size N's





# Failure Models for SRAM

**Write-ability** failure occurs when the internal cell node voltage does not reach to the desired voltage level during a write operation

**Readability** failures occur when read bit lines discharge level in specified time is less than the offset of the sense amplifier.

**Read stability** failures occurs when bit cells content flip accidentally during the read operation.

- The classical method of calculating SRAM cell stability is based on measuring the static noise margin (SNM) of the cross-coupled inverters in SRAM cell. The SNM is a measure of the maximum extent of noise voltage that can be allowed at the inverter nodes without flipping the cell

# Techniques for RW Assist

Modern CPU contains many operation modes such as Turbo, Sleep or Hibernate by setting different voltages for logics or arrays to save power

To address the minimum voltage requirement and parametric yield loss due to SRAM failure many techniques are available

- SRAM cell modifications - Beta (1.5 ~2) & Alpha Ratio -> usually provided by IP vendors with a few options
- Circuit techniques – main focus
- Body biasing

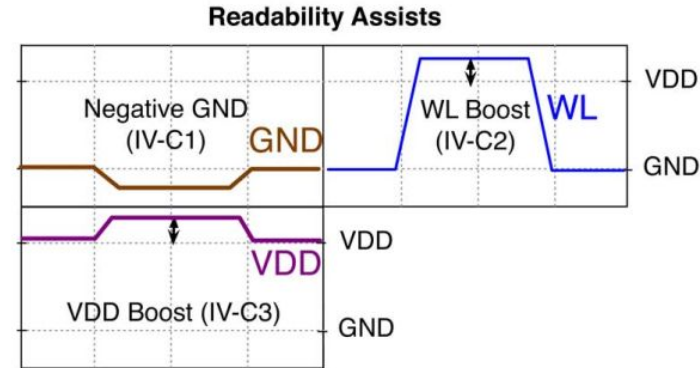
“SRAM Assist Techniques for Operation in a Wide Voltage Range in 28-nm CMOS”,  
B. Zimmer et. al., IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, VOL. 59, NO. 12,  
DECEMBER 2012

# Circuit Techniques For Readability Assist

**Word Line (WL) Boost:** Using a wordline boost can improve both the readability and write-ability of cells, but will drastically diminish the read stability

**VDD Boost (VDDBoost):** Supply voltage across the cell is increased above the VDD. It improves the  $V_{gs}$  therefore the strength of the pull-up transistor -> improves read stability but it degrades to write margin during write mode.

**Negative GND:** Reducing GND below the ground level. Negative GND is the most effective of all readability assist techniques as it increases the  $V_{gs}$  on both the pull down and Pass transistor by pulling the internal node holding '0' below ground. Unfortunately, this technique has a very high-energy cost because GND lines have large capacitance

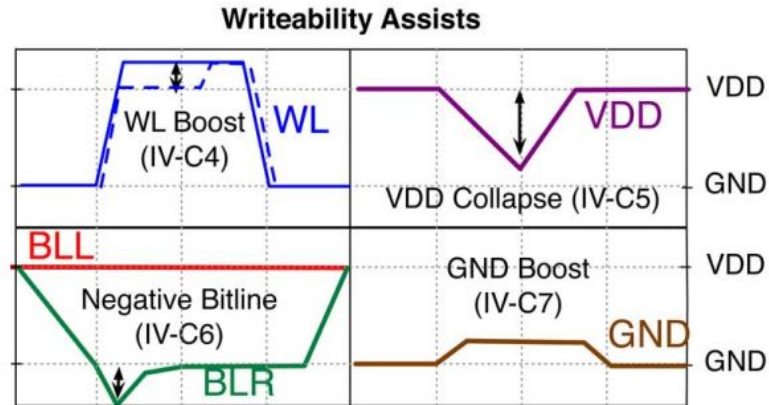


# Circuit Techniques For Write-ability Assist

**Negative Bitline Voltage:** Bit line voltage is pulled into negative (below ground potential), which will increase the strength ( $V_{gs}$ ) of the pass transistor. This negative bit line voltage helps for the faster discharge of the SRAM cell node “1” and improves the write-ability operation

**Negative VDD:** helps the PG pull the high-node low by weakening the high-node PU, a write-ability failure can still occur if the PU on the low-node side is weak

**GND Boost** (Not good)



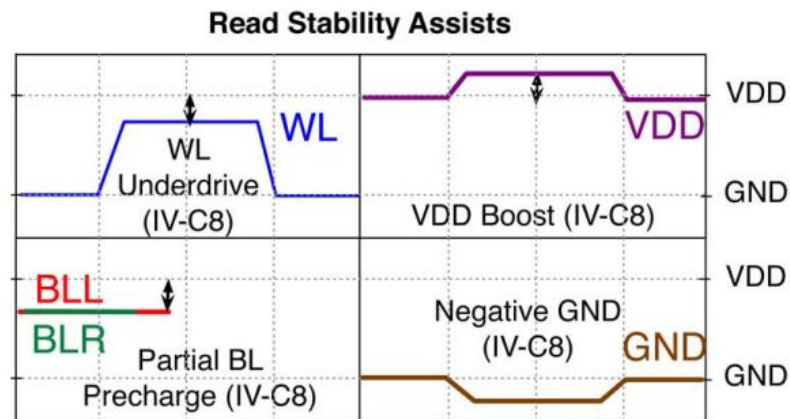
# Circuit Techniques For Read Stability Assist

**Word Line Under Drive:** Word line voltage is reduced by some 10s mV -> Pass Transistor weaker, i.e., improve Beta ratio (best)

**VDD Boost**

**Partial BL Precharge**

**Negative GND**



# Circuit Design for RWA

All these features need design support to enable various sizes of “Underdrive” or “Boost”

Trade-off between the numbers of cells for each bitline and sense amplifier and these design features

The test patterns for different RWA need to be run to determine the RWA sizes



# RWA Summary

The memory need to allow R/W operations on different modes with different bandwidth and latency requirements. Testing memory at different frequencies and different temperatures may become too time-consuming

Read/Write Assist has been shown effective ways to improve read-/write-ability and read stability in nanometer technologies



# Summary

Memory is a different beast compared with logics. Modern SOC and CPU designs contain large asset of memory. Its diagnosis and yield is equally important if not greater than logics

