Memory Testing

Mike Yang

Yangshizhan_2004@163.com

SUMMARY

- Introduction of Memory modeling
- Failure mechanisms and fault modeling
- Test algorithms for memory product

1-1:Memory Introduction

- Non-volatile Memories:
  EPROM,EEPROM & Flash memory.
- Volatile Memories : RAM ,SRAM,DRAM.
- Memories are the most numerous IPs used in SOC designs .

1-2:Main types of semiconductor memory

- Electrically Erasable Programmable Read-Only Memory (EEPROM)
  - The data can be a just erased a byte.
  - Long write/read data time.
- Flash Memory
  - *The data is erased in large blocks.*
  - *Fastest.*

1-3: Main types of semiconductor memory

- Dynamic Random Acess Memory (DRAM)
  - Highest possible density
  - Slow access time (typically 20ns)
  - Information stored as charge on capacitor and should be refreshed
- Static Random Acess Memory (SRAM)
  - Fastest (typically 2ns)
  - Information stored in cross coupled latches

2-1: Failure, errors and faults

- A system failure occurs when the system behaviour is incorrect
- Failures are cause by errors
- An error is a difference between the faulty value and the golden one, this is the manifestation of a fault
- A fault represents the physical difference between a good and incorrect system
- Faults can be permanent or non-permanent

2-2: Failure Mechanisms

- Corrosion
- Electromigration (burning out of wires due to collision of electrons and Al grains)
- Bonding deterioration (open due to interdiffusion of materials i.e. Au-Al)
- Ionic contamination (modification of threshold voltages due to ion diffusion into transistor gate)
- Alloying (Al atom migration into Si)
- Radiation and cosmic rays (soft memory errors, …)
- …….

2-3:Functional faults

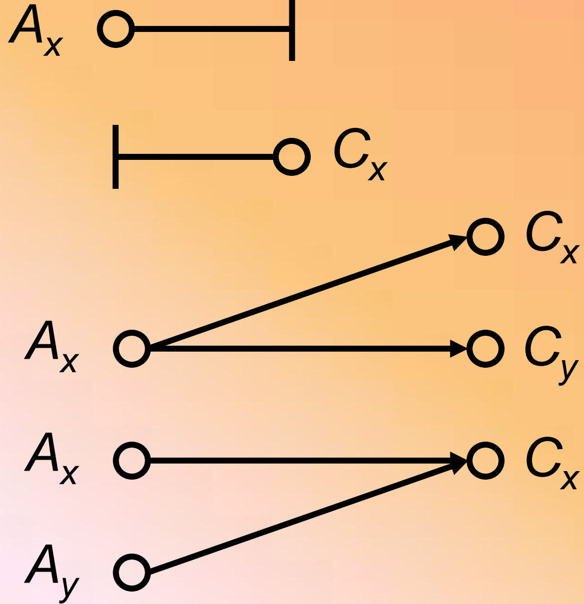
Either a memory cell or a data register

Any wiring connection in the memory

- Cell stuck
- Driver stuck
- Read/write line stuck
- Chip-select line stuck
- Data line stuck
- Open in data line
- Shorts between data lines
- Crosstalk between data lines
- Address line stuck
- Open in address line
- Shorts between address lines
- Open decoder
- Wrong access
- Multiple access
- Cell can be set to 0 and not to 1 (or vice versa)
- Pattern sensitive interaction between cells

2-4: Address decoder Fault AF

- 1. A certain address access no cell
- 2. A certain cell is never accessed
- 3. A certain address access multiple cells
- 4. A certain cell is accessed by multiple addresses



2-5: Address decoder Fault AF

4 Combinations of Address decoder Faults

- Fault A: 1+2

- Fault B: 1+3

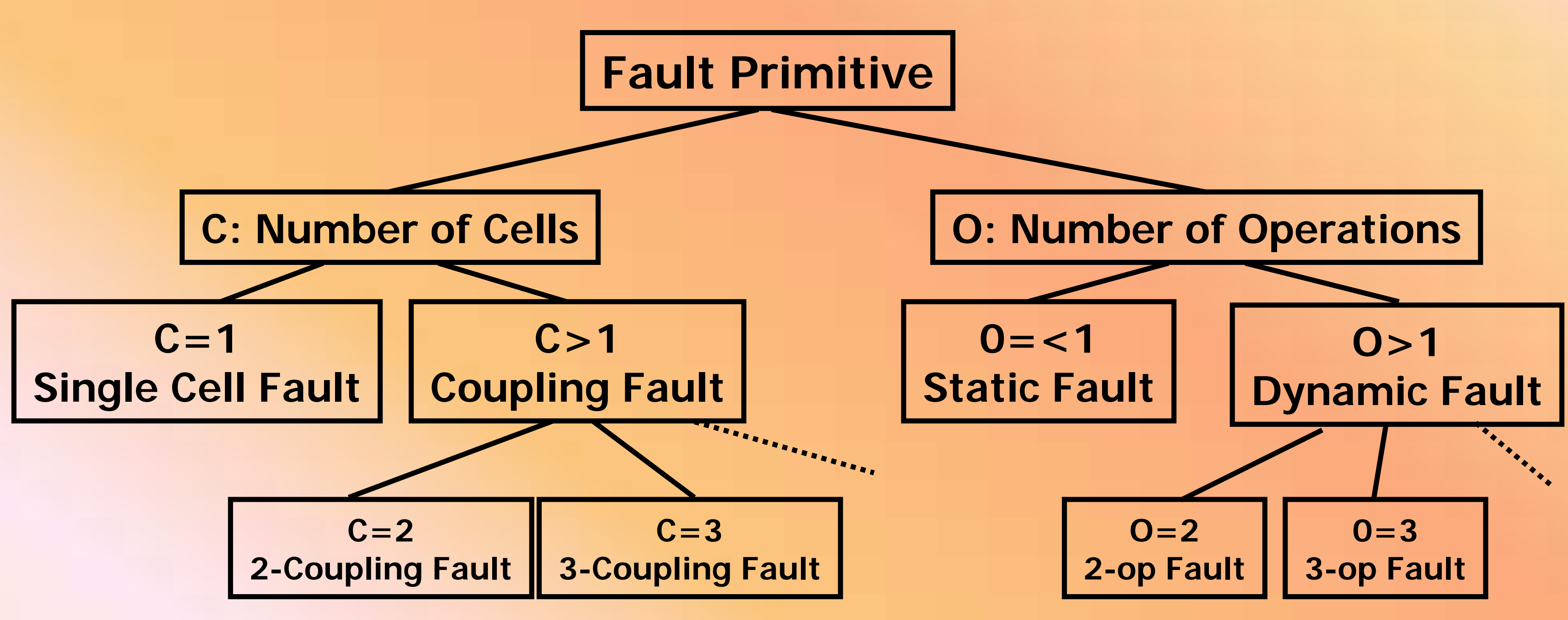
- Fault C: 2+4

- Fault D: 3+4

2-6:Functional cell Faults

- A functional fault model is a non-empty set of fault primitives
- A fault primitive is a difference between the observed and the expected memory behaviour
- A fault primitive is denoted by <S/F/R>
  - S describes the sensitizing operation sequence
  - F describes the state stored in the faulty cell
  - R describes the output of the read operation
  - For example <1/0/-> is a stuck-at 1 and <0w1/0/-> is a rising transition fault

2-7:Taxonomy of Functional Memory Faults



2-8: Static and Dynamic faults

- Static faults: sensitisation needs only one operation (Testable by common March Tests)
  - Static Single-Cell Faults (S1CF)
  - Static Two-Cell Faults (S2CF)
  - …
- Dynamic faults: sensitisation needs more than one operation (Not testable by common March Tests)
  - Dynamic Single-Cell Faults (D1CF)
  - Dynamic Two-Cell Faults (D2CF)
  - …

2-9: Stuck-at fault (SAF)

- The logic value of (a line or) a cell is always 0 (SA0) or 1 (SA1)
- To detect memory cell's SAFs:
  - SA0: Write 1 Read 1 (w1 r1)
  - SA1: Write 0 Read 0 (w0 r0)

2-10: Transition Fault

- A cell fails to undergo a 0 $\rightarrow$ 1 transition (TF_{rise}) or a 1 $\rightarrow$ 0 transition (TF_{fall}) when it is written
- To detect transition fault:
  - TF_{rise} : w0 w1 r1
  - TF_{fall} : w1 w0 r0

2-11:Read Disturb Faults (RDF)

- A Cell is said to have a RDF if the read operation performed on the cell returns an incorrect value while changing the contents of the cell to the wrong value
- To detect Read Disturb Fault from each cell a 1 and a 0 should be read
  - r0
  - r1

2-12: Deceptive Read Disturb Faults (DRDF)

- A Cell is said to have a DRDF if the read operation performed on the cell returns the expected value while changing the contents of the cell to the wrong value
- To detect Deceptive Read Disturb Fault each cell should be read twice successively. The first read sensitize the fault and the second detects it
  - r0r0
  - r1r1

2-13: Incorrect Read Faults (IRF)

- A Cell is said to have a IRF if a read operation performed on the cell returns the incorrect value while keeping the correct stored value in the cell
- To detect Incorrect Read Fault from each cell a 1 and a 0 should be read
  - r0
  - r1

2-14: Write Disturb Faults (WDF)

- A Cell is said to have a WDF if a non transition write operation causes a transition in the cell
- To detect Write Disturb Fault each cell should be read after a non-transition write
  - 0w0r0
  - 1w1r1

2-15: Coupling Fault (2 cells)

- Implies two cells: the victim cell and the aggressor cell
- Different kinds of coupling faults:
  - Inversion coupling faults
  - Idempotent coupling faults
  - State coupling faults
  - Dynamic coupling faults
  - Bridging faults
  - ……..

2-16: Inversion Coupling Fault

- Inversion Coupling Fault (CF_{in}): The content of the victim cell is inverted if the aggressor cell has a transition
- According to the kind of transition ($0 \rightarrow 1$ or $1 \rightarrow 0$) there is two possible CF_{in} types:

$$<\uparrow ; \updownarrow> \quad <\downarrow ; \updownarrow>$$

- To detect CF_{in} between cell x (victim) and y (aggressor)
  - CF_{in} (y rise $\rightarrow$ x inverted): w0x w0y w1y r0x.
  - CF_{in} (y fall $\rightarrow$ x inverted): w0x w1y w0y r0x.

2-17: Idempotent Coupling Fault

- Idempotent Coupling Fault (CF_{id}): The victim is forced to 0 or 1 if the aggressor has a 0 $\rightarrow$ 1 or 1 $\rightarrow$ 0 transition
- According to the kind of transition (0 $\rightarrow$ 1 or 1 $\rightarrow$ 0) there is four possible CF_{id} types:

  $$< \uparrow ; 0 > < \downarrow ; 0 > < \uparrow ; 1 > < \downarrow ; 1 >$$

- To detect CF_{id} between cell x (victim) and cell y (aggressor)
  - CF_{id} (y rise $\rightarrow$ x=0): w1x w0y w1y r1x
  - CF_{id} (y fall $\rightarrow$ x=1): w0x w1y w0y r0x
  - CF_{id} (y rise $\rightarrow$ x=1): w0x w0y w1y r0x
  - CF_{id} (y fall $\rightarrow$ x=0): w1x w1y w0y r1x

2-18: State Coupling Fault

- State Coupling Fault (CF_{st}): The coupled cell (victim) is forced to 0 or 1 if the coupling cell (aggressor) is in a certain state
- There is four possible CF_{st} types:

  $< 0 ; 0 > < 0 ; 1 > < 1 ; 0 > < 1 ; 1 >$

- To detect CF_{st} between cell x (victim) and y (aggressor)
  - CF_{st} ($y=0 \rightarrow x=0$): w1x w0y r1x
  - CF_{st} ($y=0 \rightarrow x=1$): w0x w0y r0x
  - CF_{st} ($y=1 \rightarrow x=0$): w1x w1y r1x
  - CF_{st} ($y=1 \rightarrow x=1$): w0x w1y r0x

2-19: Dynamic Coupling Fault

- Dynamic Coupling Fault (CF_{dyn}): The victim is forced to 0 or 1 if the aggressor cell has a read or write operation
- More general case of the Idempotent Coupling Fault (CF_{id}) because it can be sensitized by any read or write operation
- There are four CF_{dyn} faults

  <r0 | w0;0> <r0 | w0;1> <r1 | w1;0> <r1 | w1;1>

2-20: Write (read) disturb coupling fault

- Write disturb coupling (CF_{wd}): A cell is said to have a CFwd if a non transition write perform on the victim results in a transition when the aggressor is set into a logic state
- There are four types of CF_{wd} faults
- Read disturb coupling (CF_{rd}): Two cells are said to have a CFrd if a read performed on the victim destroys the data stored in the victim if a given state is present in the aggressor
- There are four types of CF_{rd} faults

2-21: Incorrect read coupling fault

- Incorrect read coupling fault (CF_{ir}): Two cells are said to have an CFir if a read performed on the victim returns the incorrect logic value when the aggressor is sent into a given state
- There are four types of CF_{ir} faults

2-22: Deceptive read disturb coupling fault

- Deceptive read disturb coupling fault (CF_{dr}): A cells is said to have an CFdr if a read performed on the victim returns the correct logic value and changes the content of the victim, when the aggressor is sent into a given state
- There are four types of CF_{dr} faults

2-23: Bridging Faults (BF)

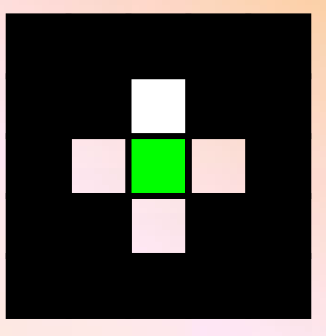
- A short between cells or lines (2 or more)
- This is a bidirectional fault caused by logic level rather than a transition
- Two sorts of bridging faults (BFs) exist:
  - AND-type (ABF): the shorted cells/lines take the AND value of their fault-free values (four possible ABFs)
  - OR-type (OBF): the shorted cells/lines take the OR value of their fault-free values (four possible OBFs)
- Can be made equivalent to a number of linked CFs, i.e.
  - "Cell x is the victim of cell y" and "Cell y is the victim of cell x"
  - The faults are linked. It is possible that they will hide each other.
- To detect a BF, it might be necessary to write a certain pattern on adjacent memory cells, see checkerboard algorithm using (“010101…”) pattern.

2-24: Pattern Sensitive (PSF)

- The victim cell is forced to 0 or 1 if a certain number of neighbors show a particular pattern
- The PSF is the most general k-coupling fault with k=n
- With the Neighborhood Pattern Sensitive Fault (NPSF), the neighborhood is limited to all the cells in a single position surrounding the base cell
- Equivalent to an N-coupling fault involving more than one aggressor (up to 8 adjacent locations)
- Extremely hard to detect
  - For each memory cell: the effect of all the possible combinations (2^8) of the adjacent cells should be tested.

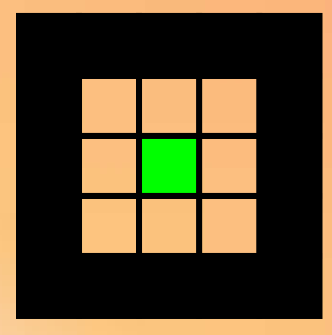
2-25: Neighborhood Pattern Sensitive (NPSF)

- In practice two types of NPSF are used
- The type-1 NPSF with 4 neighborhood cells (north, west, south, east)
- The type-2 NPSF with 8 neighborhood cells



Base cell
Type-1 neighborhood cell

Type-1



Base cell
Type-2 neighborhood cell

Type-2

2-26:Neighborhood Pattern Sensitive (NPSF)

- Active NPSF (ANPSF):
  - change of the base cell due to a transition in the neighborhood cells
  - Each base cell must be read in state 0 and in state 1, for all possible changes in the neighborhood pattern
- Passive NPSF (PNPSF):
  - the change of the base cell is impossible due to a certain neighborhood cells configuration
  - Each base cell must be written and read in state 0 and in state 1, for all permutations in the neighborhood pattern
- Static NPSF (SNPSF):
  - The content of the base cell is forced to a certain state due to a certain neighborhood pattern
  - Each base cell must be read in state 0 and in state 1, for all permutations in the neighborhood pattern

2-27: Dynamic Fault models: examples

- Sense Amplifier (SA) Recovery Fault
  - The SA saturates after a long sequence of 0s or 1s
- Write Recovery Fault (addressing fault in the decoder)
  - Write followed by a read/write to another location affects the previously accessed location
- Detection needs At-Speed testing !!

2-28: Single-Cell Dynamic Fault Models: Dynamic RDF, IRF and DRDF

Definition similar to the static Read Disturb Fault, Incorrect Read Fault and Deceptive Read Disturb Faults with an initial write follows by one or more read operation

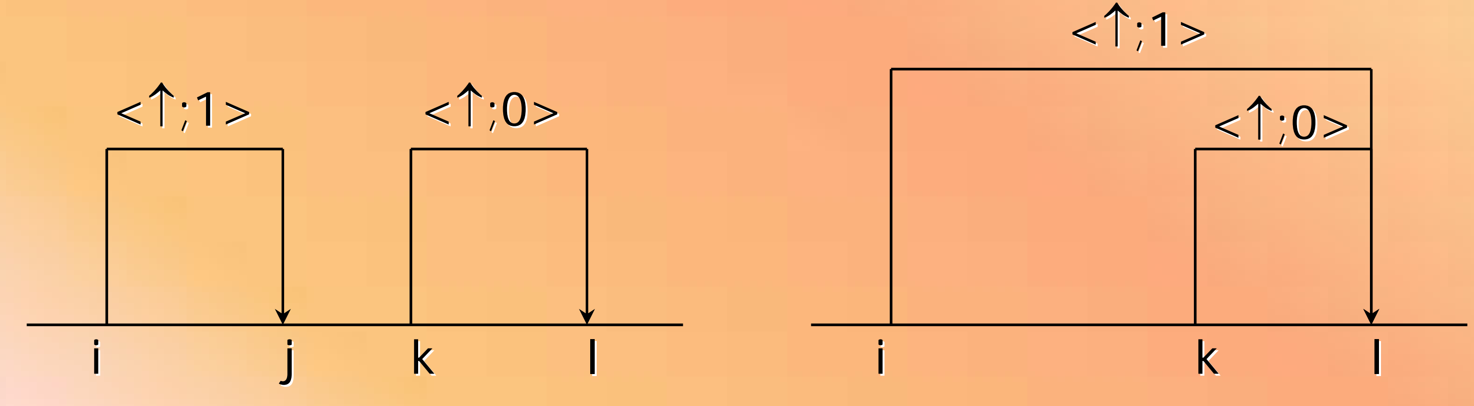
- dRDF: A Cell is said to have a dRDF if a write followed by one or more read performed on the cell returns (on the last read) an incorrect value while changing the contents of the cell to the wrong value
- dIRF: A Cell is said to have a dIRF if a write followed by one or more read performed on the cell returns (on the last read) incorrect value while keeping the correct stored value in the cell
- dDRDF: A Cell is said to have a dDRRF if a write followed by one or more read performed on the cell returns (on the last read) the expected value while changing the contents of the cell to the wrong value

2-29: Data Retention (DRF)

- A cell fails to retain its logic value after some time
- Due to a defective pull-up within a cell
- A cell loses its value due to leakage current: the cell lose its charge due to this leakage current
- Two different DRFs exist (loss of 1 and loss of 0) and may coexist
- To detect DRF => a delay has to be inserted before reading back memory content (RAM,usually ~ 10-100 ms)
- Can be easily add to any test algorithm
- Test time increase dramatically !

2-30: linked faults

- Linked faults are two or more faults (coupling faults) that affect the same cell



- Can be of the same type (i.e. CF x->y and CF z->y) or of different types (i.e. CF x->y and TF_{rise} y)

- They can hide each other (*fault masking*): Extremely difficult to detect

2-31:Relation between functional faults and fault models

- Cell stuck
- Driver stuck
- Read/write line stuck
- Chip-select line stuck
- Data line stuck
- Open in data line

SAF

- Shorts between data lines
- Crosstalk between data lines

CF

- Address line stuck
- Open in address line
- Shorts between address lines
- Open decoder
- Wrong access
- Multiple access

AF

- Cell can be set to 0 and not to 1 (or vice versa)

TF

- Pattern sensitive interaction between cells

NPSF

2-32:Relation between functional faults and fault models (examples)

- 1, 2, 3: SA0
- 4: AF for cells after the open
- 5, 6: State Coupling Faults
- 7: SAF (depending on the cell content and the technology)

2-33:Validity of fault models

- Use of Inductive Fault Analysis at the layout level
  - Generate defect sizes, location and layers (according to what may really happen in the fab.)
  - Place the defect on a model of the layout
  - Extract the schematic and electrical parameters for the defective cell
  - Deduce and check possible fault models
- Results depend on the used technology and fab. processes

Spot Size	<2µm	<9µm
SAF	51,3%	49,8%
SOF	21,0%	11,9%
TF	0%	7,0%
SCF	9,9%	13,2%
ICF	0%	3,3%

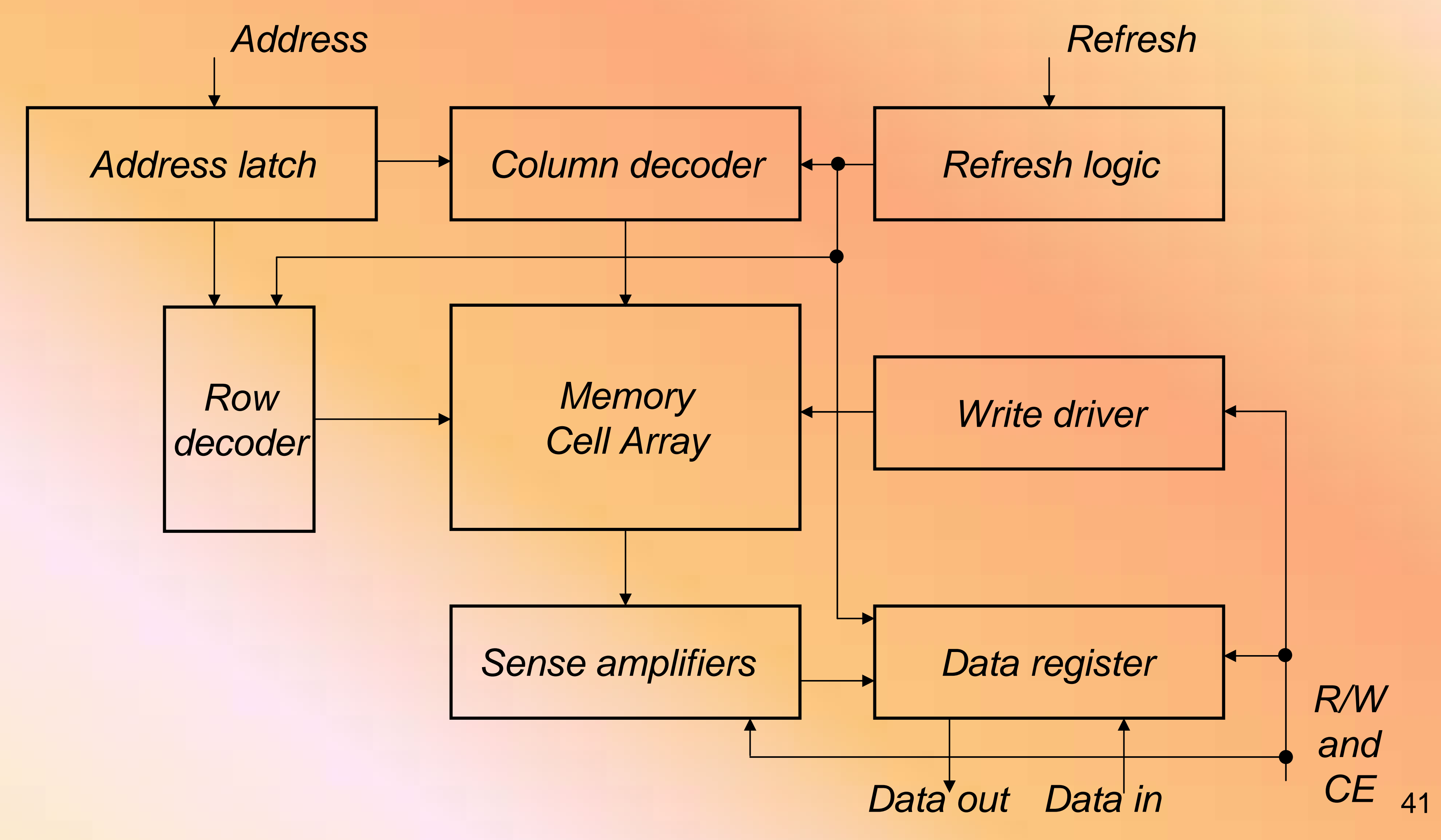
3-1:Memory testing

- Memory testing has to prove that the circuits under test behave as designed, it consists of:
  - Parametric tests which concern voltage/current levels and delays on the IO pins of the chip
  - Functional testing including dynamic testing

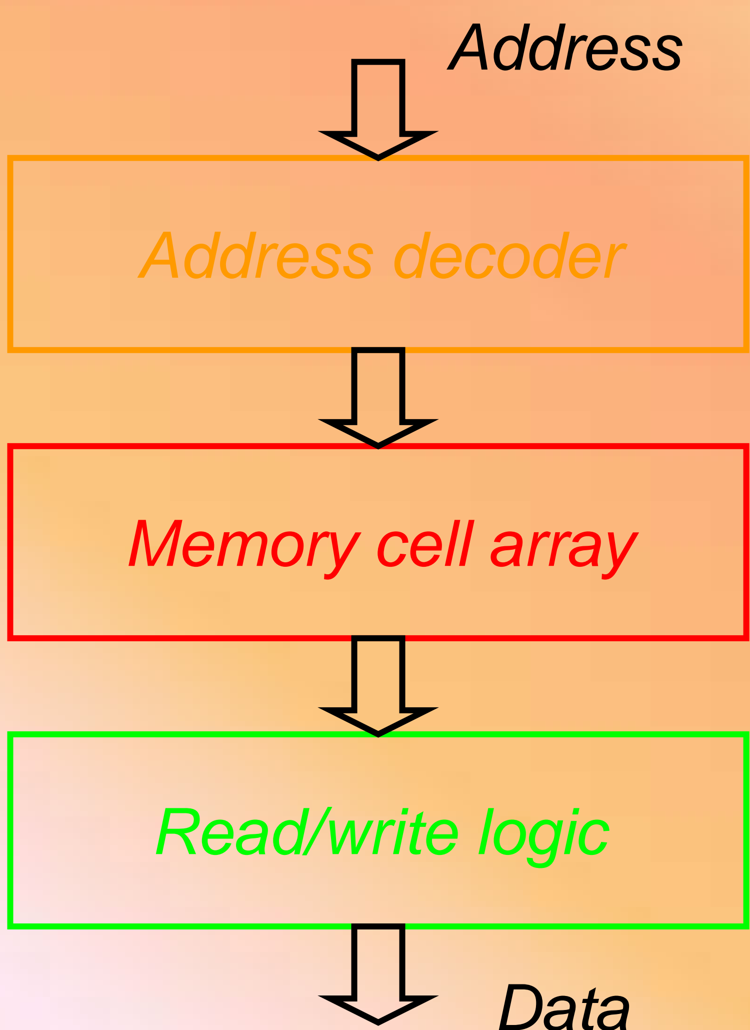
3-2:Parametric (electrical) testing

- DC Parametric testing
  - Contact test(o/s)
  - Power consumption(Idd)
  - Leakage test
  - Threshold test
  - Output drive current test
  - Output short current test
- AC Parametric testing
  - Rise and fall time
  - Setup and hold time
  - Delay test
  - Speed test

3-3:Example: memory structure



3-4:Functional Model



3-5:Test Algorithms: notations used

- ↑ : indicates address ascending order
- ↓ : indicates address descending order
- w0 : write 0 at current location
- w1 : write 1 at current location
- r0 : read current location, expecting a 0
- r1 : read current location, expecting a 1
- (….): algorithm element
- {(…),(…),…,(…)}: full algorithm

3-6: Test algorithms

- Full Behavioural test is definitively too consuming ($3.n.2^n$)
- Classical and early memory testing methods with test time proportional to
  - n (zero-one, checkerboard, …)
  - n^2 and $n.\log_2(n)$ (walking1/0, ping-pong, Galpat, Galcol, …)

3-7: Test algorithm: Zero-One

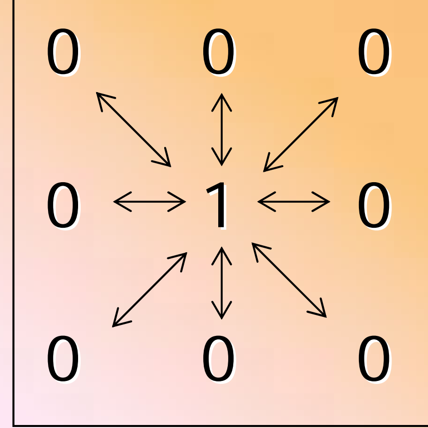
- This minimal test consists of writing 0s and 1s in the memory
  - Step1: write 0 in all cells
  - Step2: read all cells (0 expected)
  - Step3: write 1 in all cells
  - Step4: read all cells (1 expected)
- *O(n)* test
- Fault coverage:
  - Not all AFs detected
  - SAFs detected if the adress decoder is fault free
  - Not all TFs and CFs detected

3-8: Test algorithm: Checkerboard

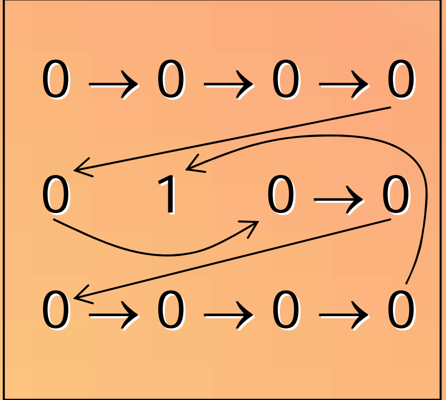
- Cell are divided in two groups
  - Step1: write 1 in all green cells and 0 in pink cells
  - Step2: read all cells
  - Step3: write 0 in all green cells and 1 in pink cells
  - Step4: read all cells
- *O(n)* test
- Fault coverage:
  - Not all AFs detected
  - SAFs detected if the adress decoder is fault free
  - Not all TFs and CFs detected
- This test is able to detect bridging faults

3-9: Test algorithms: GALPAT and Walking 1/0

- Memory is filled with 0s (or 1s) except for the base-cell which contains a 1 (0)
- During the test the base cell walks through the memory
- Difference between GALPAT and W 1/0 is in reading the base cell

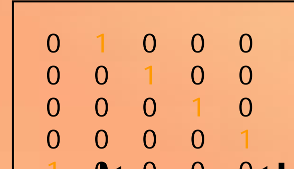


GALPAT



Walking 1/0

3-10:Test algorithms: GALPAT and Walking 1/0



- All AFs are detected and located
- All SAFs are detected and located
- All TFs are detected and located
- All CFs are also detected and located
- But both are *$O(n^2)$ tests*
- other tests have been proposed as a shorter alternative $(0(n^{3/2})$:
  - Sliding diagonal
  - Butterfly (only neighboorhood cells of the base cell are read)
  - GALCOL

3-11:Test algorithms test time

	Number of operations		
n	n	$n.\log_2 n$	n^2 (hr!!)
1Mb	0.063	1.26	18.33
16Mb	1.01	24.16	4691.3
256Mb	16.11	451	1200959.9
2Gb	128.9	3994.4	76861433.7

3-12: March tests

- The test is "marching" through the memory
- The test is composed of March elements represented between (0,1)
- March tests are the simplest tests (optimal ?) to detect most of the functional faults

3-13: March tests: example of MATS++

$\{\uparrow(w0); \uparrow(r0,w1); \downarrow(r1,w0,r0)\}$

- For i=0 to n-1
  - Write 0 in cell C_i
- For i=0 to n-1
  - Read cell C_i and check its content (0 expected)
  - Write 0 in cell C_i
- For i=n-1 to 0
  - Read cell C_i and check its content (1 expected)
  - Write 0 in cell C_i
  - Read cell C_i and check its content (0 expected)

3-14: March tests: example of MATS++

{↑(w0); ↑(r0,w1); ↓(r1,w0,r0)}

0 1

- Takes 6n operations when used bit-wise
- All SAFs are detected
- All AFs unlinked with TFs are detected
- All AFs linked with TFs are detected
- All TFs are detected

3-15:Testing of Neighborhood Pattern-Sensitive Faults

- Active NPSF (ANPSF):
  - Each base cell must be read in state 0 and in state 1, for all possible transitions in the neighborhood pattern
- Passive NPSF (PNPSF):
  - Each base cell must be written and read in state 0 and in state 1, for all permutations in the neighborhood pattern
- Static NPSF (SNPSF):
  - Each base cell must be read in state 0 and in state 1, for all permutations in the neighborhood pattern

3-16: Testing of Neighborhood Pattern-Sensitive Faults

It is essential to minimize the number of writes during NPSF testing

- SNPSF patterns are produced following an Hamiltonian sequence (hamiltonian distance of 1 between patterns, Gray code for example): $k+2^k-1$ writes
- ANPSF and PNPSF patterns are produced following an Eulerian sequence : $k+k.2^k$ writes

3-17: Test of Word-Oriented Memories

- For SAFs and TFs which involve only one cell use data background and its complement instead of 0 and 1
- MATS+ for a 4 bit memory

Data background = 0101

{↑(w0101); ↑(r0101,w1010); ↓(r1010,w0101)}

3-18: Test of Word-Oriented Memories

- For CFs involving cells in different words use data background and its complement instead of 0 and 1
- For CFs involving cells in the same word:
  - If the write operation dominates the CF, no problem
  - If the CF dominates the write operation, CF_{in} are detected but for CFid data background has to be replaced by B data background
- For SCFs and BFs data background has also to be replaced by $\log_2 B+1$ data background

Thank You