Transparent BIST for RAMs

Jin-Fu Li
Advanced Reliable Systems (ARES) Lab.
Department of Electrical Engineering
National Central University
Jhongli, Taiwan

Outline

- Introduction
- Concept of Transparent Test
- Transparent Test Techniques
- Conclusions

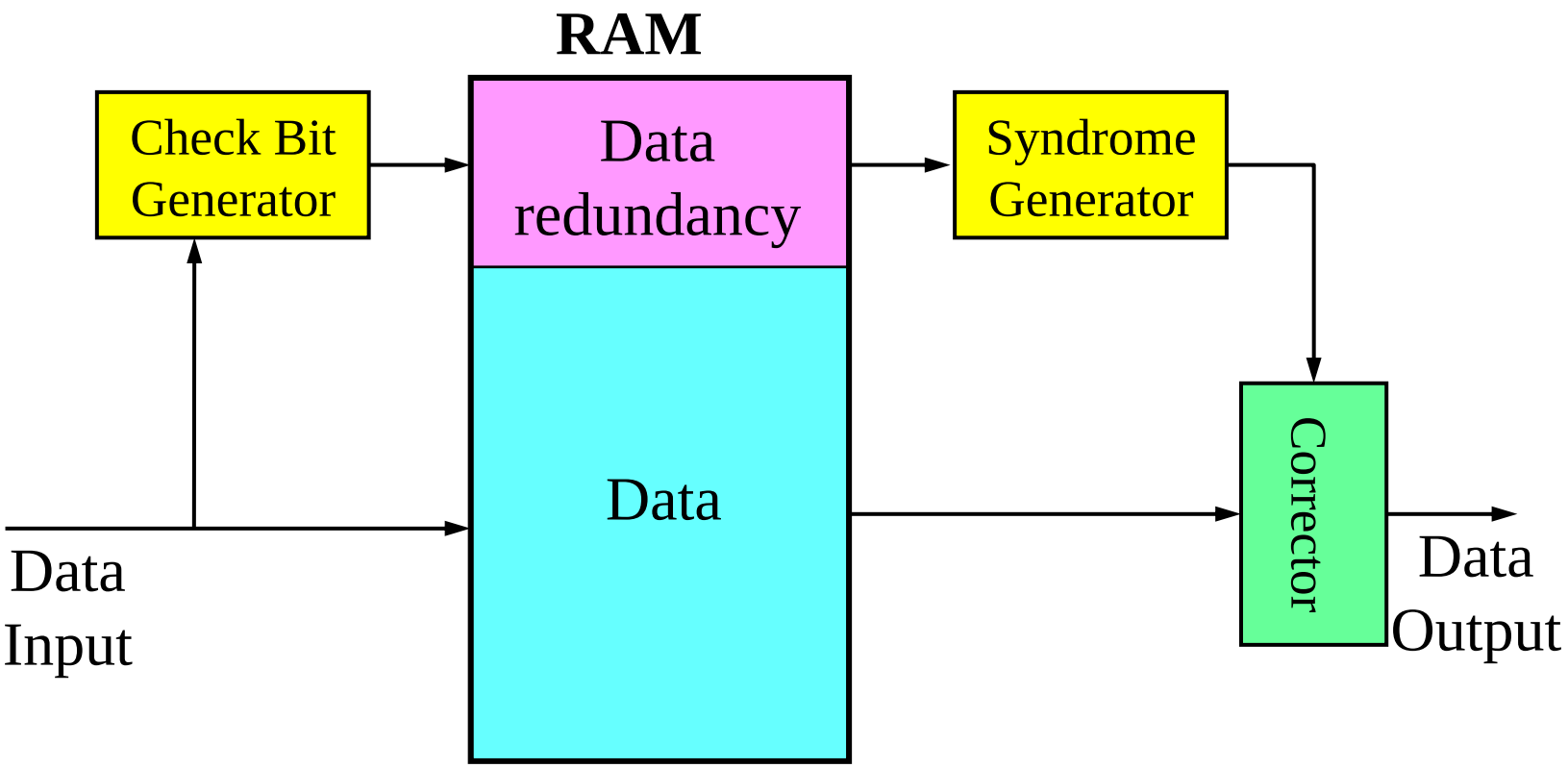
Reliability-Enhancement Techniques

- Fault-tolerant techniques are widely used to improve the reliability of systems
- All fault-tolerant techniques require redundancy
  - Redundancy is simply the addition of information, resources, or time beyond what is needed for normal system operation
- Types of redundancy
  - Hardware redundancy
  - Software redundancy
  - Information redundancy
  - Time redundancy

Memory Reliability-Enhancement Techniques

- Hardware redundancy
  - Built-in self-repair technique
- Error correction code
  - Use information redundancy to protect stored data from soft error
- Periodic transparent testing
  - Periodically apply tests to detect hard faults manifested by latent faults

Typical Error-Correction-Code Scheme



Hamming Error-Correction Code

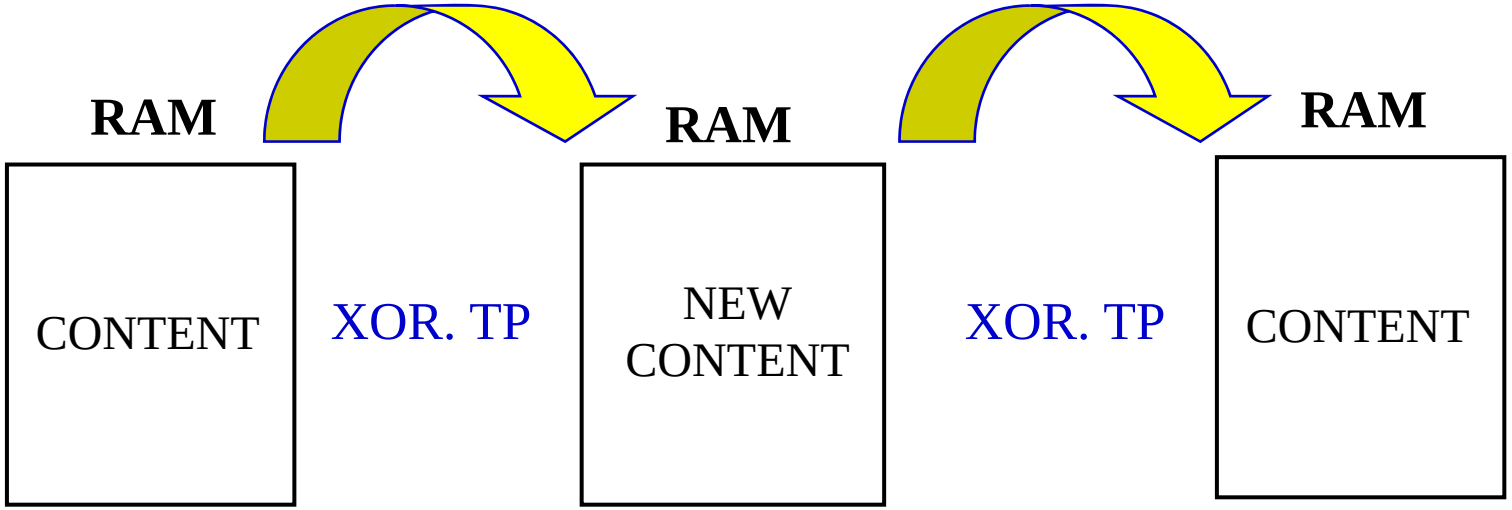
- The Hamming single error-correction code uses c parity check bits to protect k bits of information. The relationship between the values of c and k is
  - $2^c \geq c + k + 1$
- Suppose that there are four information bits (d_3, d_2, d_1, d_0) and, as a result, three parity check bits (c_1, c_2, c_3). The bits are partitioned into groups as (d_3, d_1, d_0, c_1), (d_3, d_2, d_0, c_2), and (d_3, d_2, d_1, c_3). Each check bit is specified to set the parity of its respective group, i.e.,

$$c_1 = d_3 + d_1 + d_0$$
$$c_2 = d_3 + d_2 + d_0$$
$$c_3 = d_3 + d_2 + d_1$$

What is Transparent Test?

- Transparent testing
  - Leave the original content of the circuit under test unchanged after the testing is completed if no faults are presented
- Features
  - Ensure the reliability of stored data throughout its life time
  - Provide better fault coverage than non-transparent testing for unmodeled faults
- Limitation
  - Must be performed while systems are idle

Principle of Transparent Testing



1. Read (CONTENT), take signature S(CONTENT)
2. Read (CONTENT), Write (CONTENT. XOR. TP)=NEW_CONTENT
3. Read (NEW_CONTENT), take new signature S(NEW_CONTENT)
4. Write (NEW_CONTENT. XOR. TP)

NEW_CONTENT. XOR. TP=CONTENT. XOR. TP. XOR. TP=CONTENT

S(NEW_CONTENT)=S(CONTENT. XOR. TP)=S(CONTENT). XOR. S(TP)

Issues of Transparent Testing

- Test interrupts
  - In comparison with manufacturing testing, one special issue of transparent testing is that the transparent testing process may be interrupted
- Aliasing
  - If a transparent built-in self-test scheme is considered, the signature generation typically is done by a MISR
- Fault location
  - If a fault is detected, it is very difficult to locate the fault

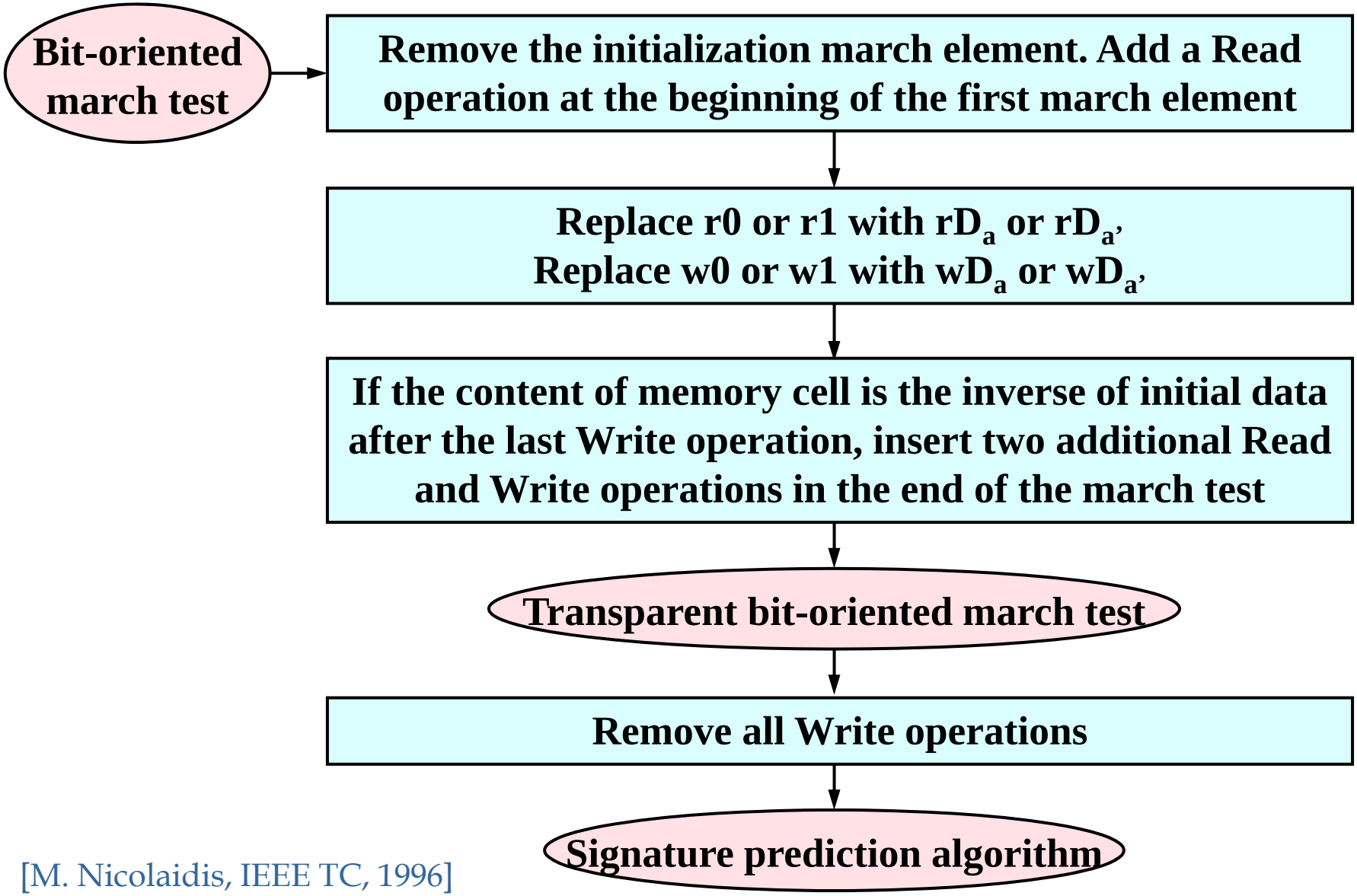
A Typical Transparent March Test

- A typical transparent March test consists of two-phase tests
  - Signature-prediction test
  - Transparent March test
- Types of transparent test schemes
  - Transparent March tests
  - Symmetric transparent March tests
  - Combination of Transparent March tests and ECCs

Notation

- In a test algorithm
  - D denotes the initial content of a cell or a word for bit-oriented or word-oriented memories
  - D_a is data of the bit-wise XOR operation on D and a
  - $\Uparrow(\Downarrow)$ represents the ascending (descending) address sequence
  - $\Updownarrow$ denotes either ascending or descending address sequence
  - wX denotes a write X operation
  - rX denotes a read operation with expect data X

A Typical Transformation Method



[M. Nicolaidis, IEEE TC, 1996]

An Example

- Consider the March C- test:
  - $\{\Updownarrow(w0);\Uparrow(r0,w1);\Uparrow(r1,w0);\Downarrow(r1,w0);\Downarrow(r0,w1);\Updownarrow(r0)\}$
- After Step 1 transformation:
  - $\{\Uparrow(r0,w1);\Uparrow(r1,w0);\Downarrow(r1,w0);\Downarrow(r0,w1);\Updownarrow(r0)\}$
- After Step 2 transformation:
  - $\{\Uparrow(rD_a,wD_{\bar{a}});\Uparrow(rD_{\bar{a}},wD_a);\Downarrow(rD_a,wD_{\bar{a}});\Downarrow(rD_{\bar{a}},wD_a);\Updownarrow(rD_a)\}$
- The content of memory cell after the last operation is the same as the initial state. Step 3 is omitted.
- Thus, the transparent March C- test is follows:
  - $\{\Uparrow(rD_a,wD_{\bar{a}});\Uparrow(rD_{\bar{a}},wD_a);\Downarrow(rD_a,wD_{\bar{a}});\Downarrow(rD_{\bar{a}},wD_a);\Updownarrow(rD_a)\}$
- Remove the Write operations. The signature prediction algorithm is as follows:
  - $\{\Uparrow(rD_a);\Uparrow(rD_{\bar{a}});\Downarrow(rD_a);\Downarrow(rD_{\bar{a}});\Updownarrow(rD_a)\}$

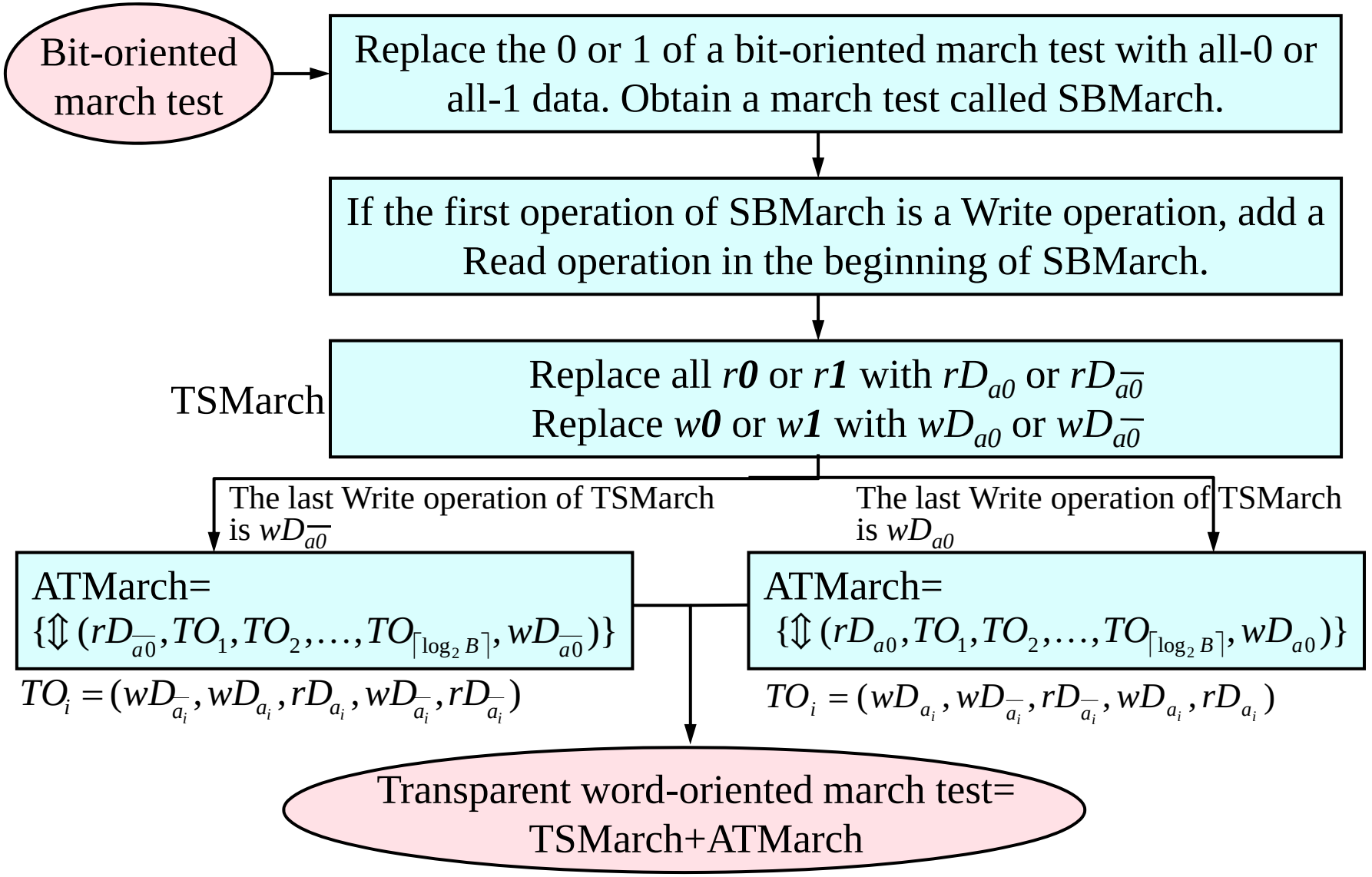
Word-Oriented Transparent Tests

- Word-oriented transparent test can be obtained
  - By applying the transformation rules to all the bits of each word [Nicolaids, ITC92].
- E.g., a word-oriented March C- for 4-bit words
  - T1: $\{\Updownarrow(w0000);\Uparrow(r0000,w1111);\Uparrow(r1111,w0000);\Downarrow(r1111,w0000);\Downarrow(r0000,w1111);\Updownarrow(r0000)\}$
  - T2: $\{\Updownarrow(w0101);\Uparrow(r0101,w1010);\Uparrow(r1010,w0101);\Downarrow(r0101,w1010);\Downarrow(r1010,w0101);\Updownarrow(r0101)\}$
  - T3: $\{\Updownarrow(w0011);\Uparrow(r0011,w1100);\Uparrow(r1100,w0011);\Downarrow(r0011,w1100);\Downarrow(r1100,w0011);\Updownarrow(r0011)\}$
- Thus, the transparent word-oriented March C-
  - T1': $\{\Uparrow(rD_{a0},wD_{\overline{a0}});\Uparrow(rD_{\overline{a0}},wD_{a0});\Downarrow(rD_{a0},wD_{\overline{a0}});\Downarrow(rD_{\overline{a0}},wD_{a0});\Updownarrow(rD_{a0},wD_{a1})\}$
  - T2': $\{\Uparrow(rD_{a1},wD_{\overline{a1}});\Uparrow(rD_{\overline{a1}},wD_{a1});\Downarrow(rD_{a1},wD_{\overline{a1}});\Downarrow(rD_{\overline{a1}},wD_{a1});\Updownarrow(rD_{a1},wD_{a2})\}$
  - T3': $\{\Uparrow(rD_{a2},wD_{\overline{a2}});\Uparrow(rD_{\overline{a2}},wD_{a2});\Downarrow(rD_{a2},wD_{\overline{a2}});\Downarrow(rD_{\overline{a2}},wD_{a2});\Updownarrow(rD_{a2},wD_{a0})\}$

Problem

- Transparent tests are usually applied in the idle state of systems or components
- Reducing the test time is very important
  - Avoiding the interrupt of testing
- However, conventional transparent word-oriented march tests are directly obtained
  - By executing the corresponding bit-oriented march test on each bit of word
- Thus, conventional transformation does not generate a time-efficiency word-oriented march test

Efficient Word-Oriented Transparent Tests



Source: J.-F. Li, IEEE TCAD, 2007 (accepted)

Example

- Consider a bit-oriented March U [15]
  - $\{\Updownarrow(w0);\Uparrow(r0,w1,r1,w0);\Uparrow(r0,w1);\Downarrow(r1,w0,r0,w1);\Downarrow(r1,w0)\}$
- Then, the solid March U (SBMarch U) is as follows
  - $\{\Updownarrow(w\vec{0});\Uparrow(r\vec{0},w\vec{1},r\vec{1},w\vec{0});\Uparrow(r\vec{0},w\vec{1});\Downarrow(r\vec{1},w\vec{0},r\vec{0},w\vec{1});\Downarrow(r\vec{1},w\vec{0})\}$
  - where $\vec{0}$ and $\vec{1}$ denote all-0 and all-1 data
- According to the transformation rules described above, the transparent SBMarch U (TSMarch U) is
  - $\{\Uparrow(rD_{a0},wD_{\overline{a0}},rD_{\overline{a0}},wD_{a0});\Uparrow(rD_{a0},wD_{\overline{a0}});\Downarrow(rD_{\overline{a0}},wD_{a0},rD_{a0},wD_{\overline{a0}});\Downarrow(rD_{\overline{a0}},wD_{a0})\}$
  - where *a0* denotes all-0 data
- The last operation of TSMarch U is wD_{a0}
  - ATMarch= $\Updownarrow(rD_{a_0},wD_{a_1},wD_{\overline{a_1}},rD_{\overline{a_1}},wD_{a_1},rD_{a_1},wD_{a_2},wD_{\overline{a_2}},rD_{\overline{a_2}},wD_{a_2},rD_{a_2},wD_{a_3},wD_{\overline{a_3}},rD_{\overline{a_3}},wD_{a_3},rD_{a_3},wD_{a_0})$

Symmetric Transparent Tests

- Feature
  - The symmetric transparent test method take advantage of the symmetric characteristic of a signature analyzer to eliminate the signature prediction phase
- Symmetric characteristic of a signature analyzer
  - Let $sig(z, S, h)=u$ denote a serial signature analyzer which has an initial state S, a feedback polynomial h, a data string for analysis z, and the corresponding signature u. Then we can obtain $sig(z^*, u^*, h^*)=S^*$, where z^*, u^*, h^*, and S^* denote the reverse of z, u, h, and S, respectively [V. N. Yarmolik and S. Hellebrand, DATE99]

An Example

- A $2n$-bit data string $Z=(x_{2n-1}x_{2n-2}\ldots x_n x_{n-1}\ldots x_1 x_0)$ is called a symmetric data string if
  - $x_{n-1}=x_n, x_{n-2}=x_{n+1}, \ldots x_1=x_{2n-2}, x_0=x_{2n-1}$
  - or $x_{n-1}=\overline{x_n}, x_{n-2}=\overline{x_{n+1}}, \ldots, x_1=\overline{x_{2n-2}}, x_0=\overline{x_{2n-1}}$
- Consider a symmetric data string $Z=(zz^*)$. Assume that a reconfigurable signature analyzer $sig(-, \boldsymbol{0}, h)$ is used to analyze the symmetric data string Z
  - Step 1: z is analyzed and $sig(z, \boldsymbol{0}, h)=u$
  - Step 2: analyzer is configured as $sig(-, u^*, h^*)$
  - Step 3: z^* is analyzed and $sig(z^*, u^*, h^*)=\boldsymbol{0}^*=\boldsymbol{0}$

Symmetric Transparent March Tests

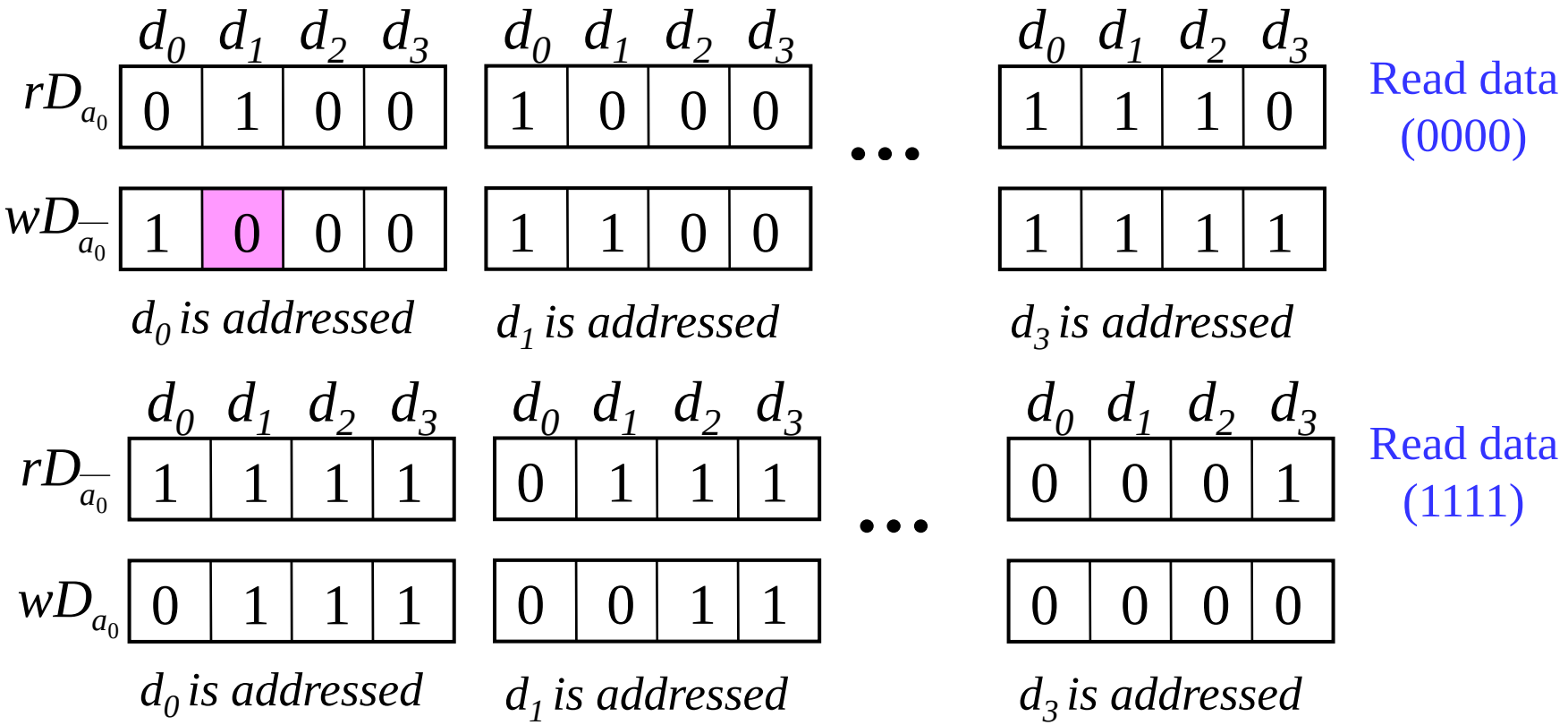
- A transparent March test is a symmetric transparent March test if the read data of the Read operations of the transparent March test is a symmetric data string Z
- For example, consider the March test MATS+
  - $\{\Updownarrow(w0);\Uparrow(r0,w1);\Downarrow(r1,w0)\}$
- It can be transformed to a transparent March test
  - $\{\Uparrow(rD_{a_0},wD_{\overline{a_0}});\Downarrow(rD_{\overline{a_0}},wD_{a_0})\}$
  - The read data can be expressed as $Z=(z,z^{*c})$

Limitations

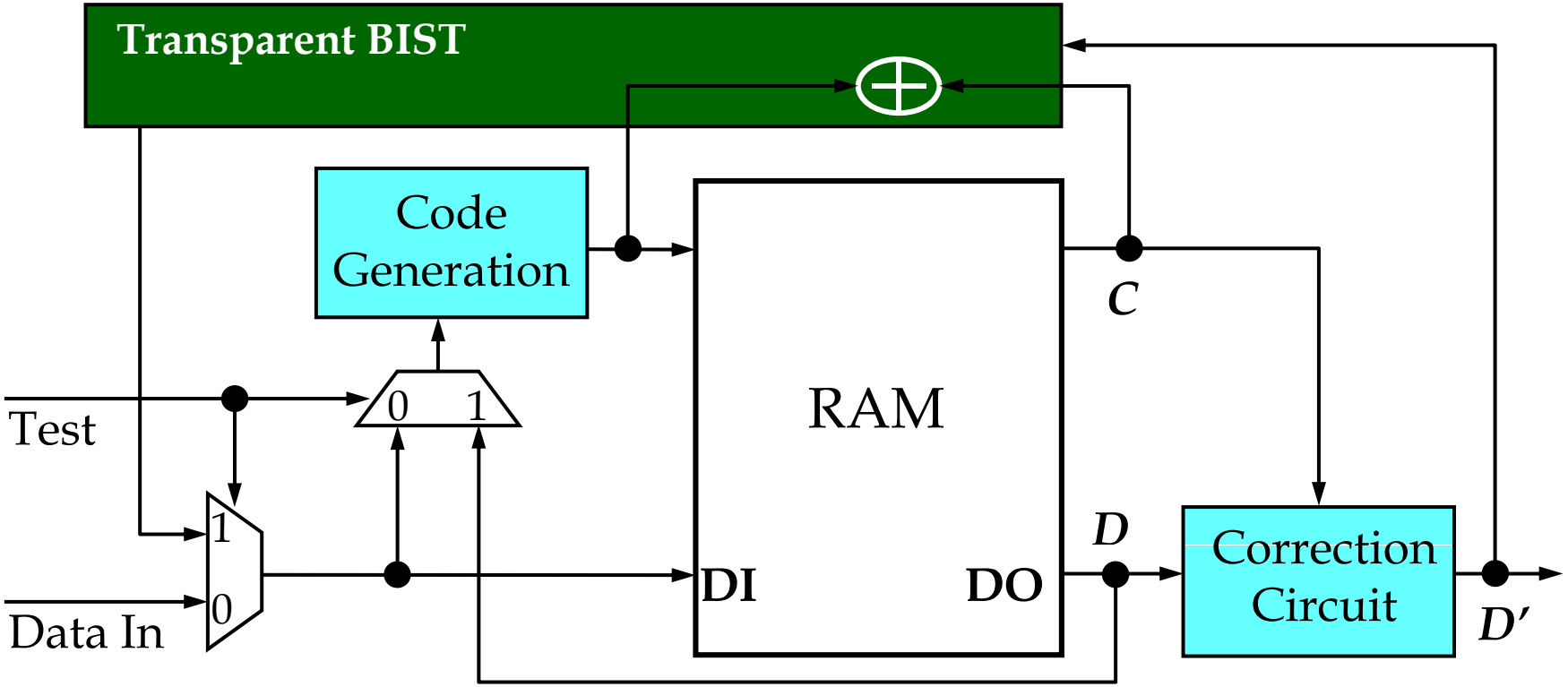
- Symmetric transparent March tests have two major limitations
  - Fault masking effect
  - Test interrupts cause the symmetric characteristic to be invalid
- Consider a 4-bit memory with initial content $(d_0 d_1 d_2 d_3)$=(0100). Assume that the memory has an idempotent coupling fault in which the aggressor and victim are at d_0 and d_1. Also, the value of the victim is forced to 0 while the aggressor has a 0 to 1 transition

Fault Masking Effect

- Assume that the symmetric transparent MATS+ is used to test a 4-bit memory with a CFid
  - Transparent MATS+: $\{\Uparrow(rD_{a_0}, wD_{\overline{a_0}}); \Downarrow(rD_{\overline{a_0}}, wD_{a_0})\}$



Transparent Test Scheme for a RAM with ECC



Read: Read the data D at DO; Check if Code_Gen(D)=C
Write: Write the data D'. XOR. TP

Source: J.-F. Li, IEEE TCAD, 2007 (accepted)

Features

- No signature prediction phase is needed. This shortens the testing time such that the probability of an interruption is reduced
- Really restoring the original content of the memory under test is achieved if the number of faulty bits of a word is less than the correction capability of the applied ECC
- It can locate the faulty bit of the faulty word by the checking response. The fault location capability is also related to the correction capability of the applied ECC

Example

- Consider a 3x4-bit memory with Hamming ECC. Also, the transparent March MATS+ is used to test the memory
  - Transparent March MATS+: $\{\Uparrow(rD_{a_0}, wD_{\overline{a_0}}); \Downarrow(rD_{\overline{a_0}}, wD_{a_0})\}$
  - Assume that *d2* of the first word has a stuck-at-0 fault

	$d_3\ d_2\ d_1\ d_0$	$h_2\ h_1\ h_0$
W_0	1 0 0 1	1 0 0
W_1	1 1 1 0	1 0 0
W_2	1 0 1 0	0 1 0

	$d_3\ d_2\ d_1\ d_0$	$h_2\ h_1\ h_0$
W_0	0 0 1 0	0 1 1
W_1	0 0 0 1	0 1 1
W_2	0 1 0 1	1 0 1

	$d_3\ d_2\ d_1\ d_0$	$h_2\ h_1\ h_0$
W_0	1 0 0 1	1 0 0
W_1	1 1 1 0	1 0 0
W_2	1 0 1 0	0 1 0

$h_0 = d_3 + d_1 + d_0$
$h_1 = d_3 + d_2 + d_0$
$h_2 = d_3 + d_2 + d_1$

Conclusions

- With the advent CMOS technology, enhancing the reliability of an integrated circuit becomes one major challenge
  - Effective and efficient reliability-enhancement techniques must be developed
- Various transparent test techniques have been presented

References

1. M. Nicolaidis,“Theory of transparent BIST for RAMs,” *IEEE Trans. on Computers*,vol. 45, no. 10, pp. 1141–1156, Oct. 1996.
2. V. N. Yarmolik and S. Hellebrand, “Symmetric transparent BIST for RAMs,” in *Proc. Conf. Design, Automation, and Test in Europe (DATE)*, 1999, pp. 702–707.
3. J.-F. Li,”Transparent test methodologies for random access memories with/without ECC,” *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, vol.26, no.10, pp. 1888-1893, Oct. 2007.