Memory Testing

- Introduction
- Memory Architecture & Fault Models
- Test Algorithms
- DC / AC / Dynamic Tests
- Built-in Self Testing Schemes
- Built-in Self Repair Schemes

Memory Market Share in 1999

• DRAM:	8×10^{17}
• Flash:	6×10^{16}
• ROM:	2×10^{16}
• SRAM:	9×10^{15}

DRAM Price per Bit

1991: US$ 400 / Mega bits

1995: US$ 3.75 / Mega bits

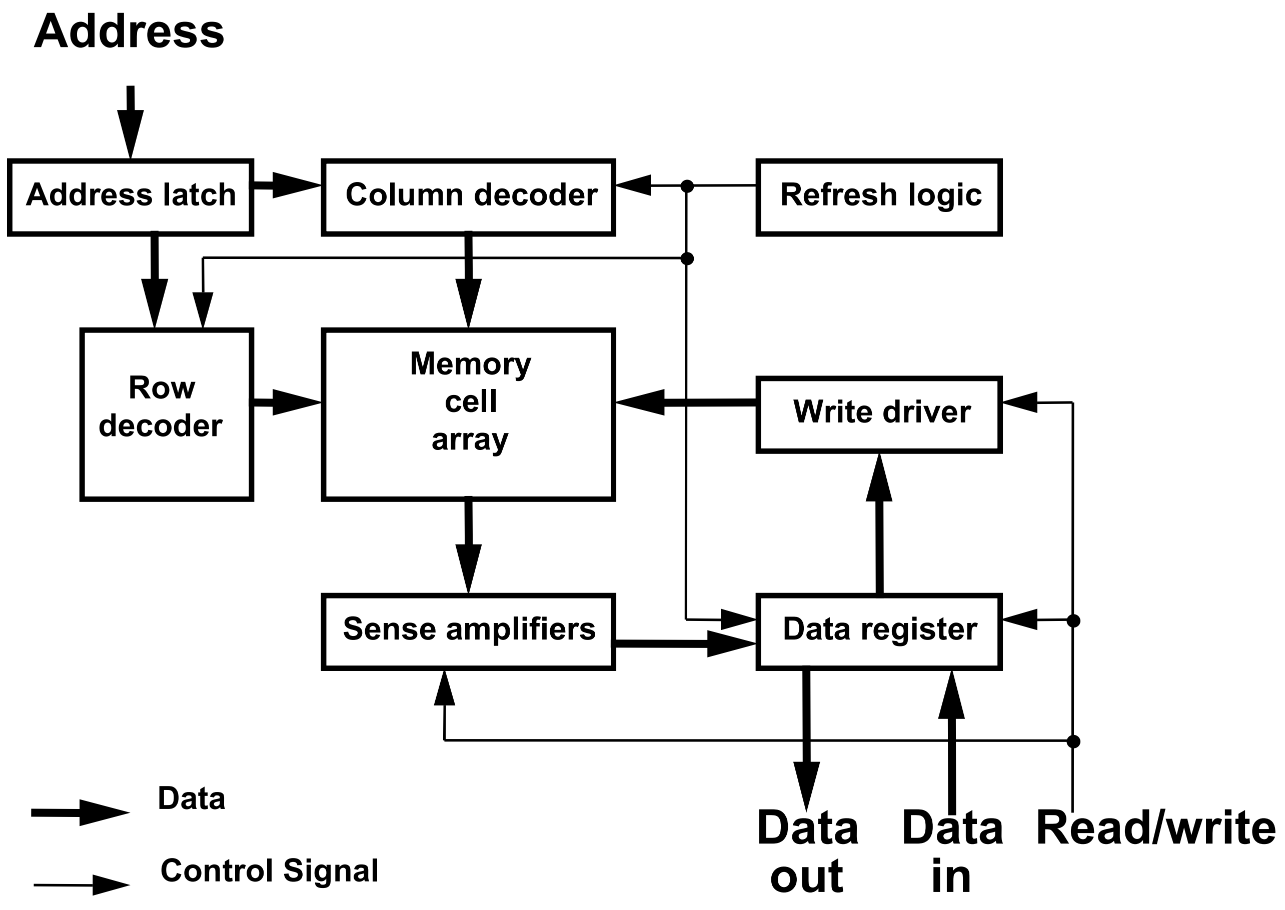
1999: US$ 0.1~0.3 / Mega bits

Test Time as a Function of Memory Size

Cycle time: 10 ns

Size n	Testing time (in seconds)			
	$64n$	$n \log_2 n$	$n^{3/2}$	n^2
16k	0.01	0.0023	0.021	2.7
64k	0.04	0.01	0.168	42
256k	0.17	0.047	1.34	11.4 Mins
1M	0.67	0.21	10.7	183 Mins
4M	2.68	0.92	85.9	49.2 Hrs
16M	10.8	4.03	11.4 Mins	36.5 Days
64M	43.2	16.2	91.6 Mins	584 Days

Architecture of a DRAM Chip



Fault Models

1.	SAF	Stuck-At Fault
2.	TF	Transition Fault
3.	CF	Coupling Fault
4.	NPSF	Neighborhood Pattern Sensitive Fault
5.	AF	Address decoding fault

Stuck-At Fault

- The logic value of a cell or a line is always 0 or 1.

Transition Fault

- A cell or a line that fails to undergo a $0 \rightarrow 1$ or a $1 \rightarrow 0$ transition.

Coupling Fault

- A write operation to one cell changes the content of a second cell.

Neighborhood Pattern Sensitive Fault

- **The content of a cell, or the ability to change its content, is influenced by the contents of some other cells in the memory.**

Address Decoder Fault (AF)

- **Any fault that affects address decoder:**
  - With a certain address, no cell will be accessed.
  - A certain cell is never accessed.
  - With a certain address, multiple cells are accessed simultaneously.
  - A certain cell can be accessed by multiple addresses.

Memory Chip Test Algorithms

- Traditional tests
- Tests for stuck-at, transition and coupling faults
- Tests for neighborhood pattern sensitive faults

Traditional Tests

Algorithm	Test length	Test Time Order
• Zero-One	$4n$	$O(n)$
• Checkerboard	$4n$	$O(n)$
• GALPAT	$2(n+2n^2)$	$O(n^2)$
• Walking 1/0	$2(3n+n^2)$	$O(n^2)$
• Sliding Diagonal	$6n+2n\cdot\sqrt{n}$	$O(n\cdot\sqrt{n})$
• Butterfly	$2[3n+5n(n/2-1)]$	$O(n\cdot\log_2 n)$

- n is the number of bits of the memory array.

March Algorithms

Algorithm March X

Step1: **write** 0 with up addressing order;

Step2: **read** 0 and **write** 1 with up addressing order;

Step3: **read** 1 and **write** 0 with down addressing order;

Step4: **read** 0 with down addressing order.

Notation of March Algorithms

$\Uparrow$: address 0 to address n-1

$\Downarrow$: address n-1 to address 0

$\Updownarrow$: either way

w0 : write 0
w1 : write 1
r0 : read a cell whose value should be 0
r1 : read a cell whose value should be 1

March Algorithms

EX:

MATS (modified algorithmic Test Sequence)

$\Updownarrow$ (w0); $\Updownarrow$ (r0,w1); $\Updownarrow$ (r1);

```
s1: write 0 to all cells
s2: for each cell
        read 0 ;
        write 1;
s3: read 1 from all cells
```

Some March Algorithms

MATS : $\Updownarrow$ **(w0);** $\Updownarrow$ **(r0,w1);** $\Updownarrow$ **(r1)**
MATS+: $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1);** $\Downarrow$ **(r1,w0)**
Marching 1/0 : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1,r1);** $\Downarrow$ **(r1,w0,r0);**
$\Updownarrow$ **(w1);** $\Uparrow$ **(r1,w0,r0);** $\Downarrow$ **(r0, w1, r1);**

MATS++ : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1);** $\Downarrow$ **(r1,w0,r0);**
MARCH X : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1);** $\Downarrow$ **(r1,w0);** $\Updownarrow$ **(r0)**
MARCH C : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1);** $\Uparrow$ **(r1,w0);** $\Updownarrow$ **(r0);**
$\Downarrow$ **(r0,w1);** $\Downarrow$ **(r1,w0);** $\Updownarrow$ **(r0);**

Some March Algorithms (Cont.)

MARCH A : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1,w0,w1);** $\Uparrow$ **(r1,w0,w1);**
$\Downarrow$ **(r1,w0,w1,w0);** $\Downarrow$ **(r0,w1,w0);**

MARCH Y : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1,r1);** $\Downarrow$ **(r1,w0,r0);** $\Updownarrow$ **(r0)**

MARCH B : $\Updownarrow$ **(w0);** $\Uparrow$ **(r0,w1,r1,w0,r0,w1);** $\Uparrow$ **(r1,w0,w1);**
$\Downarrow$ **(r1,w0,w1,w0);** $\Downarrow$ **(r0,w1,w0)**

Tests for Stuck-At, Transition and Coupling Faults

March alg.	Test len.	Fault coverage
MATS	4n	Some AFs, SAFs
MATS+	5n	AFs, SAFs
Marching 1/0	14n	AFs, SAFs, TFs
MATS++	6n	AFs, SAFs, TFs
March X	6n	AFs, SAFs, TFs, Some CFs
March C-	10n	AFs, SAFs, TFs, Some CFs
March A	15n	AFs, SAFs, TFs, Some CFs
March Y	8n	AFs, SAFs, TFs, Some CFs
March B	17n	AFs, SAFs, TFs, Some CFs

NPSF

n	n	n
n	b	n
n	n	n

b: base cell
n: neighbor cells

ANPSF:

Active Neighborhood

Pattern Sensitive Fault

n changes

$\Rightarrow$ b changes

Ex:

n: 0 $\rightarrow$ 1
b: 1 $\rightarrow$ 0

PNPSF:

Passive Neighborhood

Pattern Sensitive Fault

Contain n patterns

$\Rightarrow$ b cannot change

Ex:

n: 00000000
b: 0 or 1

SNPSF:

Static Neighborhood

Pattern Sensitive Fault

Contain n patterns

$\Rightarrow$ b is forced to a certain value

Ex:

n: 11111111
b: 1

DC Parametric Testing

- Contains:

  1. Open / Short test.
  2. Power consumption test.
  3. Leakage test.
  4. Threshold test.
  5. Output drive current test.
  6. Output short current test.

AC Parametric Testing

- Output signal: - the rise & fall times.
- Relationship between input signals:
  - the setup & hold times.
- Relationship between input and output signals:
  - the delay & access times.
- Successive relationship between input and output signals:
  - the speed test.

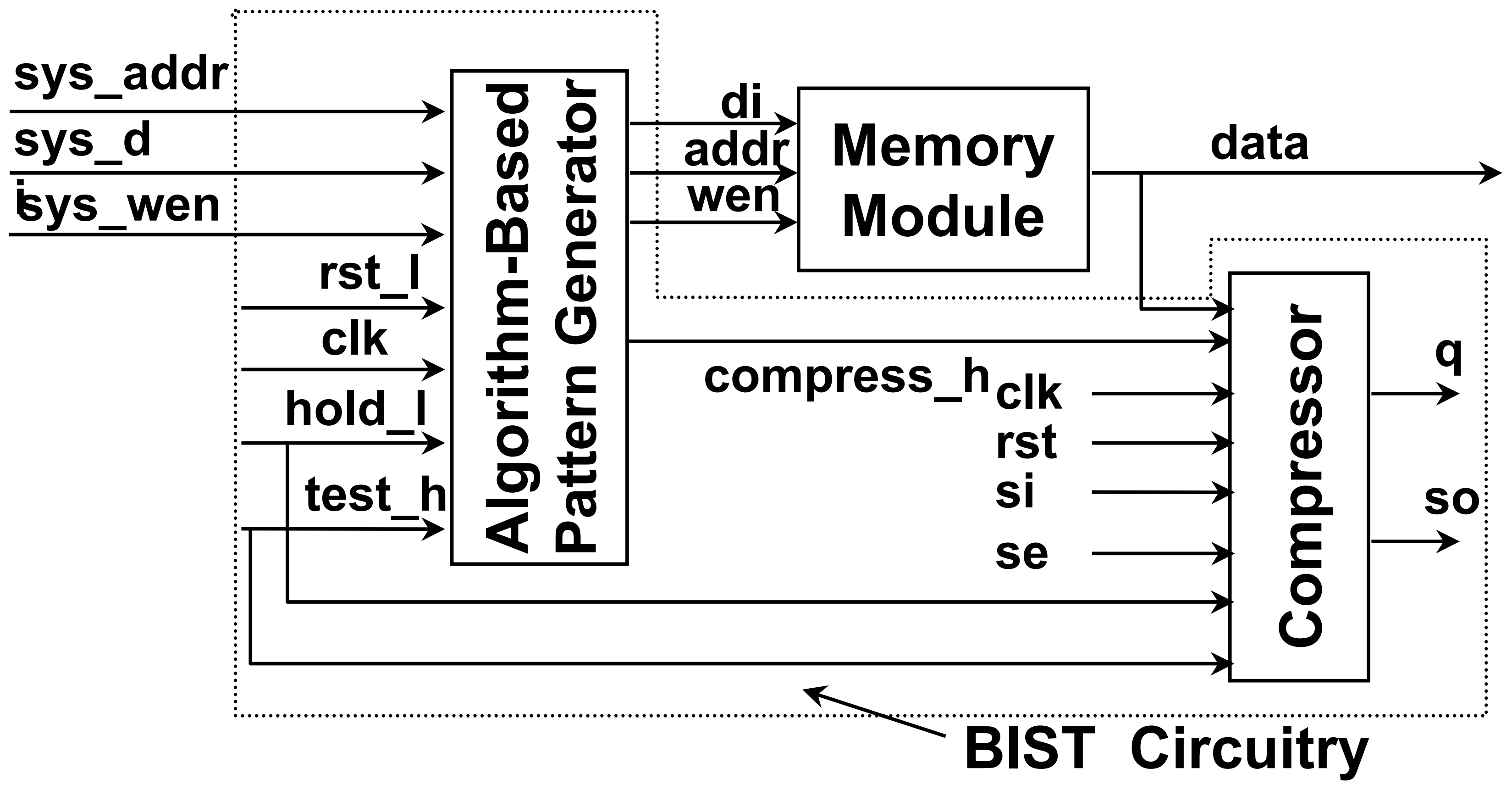
Dynamic Faults

- Recovery faults:
  - Sense amplifier recovery
  - Write recovery.
- Retention faults:
  - Sleeping sickness
  - Refresh line stuck-at
  - Static data loss.
- Bit-line precharge voltage imbalance faults.

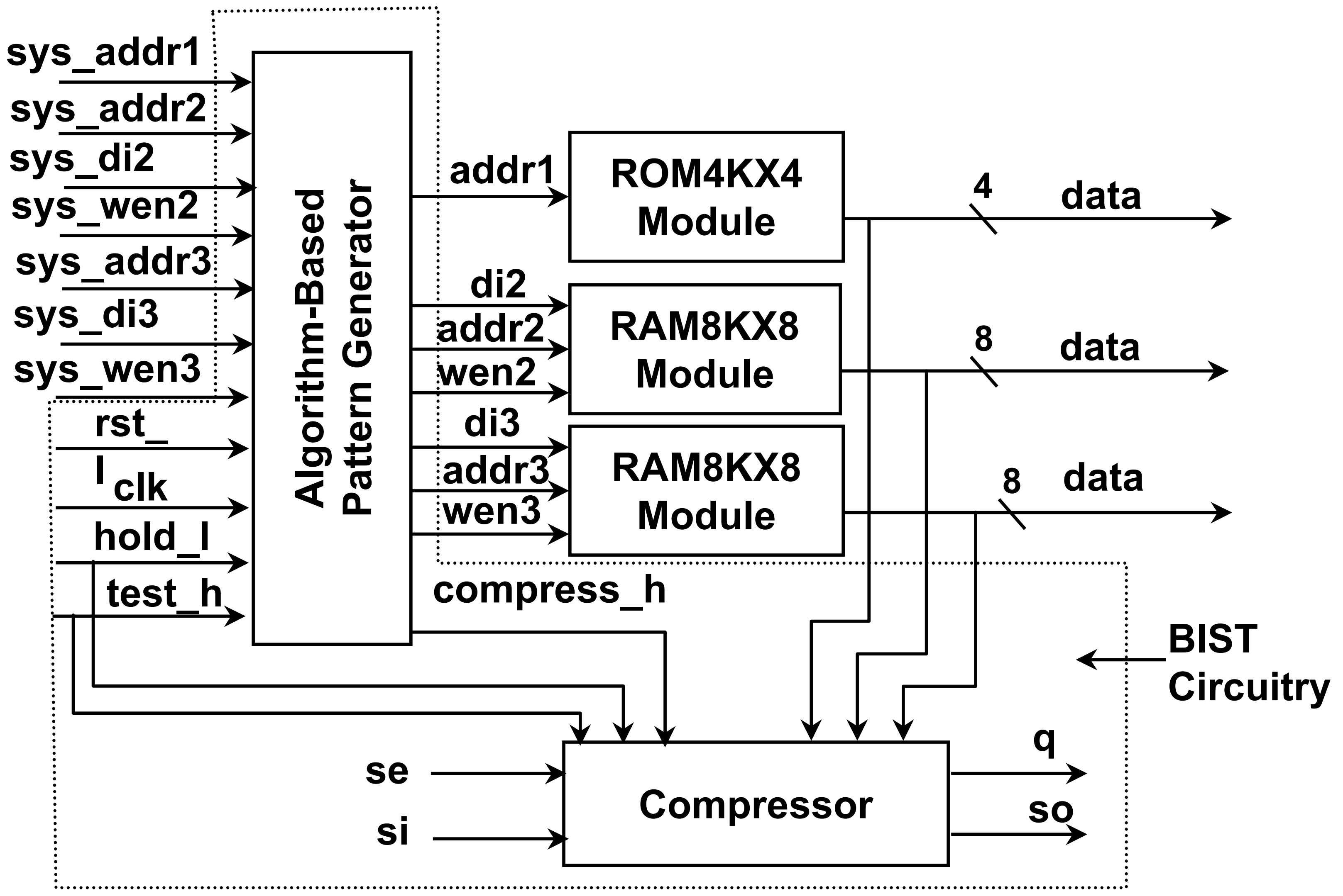
BIST: Pros & Cons

- Advantages:
  - Minimal use of testers.
  - Can be used for embedded RAMs.
- Disadvantages:
  - Silicon area overhead.
  - Speed; slow access time.
  - Extra pins or multiplexing pins.
  - Testability of the test hardware itself.
  - A high fault coverage is a challenge.

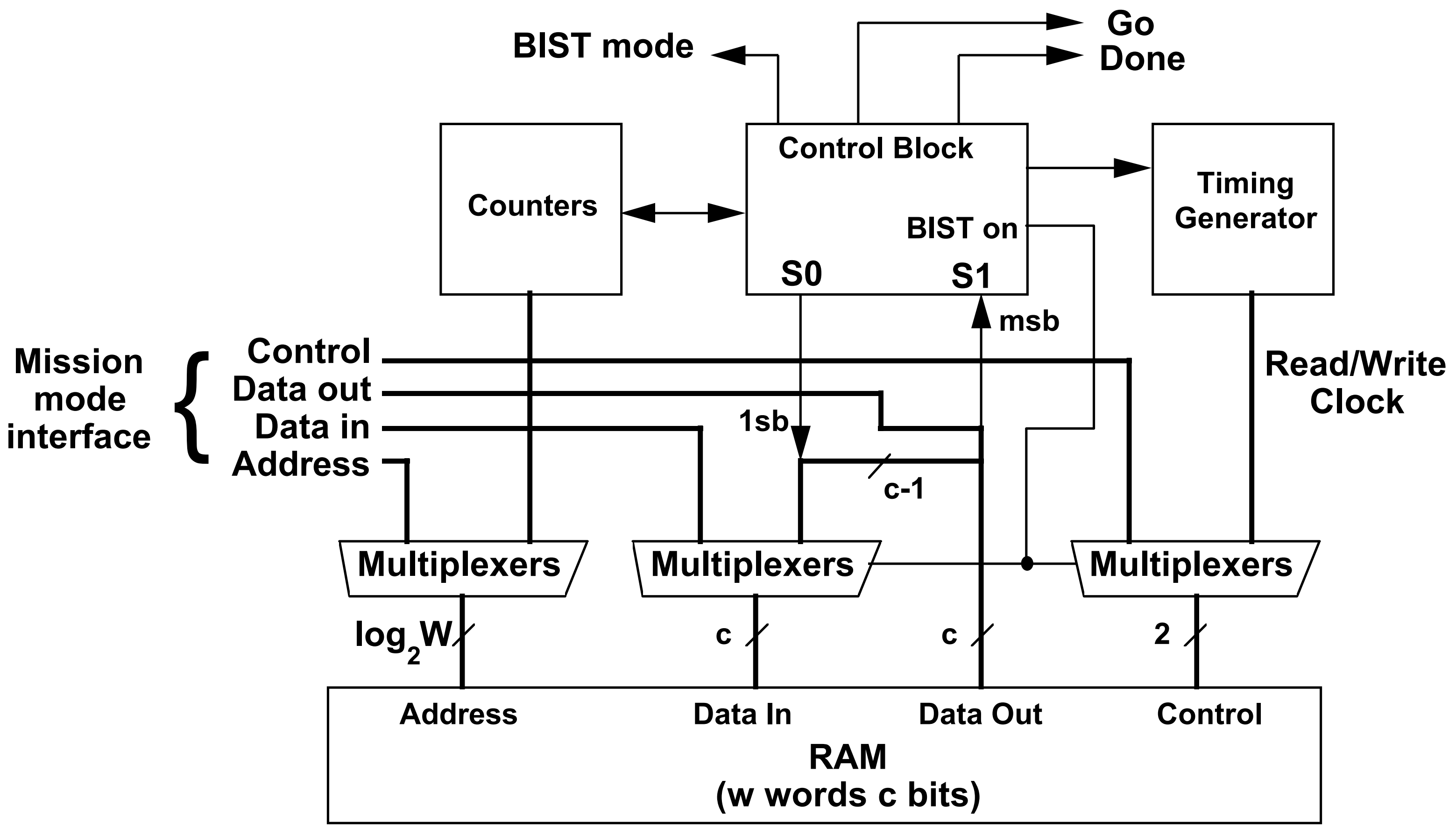
Typical Memory BIST Architecture Using Mentor's Architecture



Multiple Memory BIST Architecture



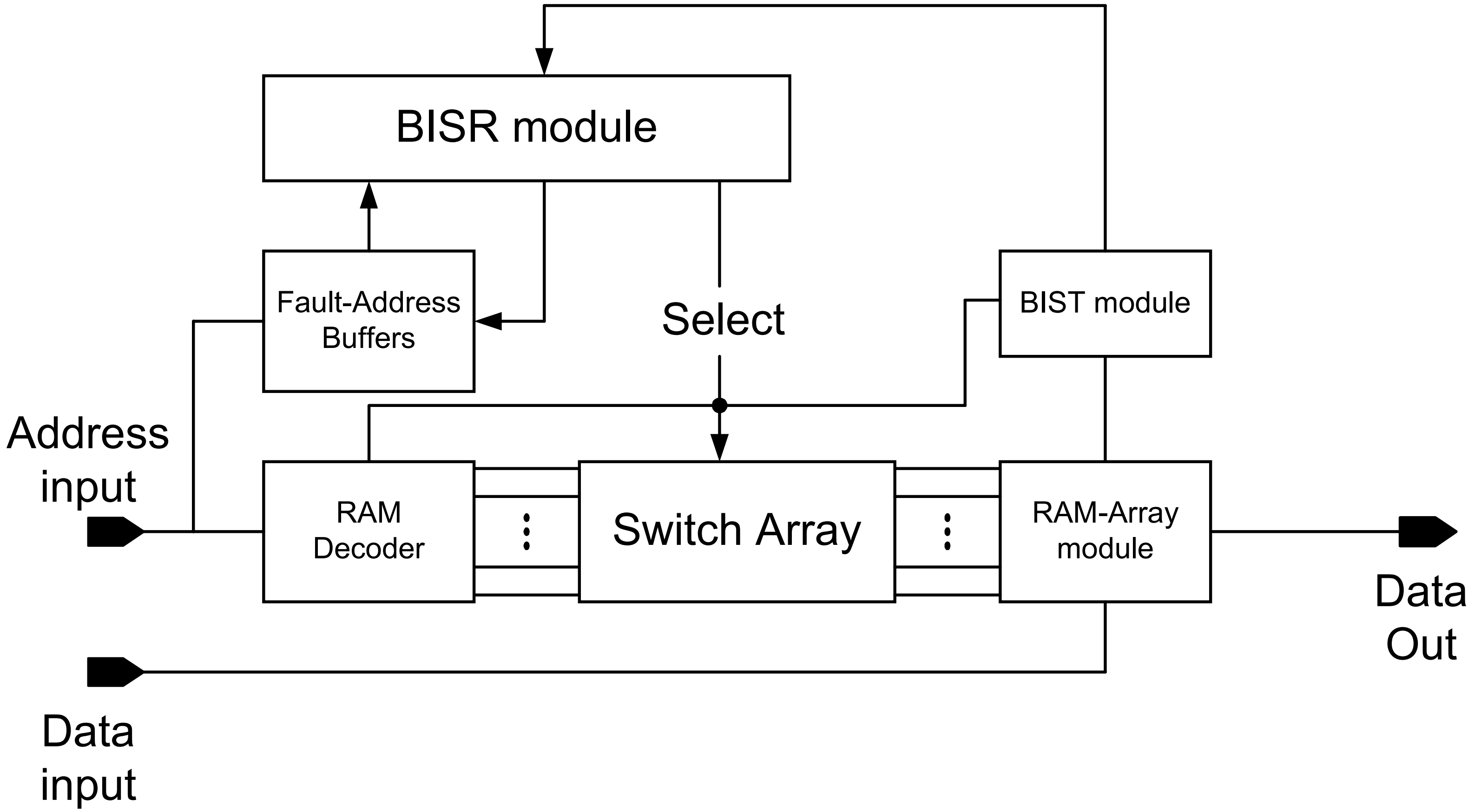
Serial Testing of Embedded RAM



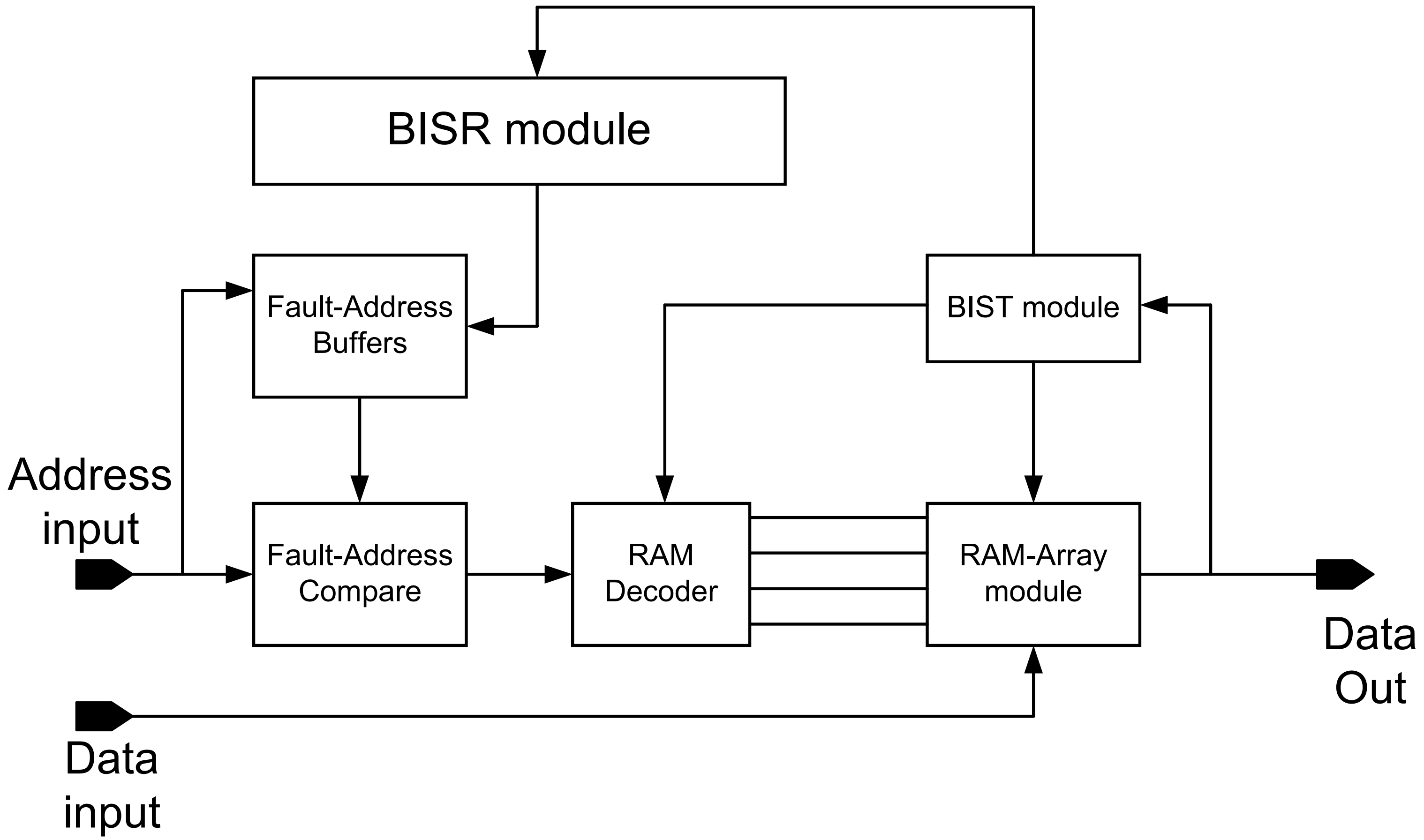
Built-in Self-Repair

- BIST can only identify faulty chip.
- Laser cut may be infeasible in some cases, e.g., field testing.
- Two types:
  - Use fault-array comparator
    - Repair by cell
    - Repair by column (or row)
  - Use switch array

BISR Using Switch Array



BISR via Fault-Address Comparison


BISR module
Fault-Address Buffers
BIST module
Address input
Fault-Address Compare
RAM Decoder
RAM-Array module
Data Out
Data input