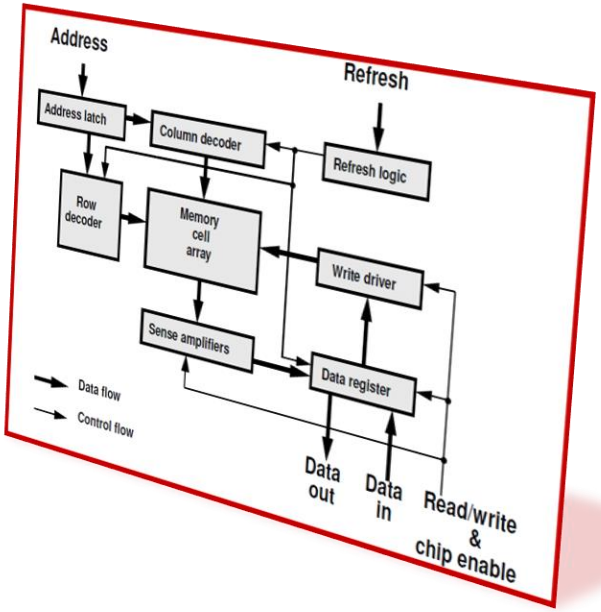
Outline

- Introduction
- Memory Fault Model
- Memory Test Algorithms
- Memory Fault Simulation (*not in exam)
- Memory Test Generation (*not in exam)
- Memory BIST (*not in exam)



Memory Fault Simulation

- What is *memory fault simulation*?
  - Given test algorithm, memory architecture
    - find fault coverage of different fault models

- Why fault simulation?
  - Evaluate fault coverage efficiently,
    - especially when many fault models
  - Help test algorithm design and optimization
  - Fault dictionary can be constructed for easy diagnosis

Sequential Memory Fault Simulation

```
For each fault    /* N^2 for 2-cell CF */
Inject fault;
For each test element    /* N for March */
            {
              Apply test element;
              Report error output
            }
```

- Complexity is N^3 for 2-cell CF
  - This is very slow

Parallel Fault Simulation [Wu 02]

- **Random Access Memory Simulator for Error Screening (RAMSES)**
- **Consider all faults at a time**
- **Complexity N^2**

```
# S/1
AGR := w0
SPT := @          /* Single-cell fault */
VTM := r0
RCV := w1

# CFst <0;s/1>
AGR := v0
SPT := *          /* All other cells are suspects */
VTM := r0
RCV := w1
```

For each test operation

{

If op is AGR then mark victim cells;

If op is RCV then release victim cells;

If op is VTM then report error;

}

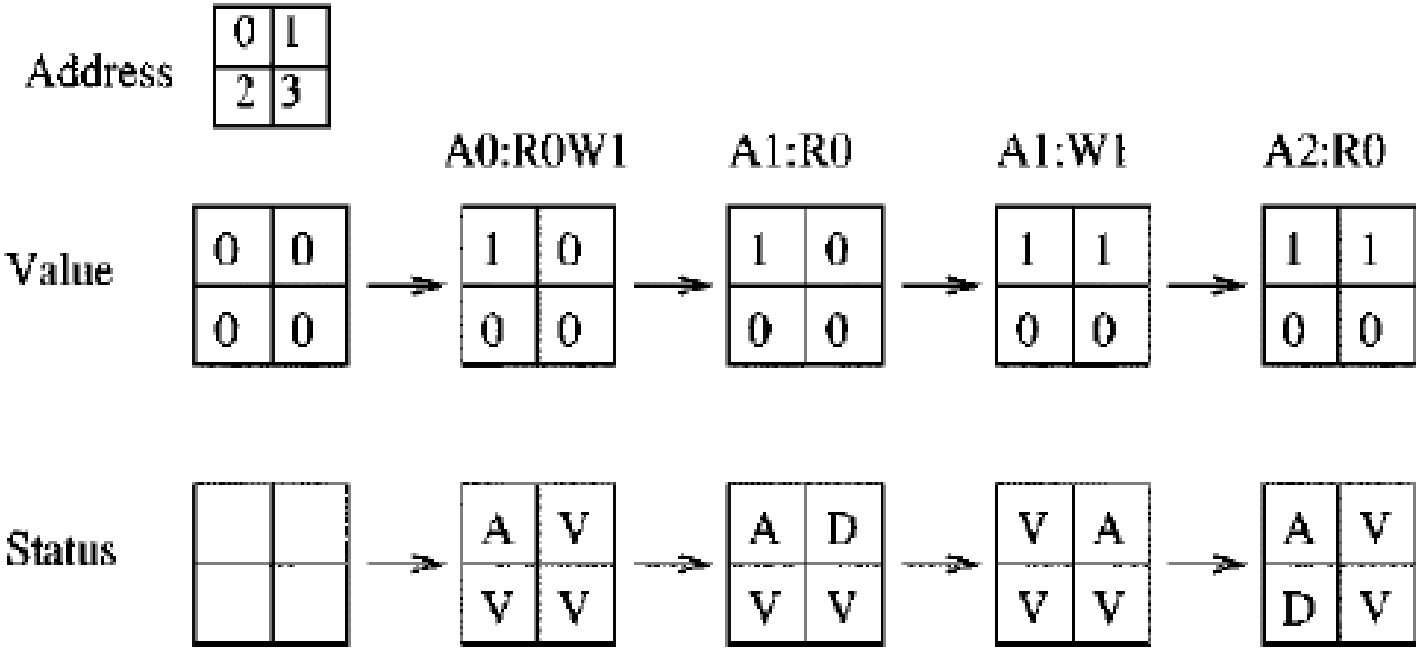
AGR=aggressor
SPT=suspects
VTM=victim
RCV=recover
@=this cell
***=all cells**

RAMSES Algorithm

```
For each fault opeartion
  set op_flags;
  if (ARG ⊂ op_flags)  {
     for each victim cell {
        set victim flags;
        set aggressor address;
     }
  }
  if (OP eq RCV)  {
     clear victim flag;
     clear aggressor entry;
  }
  else if (OP eq VTM) {
      mark detected;
  }
}
```

RAMESE Example

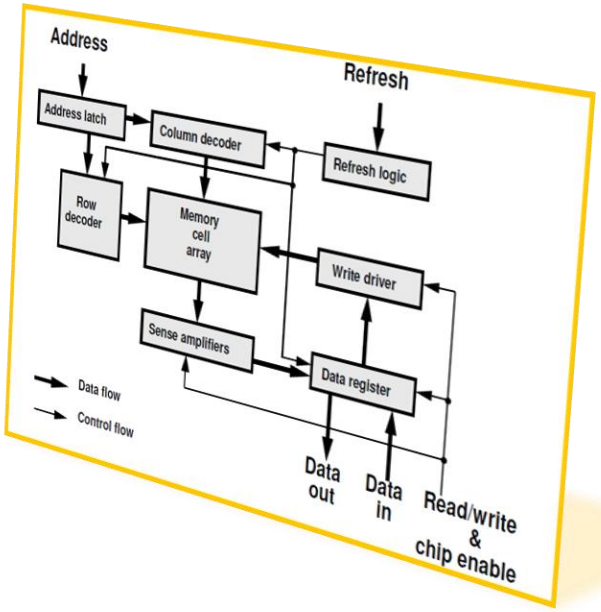
- CF_{in} <↑ / ↕ >
- March element ⇑(r0, w1)



D = detect; A = aggressor; V=victim

Outline

- Introduction
- Memory Fault Model
- Memory Test Algorithms
- Memory Fault Simulation
- Memory Test Generation (* not in exam)
- Memory BIST



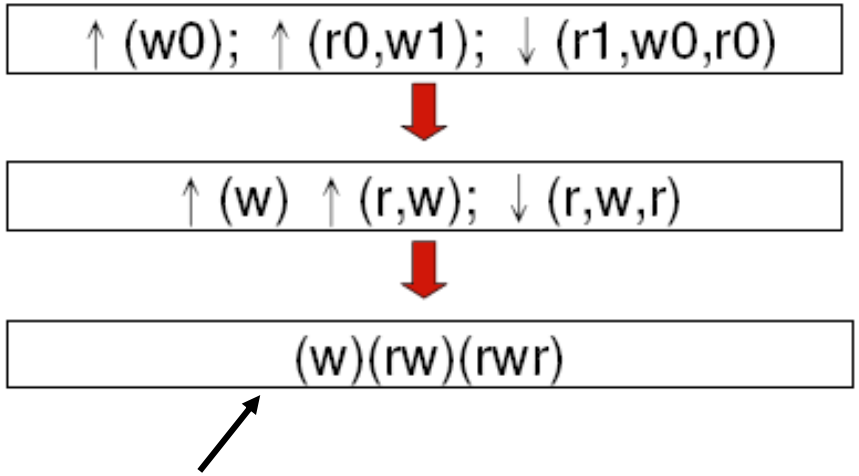
Memory Test Generation

- What is memory test algorithm generation
  - Given a set of target fault models and a test length constraint,
    - generate a test with the highest fault coverage
- Priority setting for fault models

TAGS [Wu 00]

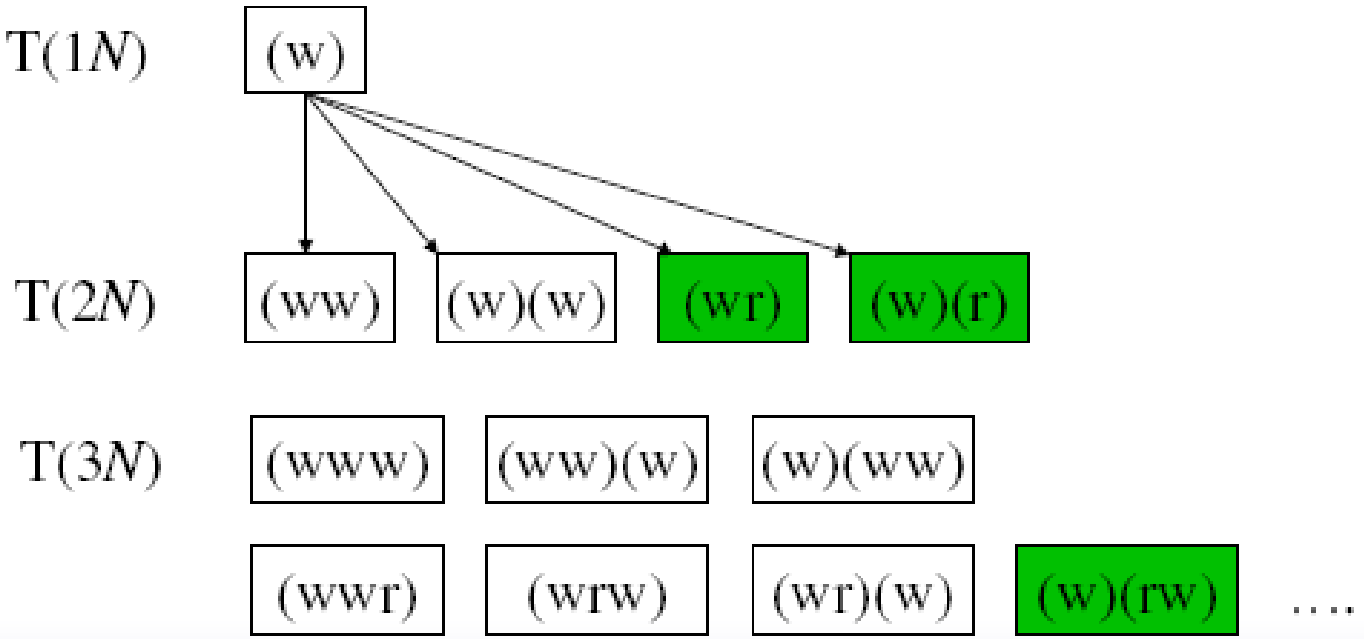
- **Test Algorithm Generation by Simulation (TAGS)**

- March template abstraction:



Template Set

- **Template set**
  - **T(*cN*) = a set of march templates of *c* read or write**
- **Exhaustive generation of template set is very expensive**
  - **e.g., 6.7 million templates when *N* = 9**
- **Heuristics should be developed to select useful templates**



TAGS Procedure

- 1. Initialize test length as 1*N*, T(1*N*) = {(w)};
- 2. Increase test length by 1*N*: apply generation options;
- 3. Apply filter options;
- 4. Assign address orders and data backgrounds;
- 5. Fault simulation using RAMSES;
- 6. Drop ineffective tests;
- 7. Repeat 2-6 using the new template set until constraints met;

Template Generation and Filtering

- **Generation heuristics:**
  - **(r) insertion**
  - **(…r), (r…) expansion**
  - **(w) insertion**
  - **(…w), (w…) expansion**
- **Filtering heuristics:**
  - **Consecutive read: (…rr…)**
  - **Repeated read: (r)(r)**
  - **Tailing single write: …(w)**

3*N* March Test

- 3*N* March Test generated by TAGS after step 3
  - WWW Table 8.7

	Test
1	⇑ (w0) ⇑(w1, r1)
2	⇑(w0) ⇓(w1, r1)
3	⇑(w0) ⇑(w0, r0)
4	⇑(w0) ⇓(w0, r0)
5	⇑(w0) ⇑(r0, w1)
6	⇑(w0) ⇓(r0, w1)
7	⇑(w0) ⇑(r0) ⇑(r0)
8	⇑(w0) ⇑(w1) ⇑(r1)

- 3N March test selected by RAMSES
  - WWW Table 8.8

	Test
3	⇑(w0) ⇑(w0, r0)
5	⇑(w0) ⇑(r0, w1)
8	⇑(w0) ⇑(w1) ⇑(r1)

TAGS Examples (1/2)

- **WWW Table 8.9**

$T(N)$	Name	March algorithm
$1N$	MI_1^1	$\Uparrow (w0)$
$2N$	MI_1^2	$\Uparrow (w0)\ \Uparrow (r0)$
$3N$	MI_1^3	$\Uparrow (w0)\ \Uparrow (w1)\ \Uparrow (r1)$
$3N$	MI_2^3	$\Uparrow (w0)\ \Uparrow (r0, w1)$
$3N$	MI_3^3	$\Uparrow (w0)\ \Downarrow (w1)\ \Uparrow (r1)$
$3N$	MI_4^3	$\Uparrow (w0)\ \Downarrow (r0, w1)$
$4N$	MI_1^4	$\Uparrow (w0)\ \Downarrow (r0, w1)\ \Uparrow (r1)$
$4N$	MI_2^4	$\Uparrow (w0)\ \Downarrow (r0, w1, r1)$
$5N$	MI_1^5	$\Uparrow (w0)\ \Uparrow (w1)\ \Uparrow (r1, w0)\ \Uparrow (r0)$
$5N$	MI_2^5	$\Uparrow (w0)\ \Downarrow (r0, w1)\ \Uparrow (r1, w0)$
$5N$	MI_3^5	$\Uparrow (w0)\ \Uparrow (w1)\ \Uparrow (r1, w0, r0)$
$6N$	MI_1^6	$\Uparrow (w0)\ \Uparrow (w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$
$6N$	MI_2^6	$\Uparrow (w0)\ \Downarrow (r0, w1)\ \Uparrow (r1, w0)\ \Uparrow (r0)$
$6N$	MI_3^6	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Uparrow (r0)$
$6N$	MI_4^6	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0, r0)$
$6N$	MI_5^6	$\Uparrow (w0)\ \Downarrow (r0, w1)\ \Uparrow (r1, w0, r0)$
$7N$	MI_1^7	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$

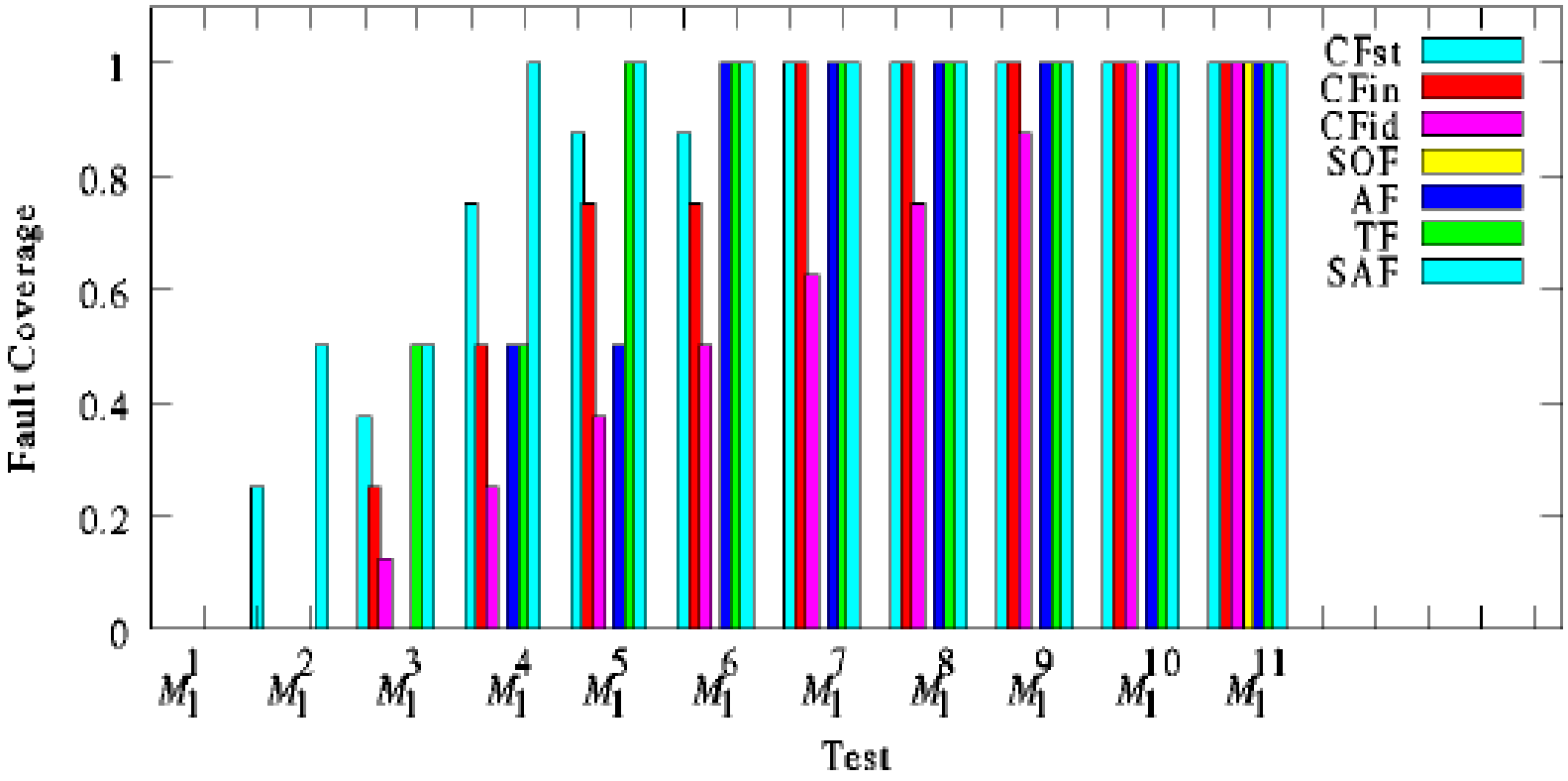
TAGS Examples (2/2)

- **WWW Table 8.9 cont'd**

$7N$	M_1^7	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$
$7N$	M_2^7	$\Uparrow (w0)\ \Uparrow (w1)\ \Downarrow (r1, w0)\ \Uparrow (r0, w1, r1)$
$7N$	M_3^7	$\Uparrow (w0)\ \Downarrow (r0, w1)\ \Uparrow (r1, w0, r0)\ \Uparrow (r0)$
$7N$	M_4^7	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0, r0)\ \Uparrow (r0)$
$8N$	M_1^8	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$ $\Uparrow (r1)$
$8N$	M_2^8	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)$ $\Downarrow (r0, w1, r1)$
$9N$	M_1^9	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$ $\Downarrow (r1, w0)$
$9N$	M_2^9	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)$ $\Downarrow (r0, w1, r1)\ \Uparrow (r1)$
$10N$	M_1^{10}	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$ $\Downarrow (r1, w0)\ \Uparrow (r0)$
$10N$	M_2^{10}	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$ $\Downarrow (r1, w0, r0)$
$11N$	M_1^{11}	$\Uparrow (w0)\ \Uparrow (r0, w1)\ \Uparrow (r1, w0)\ \Downarrow (r0, w1)$ $\Downarrow (r1, w0, r0)\ \Uparrow (r0)$

RAMSES Simulation Results

- **WWW Figure 8.13**



RAMSES Simulation Results

- **WWW Figure 8.14**
  - **Five different 6*N* tests**