Memory Built-In Self-Repair

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

Outline

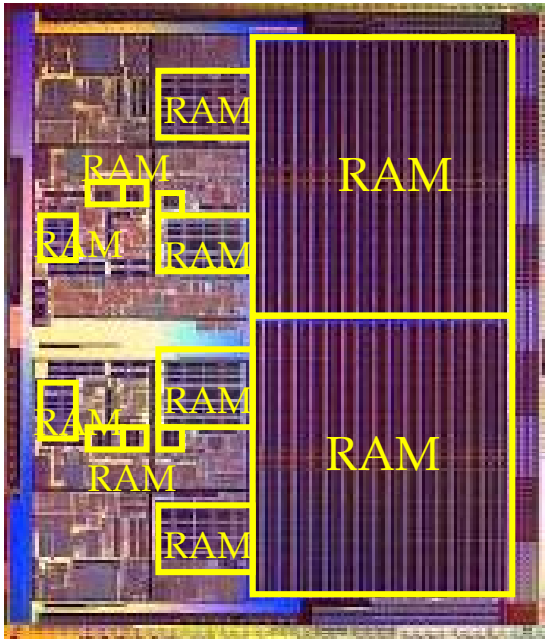
- Introduction
- Redundancy Organizations
- Built-In Redundancy Analysis Techniques
- Built-In Self-Repair Techniques
- Conclusions

Embedded Memory–Quality

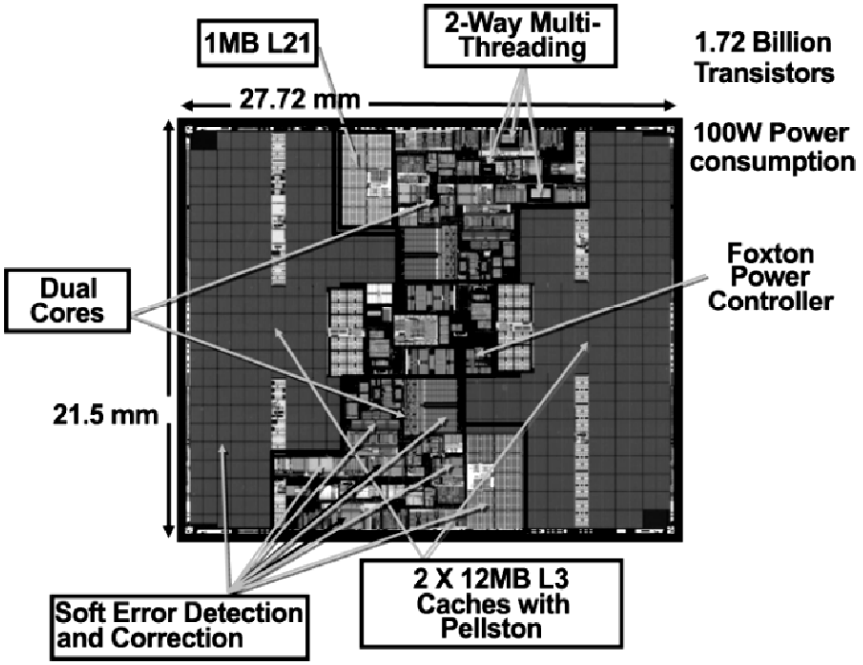
- During manufacture
  - Yield
  - Exponential yield model
  - $Y = e^{-\sqrt{AD}}$, where A and D denote the area and defect density, respectively
- After manufacture
  - Reliability
- During use
  - Soft error rate

An Explosion in Embedded Memories

- Hundreds of memory cores in a complex chip is common
- Memory cores usually represent a significant portion of the chip area



AMD dual-core Opteron™ processor

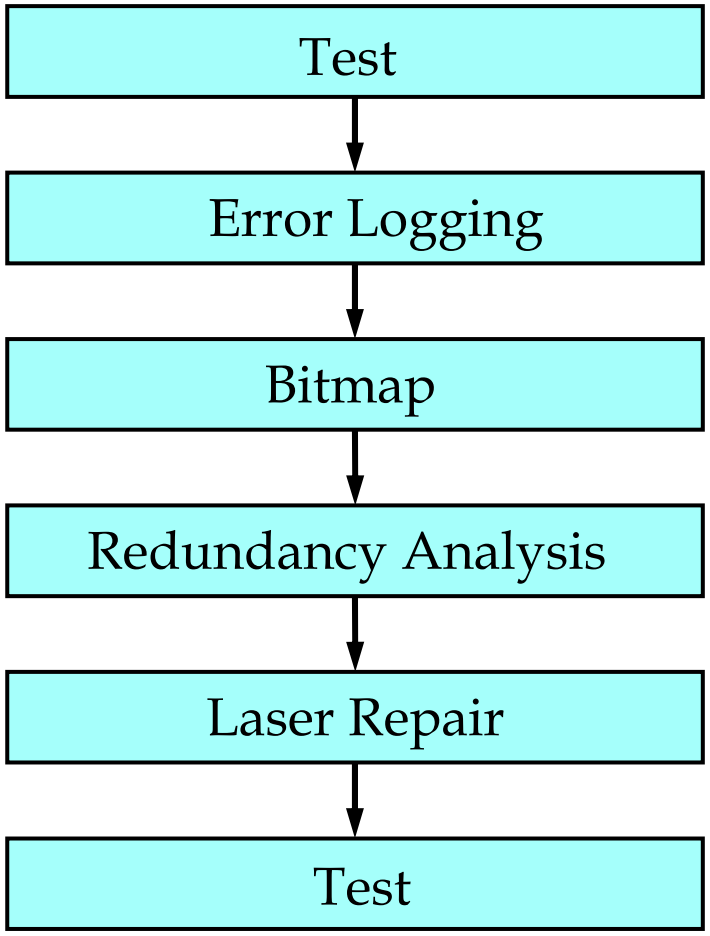


Intel dual-core Intanium processor (JSSC, 2006)

Memory Repair

- Repair is one popular technique for memory yield improvement
- Memory repair consists of three basic steps
  - Test
  - Redundancy analysis
  - Repair delivery

Conventional Memory Repair Flow



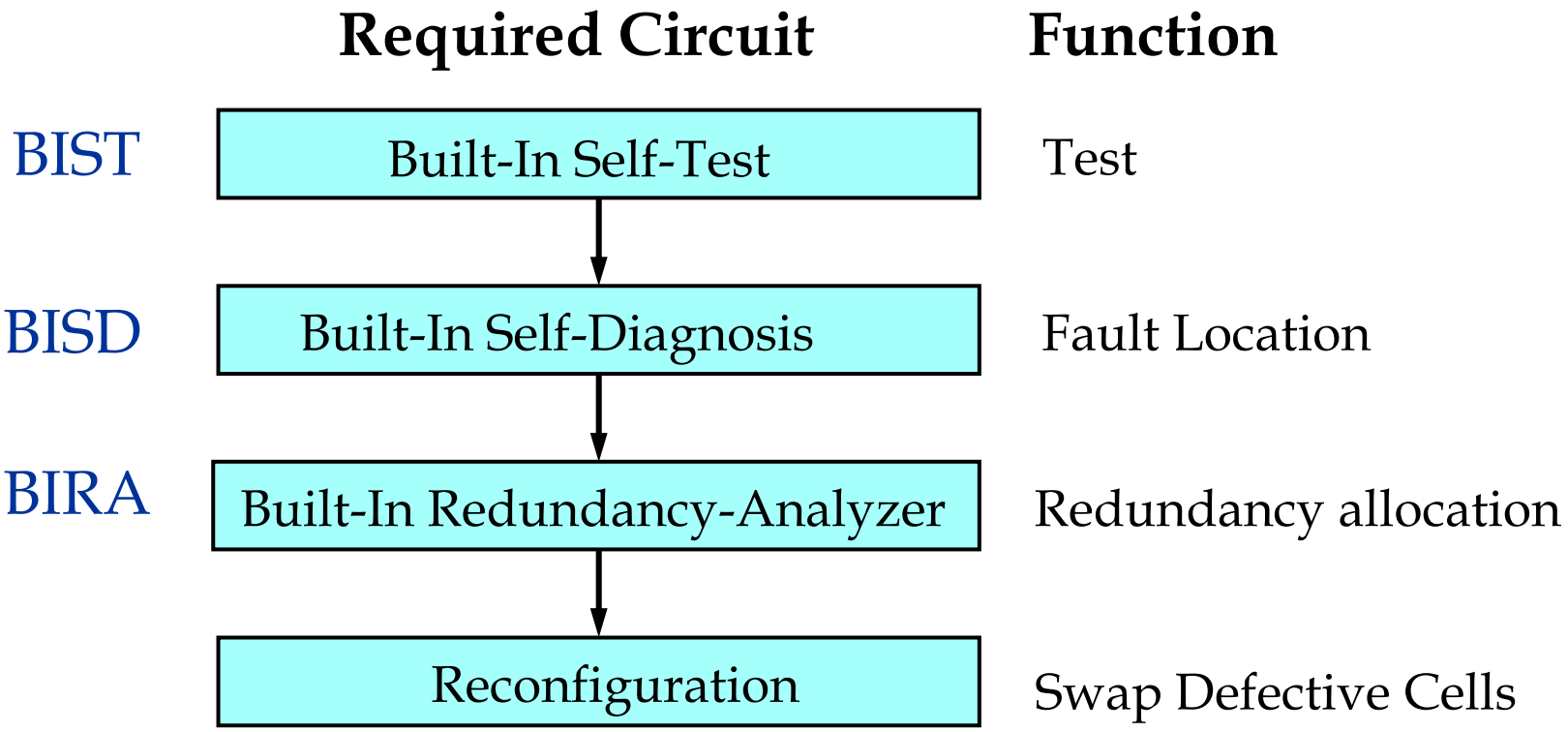
Requirements:

1. Memory tester
2. Laser repair equipment

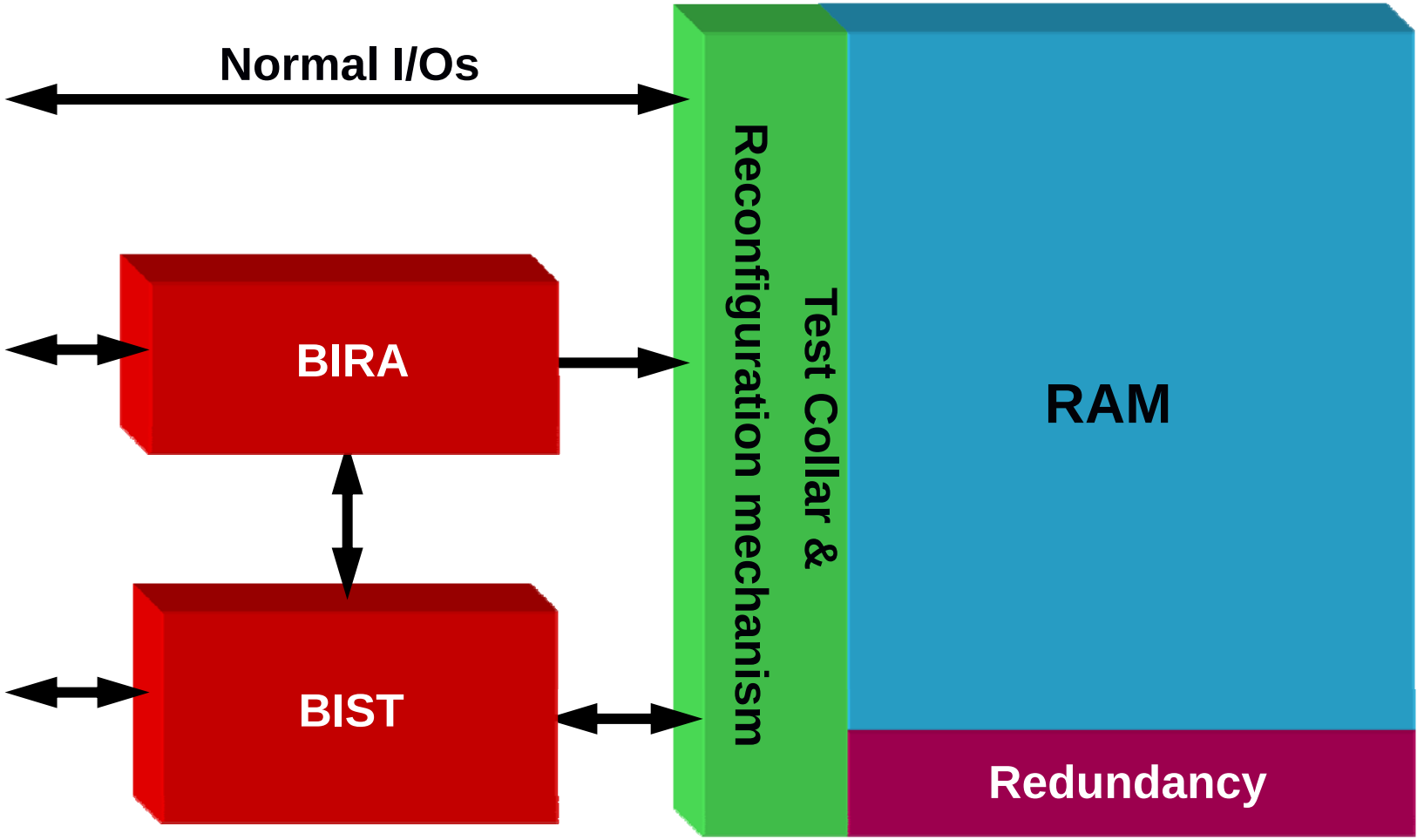
Disadvantages:

1. Time consuming
2. Expensive

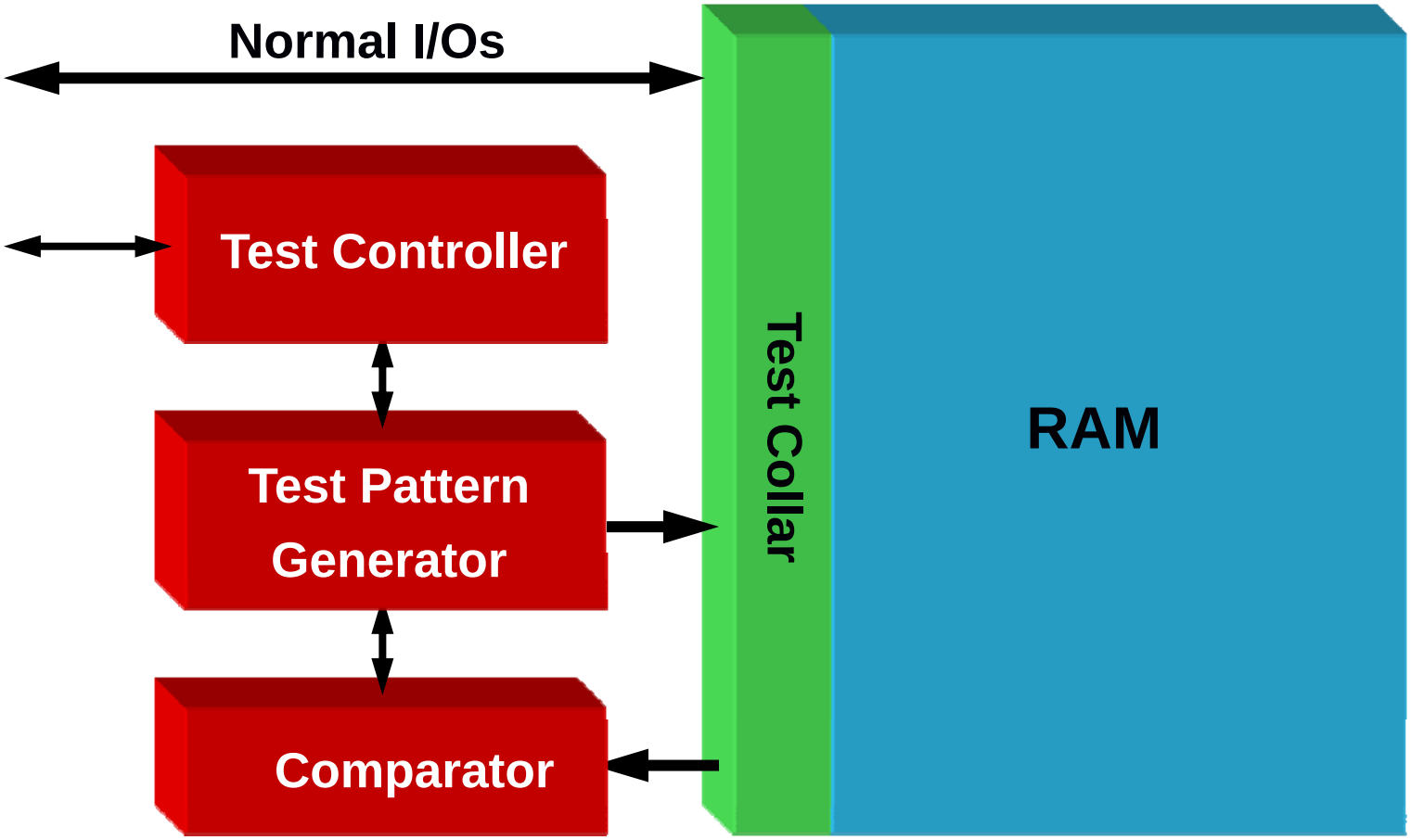
Memory BISR Flow



Typical Memory BISR Architecture

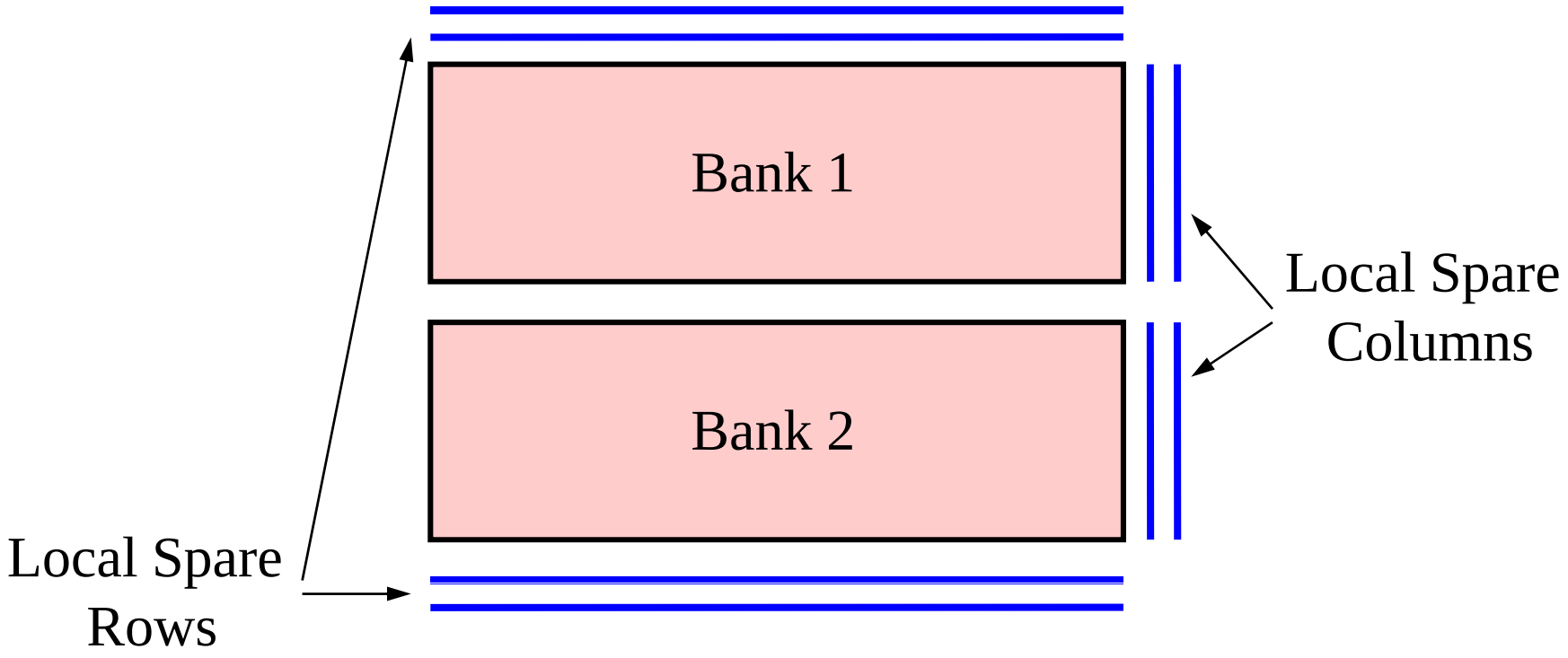


Typical Memory BIST Architecture



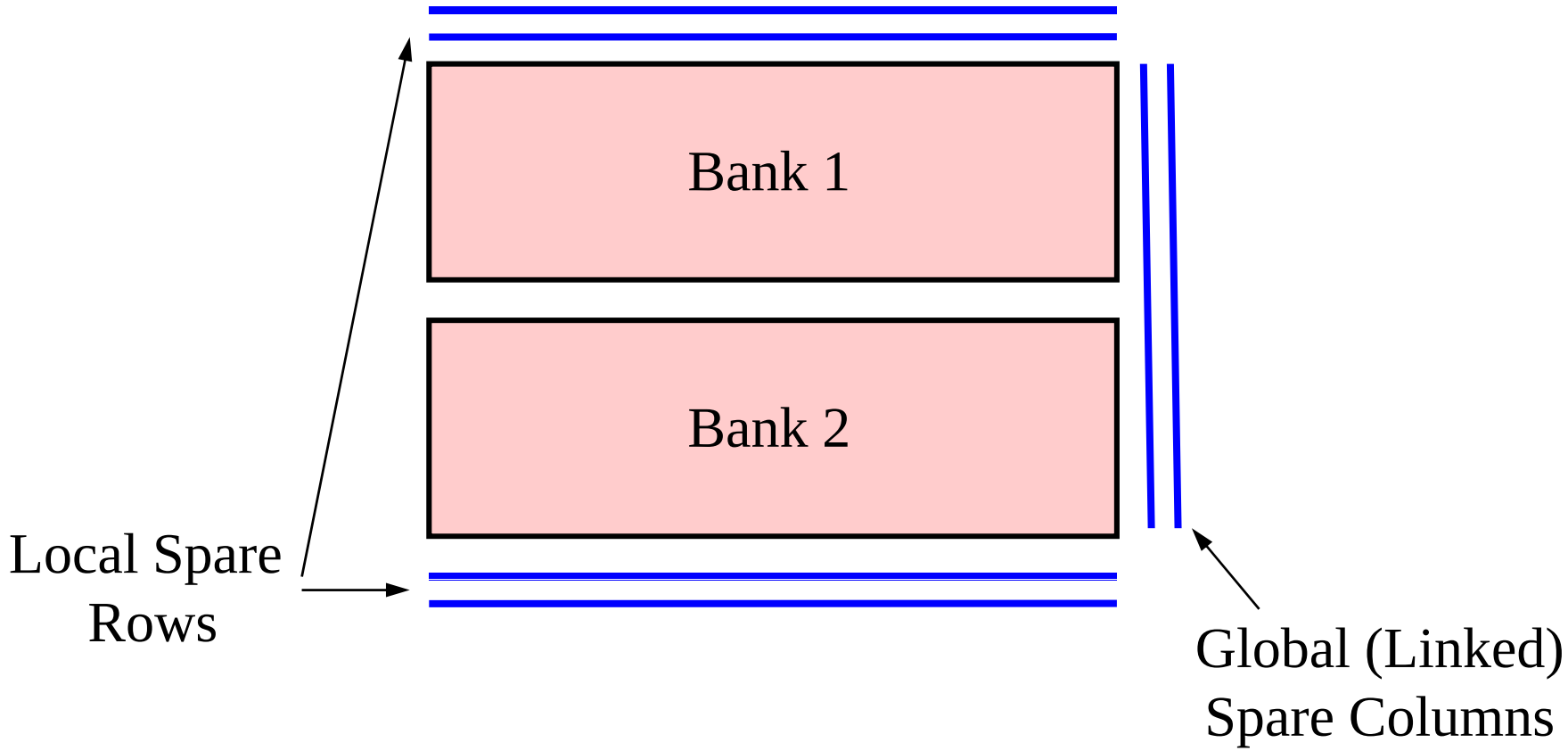
Redundancy Organizations

- A memory array with local redundancies



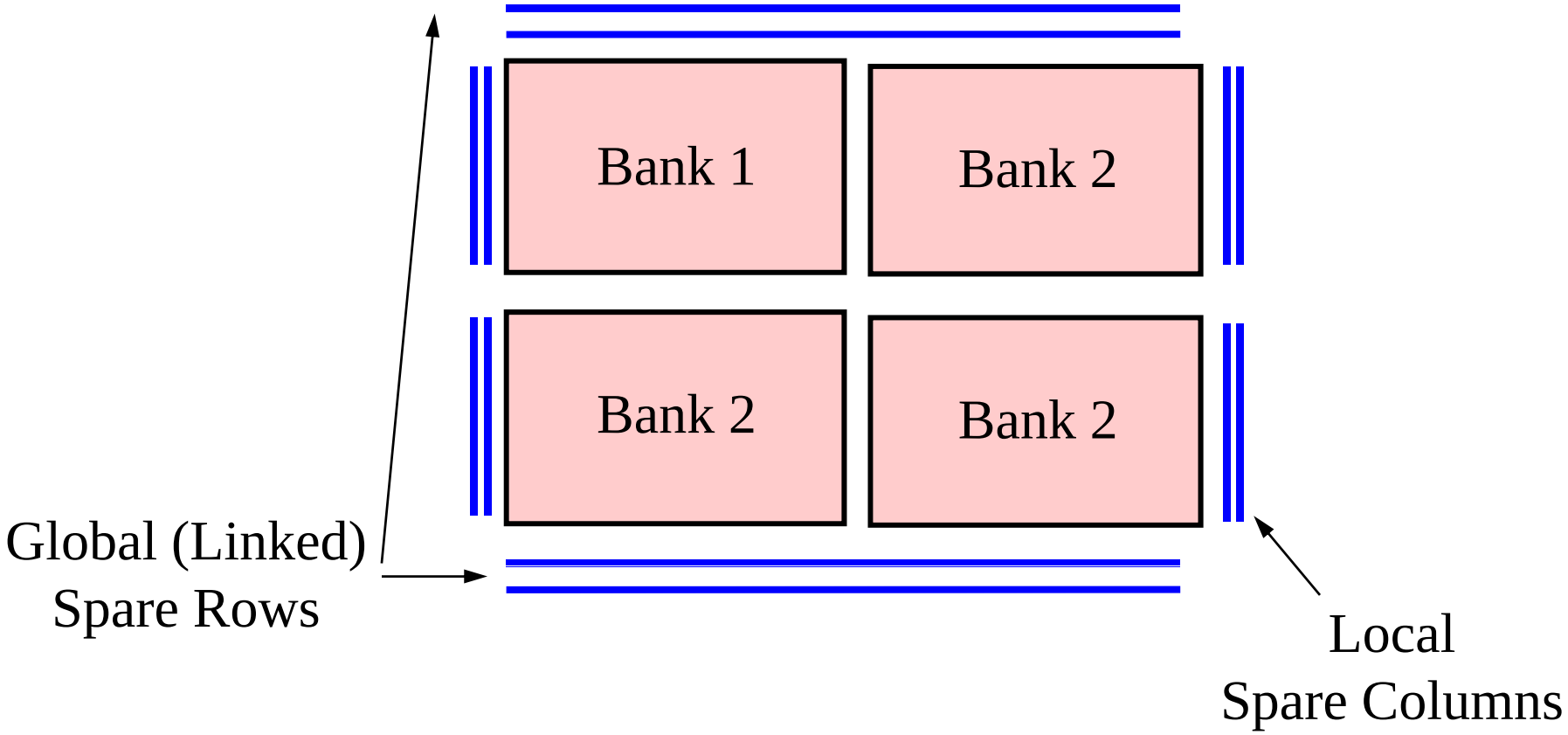
Redundancy Organizations

- A memory array with hybrid redundancies



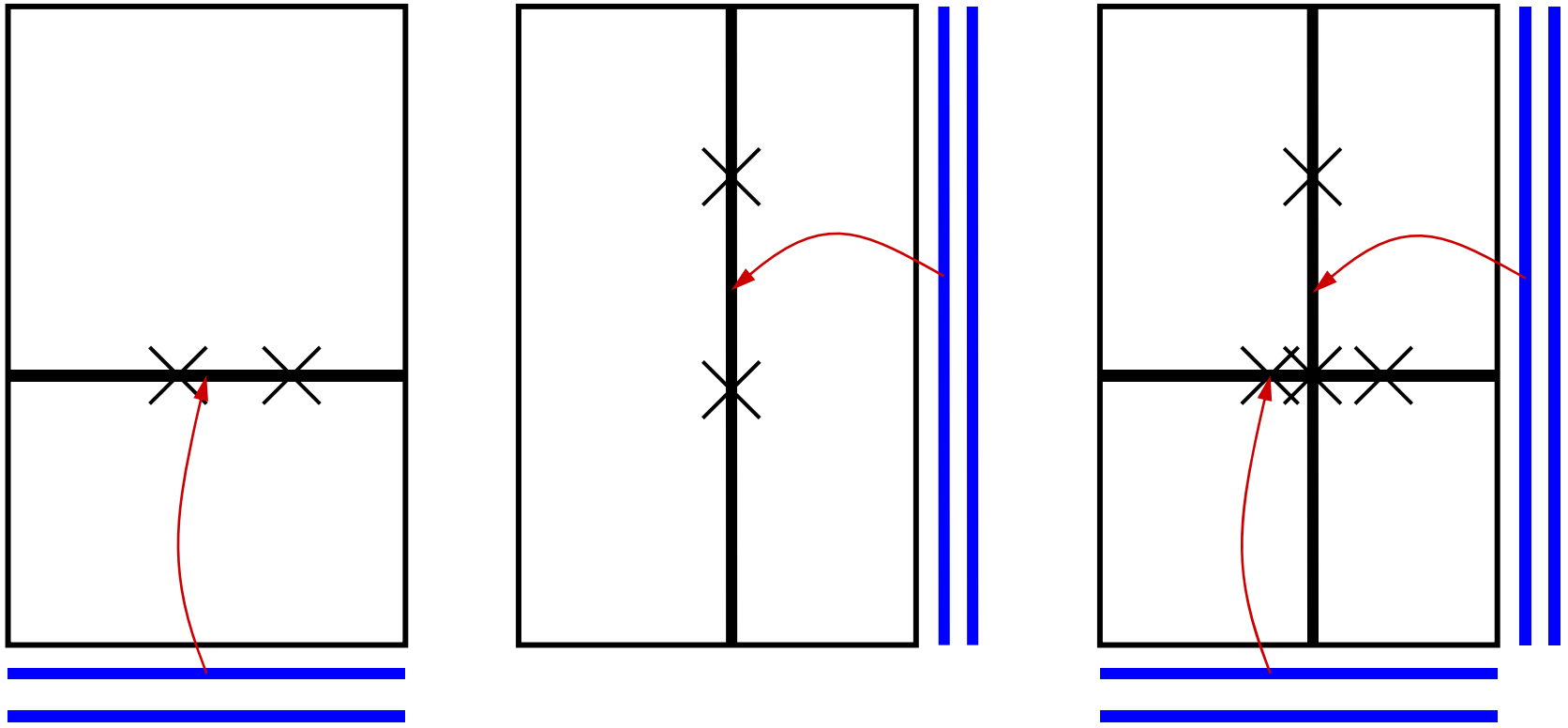
Redundancy Organizations

- A memory array with hybrid redundancies



Redundancy Scheme

- Three typical local redundancy schemes

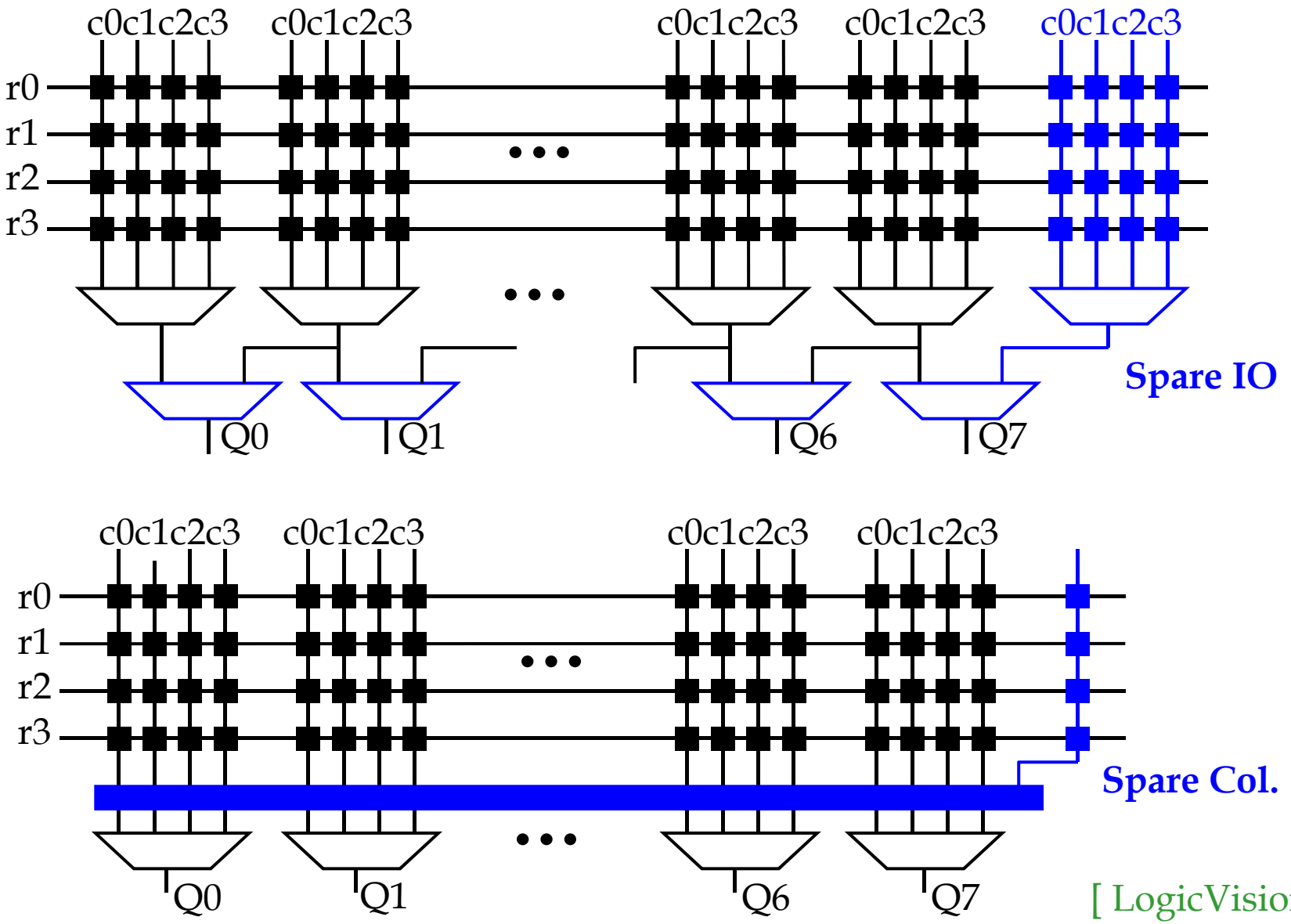


Spare rows

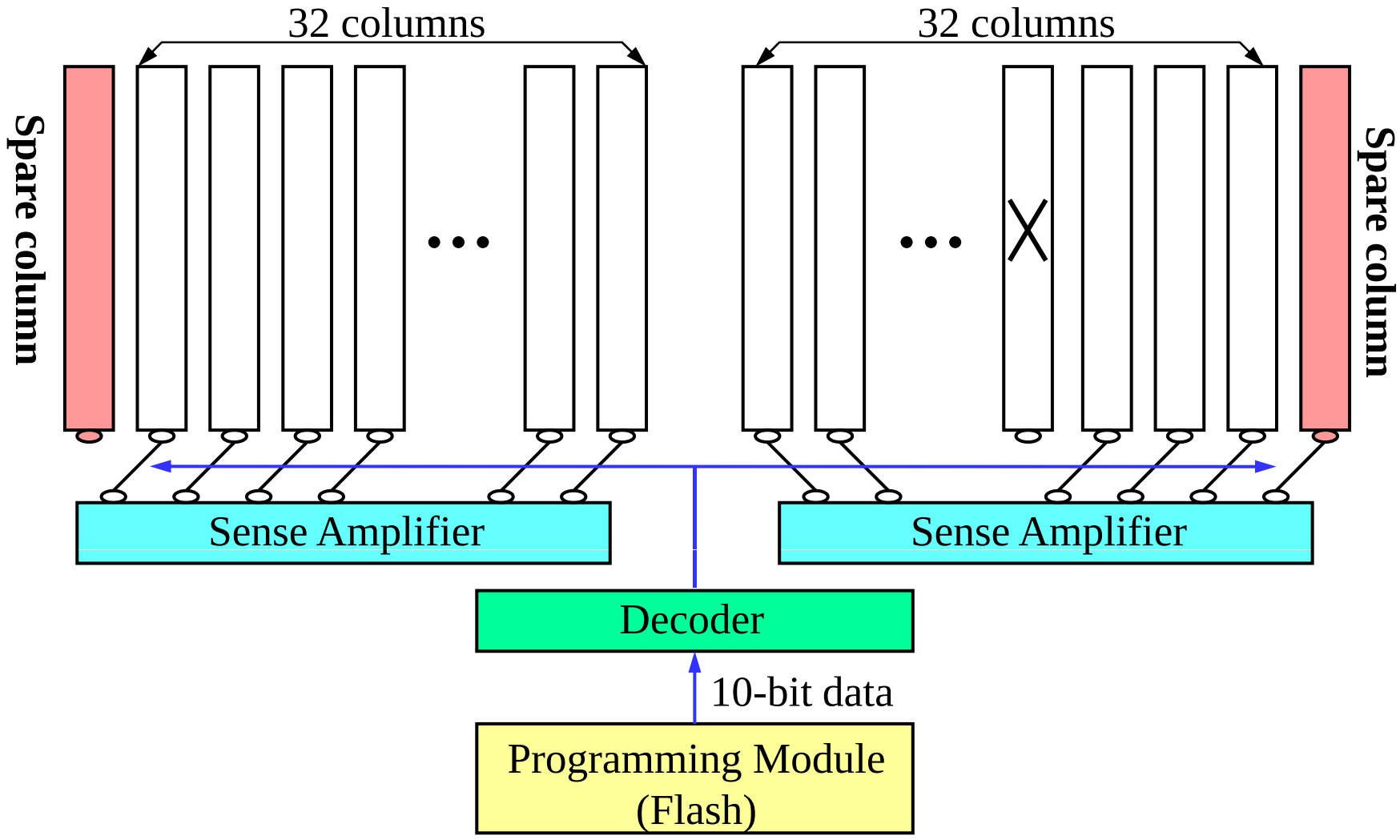
Spare columns

Spare rows and
Spare columns

Spare Column & Spare IO



Reconfiguration Scheme



[M. Yarmaoka, et al., JSSC, 2002]

Types of Reconfiguration Schemes

- Three kinds of reconfiguration techniques
  - Soft reconfiguration
    - By programming FFs to store repair information
  - Firm reconfiguration
    - By programming non-volatile memories to store repair information
  - Hard (permanent) reconfiguration
    - Laser-blown or electrically-blown polysilicon or diffusion fuses

Comparison

	Advantages	Disadvantages
Soft	1. Multi-time repair 2. Low design overhead	1. Some latent defects cannot be repaired 2. Long repair setup time
Firm	1. Multi-time repair 2. Short repair setup time	1. High-voltage programming circuit is required
Hard	1. Short repair setup time	1. One-time repair 2. Specific technology is required

Memory BISR Techniques

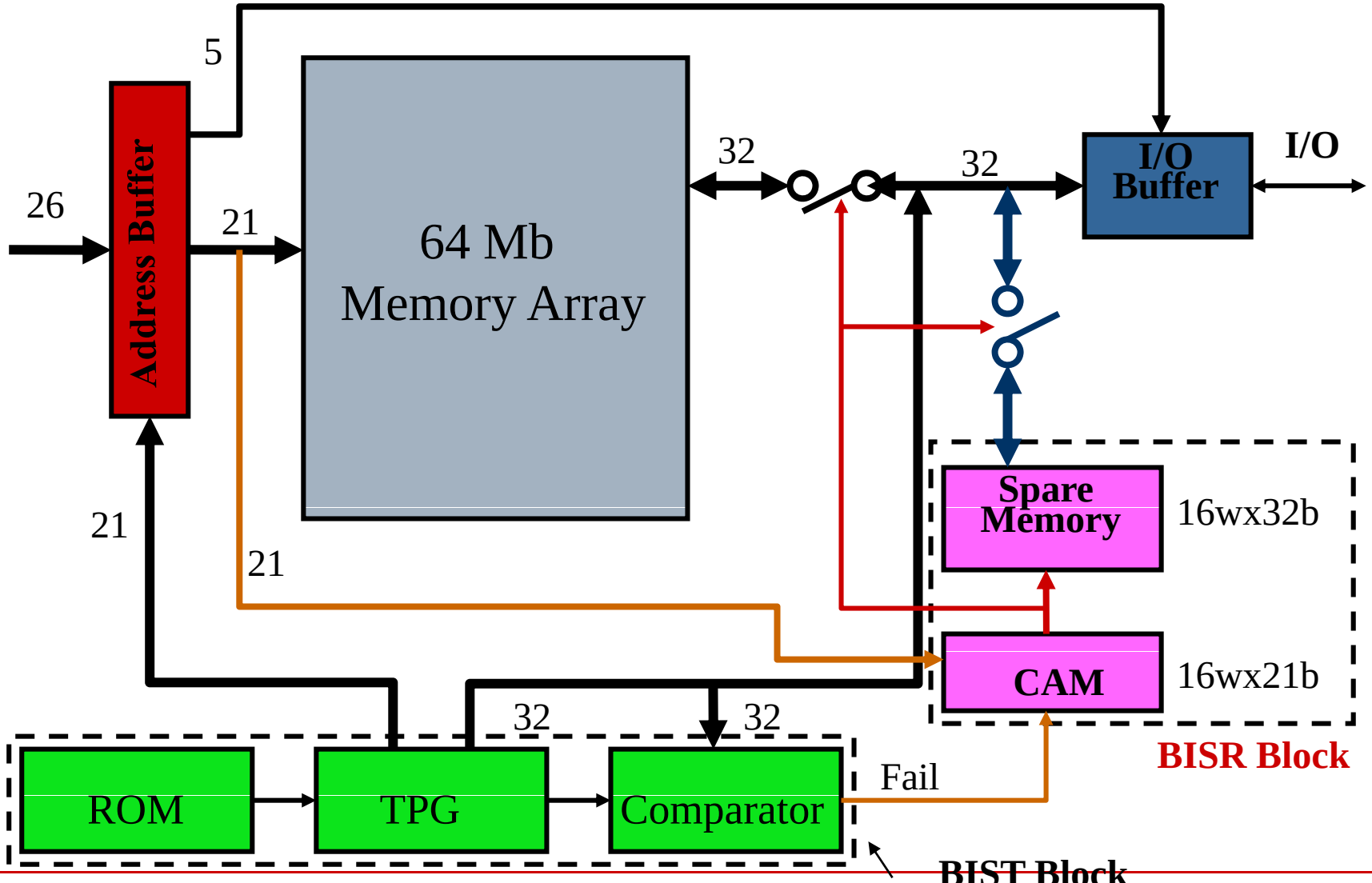
- Dedicated BISR scheme
  - A RAM has a self-contained BISR circuit
- Shared BISR scheme
  - Multiple RAMs share a BISR circuit
  - E.g., processor-based BISR scheme and IP-based BISR scheme
- BISR classification according to the capability of redundancy analysis
  - BISR with redundancy analysis capability
  - BISR without redundancy analysis capability

BISR Strategies

- Types of BISR
  - Off-line BISR
  - On-line BISR
- Off-line BISR without BIRA ability
  - BIST + reconfiguration mechanism
- Off-line BISR with BIRA ability
  - BIST + BIRA + reconfiguration mechanism
- On-line BISR

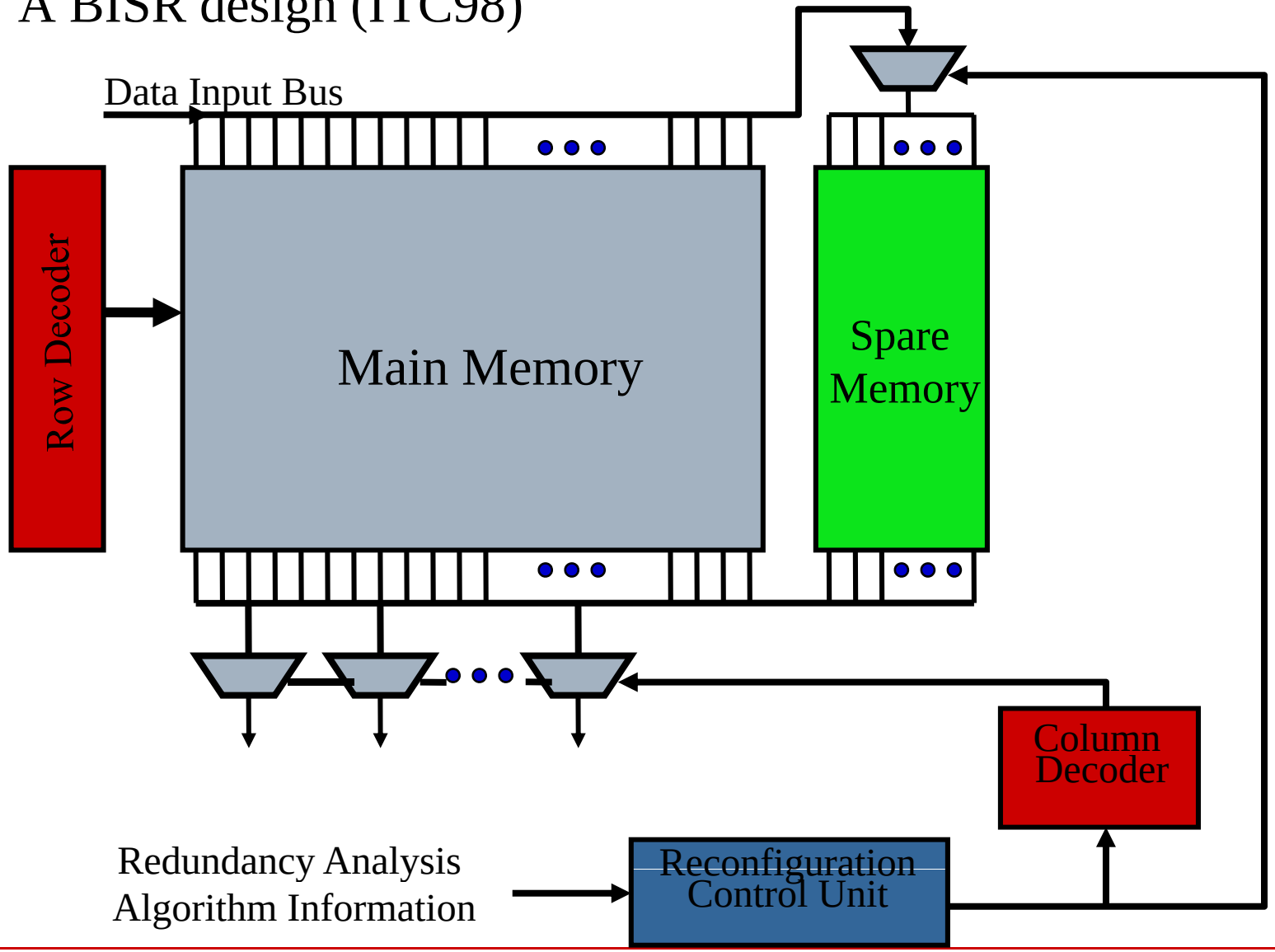
Examples of BISR Design

- NEC BISR design without BIRA (JSSC92)

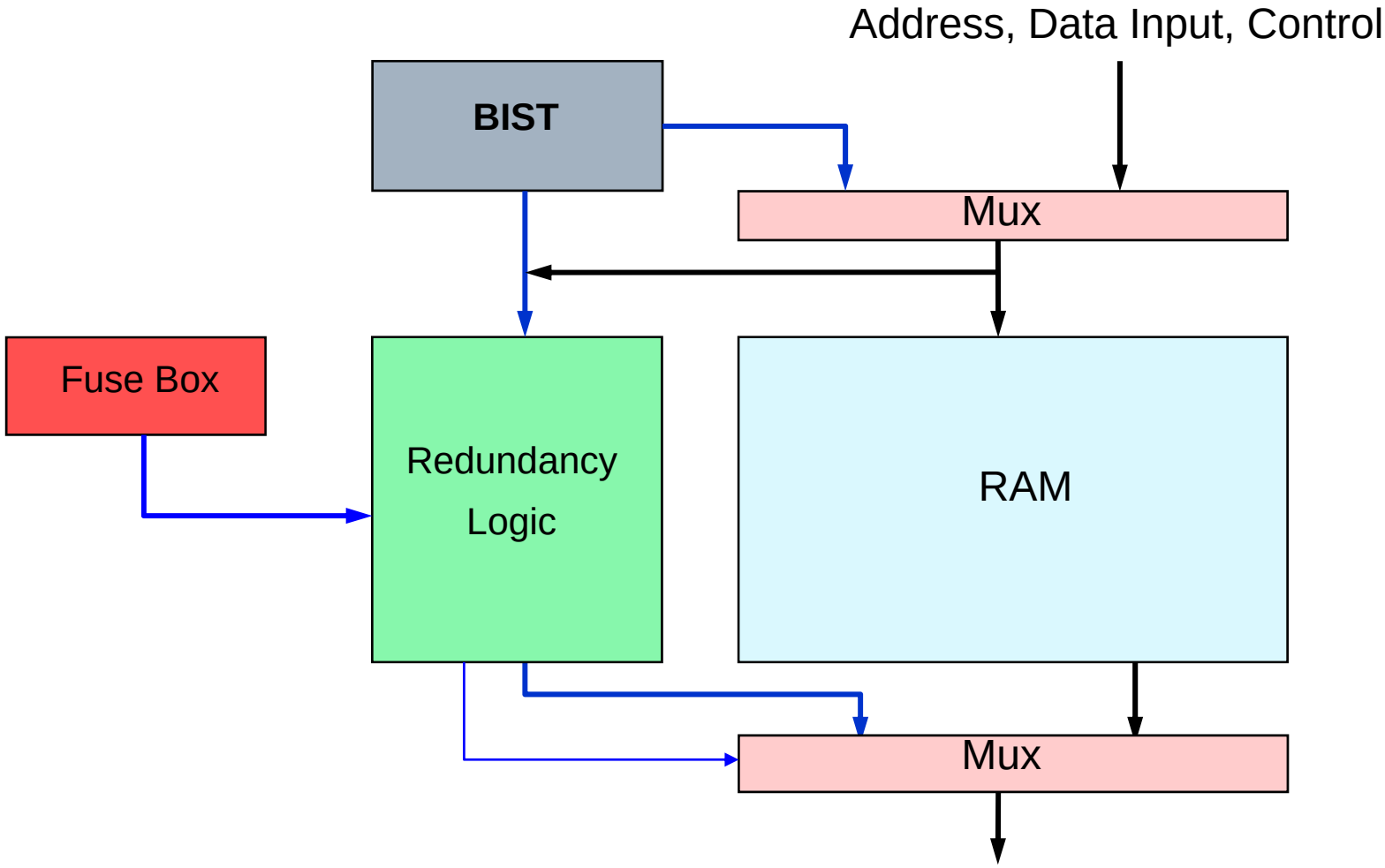


Examples of BISR Design

- A BISR design (ITC98)



RAM BISR Using Redundant Words

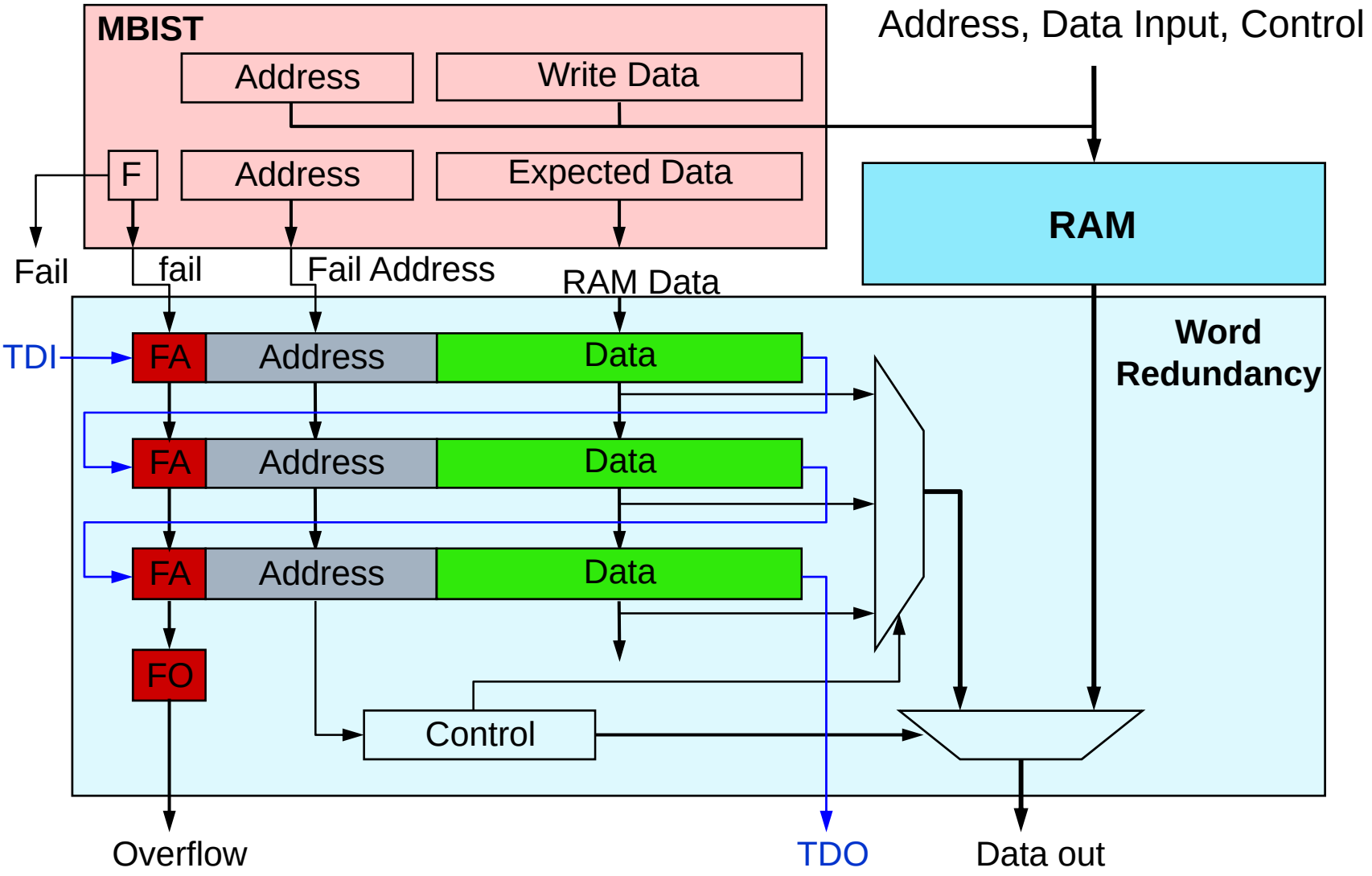


[V. Schober, et. al, ITC01]

Redundancy Wrapper Logic

- The redundancy logic consists of two basic components
  - Spare memory words
  - Logic to program the address decoding
- The address comparison is done in the redundancy logic
  - The address is compared to the addresses that are stored in the redundancy word lines
- An overflow bit identifies that there are more failing addresses than possible repair cells
- The programming of the faulty addresses is done during the memory BIST or from the fuse box during memory setup

An Array of Redundant Word Lines

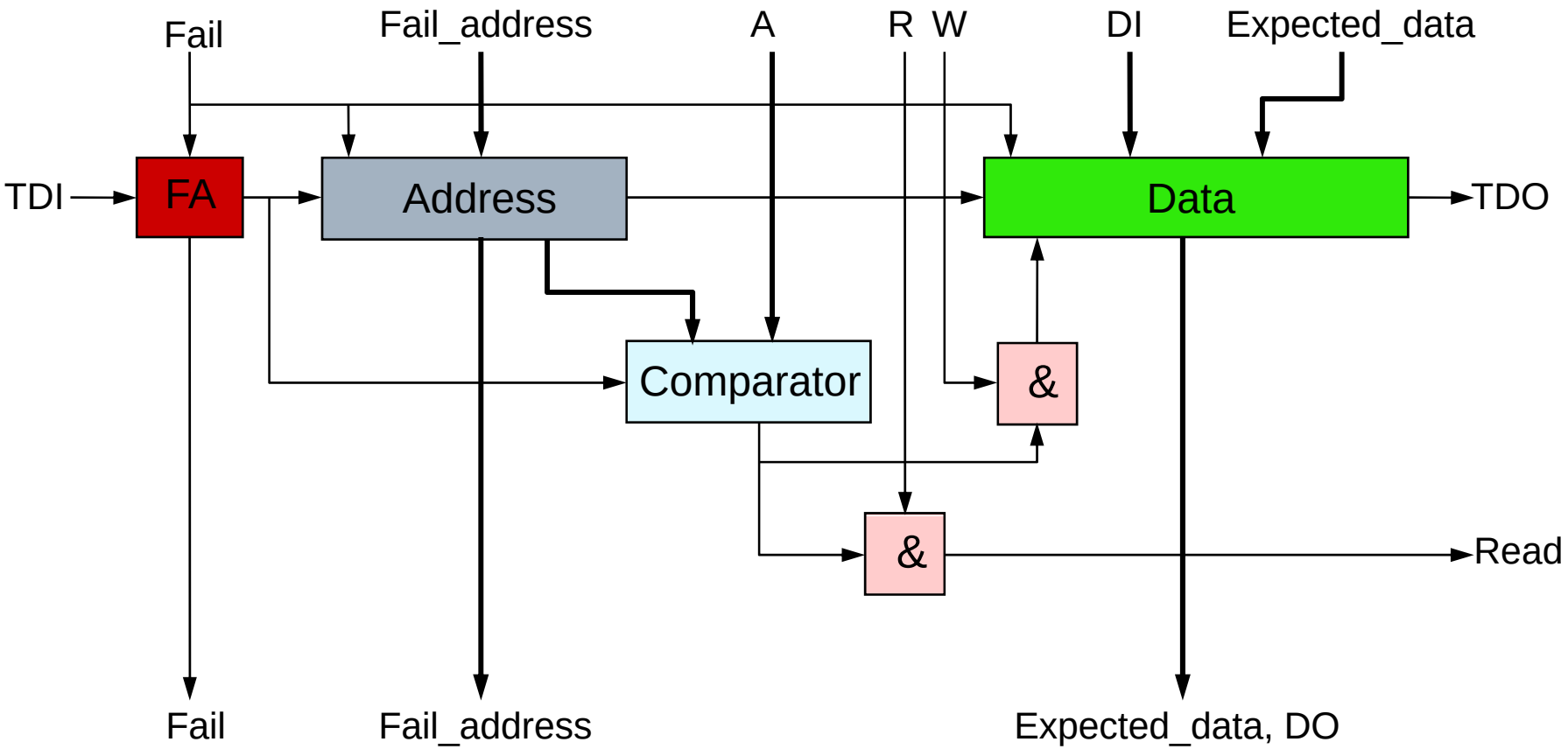


[V. Schober, et. al, ITC01]

Applications of Redundancy Logic

- Faulty addresses can be streamed out after test completion. Then the fuse box is blown accordingly in the last step of the test
  - This is called here hard repair
  - This is normally done at wafer level test
- Furthermore, the application can be started immediately after the memory BIST passes
  - This is called here soft repair

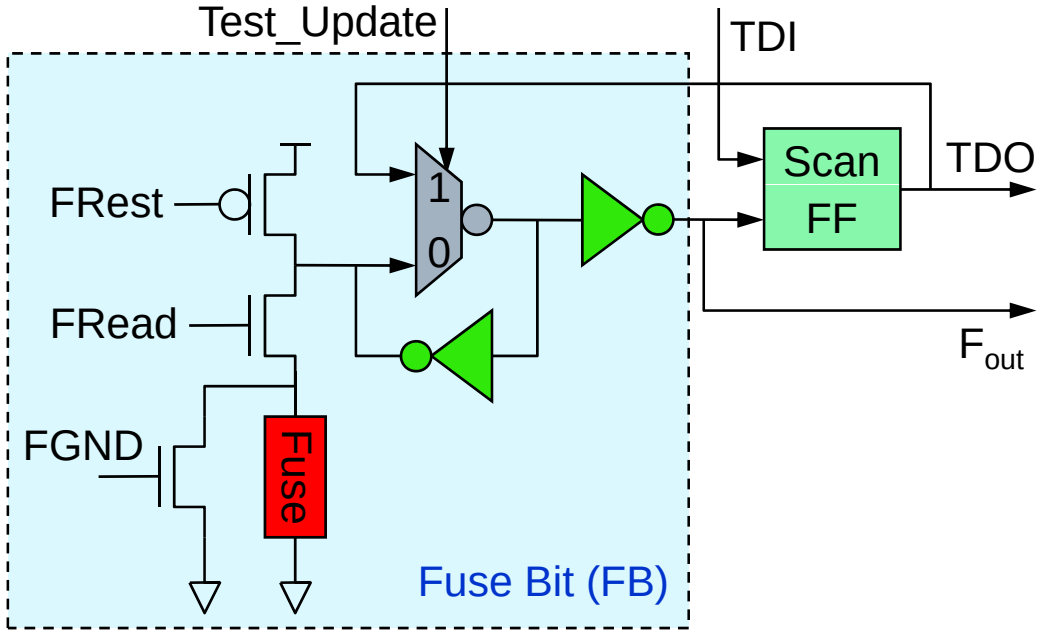
Redundancy Word Line

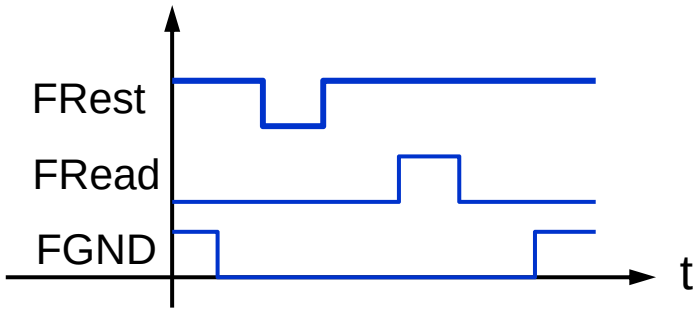


[V. Schober, et. al, ITC01]

One-Bit Fuse Box

- One-bit fuse box contains a fuse bit and a scan flip flop for controlling and observing the fuse data
  - Test_Update=0: the chain of inverters is closed (The value is latched)
  - Test_Update=1: it is possible to set the internal node from TDO
  - The ports TDI and TDO are activated at scan mode



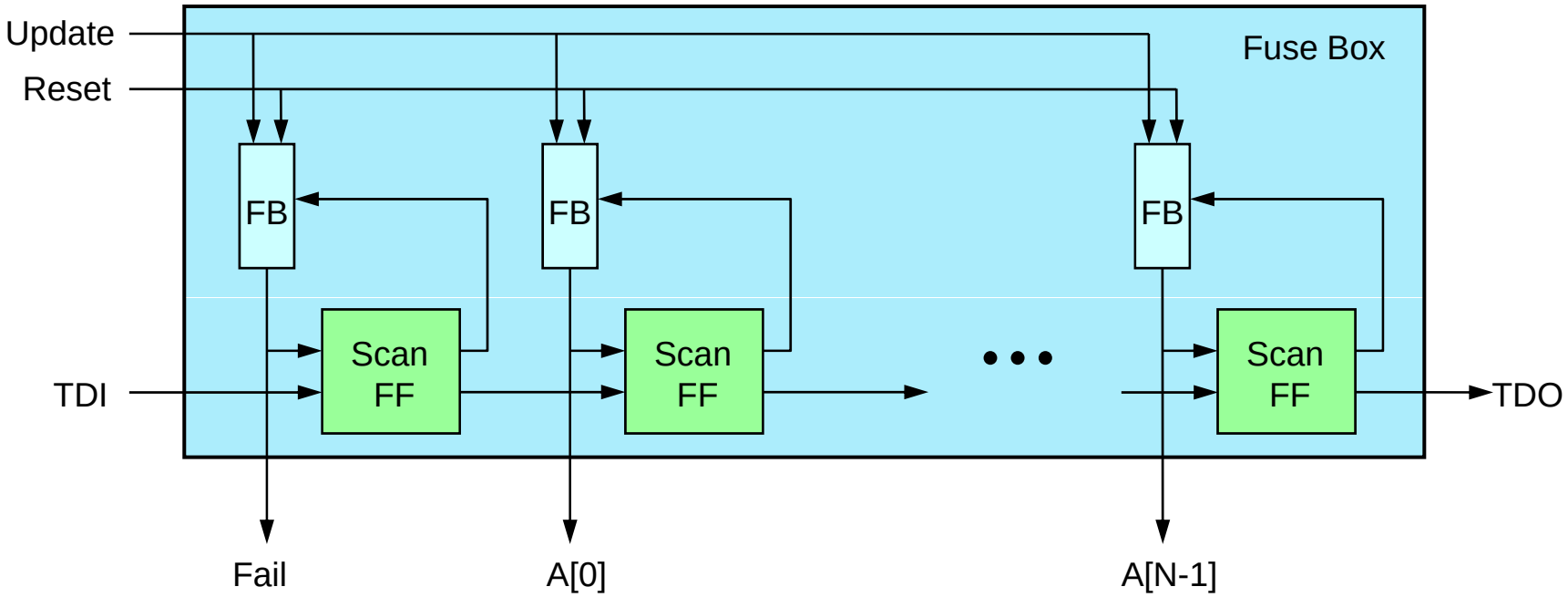


Reset cycle to read out the fuse information

[V. Schober, et. al, ITC01]

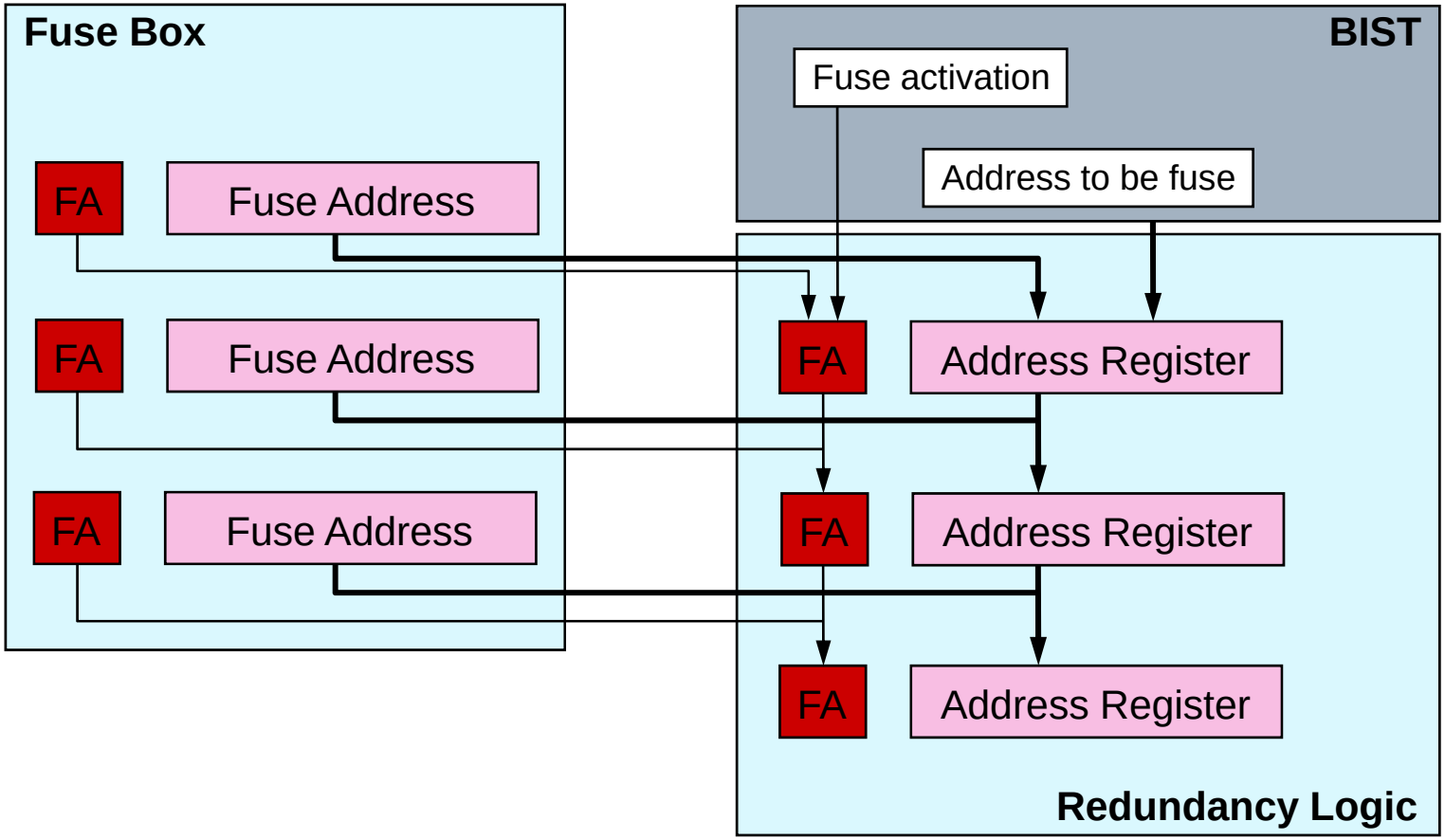
Fuse Boxes

- The fuse box can be connected to a scan register to stream in and out data during test and redundancy configuration



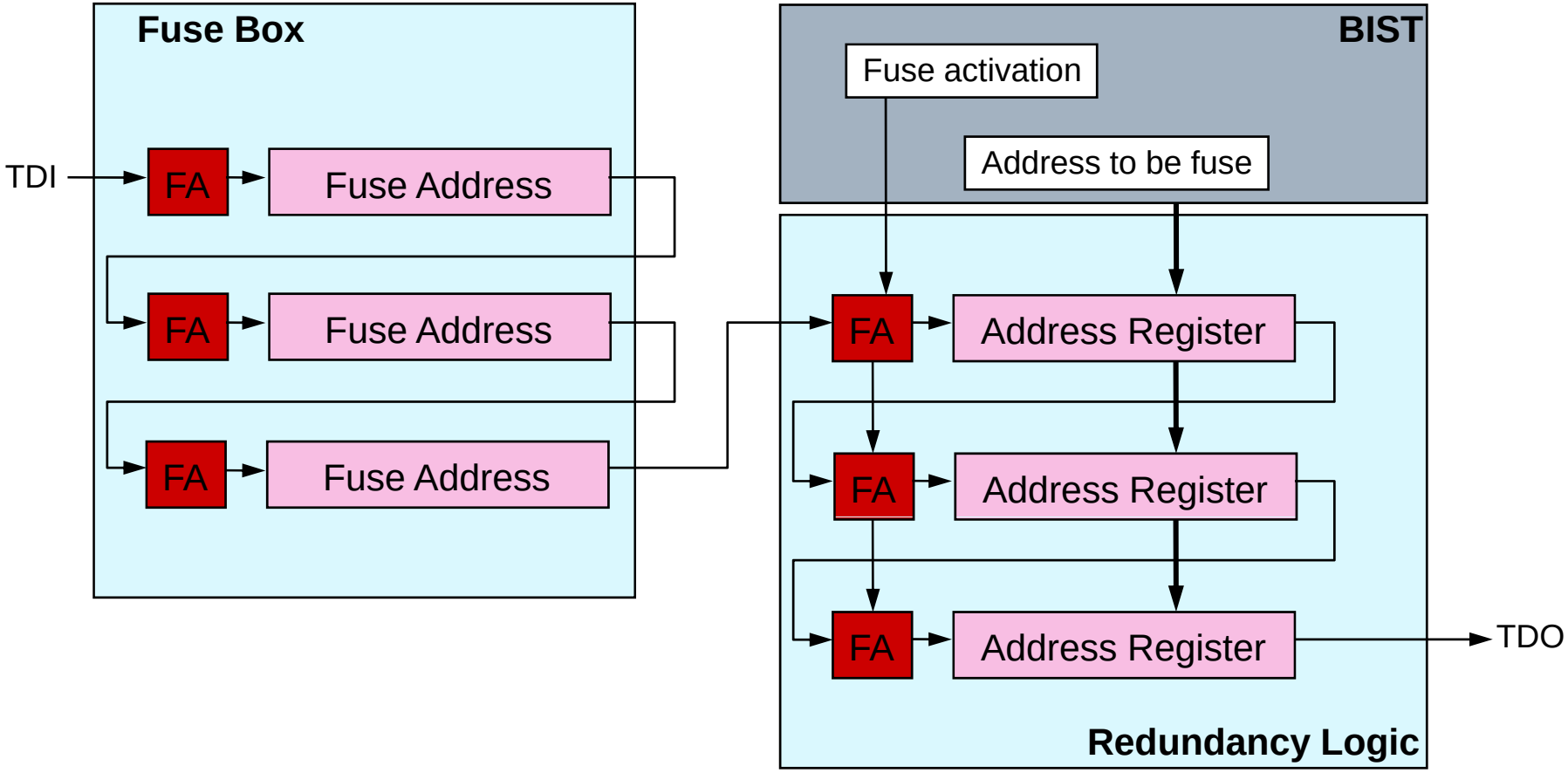
[V. Schober, et. al, ITC01]

Parallel Access of the Fuse Information



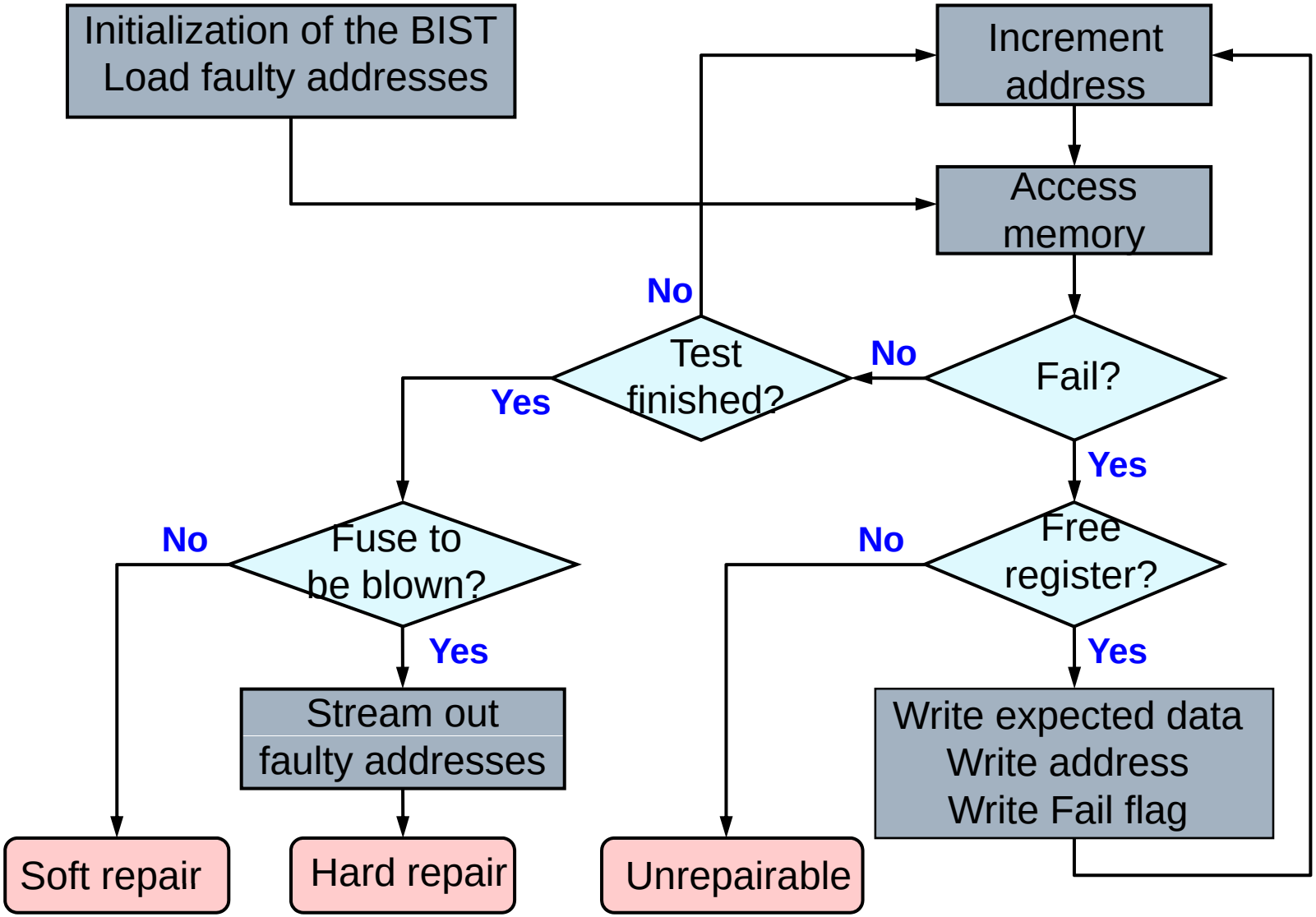
[V. Schober, et. al, ITC01]

Serial Access of the Fuse Information



[V. Schober, et. al, ITC01]

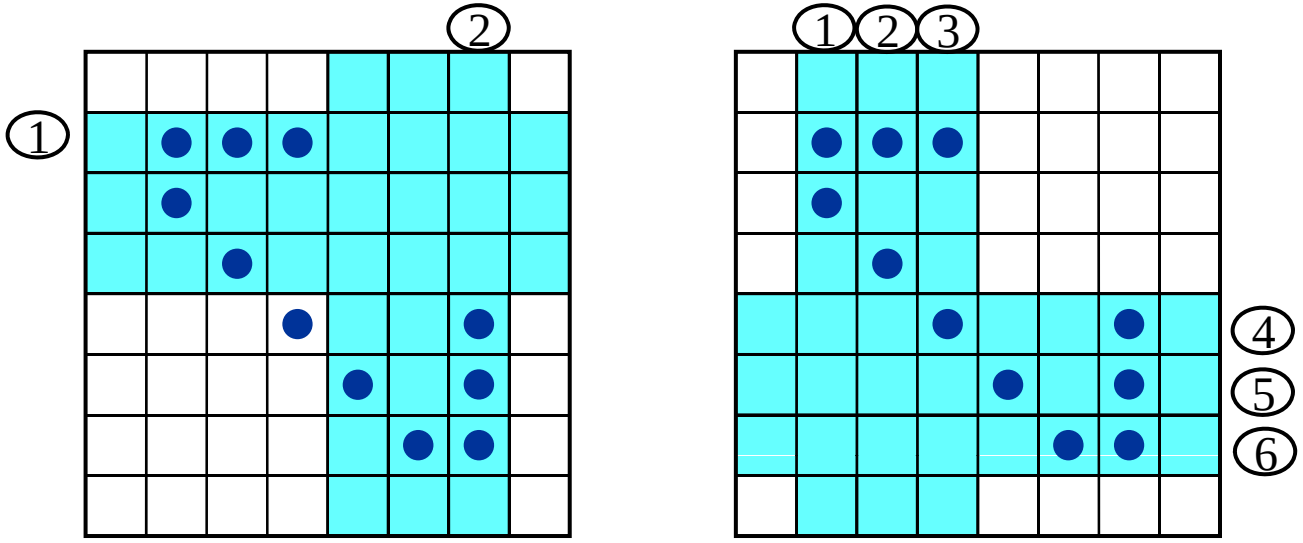
Test Flow to Activate the Redundancy



[V. Schober, et. al, ITC01]

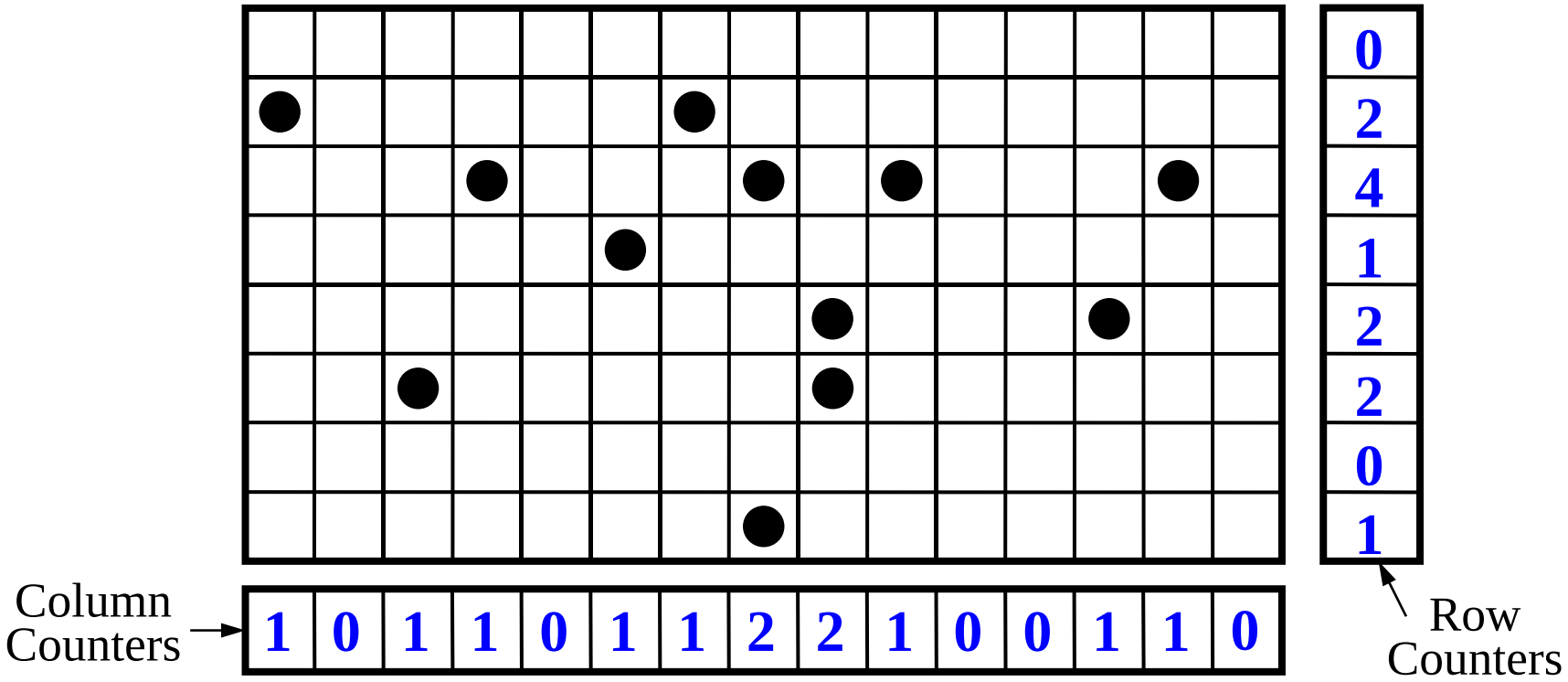
Redundancy Analysis

- A repairable memory with 1D redundancy
  - Redundancy allocation is straightforward
- A repairable memory with 2D redundancy
  - Redundancy analysis (redundancy allocation) is needed
- Redundancy analysis problem
  - Choose the minimum number of spare rows and columns that cover all the faulty cells



Redundancy Analysis Using ATE

- Create a fault map which size is the same as the memory under test



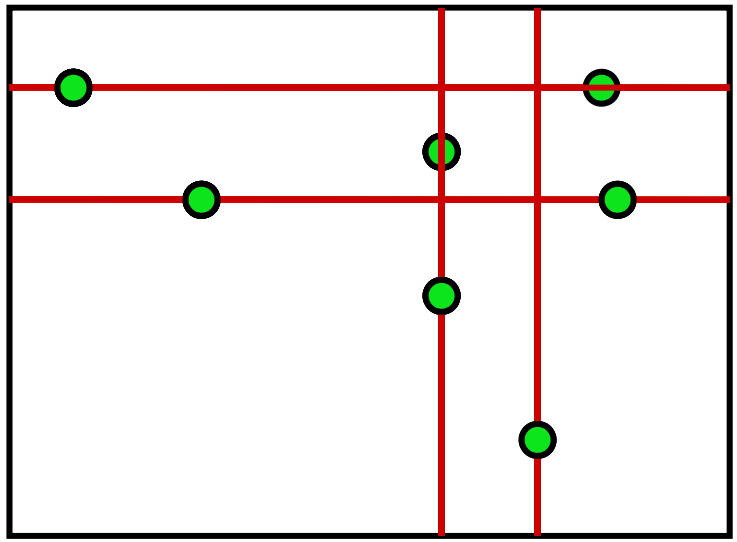
- Execute software-based redundancy analysis using computer in ATE

Redundancy Analysis Using ATE

- Hardware necessary to execute the redundancy analysis
  - A device image memory (or fault memory)
    - The size is the same as the memory under test
  - Counters that indicate the number of faults that occur in a row, or a column
- Apparently, the conventional software-based redundancy analysis algorithms are not adapted to be realized with hardware and be embedded into the SOCs
  - Hardware overhead is too large
- Efficient built-in redundancy-analysis (BIRA) algorithms are required to be developed

BIRA Algorithm – CRESTA

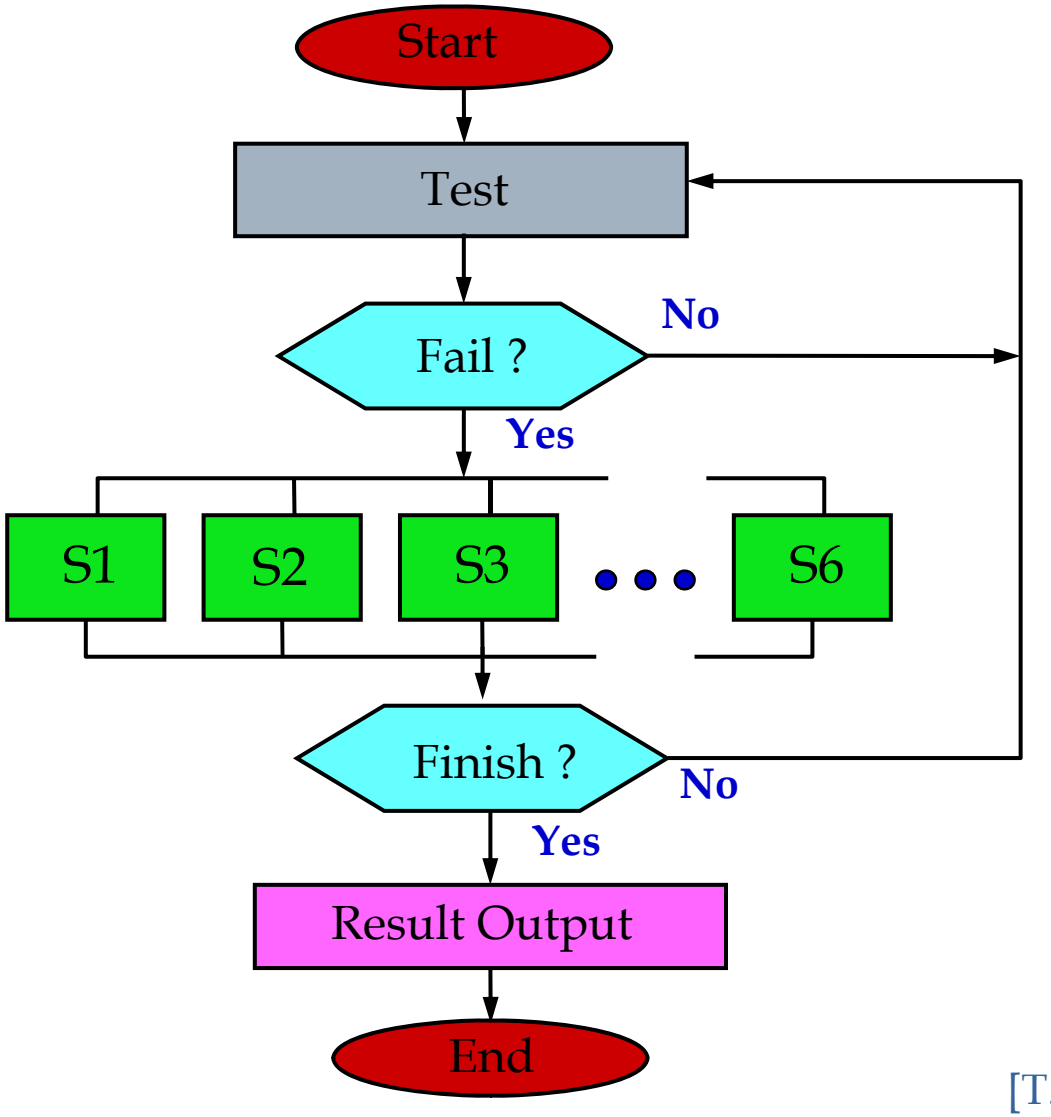
- **C**omprehensive **R**eal-time **E**xhaustive **S**earch **T**est and **A**nalysis
- Assume that a memory has 2 spare rows (Rs) & 2 spare columns (Cs), then all possible repair solutions
  - R-R-C-C (Solution 1)
  - R-C-R-C (Solution 2)
  - R-C-C-R (Solution 3)
  - C-R-R-C (Solution 4)
  - C-R-C-R (Solution 5)
  - C-C-R-R (Solution 6)



Solution 1 (R-R-C-C)

[T. Kawagoe , et. al, ITC00]

CRESTA Flow Chart



[T. Kawagoe , et. al, ITC00]

Basic Idea and Limitation of CRESTA

- Assume that there are *m* spare rows and *n* spare columns in a memory. Then a CRESTA repair analyzer contains C(*m*+*n*, *m*) sub-analyzers
  - E.g., if 2 spare rows and 2 spare columns are available, CRESTA will need C(4, 2)=6 sub-analyzers
- Each sub-analyzer analyzes in-coming row/column addresses of faulty memory cells in parallel in a different repair strategy
- Because CRESTA tries all the possible repair strategies of spare resources, it guarantees finding a solution for a repairable memory

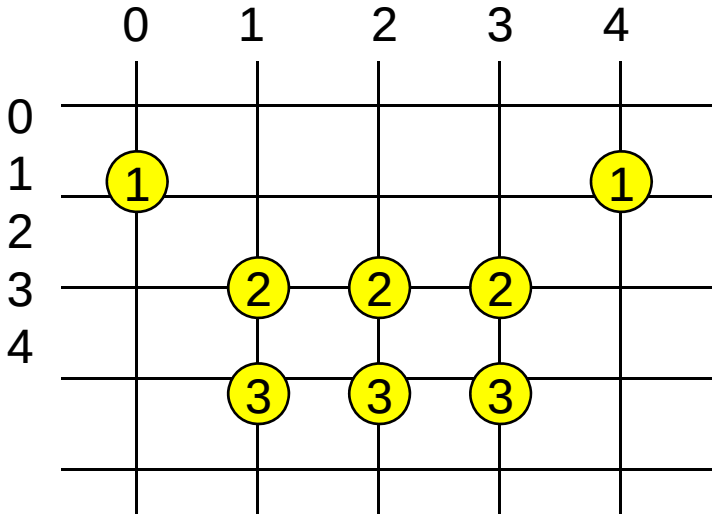
Basic Idea and Limitation of CRESTA

- Since CRESTA needs row address and column address of a faulty memory cell in order to check if the current faulty memory cell can be repaired by previously allocated spare resources
  - It is unable to handle at-speed multiple-bit failure occurring in a word-oriented memory
  - Determine the number of spare columns needed for all failure bits in a word cannot be achieved in one cycle
- In an at-speed BISR design, a column repair vector (CRV) is used to store column failure information for solving this problem
  - CRV is a column repair vector of the same size as the word width

At-Speed BIRA

- Example of redundancy allocation
  - CCRR (Unrepairable)

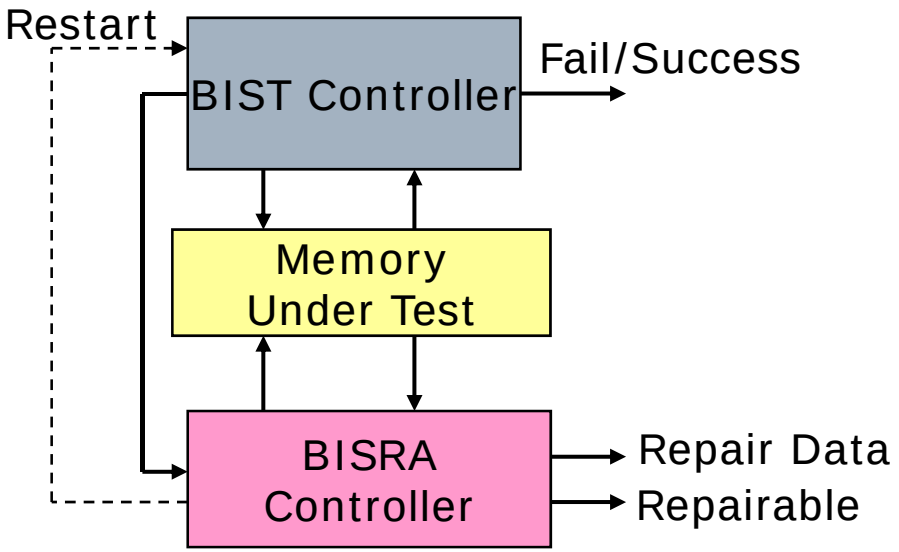
RSV	BIST Read Cycle	Fail_ Map	Current Spare	Allocated Rows	CRV
CCRR	0	00000	C1	--	00000
	1	10001	C1	--	10001
	2	01110	C2	--	11111
	3	01110	R1	--	11111
	4	00000	R1	--	11111



  - CRRC

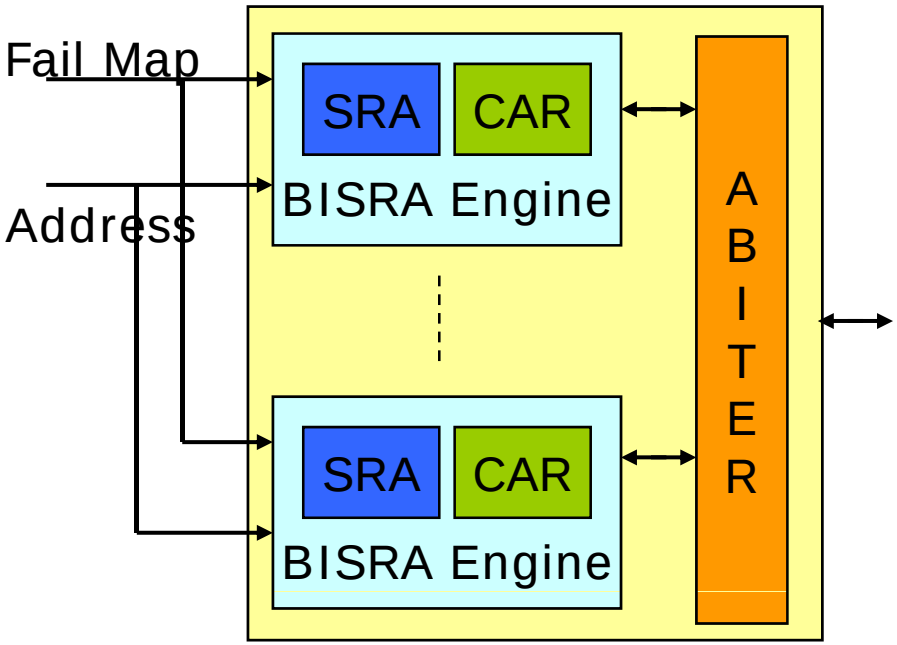
RSV	BIST Read Cycle	Fail_ Map	Current Spare	Allocated Rows	CRV
CRRC	0	00000	C1	--	00000
	1	10001	C1	--	10001
	2	01110	R1	R1	10001
	3	01110	R2	R1R2	10001
	4	00000	C2	R1R2	10001

At-Speed BIRA Implementation

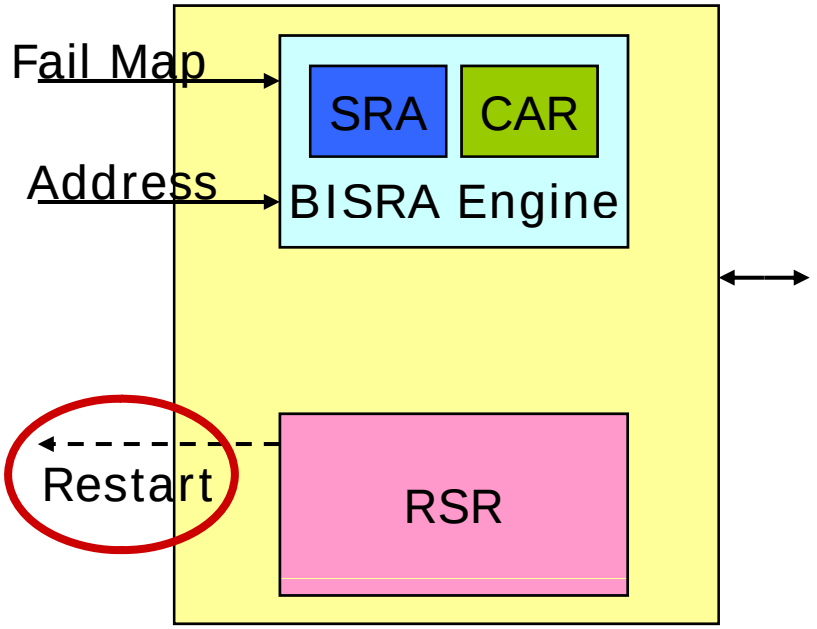


- In the BISRA, all C(m+n, m) analysis engines or just one engine can be implemented
- In one engine scheme, update the repair strategy if the current repair strategy fails and then re-run BIST and try the next repair strategy

At-Speed BIRA Implementation



BISRA controller with C(m+n, m) engines



BISRA controller with one engine

At-Speed BIRA Implementation

- Spare Resource Allocation (SRA): allocates either a spare row or a spare column according to its repair strategy
- Control and Report (CAR): checks if this repair strategy fails
  - If not, it will report the repair data, such as faulty row addresses and CRV, to BISRA controller
- Repair Strategy Reconfiguration (RSR) block: updates the repair strategy and sends a “restart” signal to BIST controller

Heuristic BIRA Algorithms

- Most of heuristic BIRA algorithms need a local bitmap for storing the information of faulty cells detected by the BIST circuit
- An example of 4×5 local bitmap

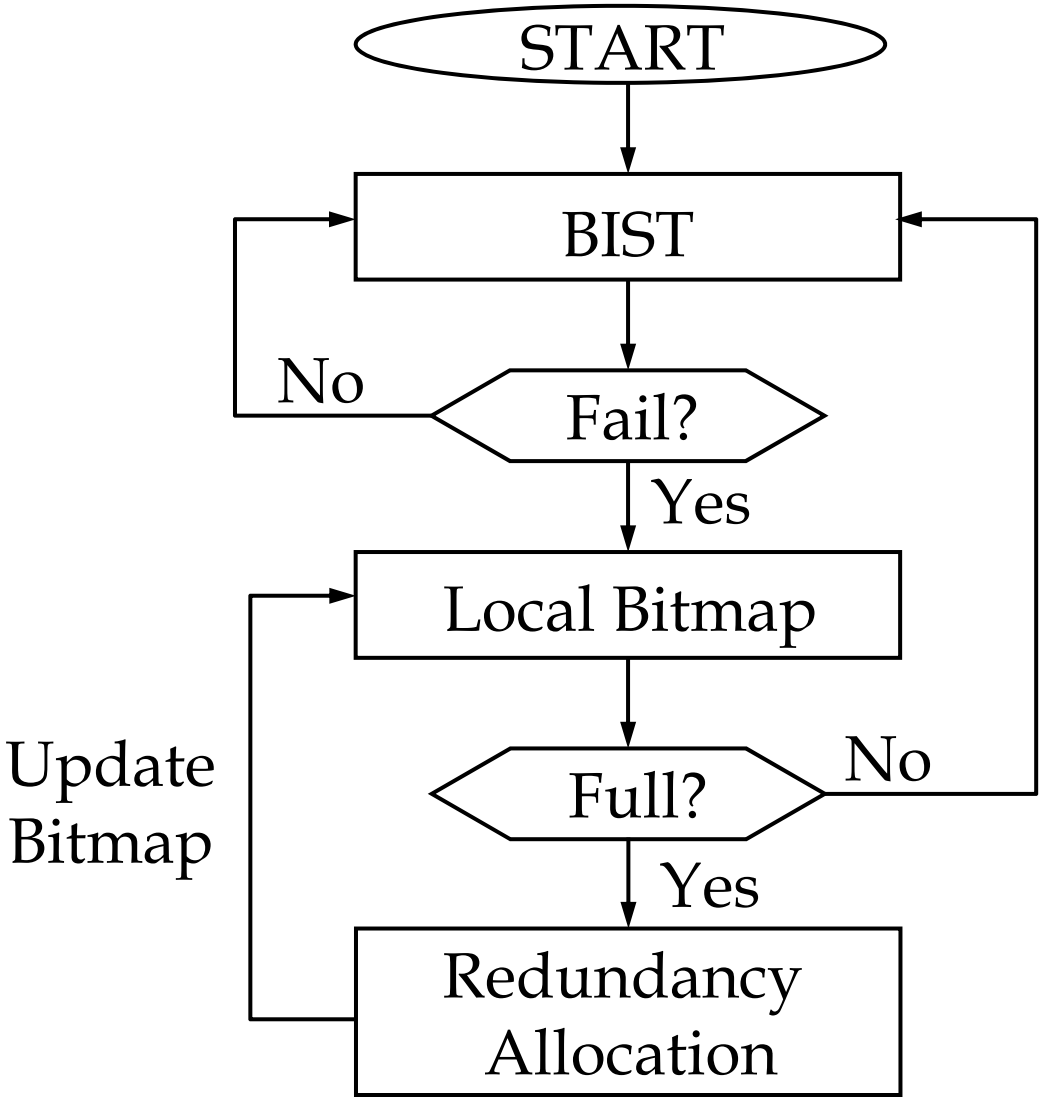
Address →

	C1	C1	C1	C1	C1
R1	1	0	0	0	0
R2	0	1	0	1	0
R3	0	0	1	0	1
R4	0	0	0	0	0

CAR

RAR

A BIRA Flow for Performing Heuristic RAs

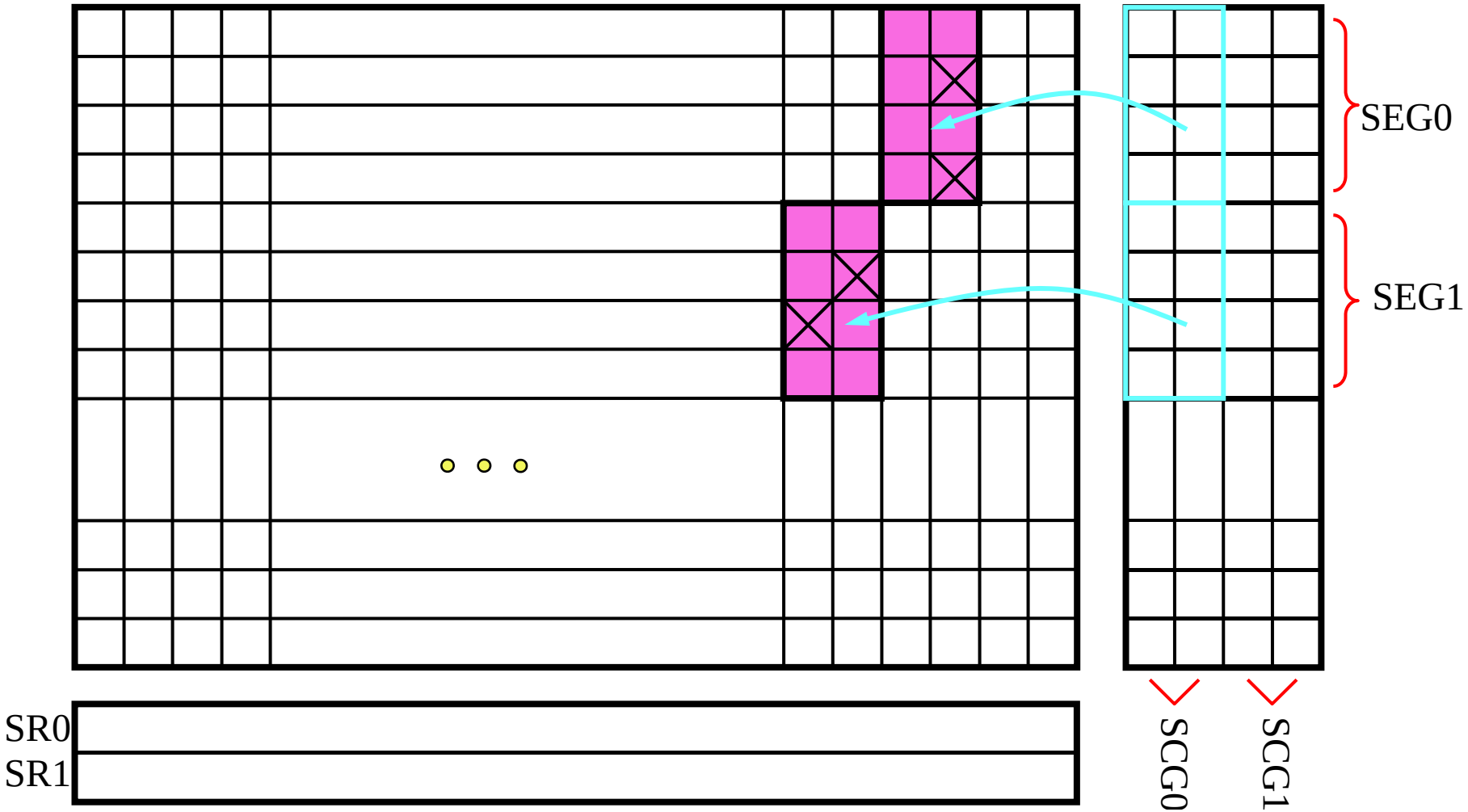


Redundancy Allocation Rules

- Typical redundancy analysis algorithms
  - Two-phase redundancy allocation procedure: *must-repair phase* and *final-repair phase*
- Must-repair phase
  - Row-must repair (column-must repair): a repair solution forced by a failure pattern with $>S_C$ ($>S_R$) defective cells in a single row (column), where S_C and S_R denote the number of available spare columns and spare rows
- Final-repair phase
  - Heuristic algorithms are usually used, e.g., repair-most rule

NTHU/ADMtek BISR Scheme

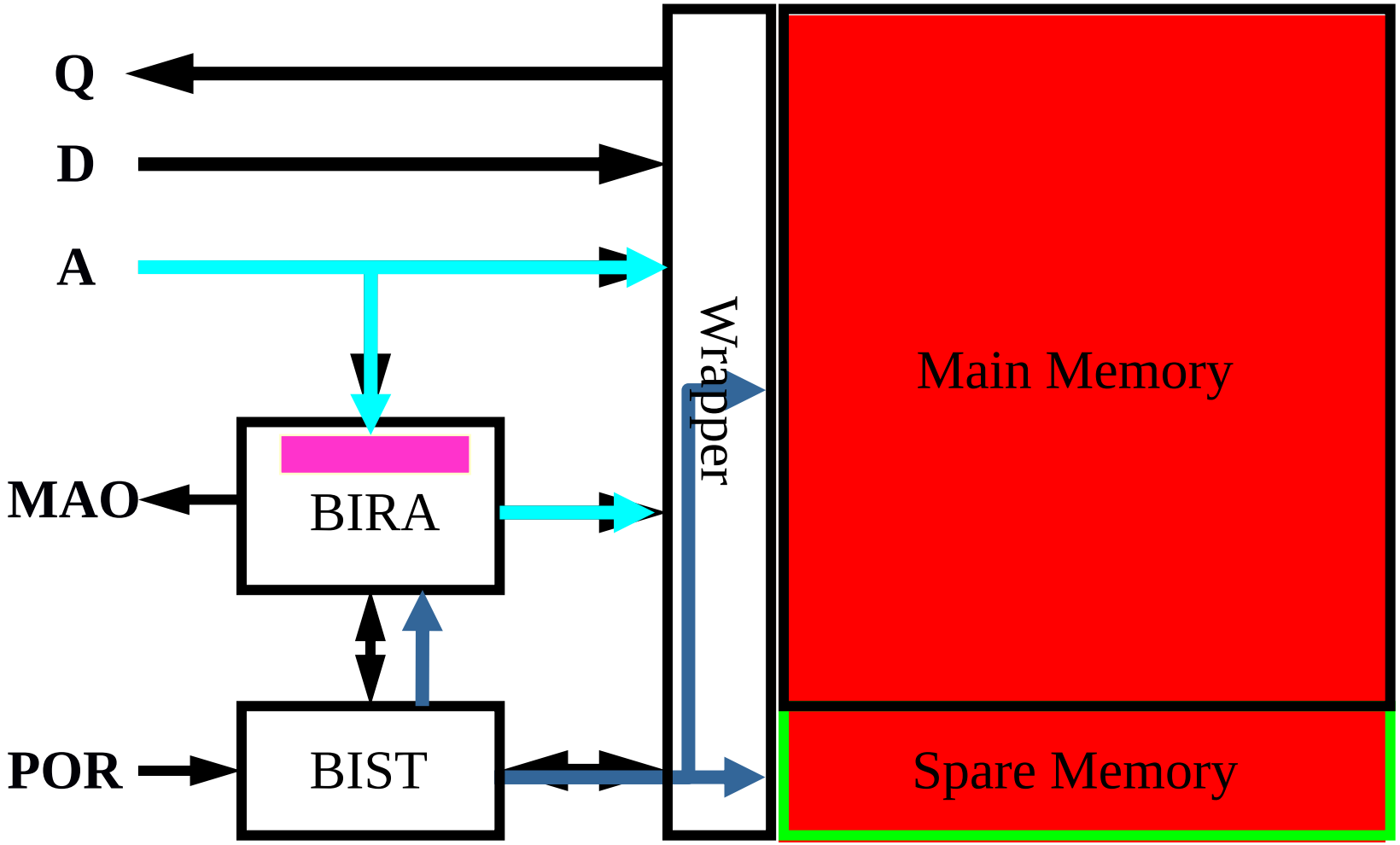
- Redundancy organization



SR: Spare Row; SCG: Spare Column Group; SEG: Segment

Dedicated BISR: NTHU/ADMtek BISR Scheme

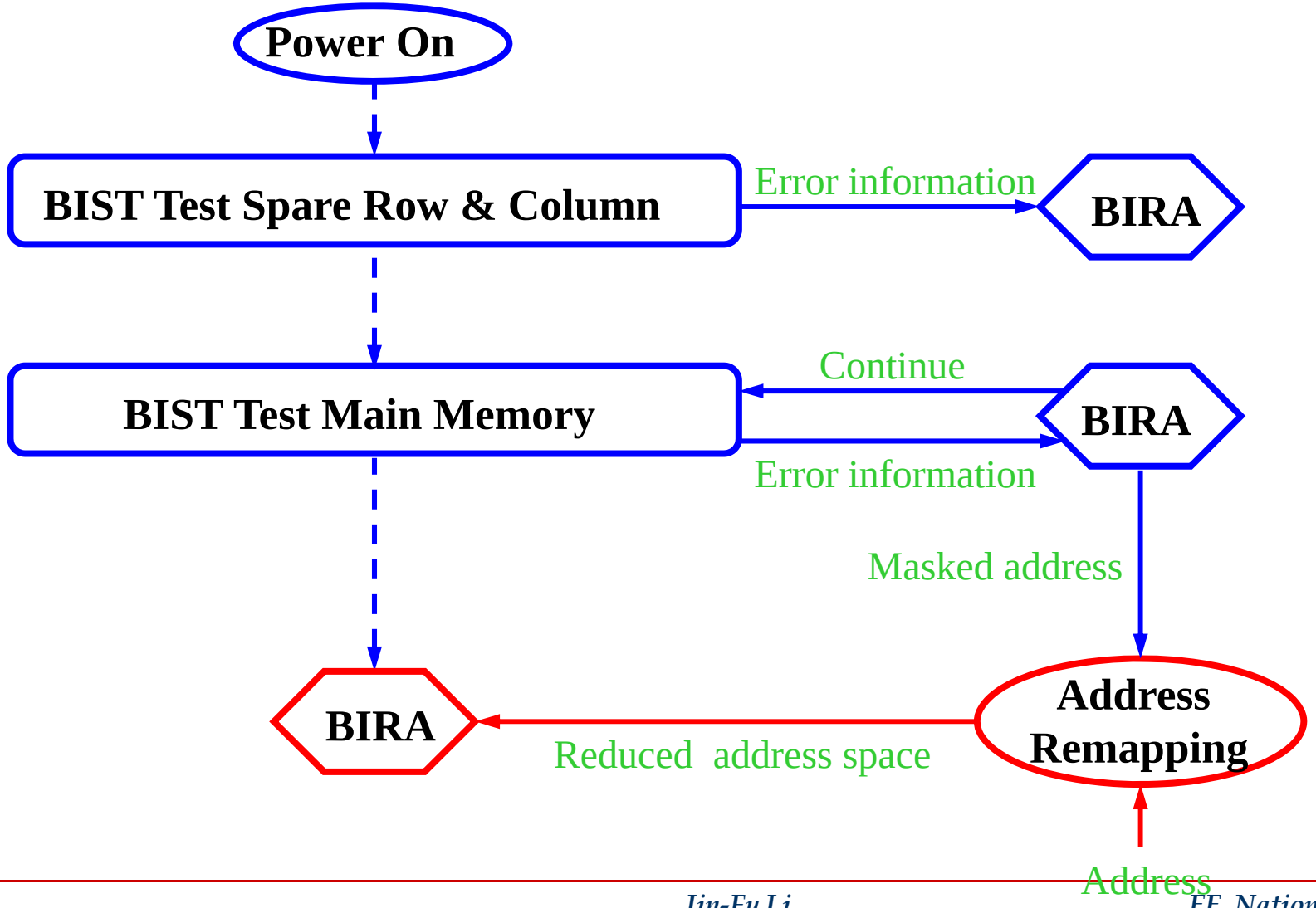
- BISR block diagram



MAO: mask address output; POR: power-on reset

BISR Flow

- Power-on BISR procedure

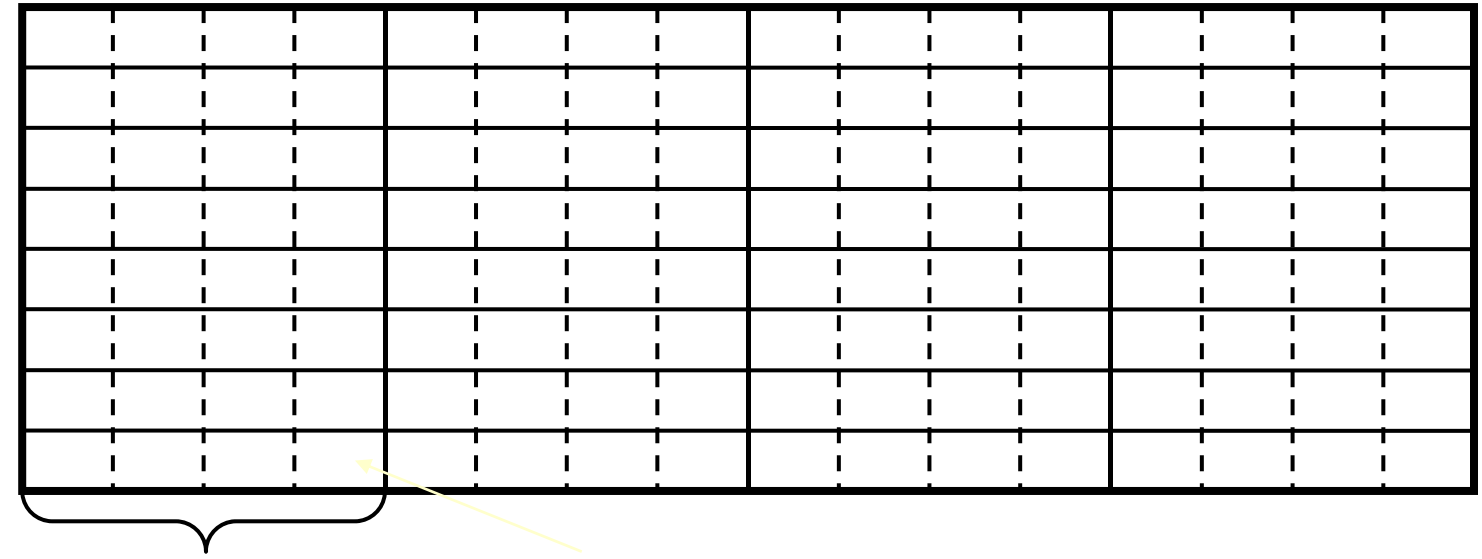


Degraded Performance

- Down-graded operation mode
  - If the spare rows are exhausted, the memory is operated at down-graded mode
    - The size of the memory is reduced
  - For example, assume that a memory with multiple blocks is used for buffering and the blocks are chained by pointers
    - If some block is faulty and should be masked, then the pointers are updated to invalidate the block
    - The system still works if a smaller buffer is allowed

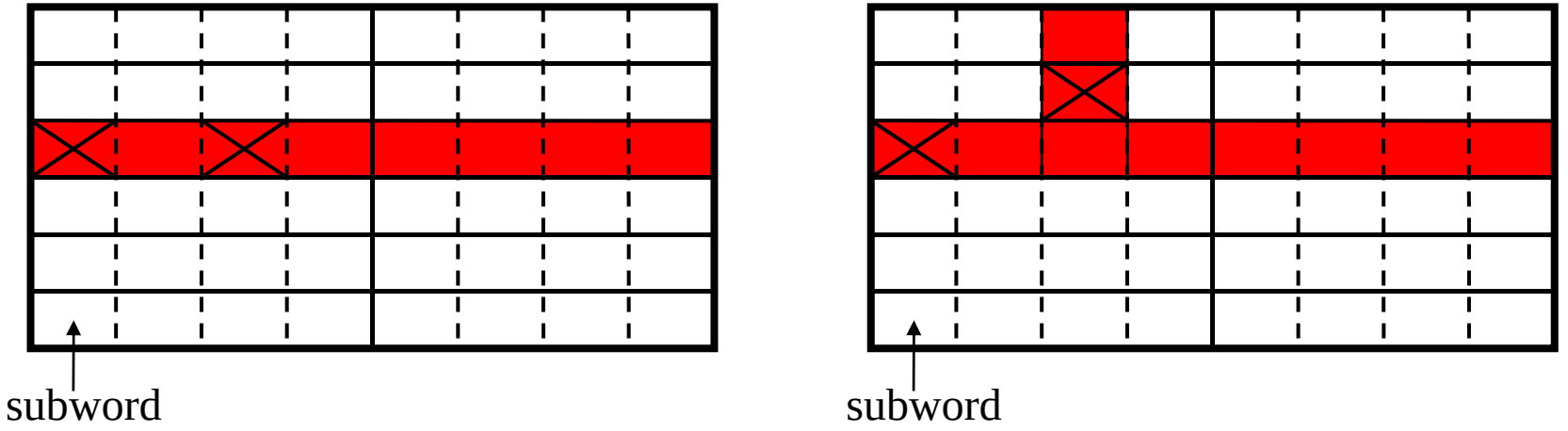
Definitions

- Definition
  - Subword
    - A subword is consecutive bits of a word
    - Its length is the same as the group size
  - Example: a 32x16 RAM with 3-bit row address and 2-bit column address



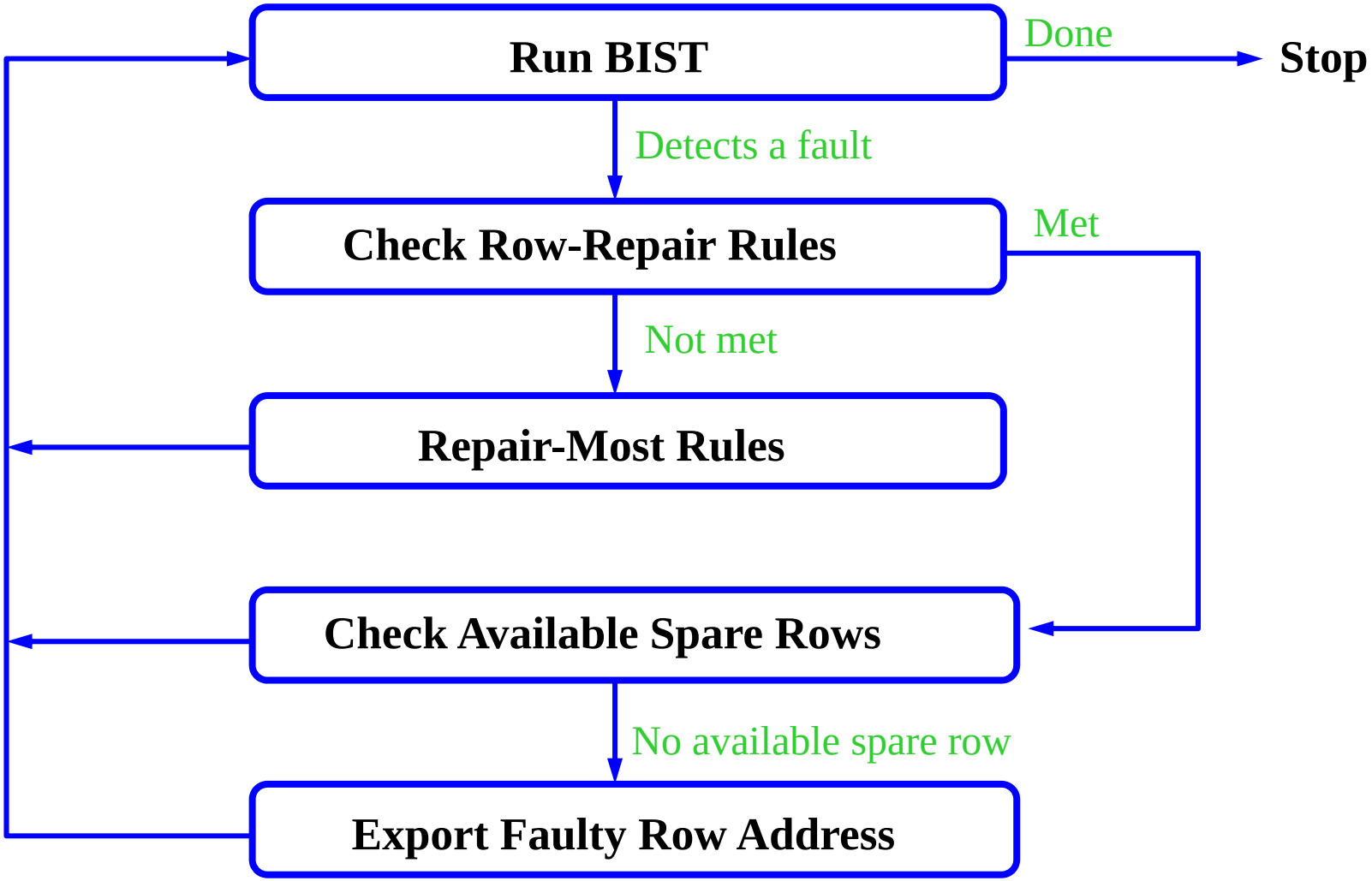
BIRA Algorithm

- Row-repair rules
  - To reduce the complexity, we use two row-repair rules
    - A row has multiple faulty subwords
    - Multiple faulty subwords with the same column address and different row addresses
  - Examples:



BIRA Procedure

- BIRA procedure



Analysis of Repair Rate

- Repair rate analysis
  - Repair rate
    - The ratio of the number of repaired memories to the number of defective memories
  - A simulator has been implemented to estimate the repair rate of the proposed BISR scheme [Huang, *et al*., MTDT, 2002]
  - Industrial case:
    - SRAM size: 8Kx64
    - # of injected random faults: 1~10
    - # of memory samples: 534
    - RA algorithms: proposed and exhaustive search algorithms

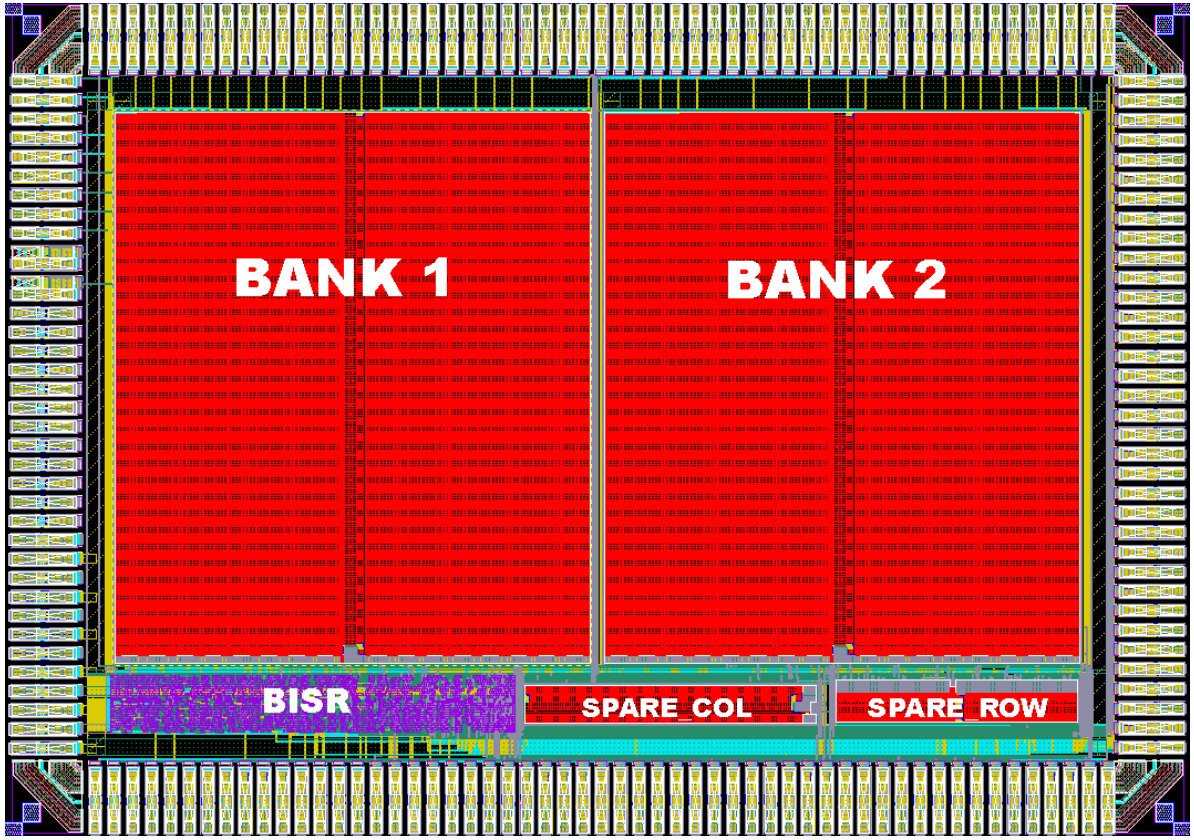
Results of Repair Rate

- Simulation results

N_{SR}	N_{SC}	N_{SCG}	RR	1MA	2MA	3MA	4MA	5MA	>5MA	RR (Best)
1	0	0	18.37%	99	191	4	69	45	32	18.54%
1	4	1	73.10%	38	40	35	16	9	7	86.14%
1	8	2	94.43%	5	7	12	1	3	2	99.81%
1	12	3	99.26%	1	1	1	1	0	0	100%
2	0	0	36.55%	192	2	71	46	18	13	37.08%
2	4	1	86.09%	36	16	12	3	8	0	94.01%
2	8	2	99.26%	3	1	0	0	0	0	100%
2	12	3	100%	0	0	0	0	0	0	100%
3	0	0	72.17%	0	75	43	18	7	7	55.06%
3	4	1	96.10%	7	5	4	3	2	0	97.38%
3	8	2	99.81%	1	0	0	0	0	0	100%
3	12	3	100%	0	0	0	0	0	0	100%
4	0	0	72.36%	73	44	18	8	5	1	71.91%
4	4	1	98.52%	4	3	0	0	0	0	98.69%
4	8	2	100%	0	0	0	0	0	0	100%
4	12	3	100%	0	0	0	0	0	0	100%
5	0	0	85.90%	44	18	7	6	1	0	85.77%
5	4	1	99.81%	1	0	0	0	0	0	99.81%
5	8	2	100%	0	0	0	0	0	0	100%
5	12	3	100%	0	0	0	0	0	0	100%

Test Chip

- Layout view of the repairable SRAM



Technology: 0.25um
SRAM area: 6.5 mm^2
BISR area : 0.3 mm^2
Spare area : 0.3 mm^2
HO_{spare}: 4.6%
HO_{bisr}: 4.6%
Repair rate: 100% (if # random faults is no more than 10)

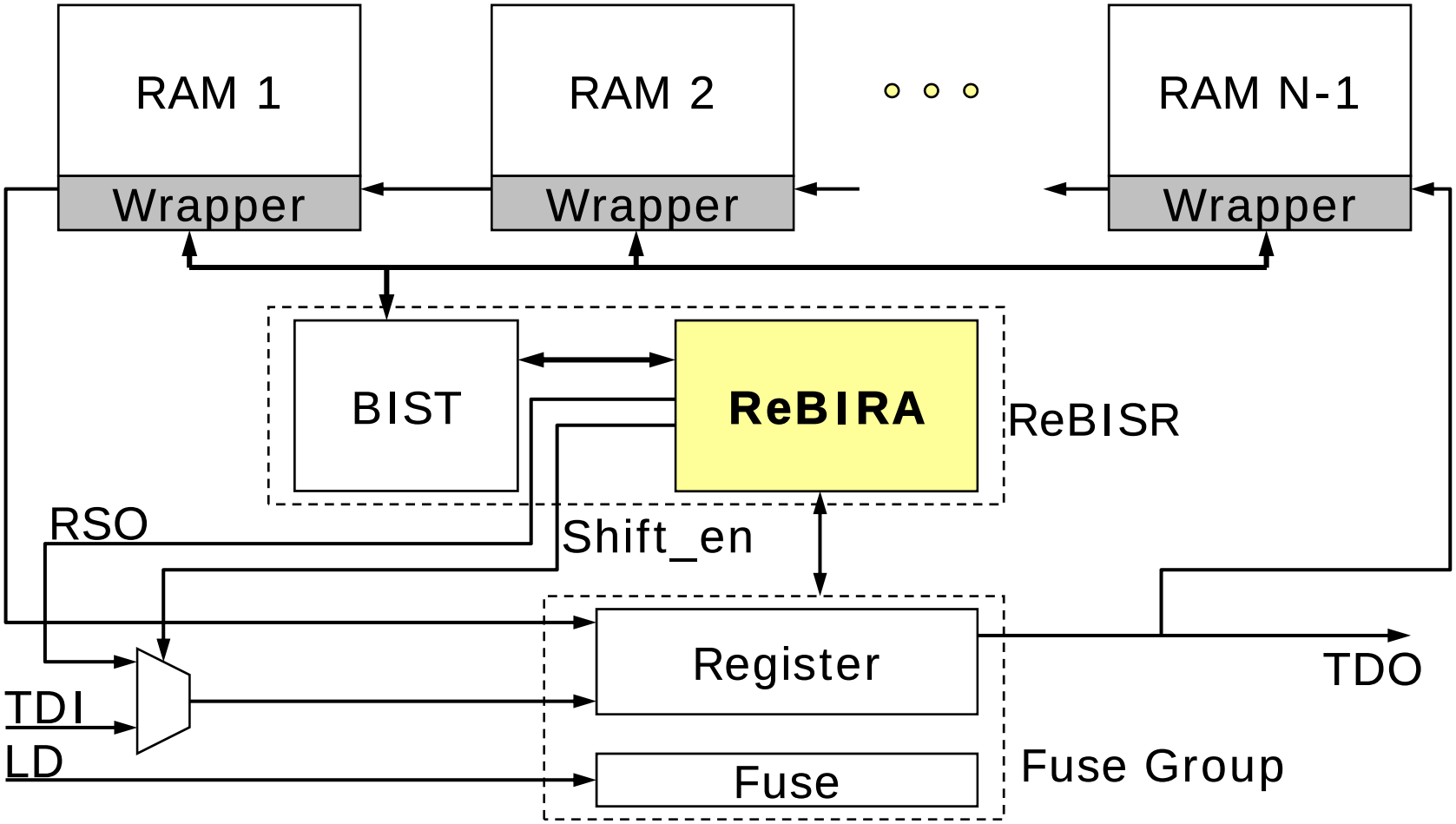
Redundancy: 4 spare rows and 2 spare column groups
Group size: 4

Shared BISR Techniques

- A complex SOC usually has many RAMs with different sizes
- Each repairable RAM has a dedicated BISR circuit
  - Area cost is high
- If a BISR circuit can be shared by multiple RAMs, then the area cost of the BISR circuit can drastically be reduced
- Shared BISR techniques
  - Reconfigurable BISR or IP-based BISR technique

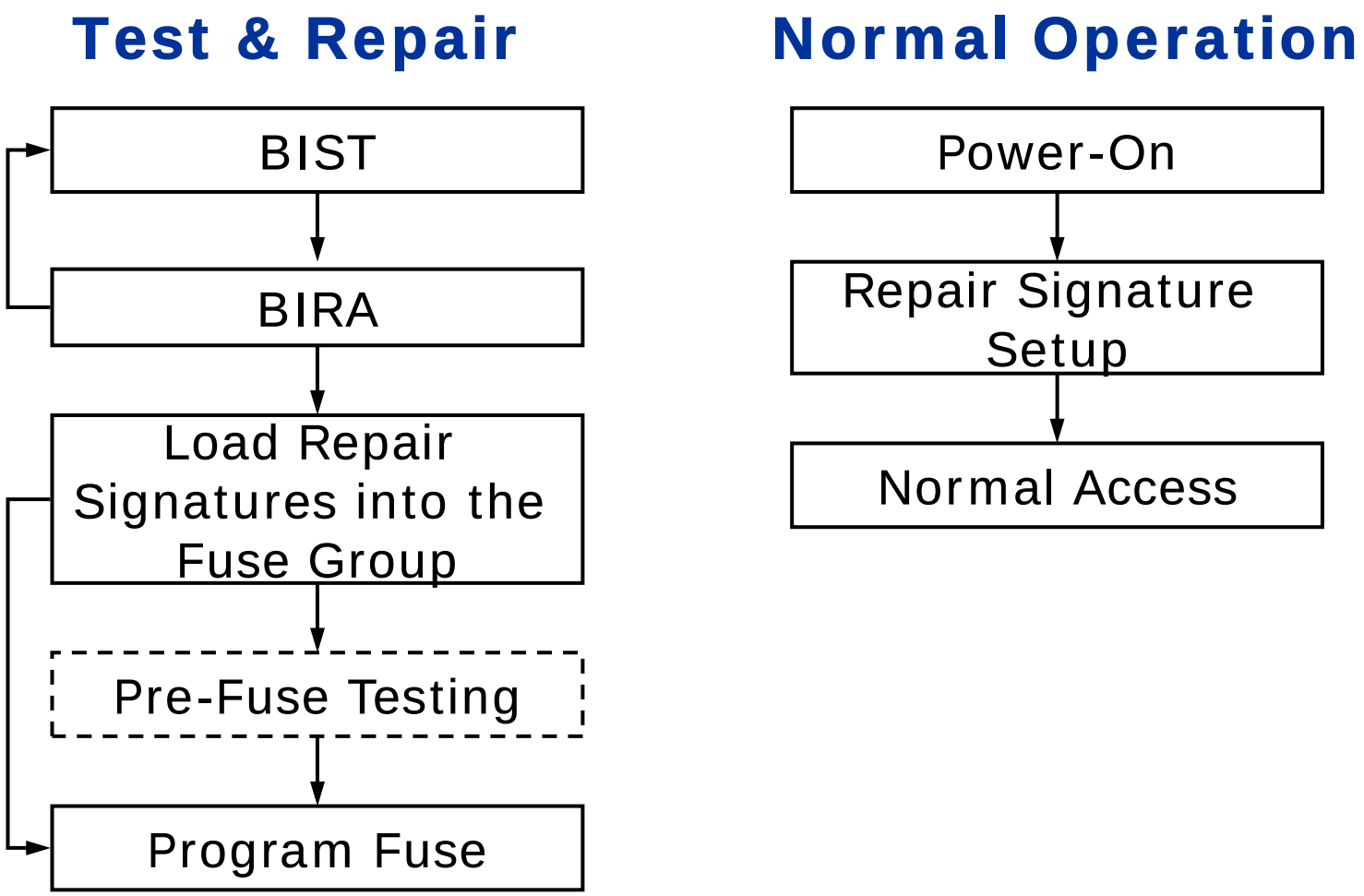
NCU/FTC BISR Scheme

- Reconfigurable BISR scheme for multiple RAMs

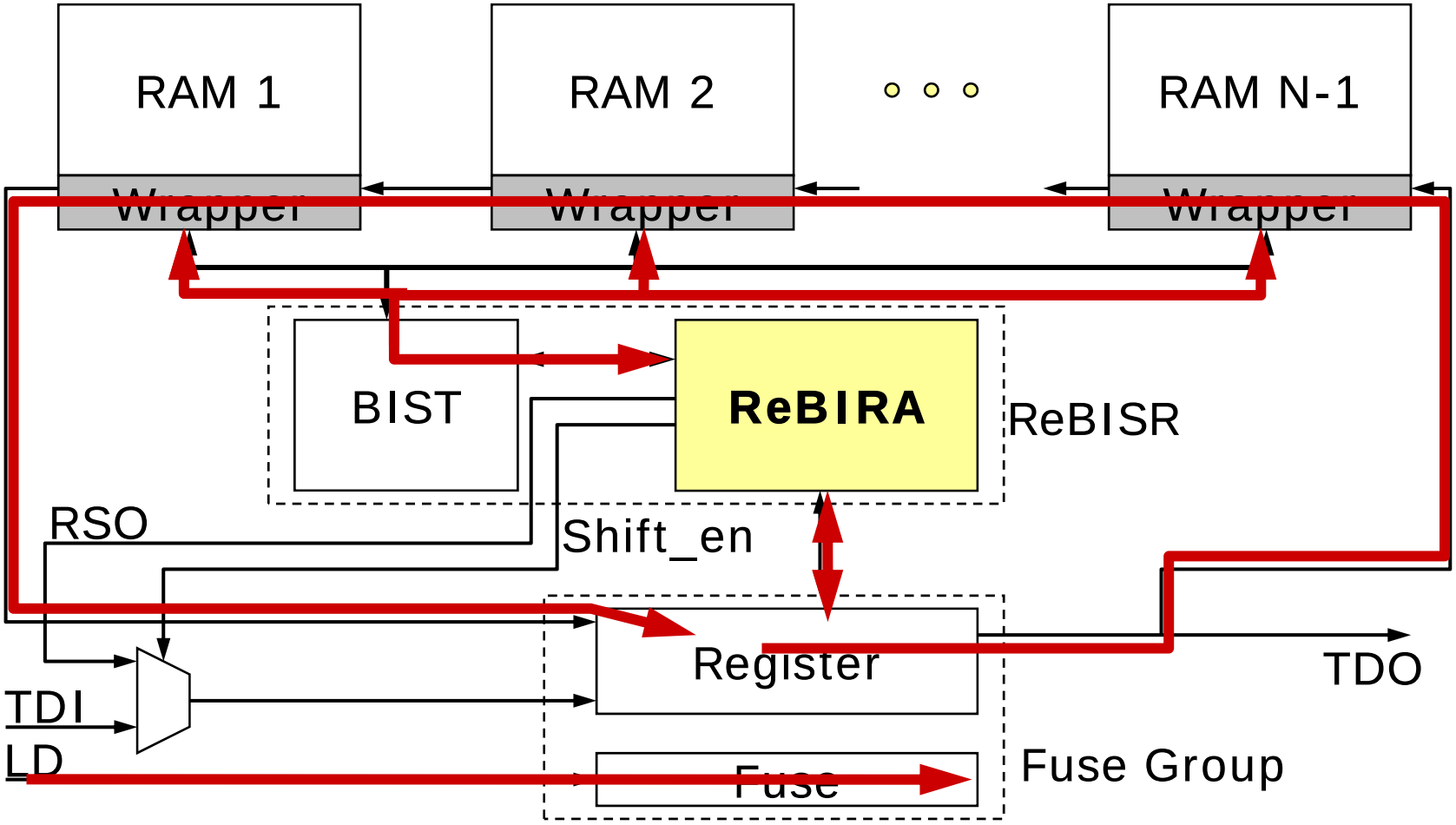


[T. W. Tseng, et. al, ITC06]

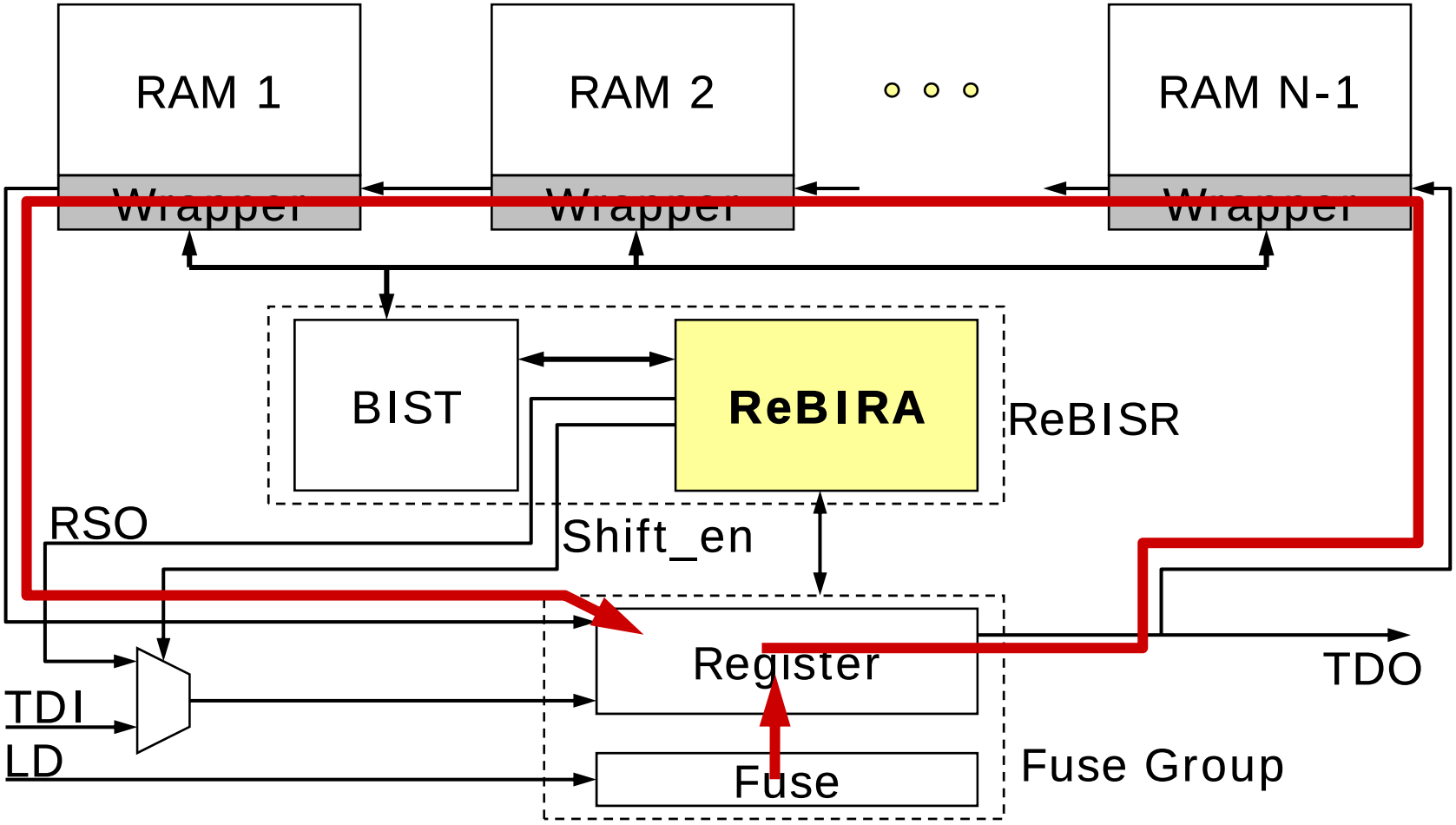
Repair Process



Test and Repair Mode

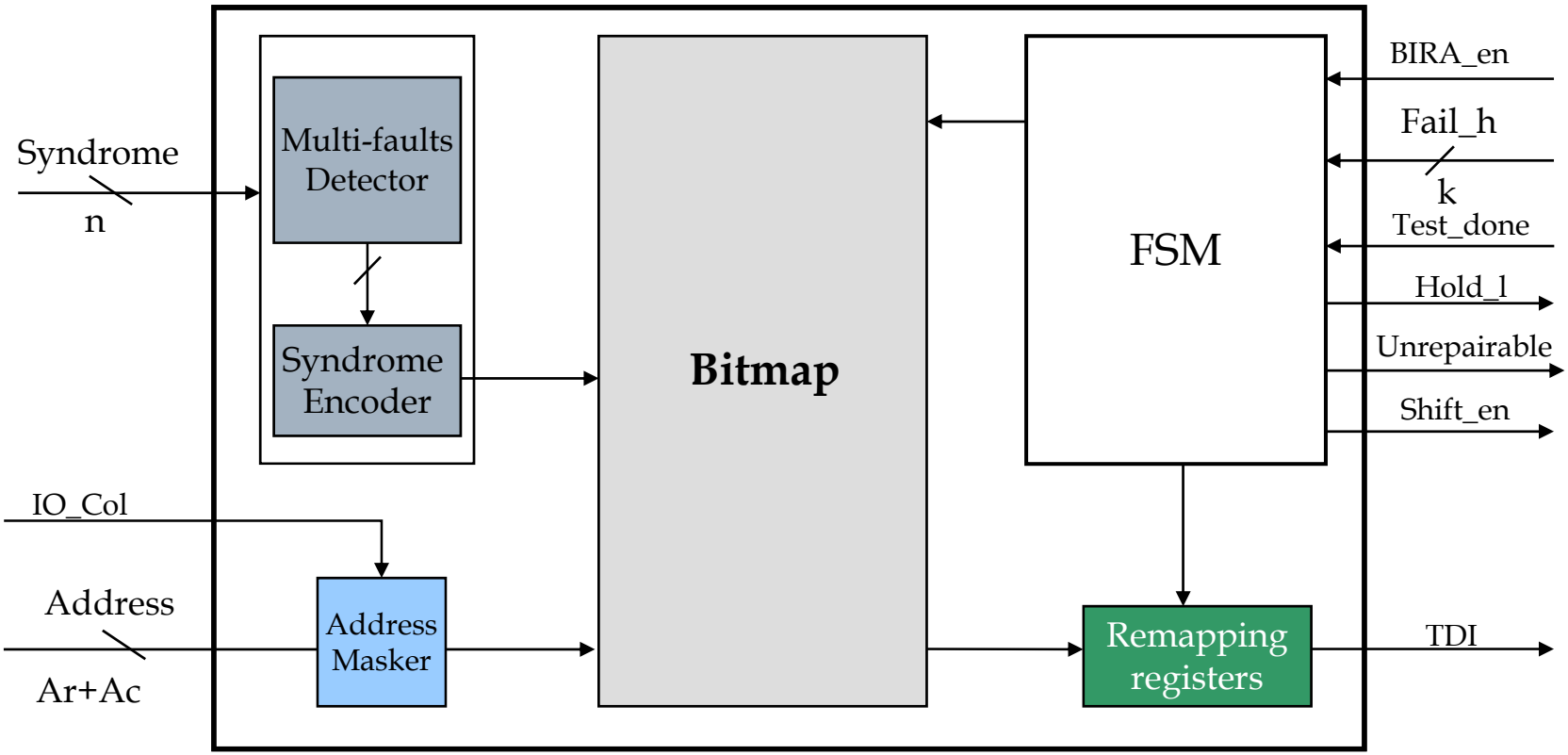


Normal Mode


RAM 1
RAM 2
RAM N-1
Wrapper
Wrapper
Wrapper
BIST
ReBIRA
ReBISR
RSO
Shift_en
TDI
LD
Register
Fuse
Fuse Group
TDO

NCU/FTC BISR Scheme

- Reconfigurable BIRA architecture



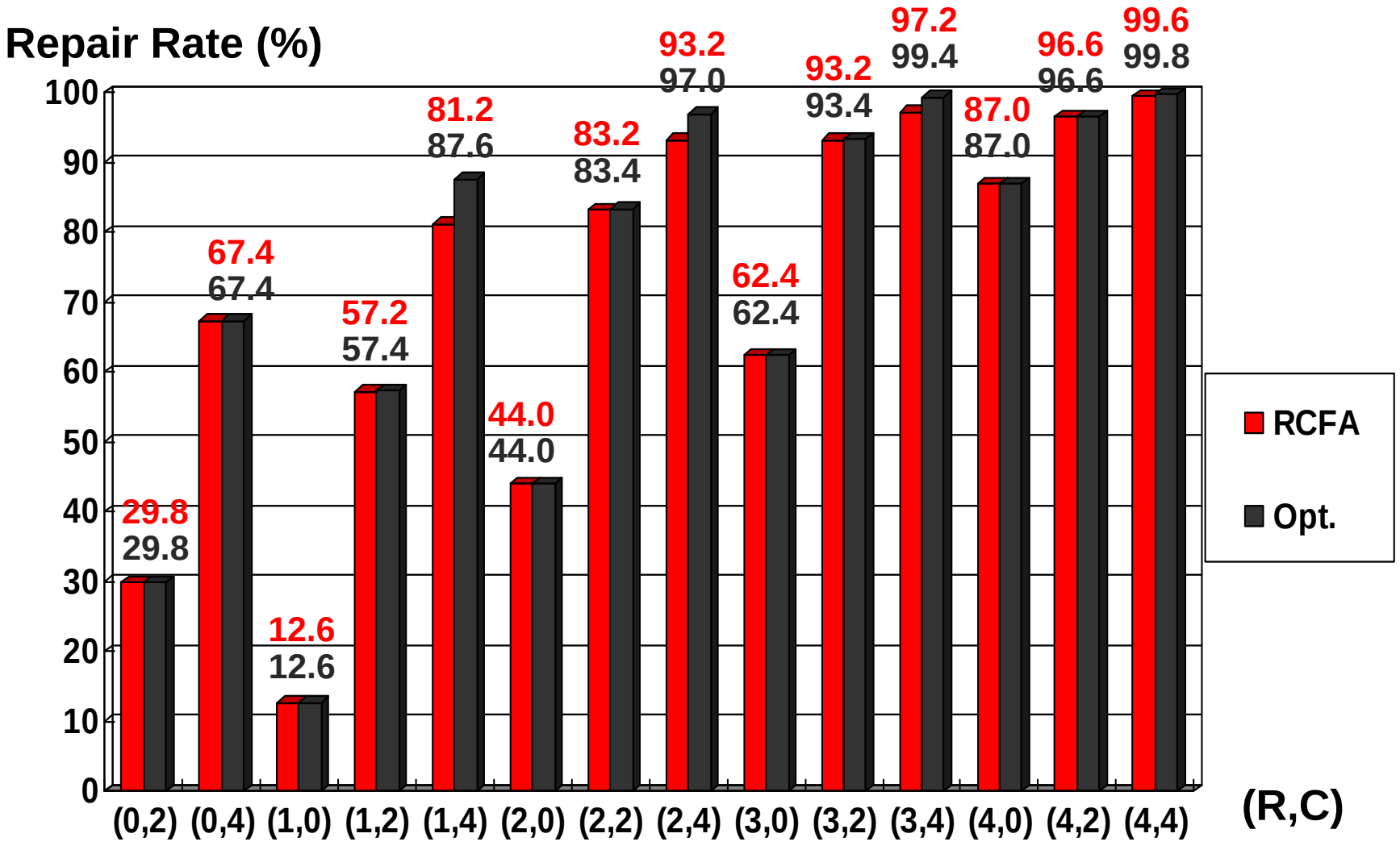
[T. W. Tseng, et. al, ITC06]

Evaluation of Repair Efficiency

- Repair rate
  - The ratio of the number of defective memories to the number of repaired memories
- A simulator was implemented to simulate the repair rate [R.-F. Huang, et. al, IEEE D&T, 2005 (accepted)
- Simulation setup
  - Simulated memory size: 4096x128
  - Simulated memory samples: 500
  - Poisson defect distribution is assumed
  - Original yield is about 60%

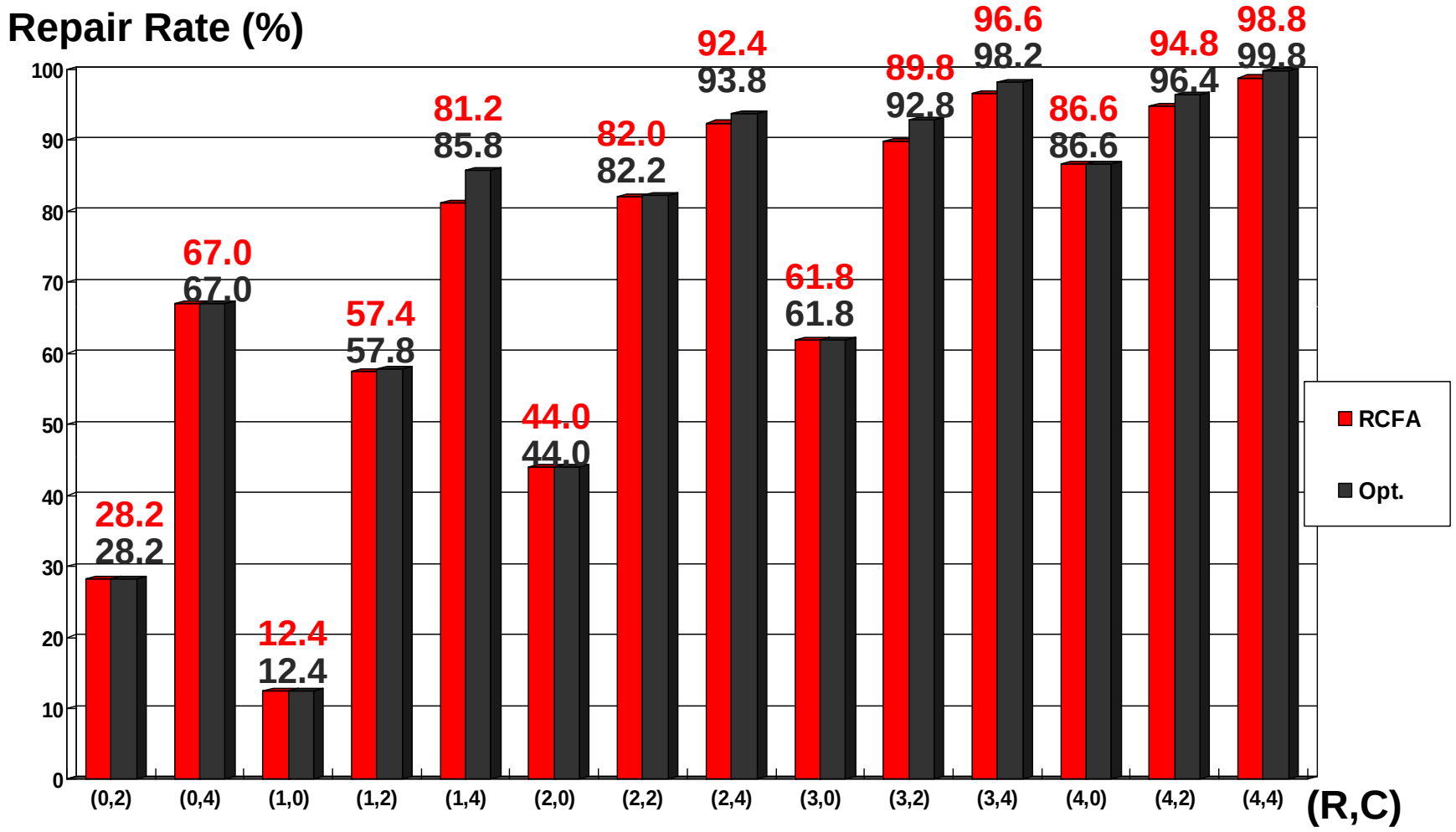
Repair Rate

- Case 1: 100% single-cell faults



Repair Rate

- Case 2: 50% single-cell faults, 20% faulty rows, 20% faulty columns, and 10% column twin-bit faults



ReBISR Implementations

- FTC 0.13*um* standard cell library is used
- Three cases are simulated

	Case 1	Case 2	Case 3
Core 0	64x2x8	64x2x16	64x2x32
Core 1	128x4x16	128x4x32	128x2x64
Core 2	256x8x32	256x8x64	256x4x128
Core 3	512x16x64	512x8x128	512x4x256

Simulation Results

- Delay and area overhead

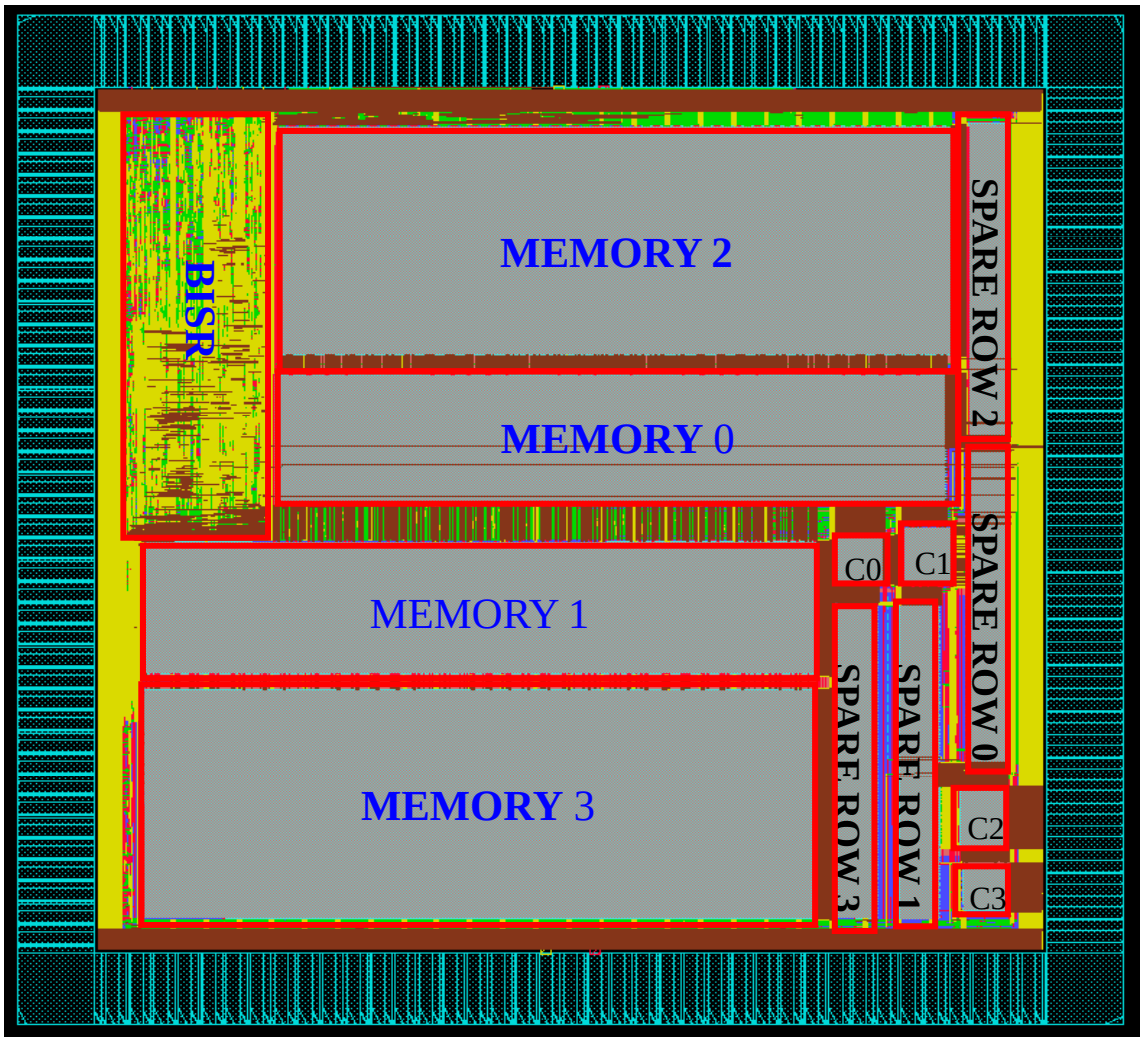
ReBIRA Parameter	Memory Area (um2)	ReBIRA Area (um2)	Ratio (%)	Delay (ns)
512x16x64	1496258.4	18766	1.25	2.5
512x8x128	1497561.6	20303	1.36	2.5
512x4x256	1528848	23255	1.52	2.5

- BIRA time overhead w.r.t. a 14N March test with solid data background

ReBIRA Parameter	Repair Rate (%)	ReBIRA Cycles	BIST Cycles	Ratio (%)
512x16x64	83.6	29952	47939584	0.06
512x8x128	82.3	30698	23605658	0.13
512x4x256	83.8	30404	12013568	0.25

NCU/FTC BISR Scheme

- Layout view for an experimental case

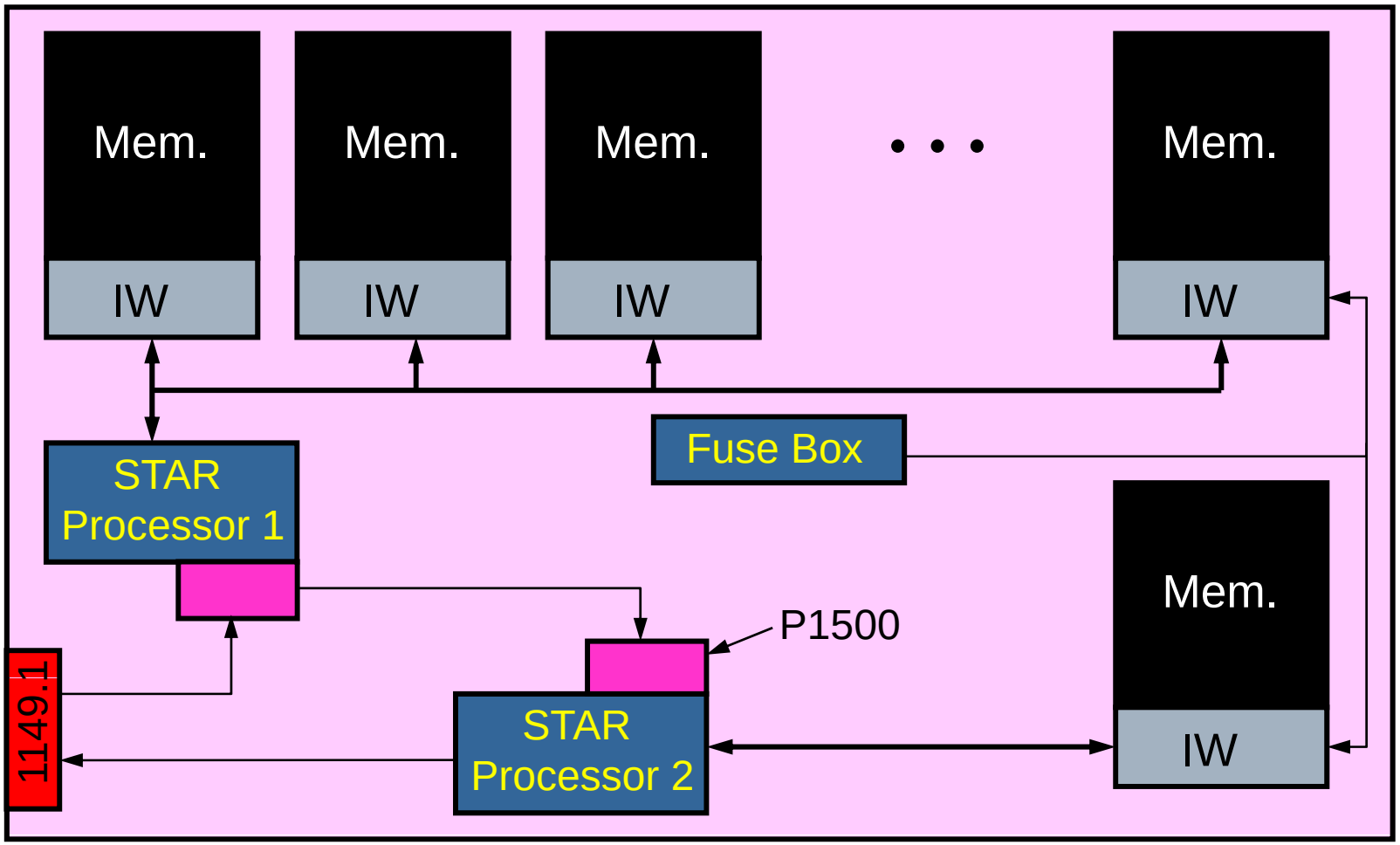


Infrastructure IP

- What is Infrastructure IP
  - Unlike the functional IP cores used in SOCs, the infrastructure IP cores do not add to the main functionality of the chip. Rather, they are intended to ensure the manufacturability of the SOC and to achieve lifetime reliability
- Examples of such infrastructure IPs
  - Process monitoring IP, test & repair IP, diagnosis IP, timing measurement IP, and fault tolerance IP

Infrastructure IP – STAR

- STAR IIP



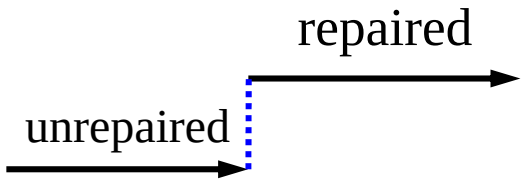
[Y. Zorian, ITC02]

Infrastructure IP – STAR

- The infrastructure IP is comprised of a number of hardware components, including
  - A STAR processor, a fuse box, and intelligent wrappers (IWs)
- The STAR Processor
  - Performs all appropriate test & repair coordination of a STAR memory
  - It is programmed by a set of instructions to control the operation of the internal modules
- The Intelligent Wrapper
  - Address counters, registers, data comparators and multiplexers

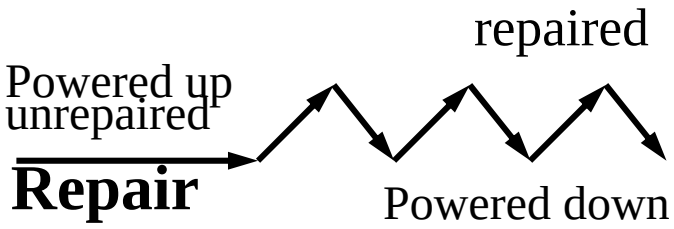
Infrastructure IP – Repair Strategies

- **Hard Repair**

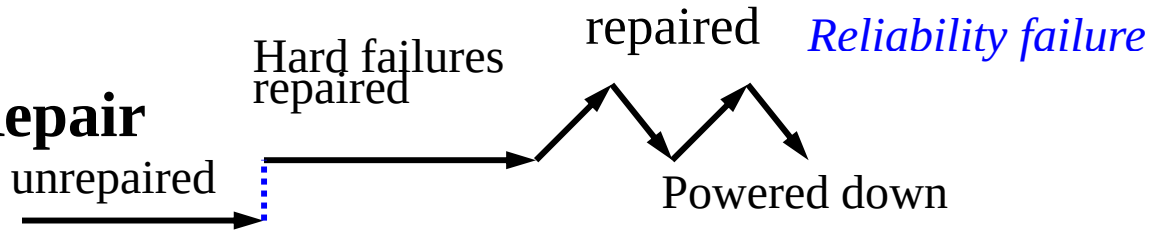


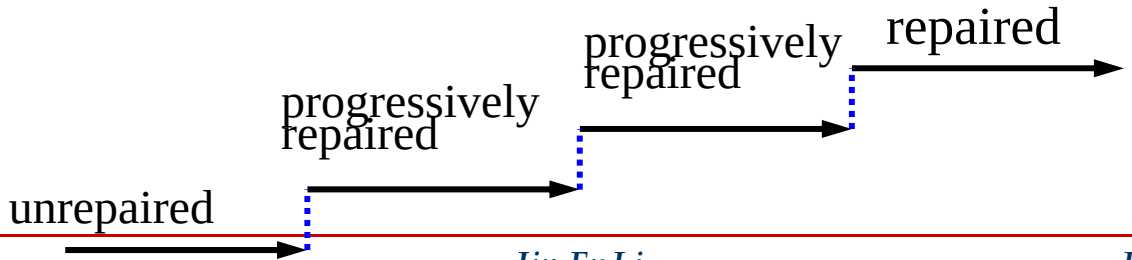
- **Soft Repair**

- **Combinational Repair**



- **Cumulative Repair**

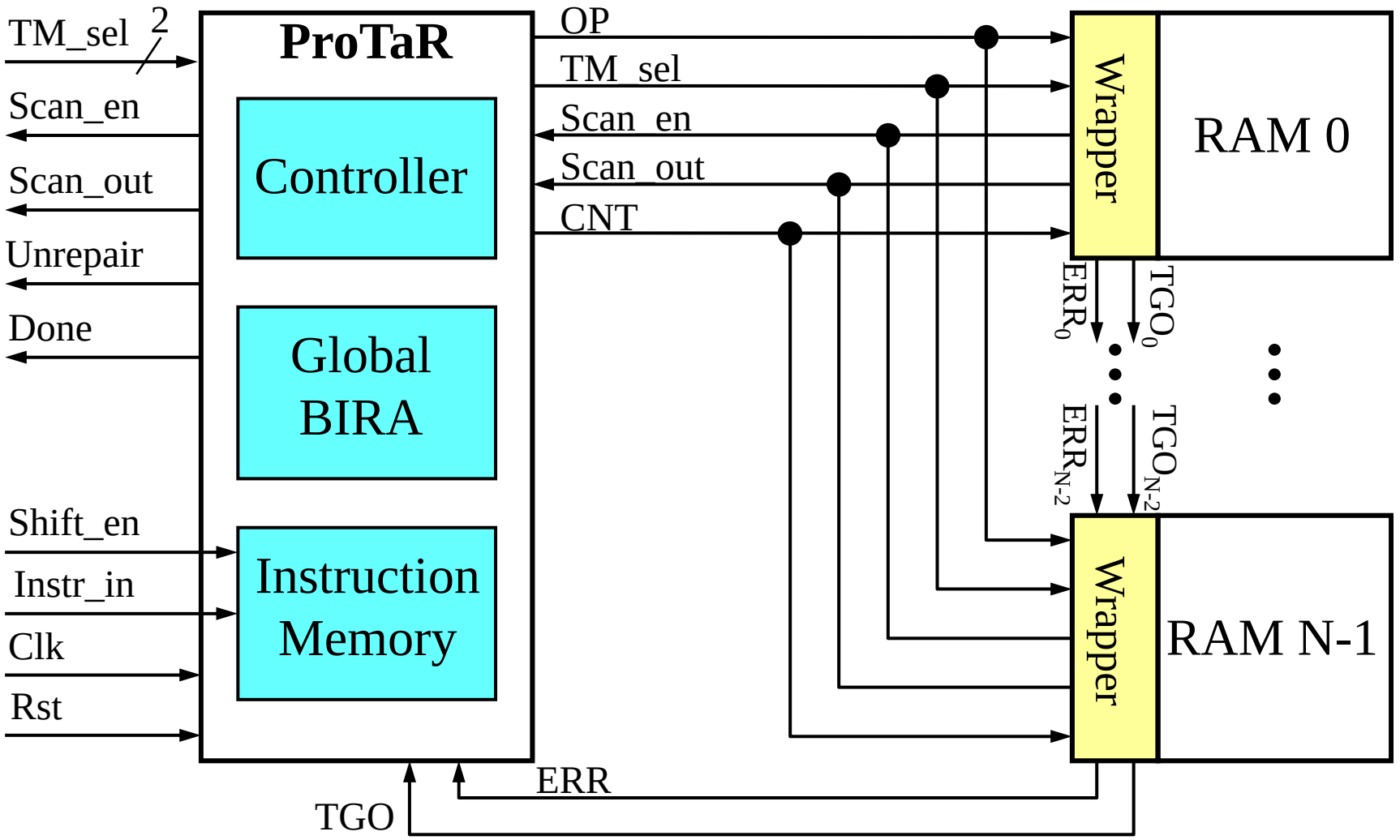




Infrastructure IP – ProTaR

- ProTaR [C.-D. Huang, J.-F. Li, and T.-W. Tseng, IEEE TVLSI, 2007 (accepted)]
  - **Pro**cessor for **T**est **a**nd **R**epair of RAMs
- The infrastructure IP is comprised of a number of hardware components, including
  - A ProTaR processor
  - A wrapper
- Features
  - Parallel test and diagnosis
  - Serial repair
  - Support multiple redundancy analysis algorithms

Architecture of the Proposed IIP



[C.-D. Huang, J.-F. Li, and T.-W. Tseng, IEEE TVLSI, 2007 (accepted)]

Multiple Redundancy Analysis Algorithms Support

- In the IIP, the ProTaR has one global BIRA module and each wrapper has one local BIRA module
- The local BIRA module performs the must-repair phase of a redundancy analysis algorithm
- Then, the global BIRA module performs the final-repair phase of the redundancy analysis algorithm

Global/Local Bitmaps and RA Instructions

Local bitmap

	C0	C1	C2	C3	CAR
R0	1	0	0	0	
R1	0	1	0	1	
R2	0	1	0	0	
R3	0	0	1	0	

RAR

Global bitmap

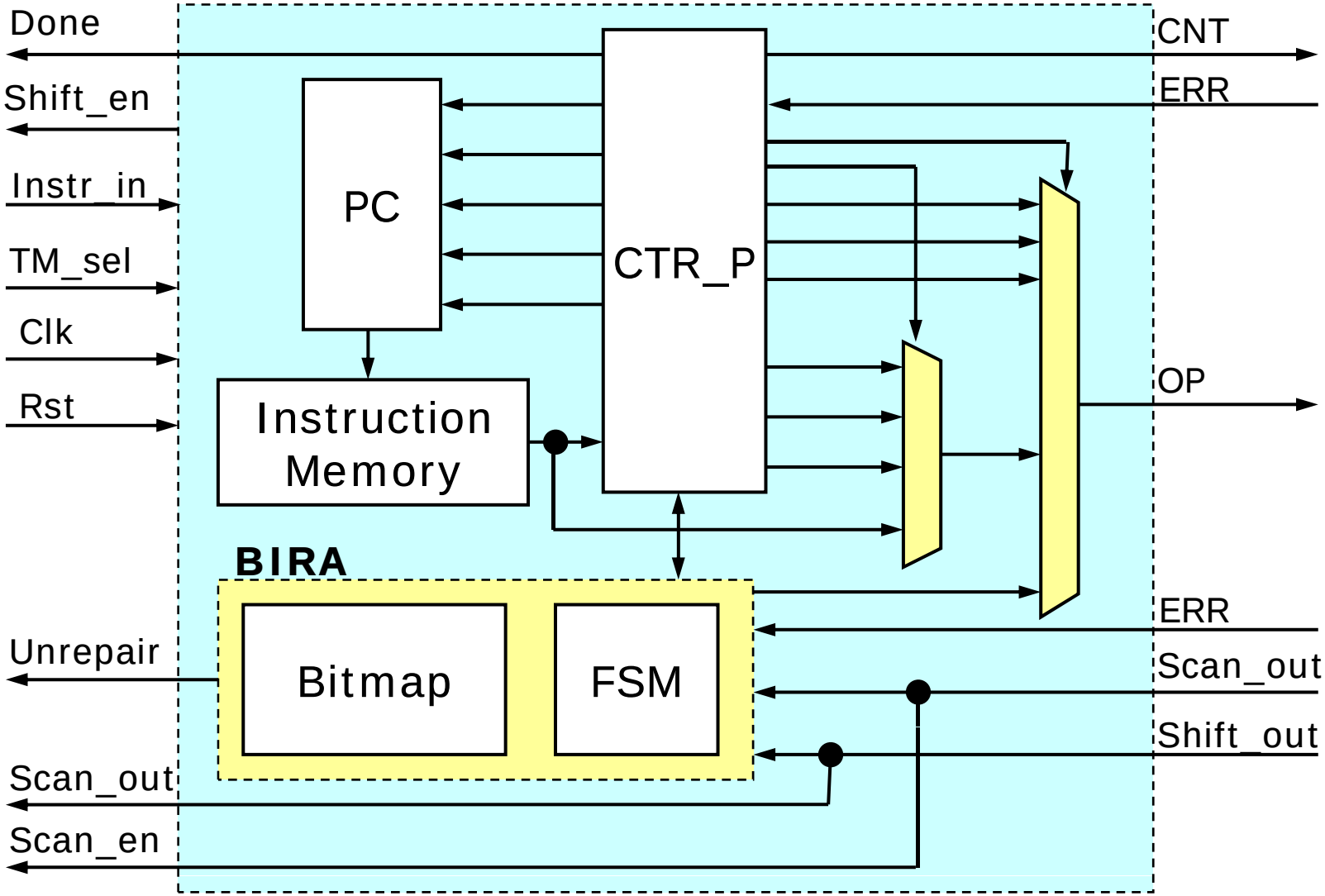
	c0	c1	c2	c3	CID
r0	1	0	0	0	
r1	0	1	0	1	
r2	0	1	0	0	
r3	0	0	1	0	

RID

RA algorithm	**Instructions**
Local repair-most (LRM) alg.	LRM
Essential spare pivoting (ESP) alg.	{FHFR, ROW_FIRST, COL_FIRST}
Row first alg.	{ROW_FIRST, COL_FIRST}
Column first alg.	{COL_FIRST, ROW_FIRST}

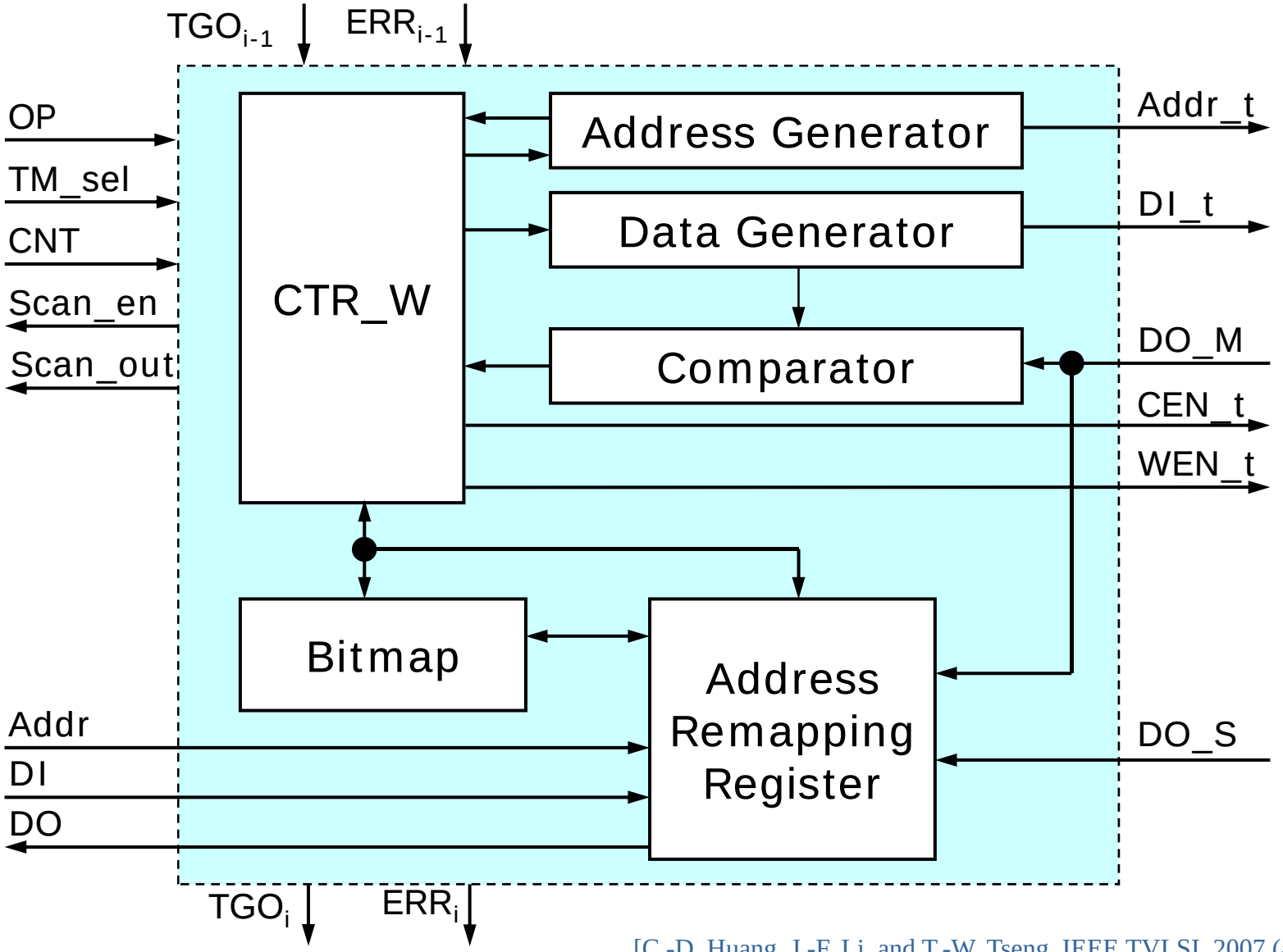
[C.-D. Huang, J.-F. Li, and T.-W. Tseng, IEEE TVLSI, 2007 (accepted)]

Block Diagram of the ProTaR



[C.-D. Huang, J.-F. Li, and T.-W. Tseng, IEEE TVLSI, 2007 (accepted)]

Block Diagram of the Wrapper



[C.-D. Huang, J.-F. Li, and T.-W. Tseng, IEEE TVLSI, 2007 (accepted)]

Area Cost of the Wrapper

- Area overhead of the Wrapper is defined as the ratio of the area of the wrapper to the area of the corresponding memory
- Experimental results for an 8Kx64-bit memory

Redundancy Configuration	Wrapper Area	Area Overhead
2R2C	6739 gates	2.3%
2R3C	7342 gates	2.5%
3R3C	8317 gates	2.8%

- Area cost of Wrappers for different memory sizes

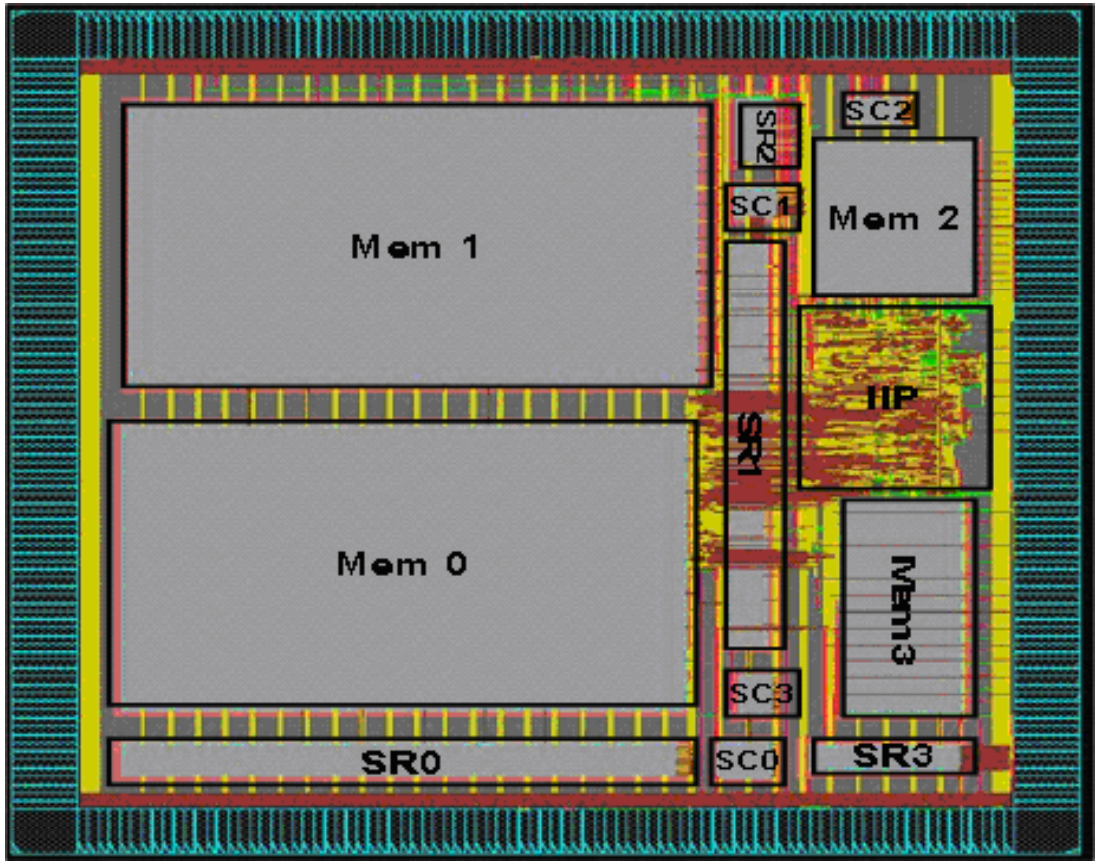
Memory Configuration	Wrapper Area	Area Overhead
8K x 16	3944 gates	4.6%
4K x 32	4825 gates	5.8%
2K x 64	6501 gates	7.1%

Area Cost of the IIP

- An IIP for four memories is implemented
  - The size of Mem0, Mem1, Mem2, and Mem3 are 8Kx64, 8Kx64, 4Kx14, and 2Kx32, respectively
  - The redundancy configurations of the Mem0, Mem1, Mem2, and Mem3 are (3x3), (2x2), (2x2), and (2x2), respectively
- The area of the four memories is 6798472um^2
- The area of all the redundancies is about 896060um^2
- The area of the IIP is only about 309893um^2
- Thus, the area overhead of the IIP is only about 4.56%

Layout View of the IIP

- Layout view of the proposed IIP for four RAMs



[C.-D. Huang, J.-F. Li, and T.-W. Tseng, IEEE TVLSI, 2007 (accepted)]

Conclusions

- Embedded memories represent more and more area of system-on-chip (SOC) designs
  - The yield of memory cores dominates the yield of chips
- Various BIRA techniques have been presented
- Different BISR techniques for memories in SOCs have been presented

References

1. T. Kawagoe, J. Ohtani, M. Niiro, T. Ooishi, M. Hamada, and H. Hidaka, “A built-in self-repair analyzer (CRESTA) for embedded DRAMs,” in *Proc. Int’l Test Conf. (ITC)*, 2000, pp. 567.574.
2. V. Schober, S. Paul, and O. Picot, “Memory built-in self-repair using redundant words,” in *Proc. Int’l Test Conf. (ITC)*, Baltimore, Oct. 2001, pp. 995-1001.
3. Y. Zorian, “Embedded memory test & repair: Infrastructure IP for SOC yield,” in *Proc. Int’l Test Conf. (ITC)*, Baltmore, Oct. 2002, pp. 340.349.
4. C.-T. Huang, C.-F. Wu, J.-F. Li and C.-W. Wu,”Built-in redundancy analysis for memory yield improvement,” *IEEE Trans. Reliability*, vol. 52, no. 4, pp. 386-399, Dec. 2003.
5. J.-F. Li, J.-C. Yeh, R.-F. Huang, C.-W. Wu, P.-Y. Tsai, A. Hsu, and E. Chow,”A built-in self-repair scheme for semiconductor memories with 2-D redundancies,” in *Proc. IEEE Int. Test Conf. (ITC)*, (Charlotte), pp. 393-402, Sept. 2003.
6. J.-F. Li, J.-C. Yeh, R.-F. Huang, and C.-W. Wu,”A built-in self-repair design for RAMs with 2-D redundancies,” *IEEE Trans. Very Large Scale Integration Systems*, vol.13, no.6, pp. 742-745, June, 2005.
7. S.-K. Lu, Y.-C. Tsai, C.-H. Hsu, K.-H. Wang, and C.-W. Wu, “Efficient built-in redundancy analysis for embedded memories with 2-D redundancy,” *IEEE Trans. on VLSI Systems*, vol. 14, no. 1, pp. 34–42, Jan. 2006.
8. T.-W. Tseng, J.-F. Li, C.-C. Hsu, A. Pao, K. Chiu, and E. Chen, "A reconfigurable built-in self-repair scheme for multiple self-repairable RAMs in SOCs," in *Proc. IEEE Int. Test Conf. (ITC)*, (Santa Clara), Paper 30.2, pp. 1-8, Oct. 2006.
9. C.-D. Huang, J.-F. Li, and T.-W. Tseng,”ProTaR: an infrastructure IP for repairing RAMs in SOCs,” *IEEE Trans. Very Large Scale Integration Systems*, vol.15, no.10, pp. 1135-1143, Oct. 2007.
10. R.-F. Huang, J.-F. Li, J.-C. Yeh, and C.-W. Wu,”RAISIN: a tool for evaluating redundancy analysis schemes in repairable embedded memories,” *IEEE Design and Test of Computers*, vol.24, no.4, pp. 386-396, July-August 2007.