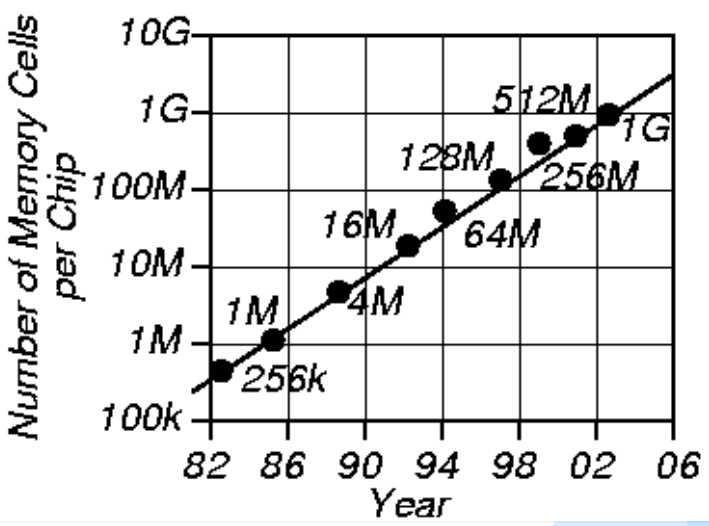
Department of Electronics Engineering
National Chiao Tung University
Hsinchu, Taiwan

Memory Test

Density and Defect Trends

- 1970 -- DRAM Invention (Intel) 1024 bits
- 1993 -- 1st 256 MBit DRAM papers
- 1997 -- 1st 256 MBit DRAM samples
  - 1 ¢/bit --> 120 X 10^{-6} ¢/bit
- Kilburn -- Ferranti Atlas computer (Manchester U.) -- Invented Virtual Memory
  - 1997 -- Cache DRAM -- SRAM cache + DRAM now on 1 chip



Memory cells per chip

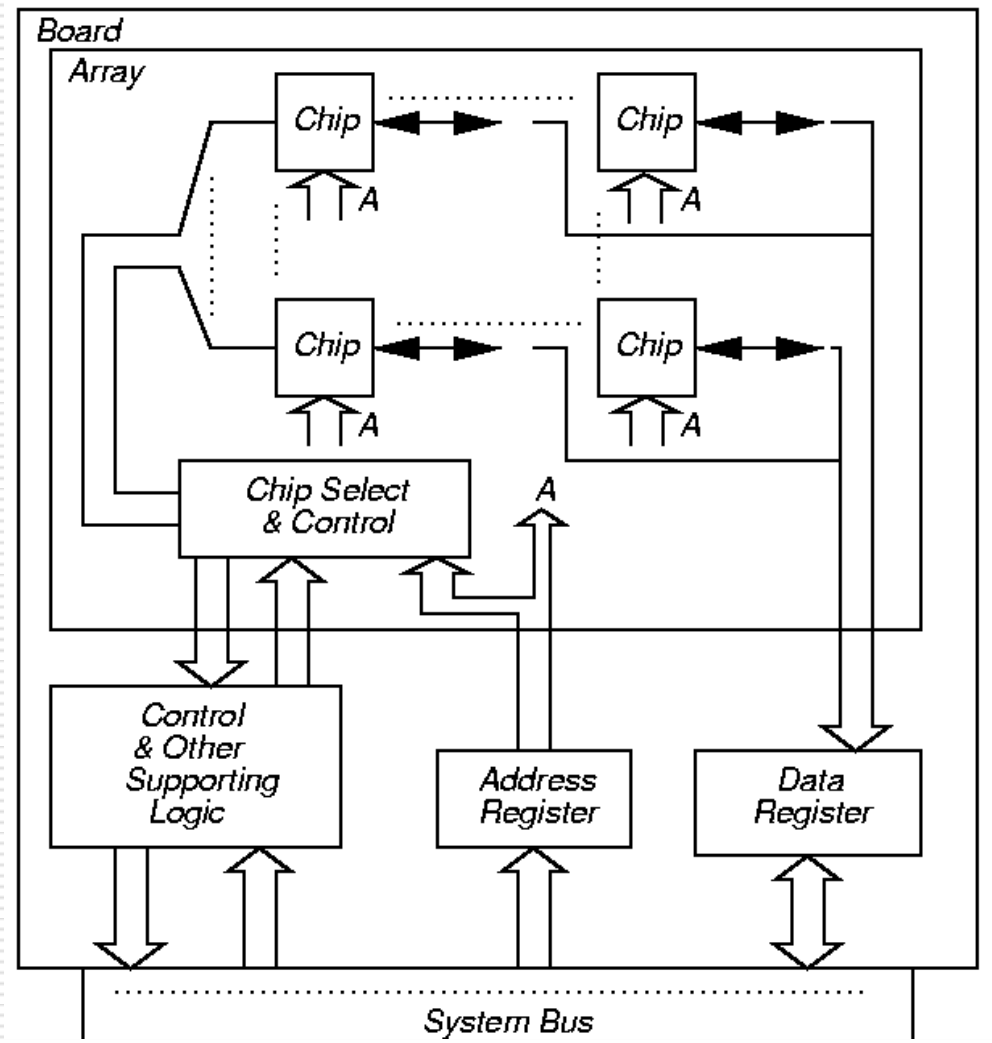
Test Time in Seconds

Size (n)	Number of Test Algorithm Operations			
	n	$nlog_2n$	$n^{3}/_{2}$	n^2
1Mb	0.06	1.26	64.5	18.3 hr
4Mb	0.25	5.54	515.4	293.2 hr
16Mb	1.01	24.16	1.2 hr	4691.3 hr
64Mb	4.03	104.7	9.2 hr	75060.0 hr
256Mb	16.11	451.0	73.3 hr	1200959.9 hr
1Gb	64.43	1932.8	586.4 hr	19215358.4 hr
2Gb	128.9	3994.4	1658.6 hr	76861433.7 hr

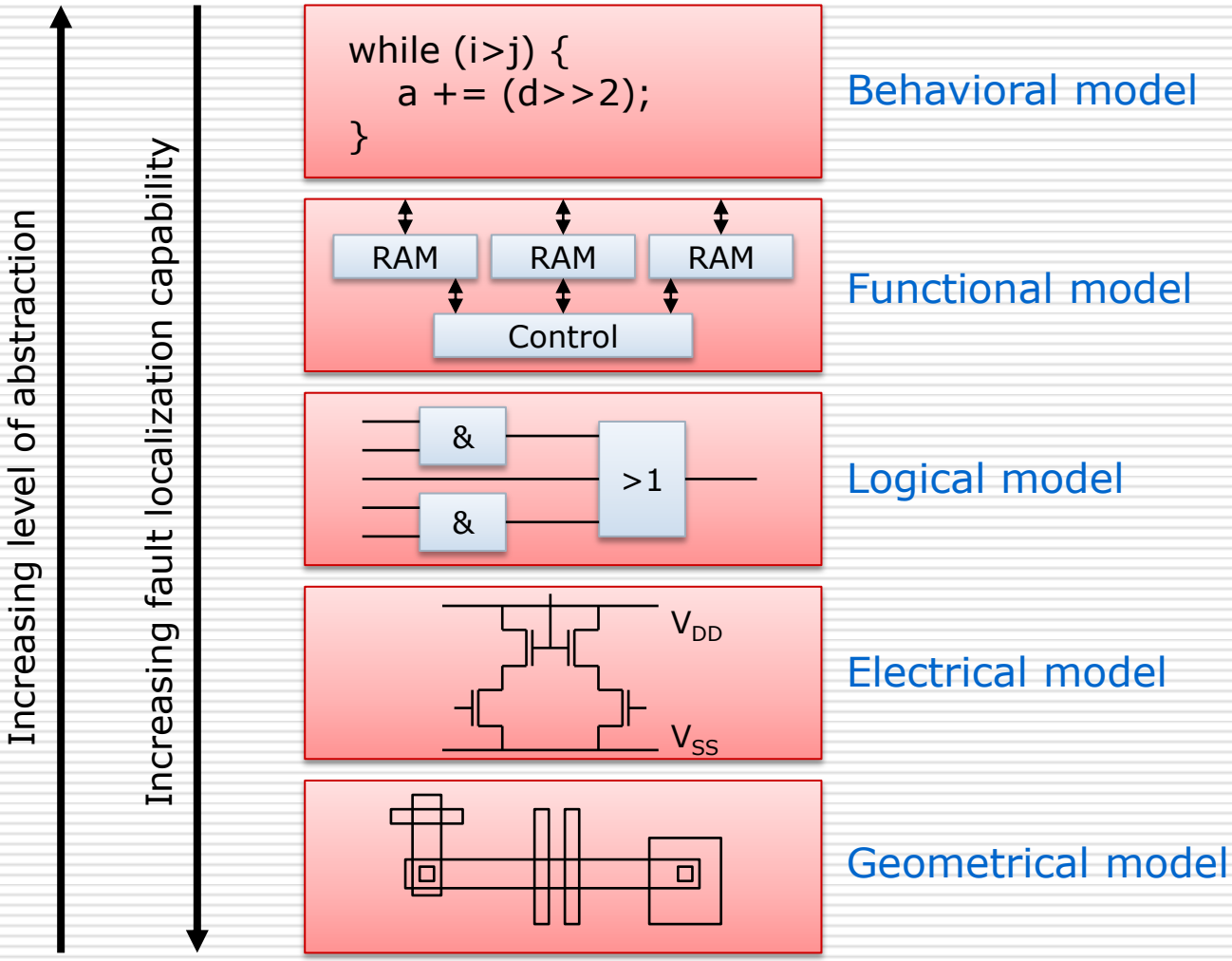
cycle time: 60ns

Memory Test Levels

- Three levels
  1. chip level
  2. array level
  3. board level

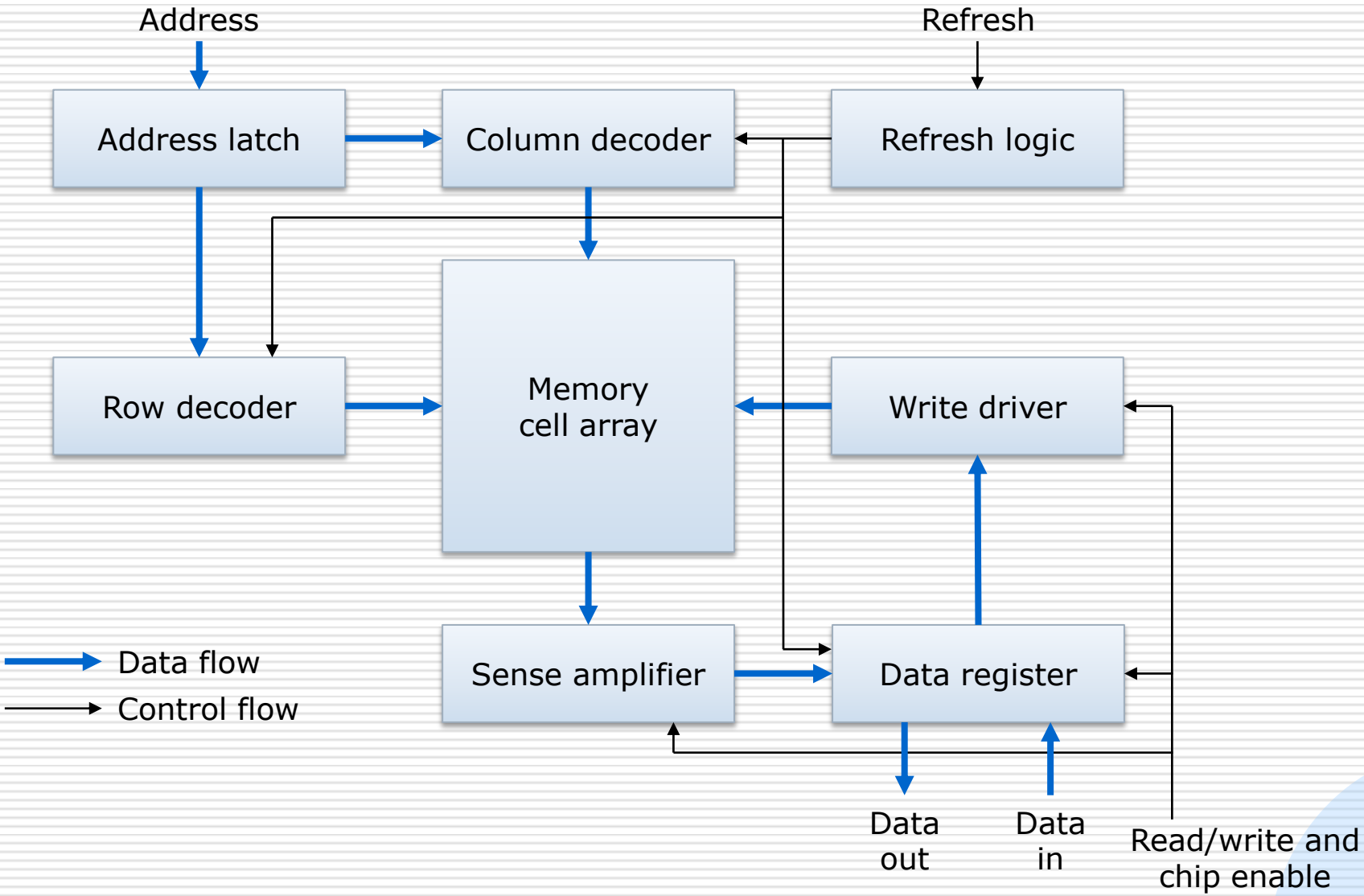


Memory Models



Models and levels of abstraction

Functional RAM Chip Model

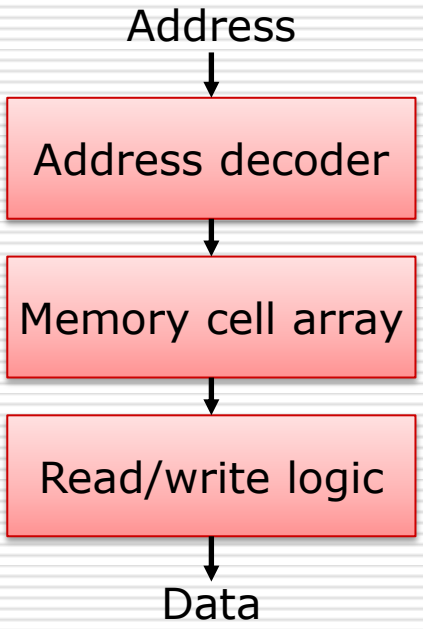


Functional Faults

	Functional fault
a	Cell stuck
b	Driver stuck
c	Read/write line stuck
d	Chip-select line stuck
e	Data line stuck
f	Open circuit in data line
g	Short circuit between data lines
h	Crosstalk between data lines
i	Address line stuck
j	Open circuit in address line
k	Shorts between address lines
l	Open circuit in decoder
m	Wrong address access
n	Multiple simultaneous address access
o	Cell can be set to 0 (1) but not to 1 (0)
p	Pattern sensitive cell interaction

Reduced Functional Model

- During chip testing one is not interested in locating a fault because a chip cannot be repaired. One is only interested in detecting a fault.
- Simplified functional model

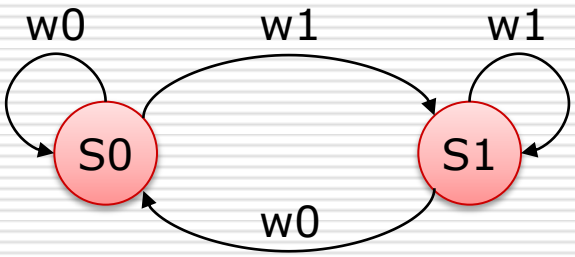


Notations

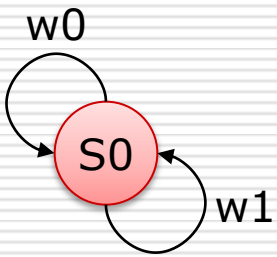
- 0 : a cell is in logical state 0
- 1 : a cell is in logical state 1
- x : a cell is in logical state x, where $x \in \{0,1\}$
- ↑ : a write 1 operation to a cell containing a 0
- ↓ : a write 0 operation to a cell containing a 1
- ↕ : a write $\overline{x}$ operation to a cell containing a x
- →: a write x operation to a cell containing a x
- ∀ : any operations; $\forall \in \{\uparrow, \downarrow, \updownarrow, \rightarrow\}$
- $< I/F >$: a fault in single cell
  - I describes the condition; F describes the value of faulty cell
- $< I1, \ldots, In-1; In/F >$: a fault involving n cells

Stuck-at Fault

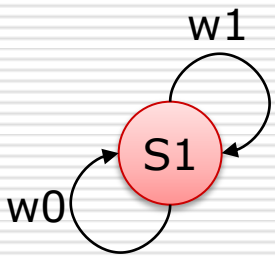
- The logic value of a stuck-at (SA) cell or line is always 0 (SA0 fault) or 1 (SA1 fault); it is always in state 0 or in state 1 and cannot be changed to the opposite state.
- $< \forall/0 >, < \forall/1 >$



state diagram of a good cell



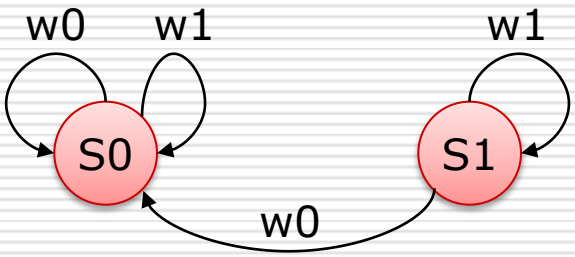
SA0 fault



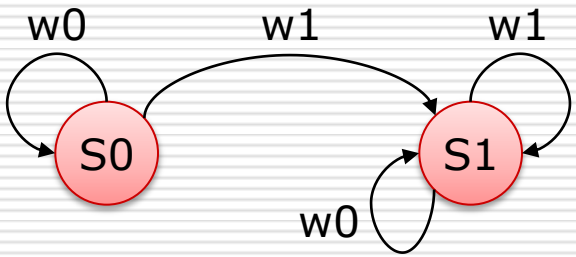
SA1 fault

Transition Fault

- A cell or line which fails to undergo a $0 \rightarrow 1$ transition when it is written is said to contain an up transition fault; similarly, a down transition fault is the impossibility of making a $1 \rightarrow 0$ transition.
- $< \uparrow/0 >, < \downarrow/1 >$



$< \uparrow/0 >$ transition fault



$< \downarrow/1 >$ transition fault

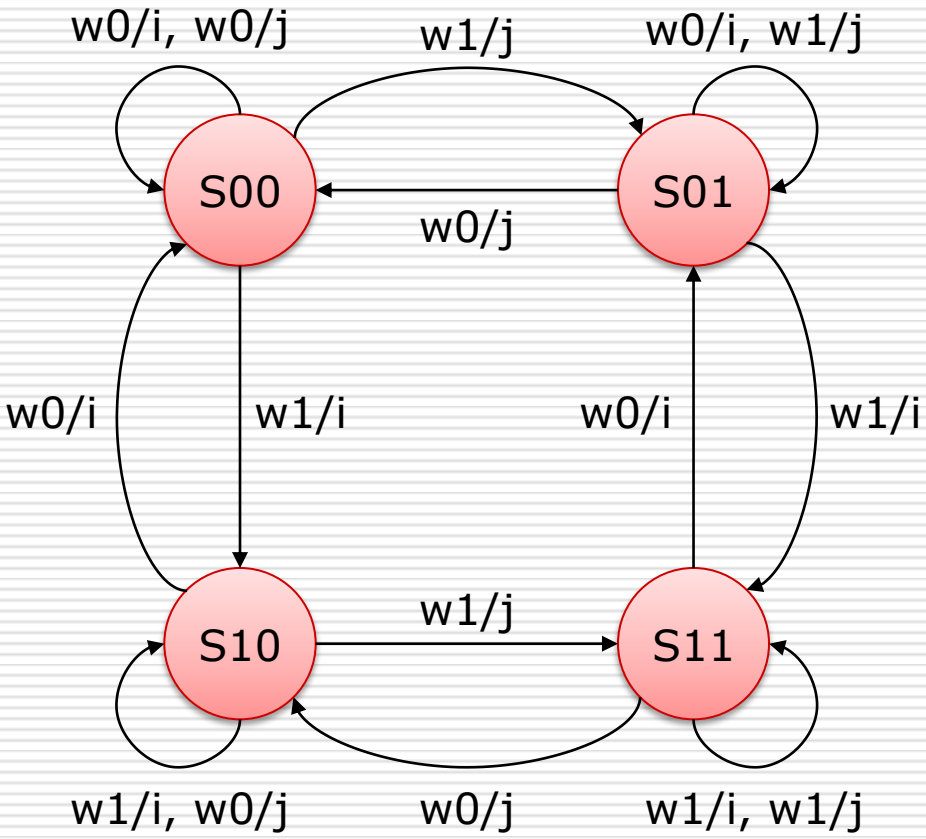
Coupling Faults

- Coupling Fault (CF): Transition in bit j causes unwanted change in bit i
- 2-Coupling Fault: Involves 2 cells, special case of k-Coupling Fault
  - Must restrict k cells to make practical
- Inversion and Idempotent CFs -- special cases of 2-Coupling Faults
- Bridging and State Coupling Faults involve any # of cells, caused by logic level
- Dynamic Coupling Fault (CFdyn) -- Read or write on j forces i to 0 or 1

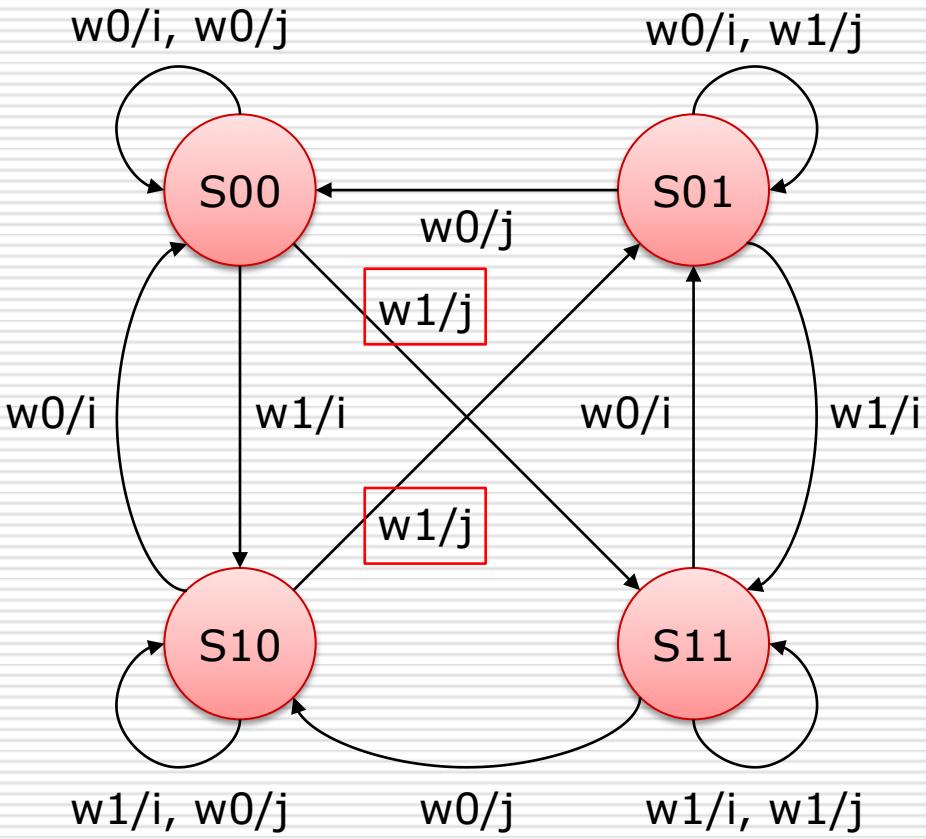
Inversion Coupling Faults (CFin)

- ↑ or ↓ in cell j inverts contents of cell i
- Condition: For all cells that are coupled, each should be read after a series of possible CFins may have occurred, and the # of coupled cell transitions must be odd (to prevent the CFins from masking each other).
- <↑;↕>, <↓;↕>

Good State Transition Diagram (2 cells)



CFin State Transition Diagram

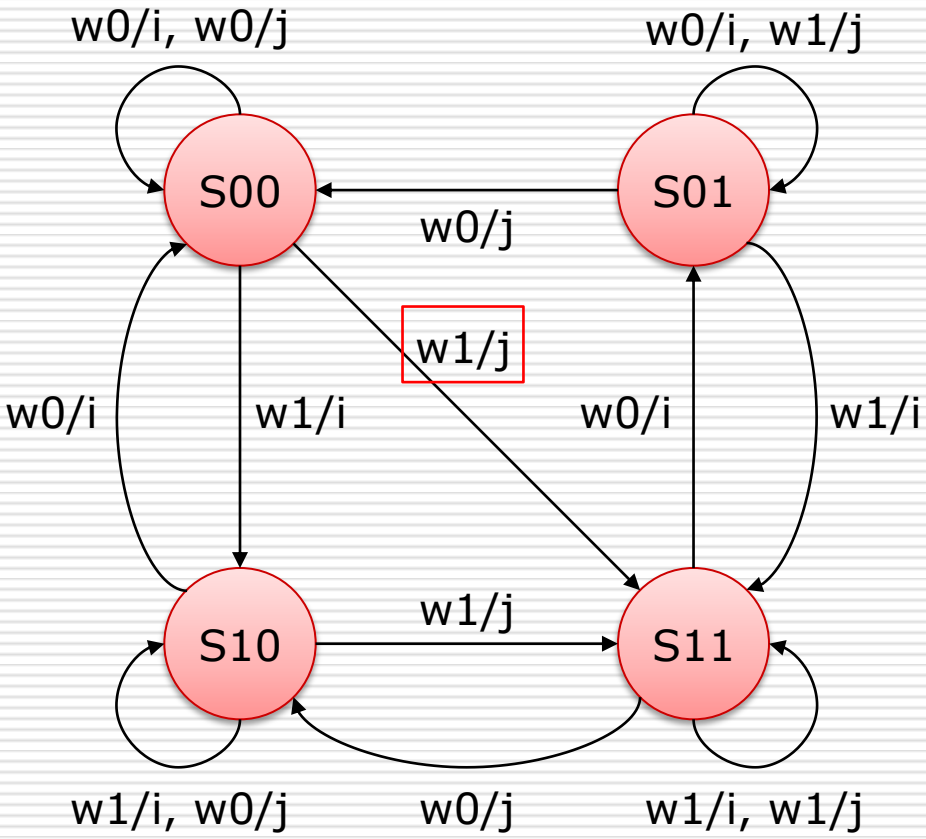


state diagram of an <↑;↕> CFin

Idempotent Coupling Faults (CFid)

- ↑ or ↓ transition in cell j sets cell i to 0 or 1
- Condition: For all coupled faults, each should be read after a series of possible CFids may have happened, such that the sensitized CFids do not mask each other.
- Asymmetric: coupled cell only does ↑ or ↓
- Symmetric: coupled cell does both due to fault
- $<\uparrow; 0>, <\uparrow; 1>, <\downarrow; 0>, <\downarrow; 1>$

CFid Example



state diagram of an <↑; 1 > CFin

Dynamic Coupling Faults (CFdyn)

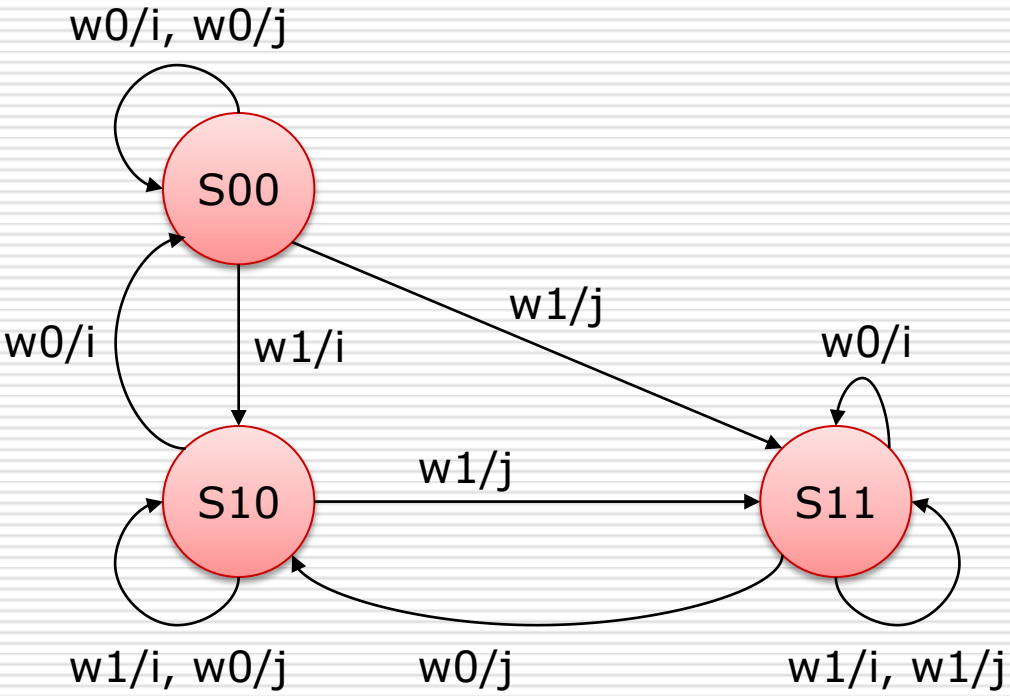
- Read or write in cell of 1 word forces cell in different word to 0 or 1
- $< r0|w0; 0 >, < r0|w0; 1 >,$
  $< r1|w1; 0 >, < r1|w1; 1 >$
  - | Denotes "OR" of two operations
- More general than CFid, because a CFdyn can be sensitized by any read or write operation

Bridging Faults

- Short circuit between 2+ cells or lines
- 0 or 1 state of coupling cell, rather than coupling cell transition, causes coupled cell change
- Bidirectional fault -- i affects j, j affects i
- AND Bridging Faults (ABF):
  - $< 0,0/0,0 >, < 0,1/0,0 >, < 1,0/0,0 >, < 1,1/1,1 >$
- OR Bridging Faults (OBF):
  - $< 0,0/0,0 >, < 0,1/1,1 >, < 1,0/1,1 >, < 1,1/1,1 >$

State Coupling Faults

- Coupling cell / line j is in a given state y that forces coupled cell / line i into state x
- $< 0; 0 >, < 0; 1 >, < 1; 0 >, < 1; 1 >$

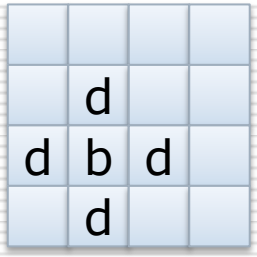


example of <1;1>

Neighborhood Pattern Sensitive Fault (1/2)

- Neighborhood Pattern Sensitive Fault (NPSF)
  - The contents of a cell, or the ability to change the contents, is influenced by the contents of all other cells in the memory.
- The NPSF can be considered the most general case of the k-coupling fault.

Memory array



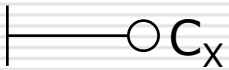
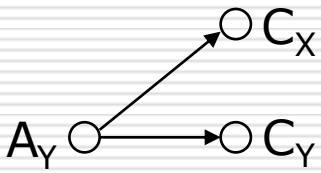
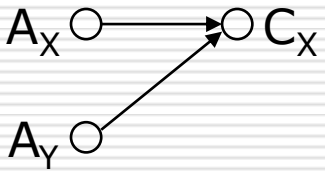
b: base cell
d: deleted neighborhood cell
b+d: neighborhood

Neighborhood Pattern Sensitive Fault (2/2)

- Active NPSF (ANPSF)
  - also called Dynamic NPSF
  - The base cell changes its contents due to a change in the deleted neighborhood pattern.
- Passive NPSF (PNPSF)
  - The content of the base cell cannot be changed (it cannot make a transition) due to a certain deleted neighborhood pattern.
- Static NPSF (SNPSF)
  - The content of a base cell is forced to a certain state due to a certain deleted neighborhood pattern.

Address Decoder Faults (ADFs)

- Address decoding error assumptions:
  - Decoder does not become sequential
  - Same behavior during both read & write
- Multiple ADFs must be tested
- Decoders have CMOS stuck-open faults

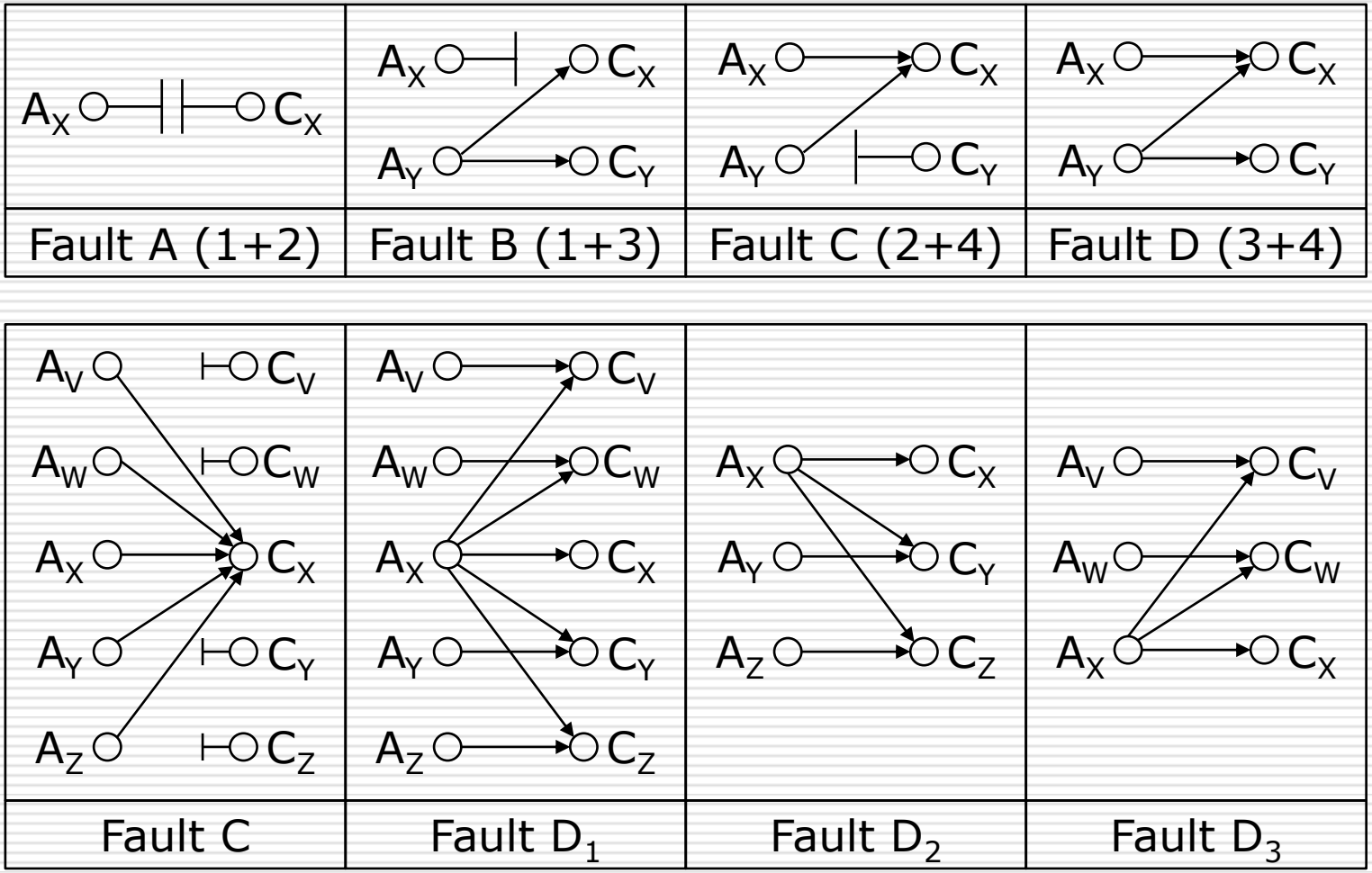
			
Fault 1	Fault 2	Fault 3	Fault 4
No cell accessed for A_X	No address to access cell C_X	Multiple cells accessed with A_Y	Multiple addresses for cell C_X

Theorem 9.2

- A March test satisfying conditions 1 & 2 detects all address decoder faults.
- ... Means any # of read or write operations
- Before condition 1, must have wx element
  - x can be 0 or 1, but must be consistent in test

Condition	March element
1	$(rx, \ldots, w\overline{x})$
2	$(r\overline{x}, \ldots, wx)$

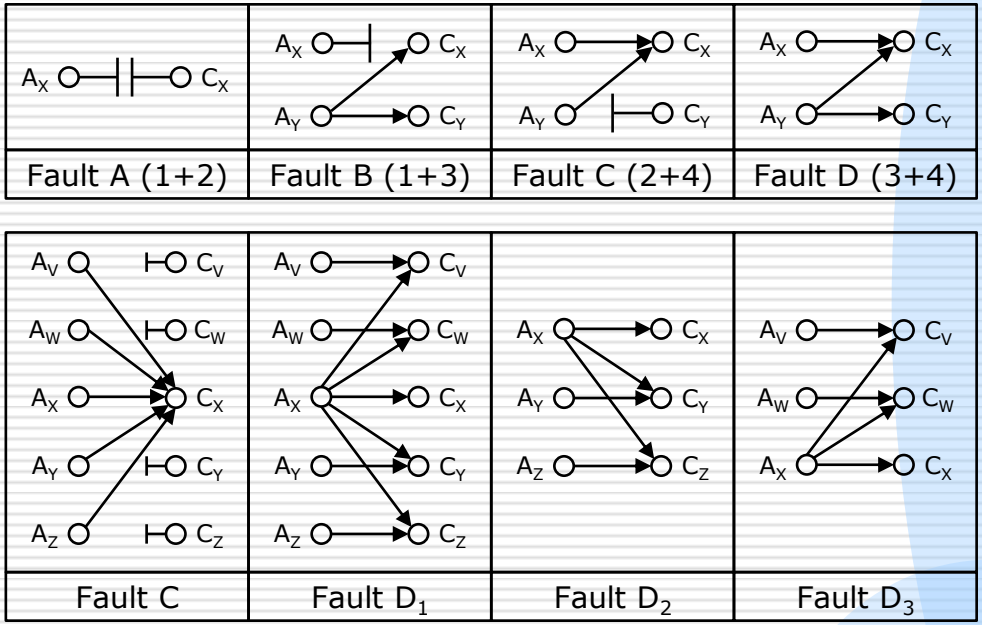
Proof Illustration



Necessity Proof

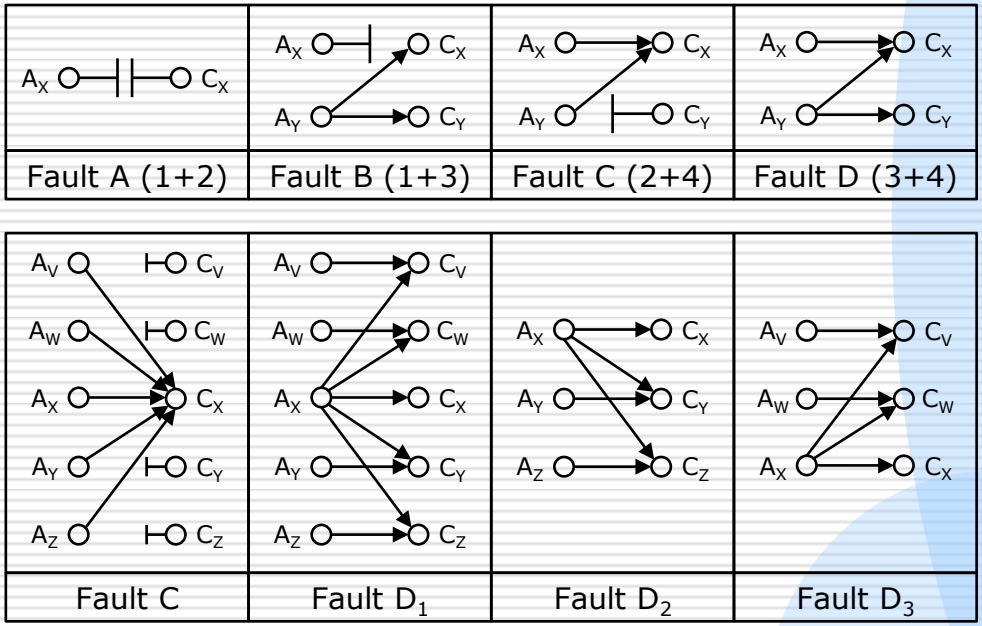
- Removing rx from Condition 1 prevents A or B fault detection when $\overline{x}$ read
- Removing $r\overline{x}$ from Condition 2 prevents A or B fault detection when x read
- Removing rx or $w\overline{x}$ from Condition 1 misses fault D2
- Removing $r\overline{x}$ or wx from condition 2 misses fault D3
- Removing both writes misses faults C and D1

Condition	March element
1	$(rx, \ldots, w\overline{x})$
2	$(r\overline{x}, \ldots, wx)$



Sufficiency Proof

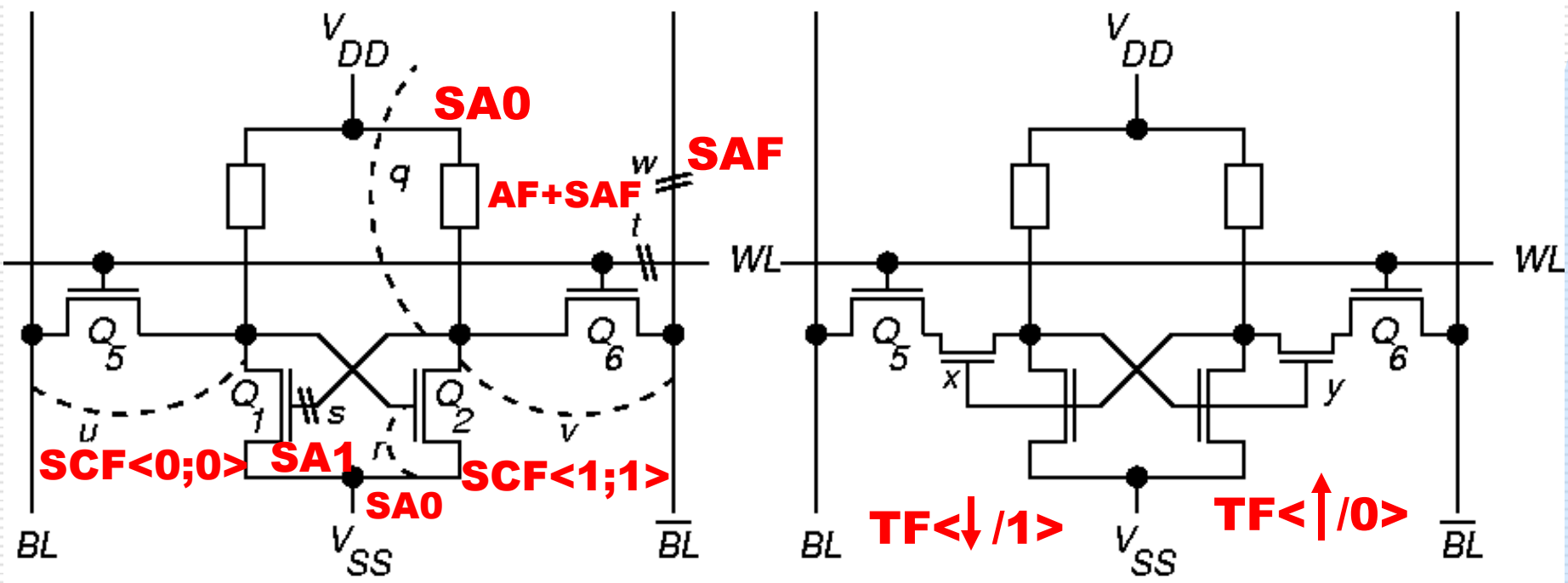
- Faults A and B: Detected by SAF test
- Fault C: Initialize memory to h (x or $\overline{x}$). Subsequent March element that reads h and writes $\overline{h}$ detects Fault C.
  - Marching ⇑ writes $\overline{h}$ to A_v. Detection: read A_w
  - Marching ⇓ writes $\overline{h}$ to A_z. Detection: read A_y
- Fault D: Memory returns random result when multiple cells read simultaneously. Generate fault by writing A_x, Detection: read A_w or A_y (⇑ or ⇓ marches)



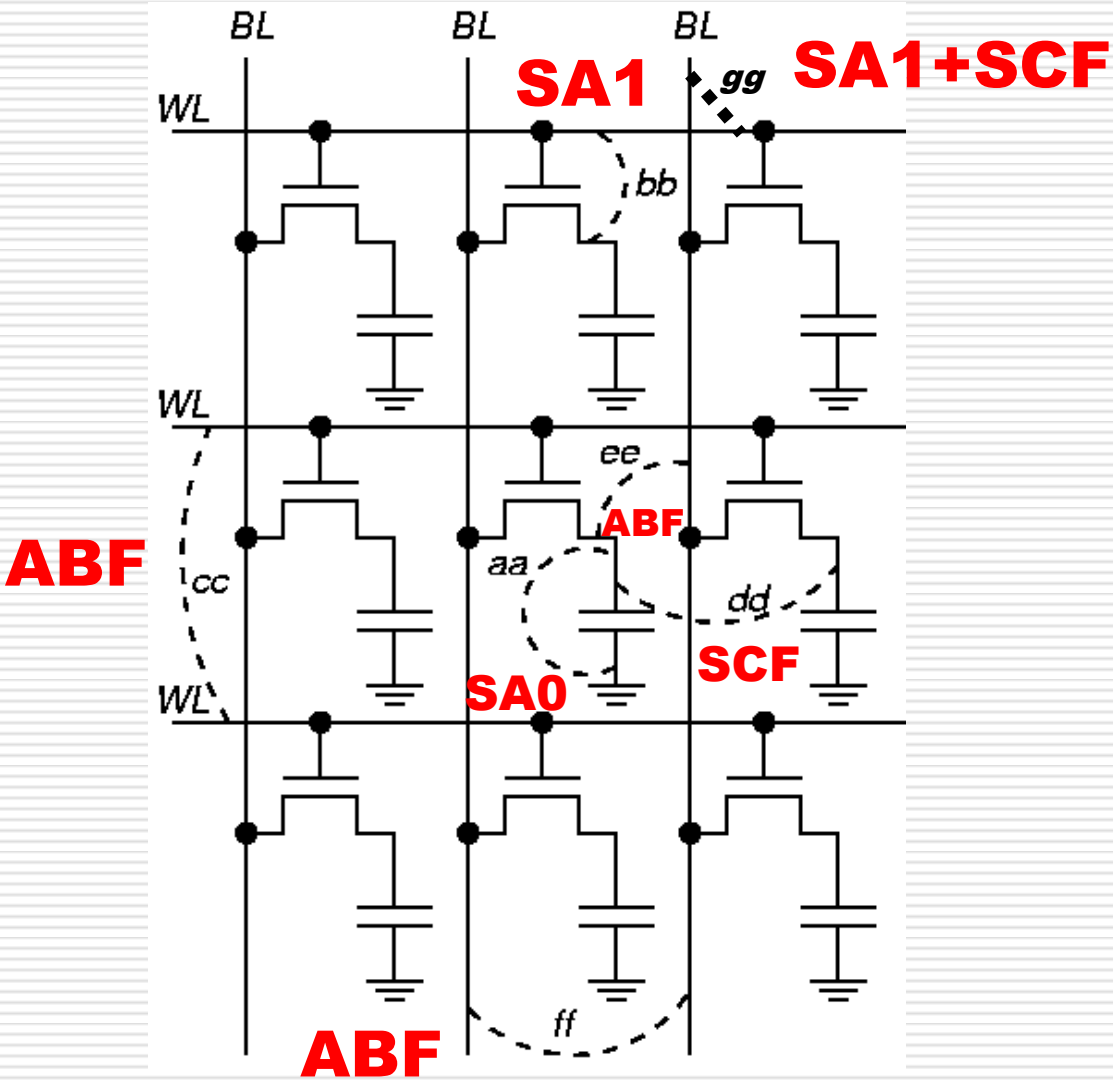
Reduced Functional Faults

Fault		Functional fault
SAF	a	Cell stuck
SAF	b	Driver stuck
SAF	c	Read/write line stuck
SAF	d	Chip-select line stuck
SAF	e	Data line stuck
SAF	f	Open circuit in data line
CF	g	Short circuit between data lines
CF	h	Crosstalk between data lines
AF	i	Address line stuck
AF	j	Open circuit in address line
AF	k	Shorts between address lines
AF	l	Open circuit in decoder
AF	m	Wrong address access
AF	n	Multiple simultaneous address access
TF	o	Cell can be set to 0 (1) but not to 1 (0)
NPSF	p	Pattern sensitive cell interaction

Fault Modeling Example 1

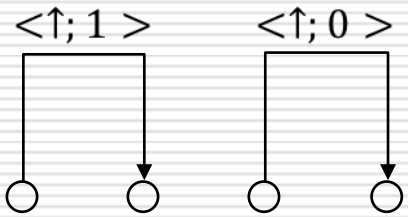


Fault Modeling Example 2

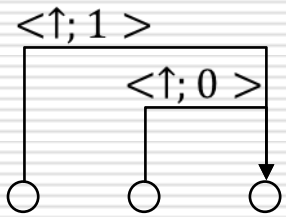


Multiple Fault Models

- Coupling Faults: In real manufacturing, any # can occur simultaneously
- Linkage: A fault influences behavior of another
- Example March test that fails:
  - $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (w0, w1); \Updownarrow (r1)\}$
  - Works only when faults not linked



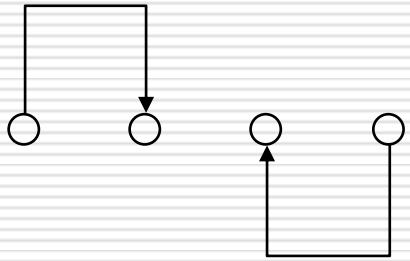
(a) Unlinked faults



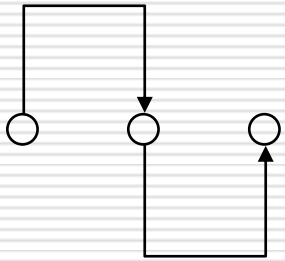
(b) Linked faults

Multiple Fault Examples

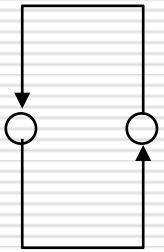
- cases 1, 2, 3 & 5 – unlinked
- cases 4 & 6 – linked



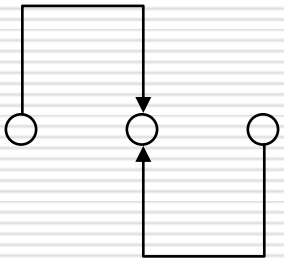
Case 1



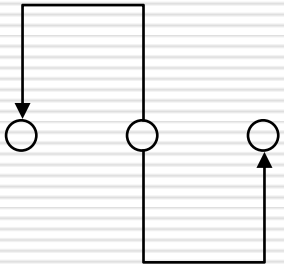
Case 2



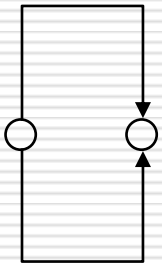
Case 3



Case 4

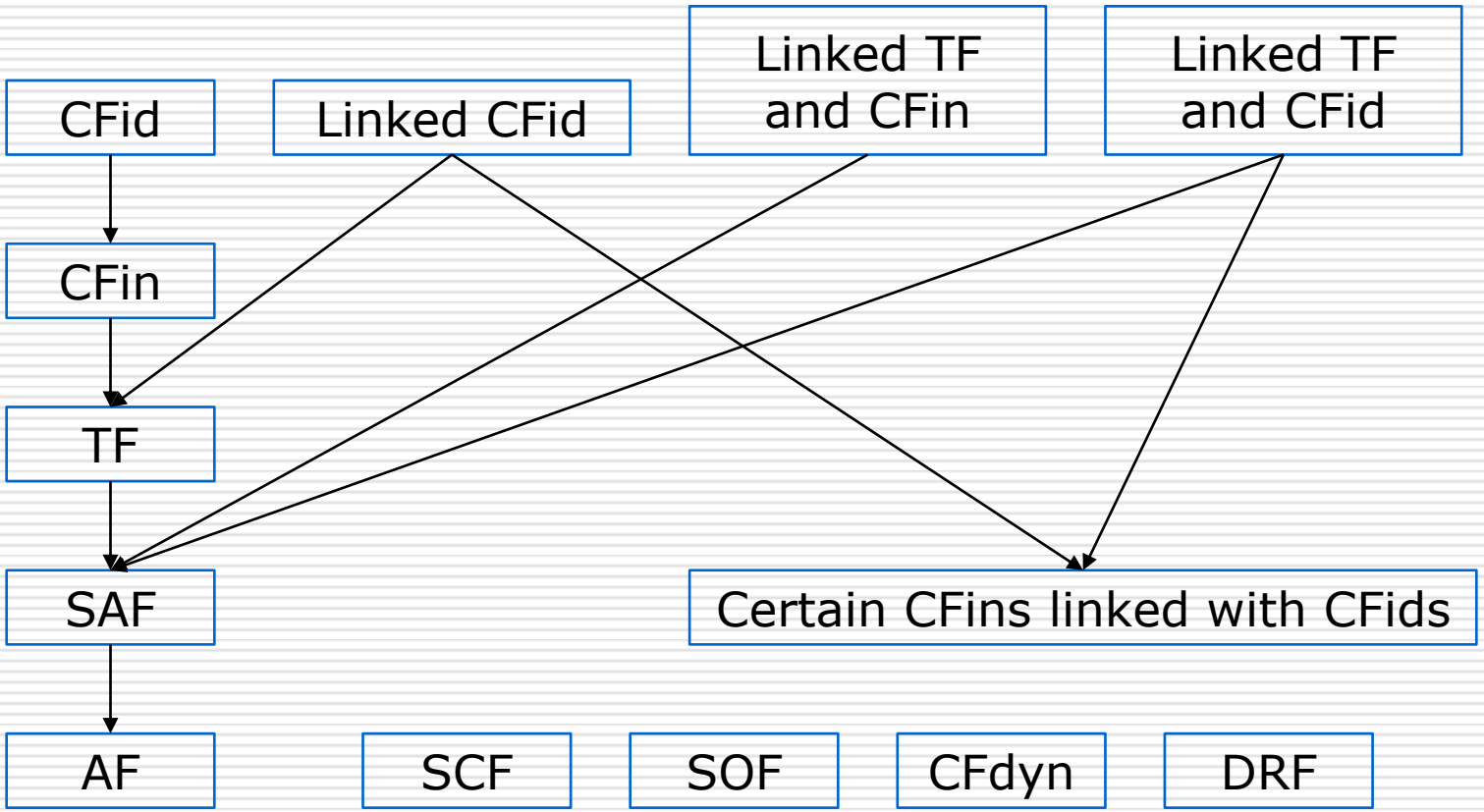


Case 5



Case 6

Fault Hierarchy



Traditional Tests

- Zero-One
- Checkerboard
- GALPAT and Walking 1/0
- Sliding Diagonal
- Butterfly

Zero-One

- This minimal test consists of writing 0s and 1s to the memory.
- This algorithm is also known under the name MSCAN (memory scan)
- Covered faults
  - some AF, SAF, some TF, some CF

Step 1: write 0 in all cells;
Step 2: read all cells;
Step 3: write 1 in all cells;
Step 4: read all cells;

Zero-One algorithm

	Step			
	1	2	3	4
Ma	w0	r0	w1	r1

Ma = all memory addresses

Zer0-One scheme

Checkerboard

- The cells of the memory are divided into two groups, cell-1 and cell-2, forming a checkerboard pattern.
- Covered faults
  - some AF, SAF, some TF, some CF

1	2	1	2
2	1	2	1
1	2	1	2
2	1	2	1

Step 1: write 1 in all cells-1 and 0 in all cell-2;
Step 2: read all cells;
Step 3: write 0 in all cells-1 and 1 in all cell-2;
Step 4: read all cells;

Checkerboard algorithm

1	0	1	0
0	1	0	1
1	0	1	0
0	1	0	1

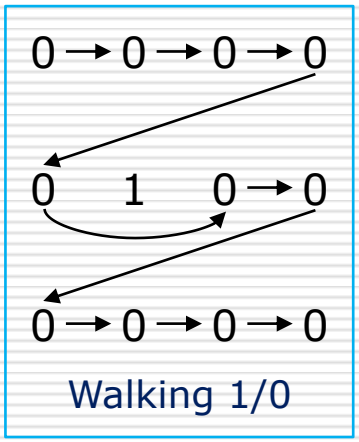
Step 1

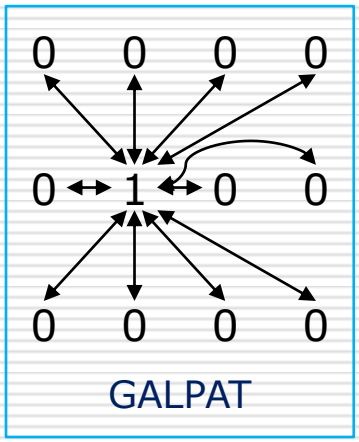
0	1	0	1
1	0	1	0
0	1	0	1
1	0	1	0

Step 3

GALPAT and Walking 1/0

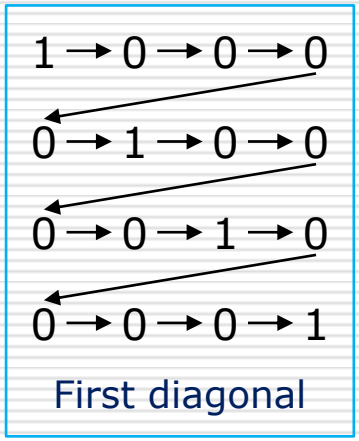
- The algorithm for GALPAT (GALloping PATtern) and Walking 1/0 are similar.
- Memory is filled with 0s (or 1s) expect for the base-cell, which contains a 1 (or 0). During the test, the base-cell walks through the memory.
- Covered faults
  - AF, SAF, TF, CF

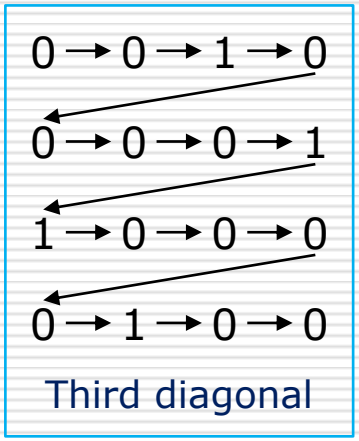




Sliding Diagonal

- The Sliding Diagonal test was developed as a shorter alternative to GALPAT; it uses a diagonal of base cells instead of a single base cell.
- Cells on the diagonal are used because each cell has a different row and column address.
- Covered faults
  - some AF, SAF, TF, some CF





Butterfly

- As in GALPAT, the memory is filled with 0s (or 1s) expect for the base cell, which contains a 1.
- During the test, the base cell walks through the memory. However, it reads cells which have the same row and column with certain distances 1, 2, 4, 8, 16, etc.
- Covered faults
  - some AF, SAF

0	N	0	0	0	0
W	b	E	E	0	W
0	S	0	0	0	0
0	S	0	0	0	0
0	0	0	0	0	0
0	N	0	0	0	0

March Test

- A march test consists of a finite sequence of march elements.
- A march element is a finite sequence of operations applied to every cell in memory before proceeding to the next cell.

March test example

$$\Updownarrow (w0); \Uparrow (r0, w1)$$

M1 M2

March element

Address	Operation
Addr 0	write 0
Addr 1	write 0
Addr 2	write 0
Addr 3	write 0
Addr 0	read 0
Addr 0	write 1
Addr 1	read 0
Addr 1	write 1
Addr 2	read 0
Addr 2	write 1
Addr 3	read 0
Addr 3	write 1

Notations for March Test

- r : read a memory location
- w : write a memory location
- r0 : read a 0 from a memory location
- r1 : read a 1 from a memory location
- w0 : write a 0 to a memory location
- w1 : write a 1 to a memory location
- ⇑ : increasing memory addressing
- ⇓ : decreasing memory addressing
- ⇕ : either increasing or decreasing

March Tests (1/2)

- MATS
  - $\{\Updownarrow(w0); \Updownarrow(r0, w1); \Updownarrow(r1)\}$
- MATS+
  - $\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0)\}$
- MATS++
  - $\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0, r0)\}$
- Marching 1/0
  - $\{\Updownarrow(w0); \Uparrow(r0, w1, r1); \Downarrow(r1, w0, r0); \Updownarrow(w1); \Uparrow(r1, w0, r0); \Downarrow(r0, w1, r1)\}$
- March X
  - $\{\Updownarrow(w0); \Uparrow(r0, w1); \Downarrow(r1, w0); \Updownarrow(r0)\}$
- March Y
  - $\{\Updownarrow(w0); \Uparrow(r0, w1, r1); \Downarrow(r1, w0, r0); \Updownarrow(r0)\}$

March Tests (2/2)

- March C
  - $\{\Updownarrow(w0); \Uparrow(r0,w1); \Uparrow(r1,w0); \Updownarrow(r0); \Downarrow(r0,w1); \Downarrow(r1,w0); \Updownarrow(r0)\}$
- March C-
  - $\{\Updownarrow(w0); \Uparrow(r0,w1); \Uparrow(r1,w0); \Downarrow(r0,w1); \Downarrow(r1,w0); \Updownarrow(r0)\}$
- March A
  - $\{\Updownarrow(w0); \Uparrow(r0,w1,w0,w1); \Uparrow(r1,w0,w1); \Downarrow(r1,w0,w1,w0); \Downarrow(r0,w1,w0)\}$
- March B
  - $\{\Updownarrow(w0); \Uparrow(r0,w1,r1,w0,r0,w1); \Uparrow(r1,w0,w1); \Downarrow(r1,w0,w1,w0); \Downarrow(r0,w1,w0)\}$

MATS+ Example (1/3)

- Cell (2,1) SA0 fault
  - MATS+ : $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$
    M1 M2 M3

Good Machine

0	0	0
0	0	0
0	0	0

after M1

1	1	1
1	1	1
1	1	1

after M2

0	0	0
0	0	0
0	0	0

after M3

Bad Machine

0	0	0
0	0	0
0	0	0

after M1

1	1	1
0	1	1
1	1	1

after M2

0	0	0
0	0	0
0	0	0

after M3

MATS+ Example (2/3)

- Cell (2,1) SA1 fault
  - MATS+ : $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$
    M1 M2 M3

Good Machine

0	0	0
0	0	0
0	0	0

after M1

1	1	1
1	1	1
1	1	1

after M2

0	0	0
0	0	0
0	0	0

after M3

Bad Machine

0	0	0
1	0	0
0	0	0

after M1

1	1	1
1	1	1
1	1	1

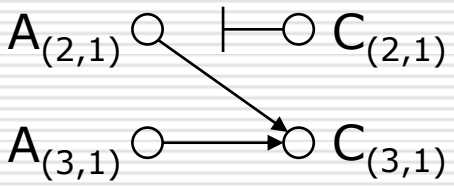
after M2

0	0	0
0	0	0
0	0	0

after M3

MATS+ Example (3/3)

- Cell (2,1) AF Type C
  - Cell (2,1) is not addressable
  - Address (2,1) maps into (3,1)
  - MATS+ : $\{\Updownarrow (w0); \Uparrow (r0, w1); \Downarrow (r1, w0)\}$
    M1 M2 M3



Good Machine

0	0	0
0	0	0
0	0	0

after M1

1	1	1
1	1	1
1	1	1

after M2

0	0	0
0	0	0
0	0	0

after M3

Bad Machine

0	0	0
X	0	0
0	0	0

after M1

1	1	1
X	0	0
1	0	0

after M2 for (2,1)

1	1	1
X	1	1
1	1	1

after M2

0	0	0
X	0	0
0	0	0

after M3

Comparison of memory tests

Algorithm	Fault coverage					Test time
	AF	SAF	TF	CF	Others	Order
Zero-One	-	L	-	-		$O(n)$
Checkerboard	-	L	-	-	Refresh	$O(n)$
Walking 1/0	L	L	L	L	SA recovery	$O(n^2)$
GALPAT	L	L	L	L	Write recovery	$O(n^2)$
Sliding Diagonal	LS	L	L	-		$O(n \cdot \sqrt{n})$
Butterfly	LS	L	-	-		$O(n \cdot log_2(n))$
MATS	DS	D	-	-		$O(n)$
MATS+	D	D	-	-		$O(n)$
MATS++	D	D	D	-		$O(n)$
Marching 1/0	D	D	D	-		$O(n)$
March X	D	D	D	D	Unlinked CFins	$O(n)$
March Y	D	D	D	D	Linked TFs	$O(n)$
March C-	D	D	D	D	Unlinked CFins	$O(n)$
March A	D	D	D	D	Unlinked CFs	$O(n)$
March B	D	D	D	D	Linked CFs	$O(n)$

L=Locate
LS=Locate Some
D=Detect
DS=Detect Some

DRAM/SRAM Fault Modeling (1/2)

DRAM or SRAM Faults	Model
Shorts & opens in memory cell array	SAF, SCF
Shorts & opens in address decoder	AF
Access time failures in address decoder	Functional
Coupling capacitances between cells	CF
Bit line shorted to word line	IDDQ
Transistor gate shorted to channel	IDDQ
Transistor stuck-open fault	SOF
Pattern sensitive fault Diode-connected transistor 2 cell short Open transistor drain Gate oxide short Bridging fault	PSF

DRAM/SRAM Fault Modeling (1/2)

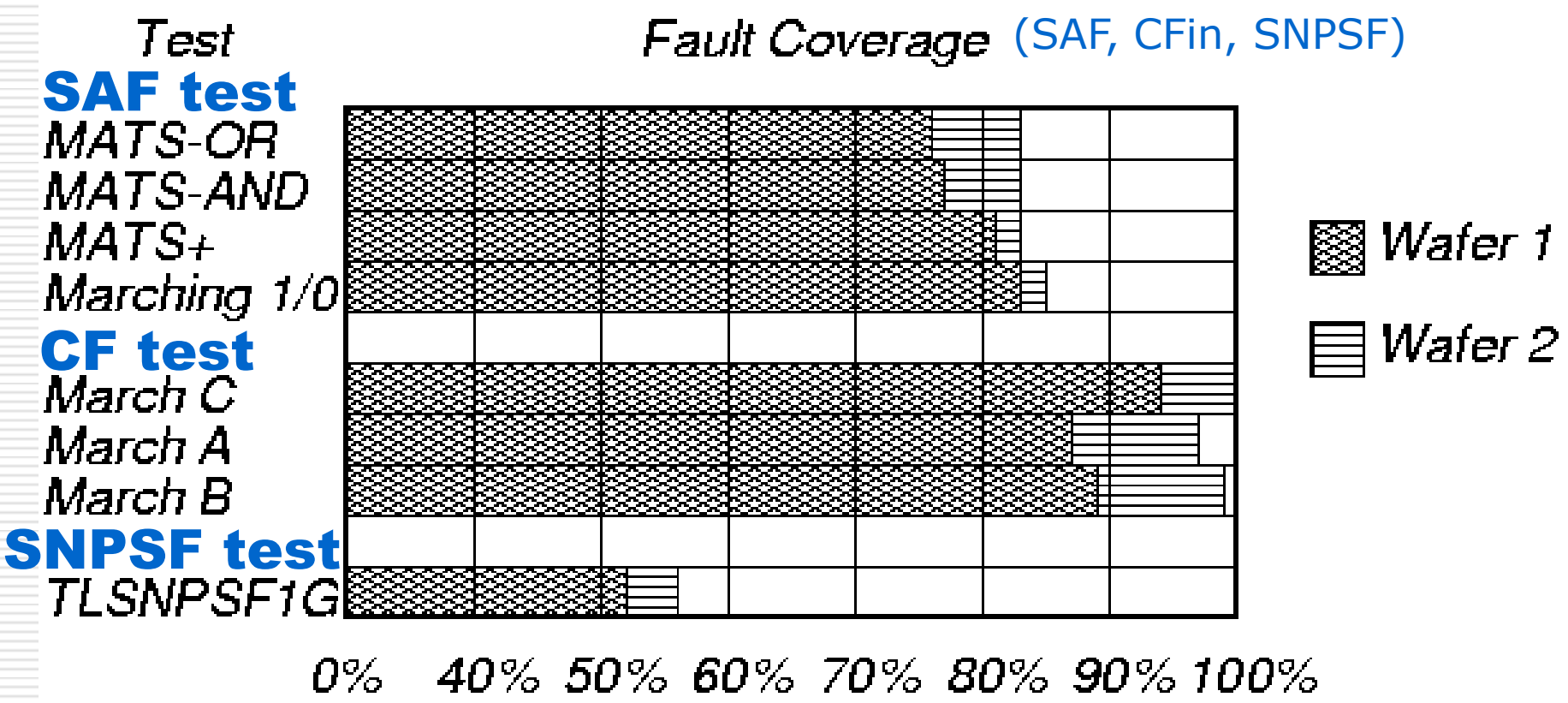
DRAM Only Fault Modeling

DRAM only Faults	Model
Data retention fault (sleeping sickness)	DRF
Refresh line stuck-at fault	SAF
Bit-line voltage imbalance fault	PSF
Coupling between word and bit line	CF
Single-ended bit-line voltage shift	PSF
Precharge and decoder clock overlap	AF

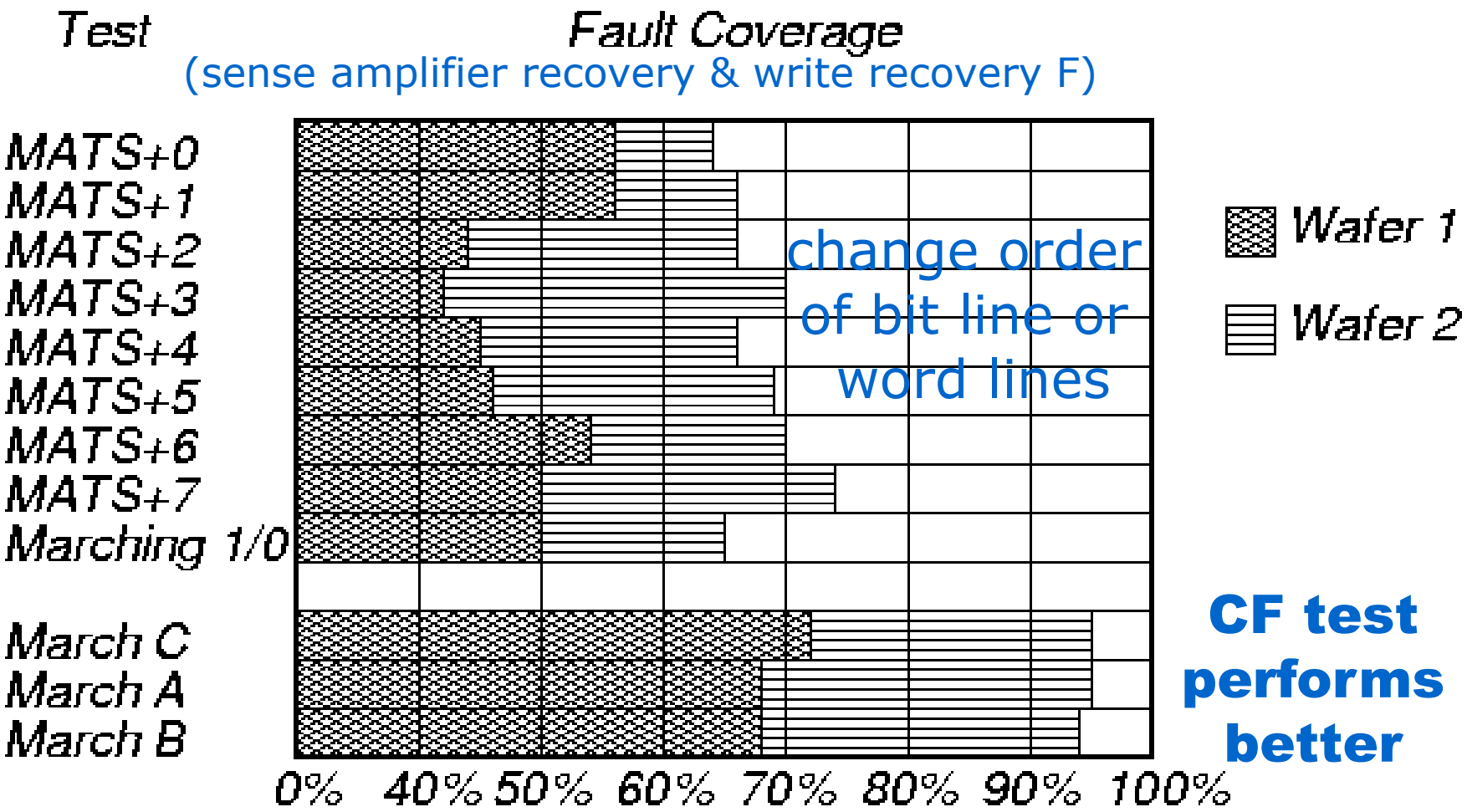
SRAM Only Fault Modeling

SRAM only Faults	Model
Open-circuited pull-up device	DRF
Excessive bit line coupling capacitance	CF

Test Influence on SRAM Fault Coverage



Influence of Addressing Order on Fault Coverage



Critical Path Length

- Length of parallel wires separated by dimension of spot defect size
- TFs and CFids happen only on long wires

Fault class	Spot defect size (μm)						
	<2	<3	<5	<7	<9	<2	<9
Stuck-at	78	213	227	269	269	51.3%	49.8%
Stuck-open	32	64	64	64	64	21.0%	11.9%
Transition	0	36	38	38	38	0%	7.0%
State coupling	15	15	51	71	71	9.9%	13.2%
Idempotent Coupling	0	0	0	0	18	0%	3.3%
Data retention	27	29	80	80	80	17.8%	14.8%
Total	152	357	460	522	540	100%	100%

Fault Frequency

- Obtained with Scanning Electron Microscope
- CFin and TF faults rarely occurred

Cluster	# Devices	Fault class
0	714	Stuck-at and Total failure
1	169	Stuck-open
2	18	Idempotent coupling
3	9	State coupling
4	8	?
5	5	?
6	26	Data retention
-	-	?
14	2	?

10% cannot be classified