Self-Repairable EPLDs: Design, Self-Repair and Evaluation Methodology

Chong H. Lee, Marek A. Perkowski,
Douglas V. Hall, David S. Jun
Portland State University
Dept. of Electrical and Computer Engineering

Topics of Discussion

- Introduction
- GAL (GAL16V8, Our Model)
- Design Methodology
  - Our Fault Model (Cross-point Stuck-at Faults in an E^2CMOS Cell of a GAL)
  - Fault Model and Design Assumptions
  - Digital System in Our Approach/Design Architecture
  - Test Generation and Fault Diagnosis/Location

Topics of Discussion

- Self-Repairing Methodology
  - Column Replacement with Extra Columns
- Evaluation Methodology
  - Assumptions and Failure Rates
  - Evaluation and Analysis of the Simulation Results
  - Hardware Overhead and Performance
- Conclusions and Future Works

Motivation and Real Problems

- **1. Lack of the Research on Self-Repair of GALs, FPGAs, or Similar Devices**
  - The high reliability and quick in-field repair of a digital system is of the primary importance.
  - The self-testing system can not be sufficient.
  - The self-repairing system would be desired.
  - **As far as we know, nothing has been yet published on self-repair of GALs or similar devices.**

Motivation and Real Problems

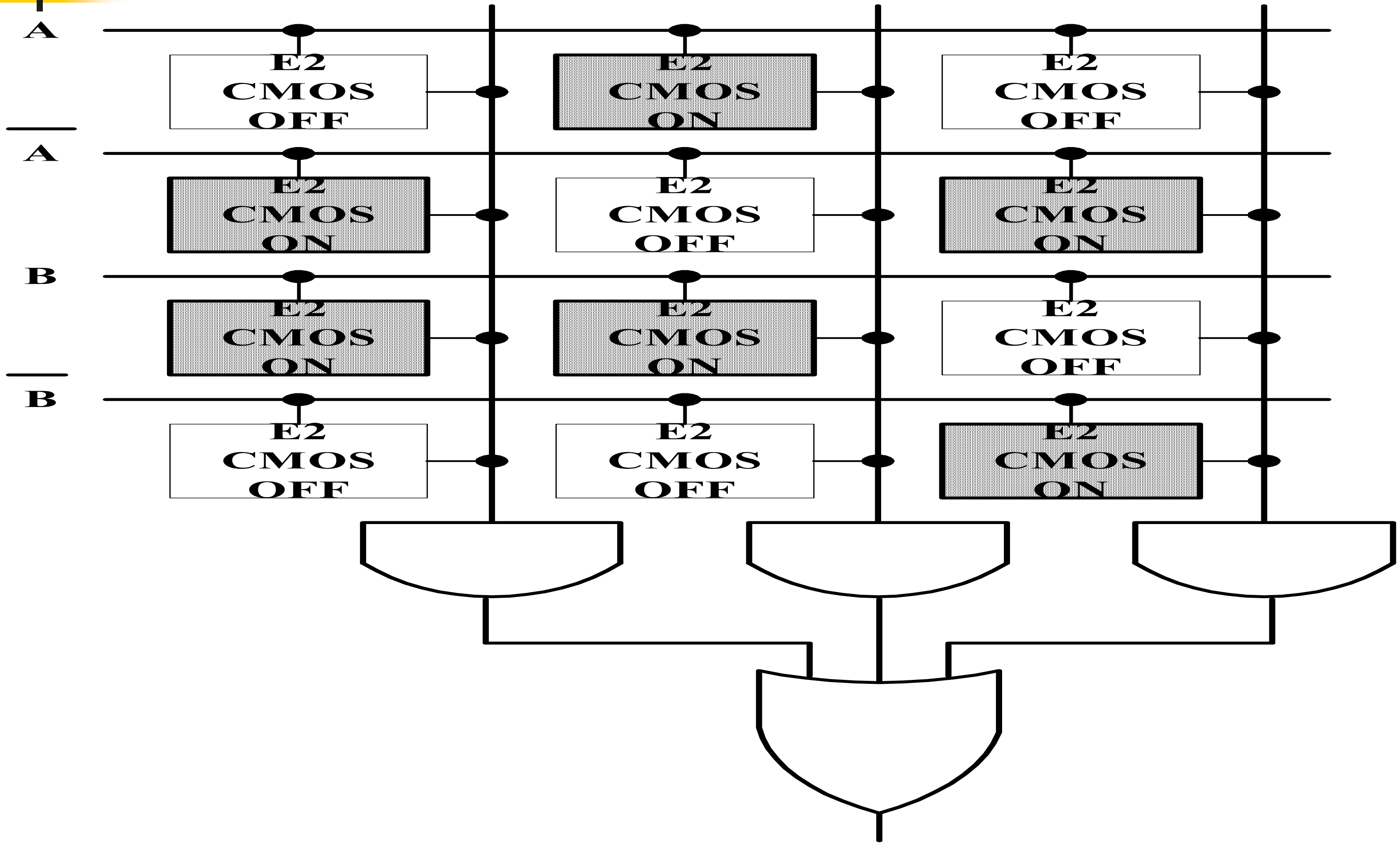
- **2. Real Problems**
  - **Faults caused by inadequate quality control during manufacture, the wear and tear of normal operation.**
  - **External disturbances such as heat, radiation, electrical and mechanical stress** also produce failures.
  - **The Early Failure Rate** (**PPM**; Parts Per Million devices from 0 year to 1 year) and **the Long Term Failure Rate** (**FIT**; Failures In Time from 0 year to 10 years).
  - **EEPROM: Early Failure Rate = 489 PPM,**

    Long Term Failure Rate = 5.07 %
  - **The PPM and FIT of EEPROM suggest that it is reasonable to invest in a repair mechanism.**

Purpose of the Research

- Introduce the **concept of the self-testable and self-repairable EPLDs** for high security and safety applications.
- **Prove that a self-repairable GAL will last longer in the field.**
- **Develop a design methodology** to allow detect, diagnose, and repair of all multiple stuck-at faults.
- **Develop a self-repairing methodology** to replace faulty elements with the new ones (extra columns) by automatic reprogramming.
- **Develop an evaluation methodology** to estimate an ideal point, where the maximum reliability can be reached with the minimum cost.

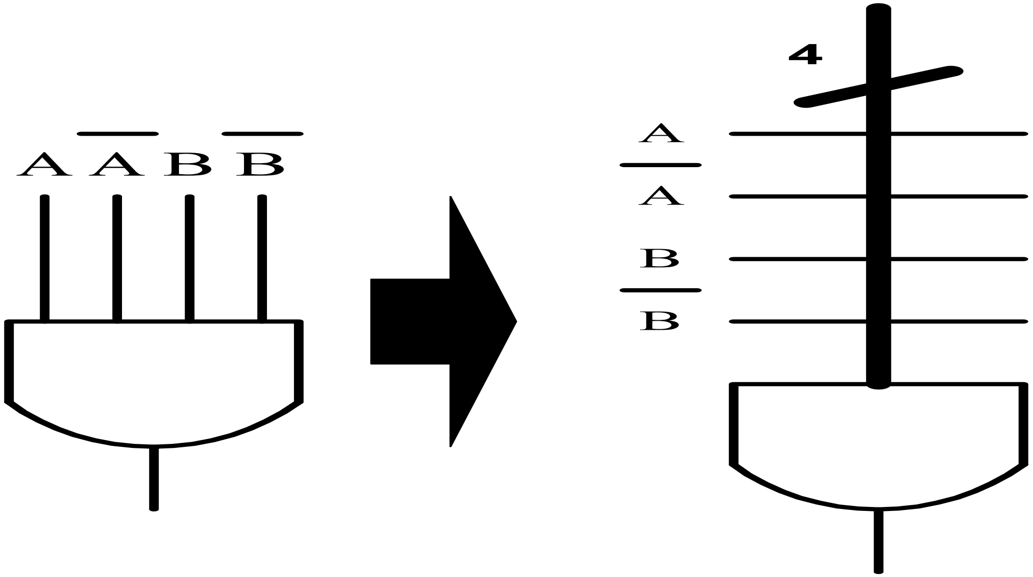
A Programmed Simple Logic Function in a GAL (Generic Array Logic)



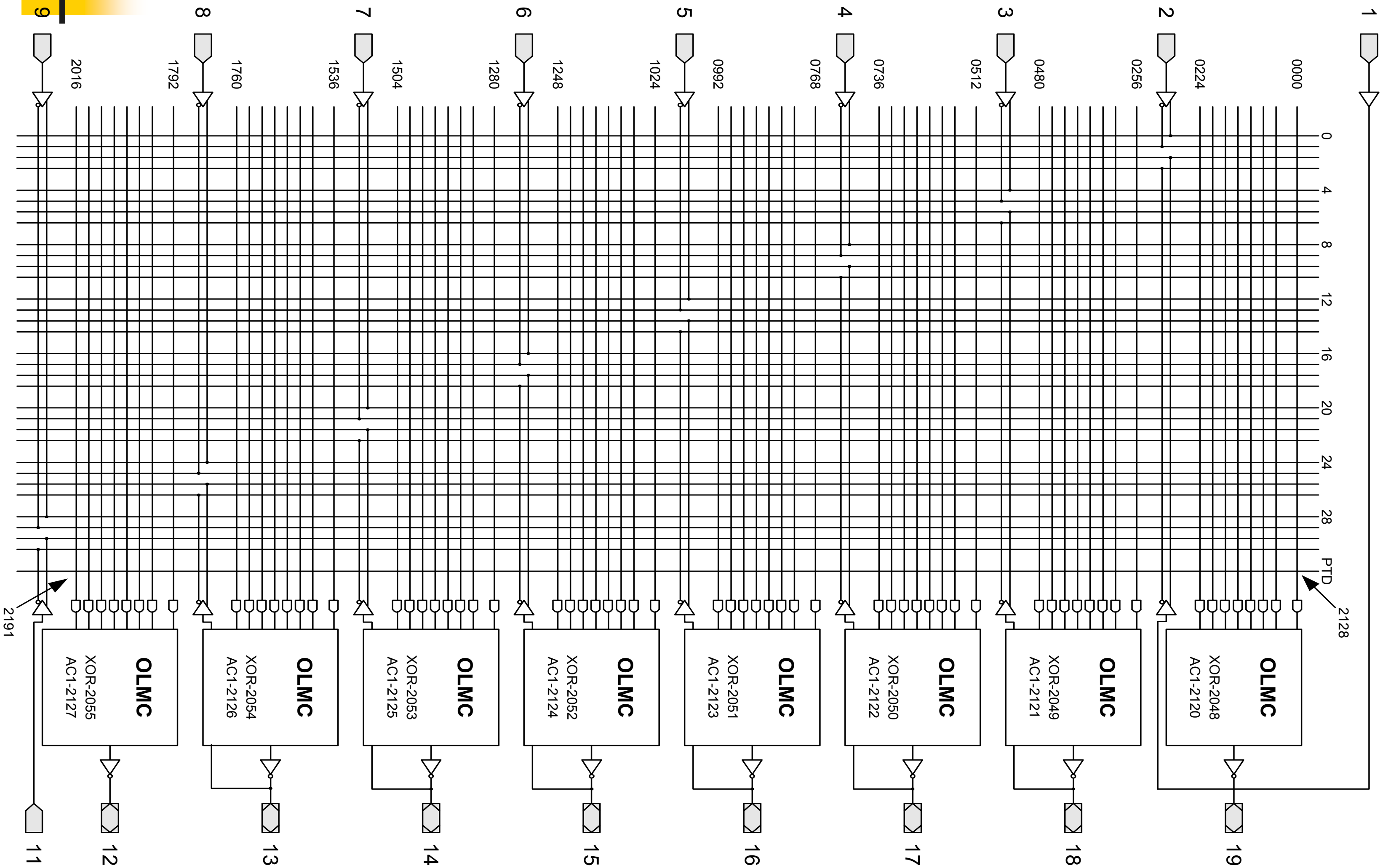
$$X = \overline{A}\ B + A\ B + \overline{A}\ \overline{B}$$

A Simplified Notation for Input Lines of an AND Gate

- Use special notation to simplify many input lines of an AND gate, likewise OR input lines.



A Structure of the GAL16V8



Our Fault Model; Cross-Point Stuck-at Faults

- **The cross-point stuck-at faults;**
  - Located in E^2CMOS cells.
  - E^2CMOS cells of an AND array form a large percentage of a GAL.
  - The same structure/technology as the EEPROM.
  - The **cross-point stuck-at-1** (simply, **s-a-1**);
    - If the E^2CMOS cell of a particular cross-point, which should be programmed as OFF, is ON, permanently.
  - The **cross-point stuck-at-0** (simply, **s-a-0**);
    - If the E^2CMOS cell of a particular cross-point, which should be programmed as ON, is OFF, permanently.

Fault Model and Design Assumptions

- No faults in an initial state after manufacturing GALs.
- No primary input/output faults in a GAL.
- No AND gate input line faults.
- The cross-point faults (s-a-0, s-a-1) as defined previous are only taken into account in this research.
- Several extra columns are pre-allocated in each OLMC in a GAL.
- At least one pair of literals of the same variables reprogrammed as ON not to affect the OR function.
- When a fault occurs in a certain column of a particular OLMC, this faulty column can be only replaced with an extra column in that OLMC.

Fault Model and Design Assumptions

- **When an E^{2}CMOS cell is OFF;**
  - Binary data '0' will be stored in memory.
  - This cell disconnects a primary input from an AND gate input.
  - '1' will be on the AND gate input.
  - Described with just intersection between rows and columns in a logic diagram.
- **When an E^{2}CMOS cell is ON;**
  - Binary data '1' will be stored in memory.
  - This cell connects a primary input from an AND gate input.
  - Denoted by 'X' between rows and columns in a logic diagram.

Digital System in Our Approach

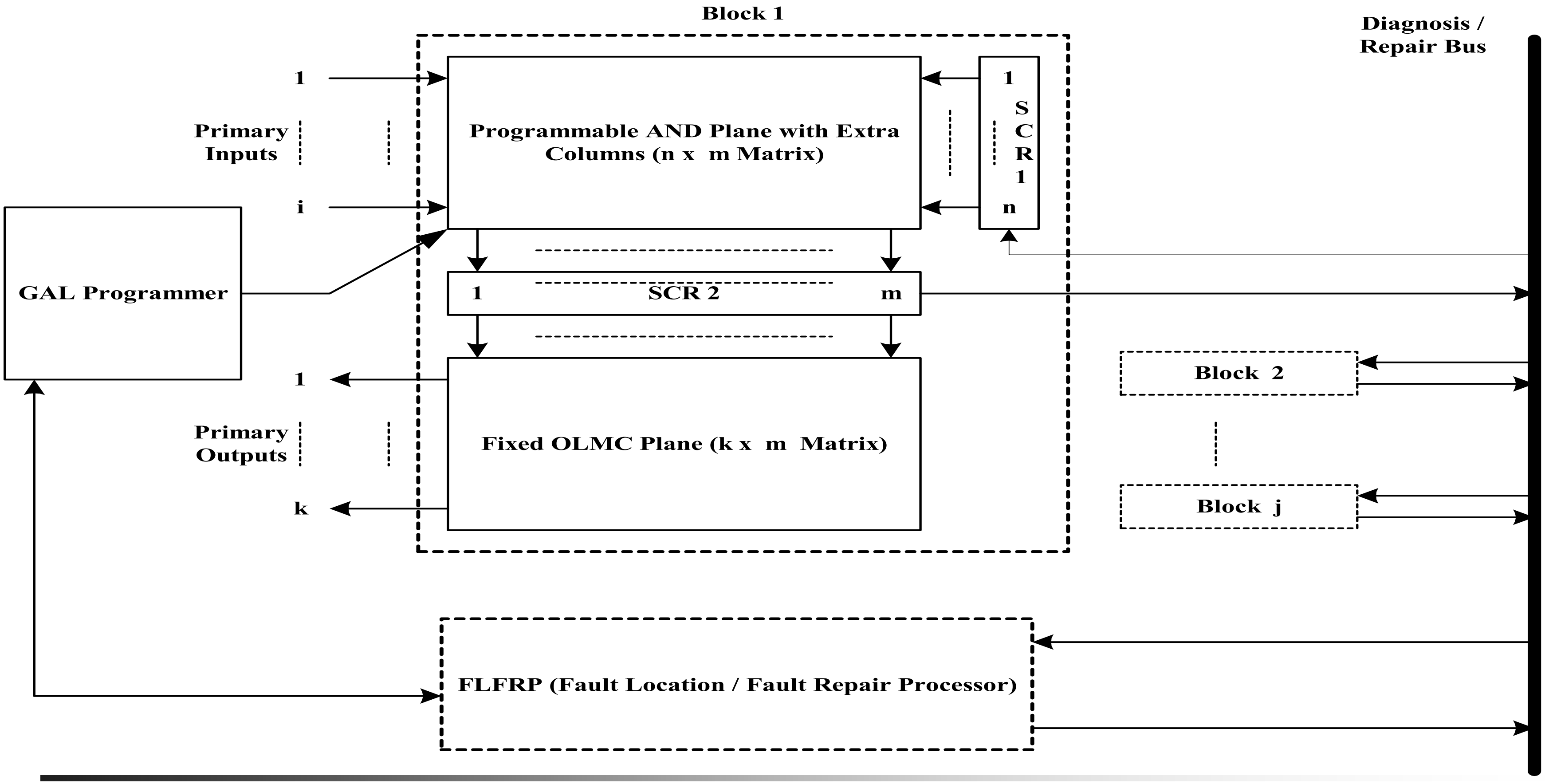
- A network of blocks realized as separate integrated circuits.
  - A two-level realization of a Boolean function, and is realized with a GAL.
  - A single module (a chip, a board) or different such modules.
  - Assume that each block includes just one self-repairable GAL.
  - Assume that the global fault of the block is signaled with go/no-go signal when there are no more spare columns to be replaced with.
  - FSMs (Finite State Machines) are composed of Boolean Logic and registers located in OLMCs.
  - This seems to be reasonable since as the EPLD/FPGA (Field Programmable Gate Array) technology advances, functions and machines of even greater sizes can be implemented in them.

Digital System in Our Approach

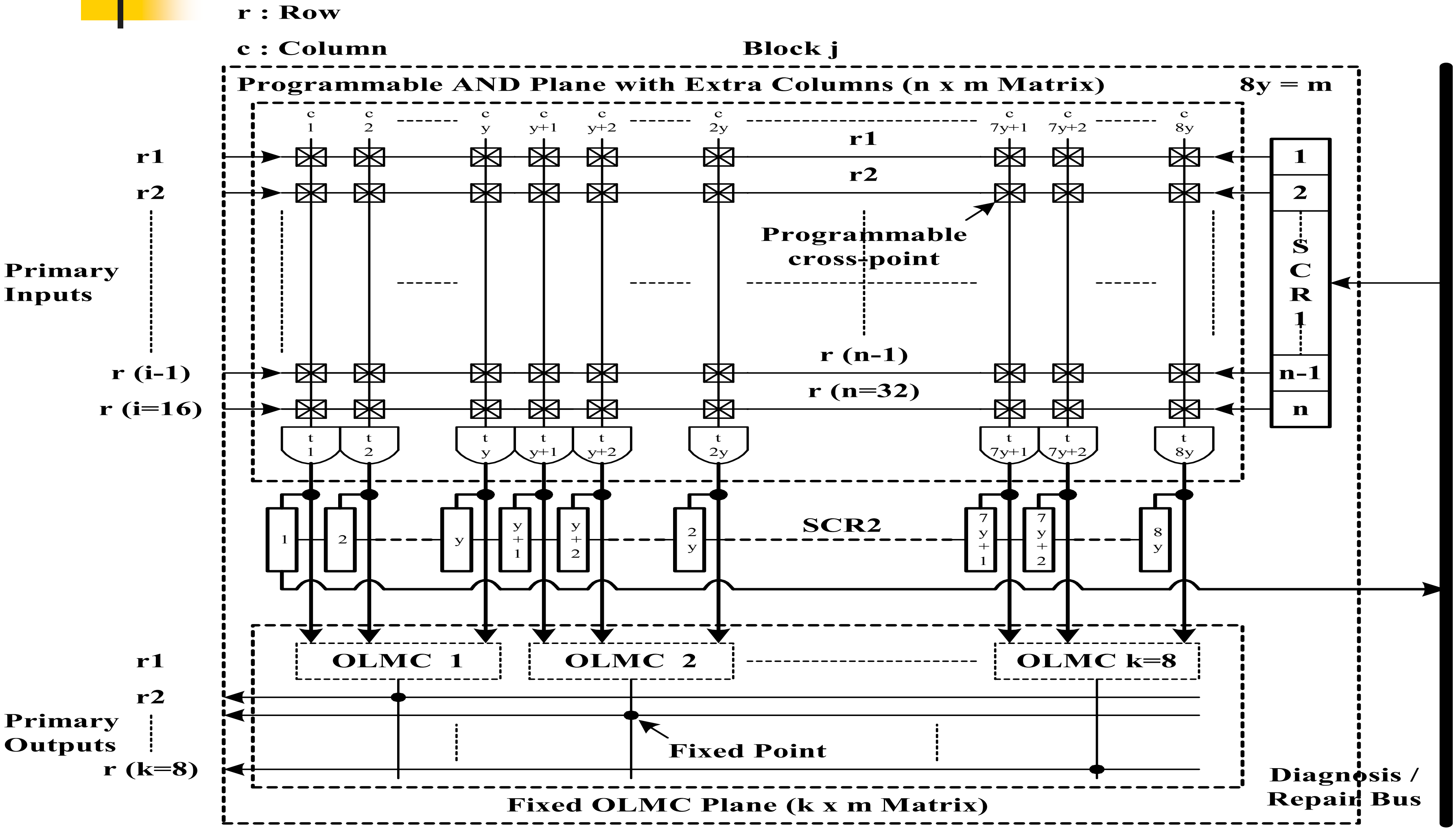
- **The problem of getting the maximum usefulness from each block (GAL) in a system at the lowest level, when the system degraded over time with new faults arising in the block.**
- Applied to military systems, satellite and astronautic systems, or medical instruments.
- Reprogramming of a GAL serves to replace gates in that GAL.
- Note that **rows will be represented as data inputs and columns will be used as product terms on the rest of figures in this presentation**.

Design Architecture for a Self-Repairable GAL

n = # of Inputs (Rows) (n = 32 in a GAL16V8) (Fixed)
k = # of Outputs (OLMCs) (k = 8 in a GAL16V8) (Fixed)
m = # of Products Terms (Columns) (m = 64 in a GAL16V8) with Extra Columns in a GAL (Variable)
y = # of Products Terms (Columns) (y = 8 in a GAL16V8) with Extra Columns in a OLMC (Variable)


Block 1
Diagnosis / Repair Bus
1
Primary Inputs
i
Programmable AND Plane with Extra Columns (n x m Matrix)
1
S C R 1
n
GAL Programmer
1
SCR 2
m
Block 2
1
Primary Outputs
k
Fixed OLMC Plane (k x m Matrix)
Block j
FLFRP (Fault Location / Fault Repair Processor)

Inner Structure of the Block



Structure of MAP (Memory AND Plane) and SAP (State AND Plane) Arrays

Inputs	c 1	c 2	------------------	c 8y-1	c 8y
r1	1	0	------------------------------	1	1
r2	0	1	------------------------------	1	1
⋮	⋮	⋮	---------------┼---------------	⋮	⋮
r (n - 1)	1	0	------------------------------	1	1
r (n = 32)	0	1	------------------------------	1	1

r : Row c : Column 1 : ON 0 : OFF 8y = m

Structure of the NC (Next Column) Register

Primary Outputs	c 1	c 2	------------------------	c y-1	c y
or1	0	0	-------------------------------------	1	1
or2	0	-1	-------------------------------------	0	1
⋮	⋮	⋮	---------------⋮---------------	⋮	⋮
or (k - 1)	-1	0	-------------------------------------	1	1
or (k = 8)	1	1	-------------------------------------	1	1

0: Column being used

1: Availble Column to replace

-1: Unavailable Column to replace

or: OR Group

c: Column in a OR gate

Test Generation

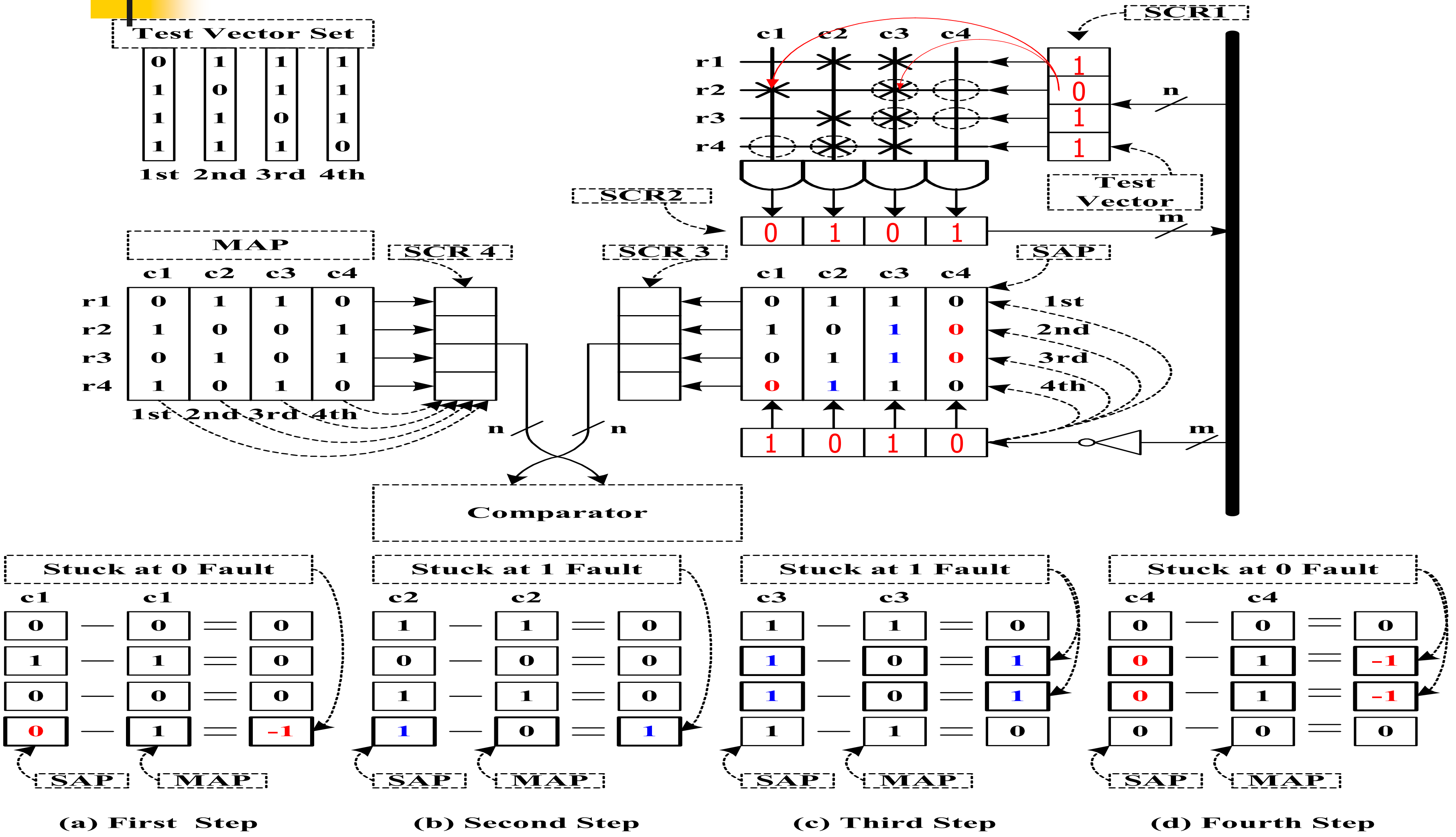
- ***Test Vector Set* (Test Pattern)**

1	2	3	…	n-1	n (inputs)
0	1	1	…	1	1
1	0	1	…	1	1
1	1	0	…	1	1
.	.	.	…	.	.
1	1	1	…	0	1
1	1	1	…	1	0

Symbol Notation

×	Programmed into ON (Connected) Cross-Point
+	Programmed into OFF (Disconnected) Cross-Point
⊕	Stuck at 0 Faulty Cross-Point
⊗	Stuck at 1 Faulty Cross-Point

Fault Diagnosis/Location of an Example with Faults



Test Vector Set

1st	2nd	3rd	4th
0	1	1	1
1	0	1	1
1	1	0	1
1	1	1	0

MAP

	c1	c2	c3	c4
r1	0	1	1	0
r2	1	0	0	1
r3	0	1	0	1
r4	1	0	1	0
	1st	2nd	3rd	4th

SAP

c1	c2	c3	c4
0	1	1	0
1	0	1	0
0	1	1	0
0	1	1	0

Stuck at 0 Fault

c1		c1		
0	−	0	=	0
1	−	1	=	0
0	−	0	=	0
0	−	1	=	−1
SAP		MAP		

(a) First Step

Stuck at 1 Fault

c2		c2		
1	−	1	=	0
0	−	0	=	0
1	−	1	=	0
1	−	0	=	1
SAP		MAP		

(b) Second Step

Stuck at 1 Fault

c3		c3		
1	−	1	=	0
1	−	0	=	1
1	−	0	=	1
1	−	1	=	0
SAP		MAP		

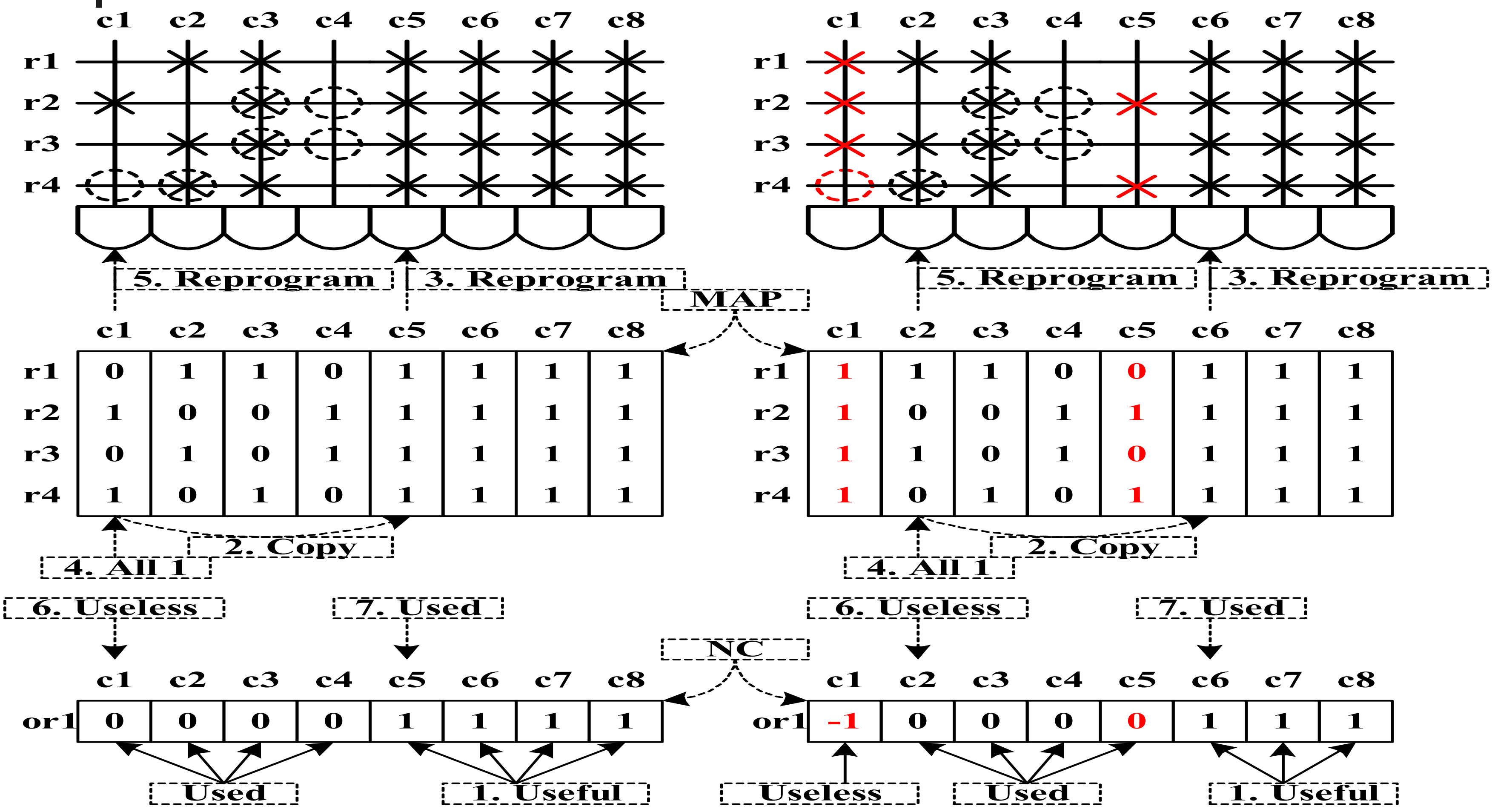
(c) Third Step

Stuck at 0 Fault

c4		c4		
0	−	0	=	0
0	−	1	=	−1
0	−	1	=	−1
0	−	0	=	0
SAP		MAP		

(d) Fourth Step

A Column Replacement Method Example of Multiple Faults



	c1	c2	c3	c4	c5	c6	c7	c8
r1	0	1	1	0	1	1	1	1
r2	1	0	0	1	1	1	1	1
r3	0	1	0	1	1	1	1	1
r4	1	0	1	0	1	1	1	1

	c1	c2	c3	c4	c5	c6	c7	c8
or1	0	0	0	0	1	1	1	1

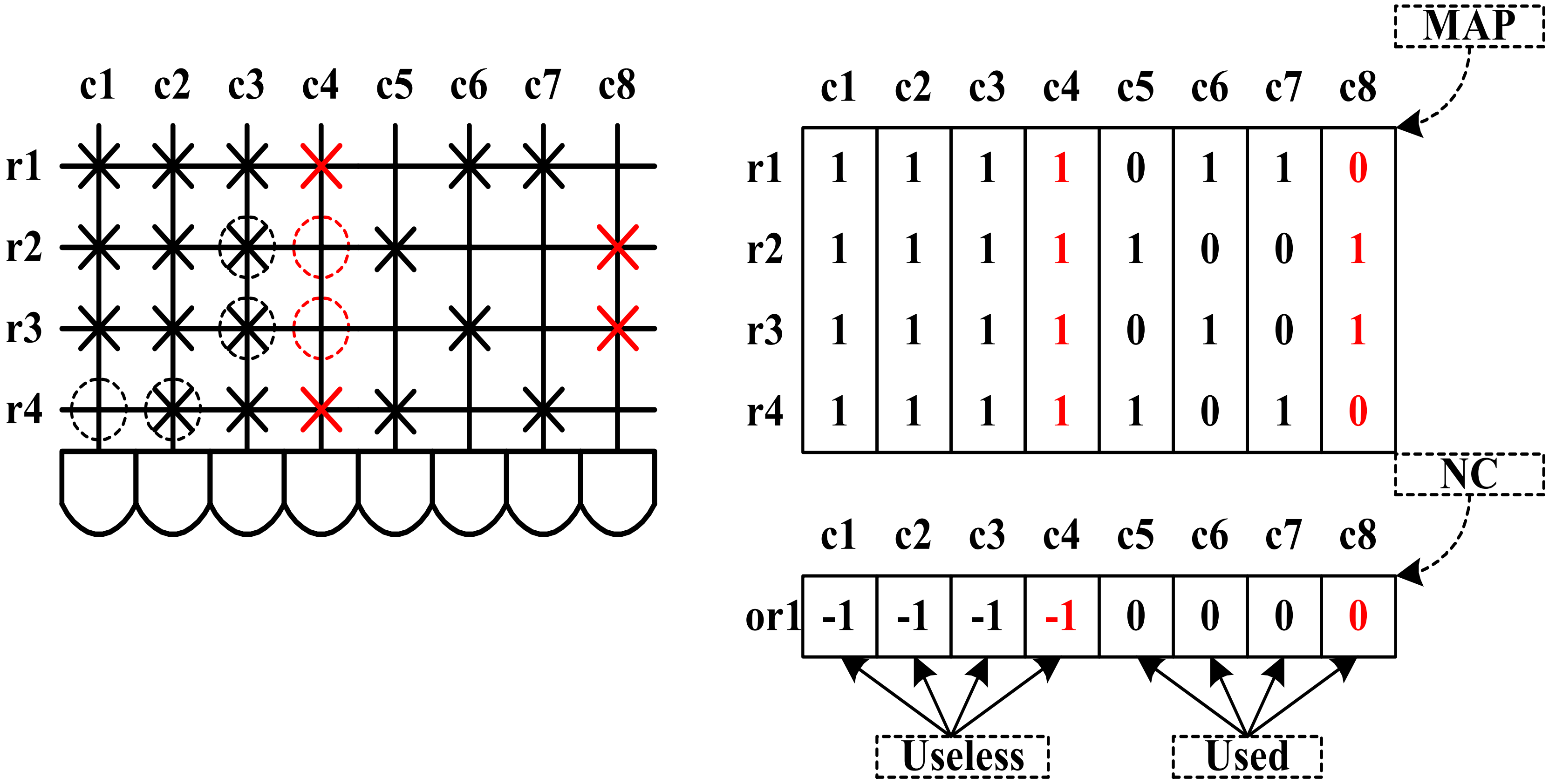
(a) c1 Replace with c5 Step

	c1	c2	c3	c4	c5	c6	c7	c8
r1	1	1	1	0	0	1	1	1
r2	1	0	0	1	1	1	1	1
r3	1	1	0	1	0	1	1	1
r4	1	0	1	0	1	1	1	1

	c1	c2	c3	c4	c5	c6	c7	c8
or1	–1	0	0	0	0	1	1	1

(b) c2 Replace with c6 Step

A Column Replacement Method Example of Multiple Faults (Continue)



	c1	c2	c3	c4	c5	c6	c7	c8
r1	1	1	1	1	0	1	1	0
r2	1	1	1	1	1	0	0	1
r3	1	1	1	1	0	1	0	1
r4	1	1	1	1	1	0	1	0

	c1	c2	c3	c4	c5	c6	c7	c8
or1	-1	-1	-1	-1	0	0	0	0

(e) Final State

The Role of Our Evaluation Methodology

- Plan how to eliminate residual defects.
- Confirm that outgoing products meet the product specification and are likely to be reliable.
- Guarantee the reliability of a GAL even in a bad condition, i.e. high temperature or deep impact, though outgoing products are satisfied with the normal product specification.

Assumptions for an Evaluation Methodology

- The same probability of stuck-at faults to occur in each cross-point.
- The failure of one cross-point is assumed to be independent of the failure of another cross-point.
- # of extra columns ≤ # of original columns.

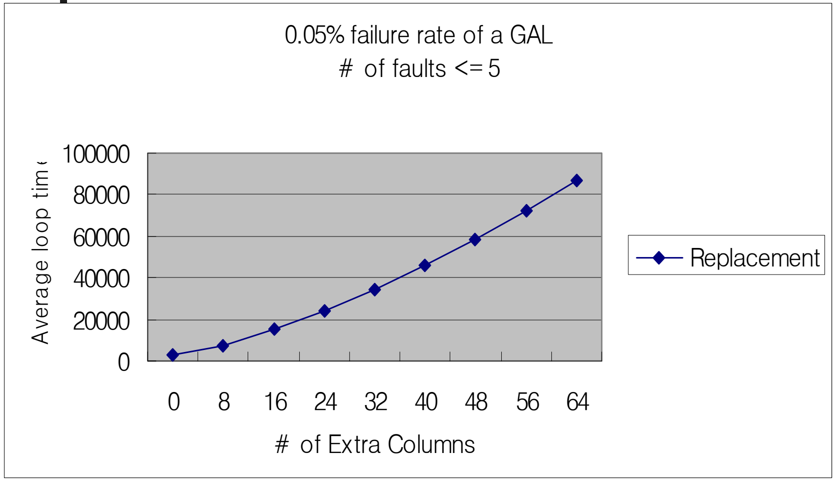
Failure Rates

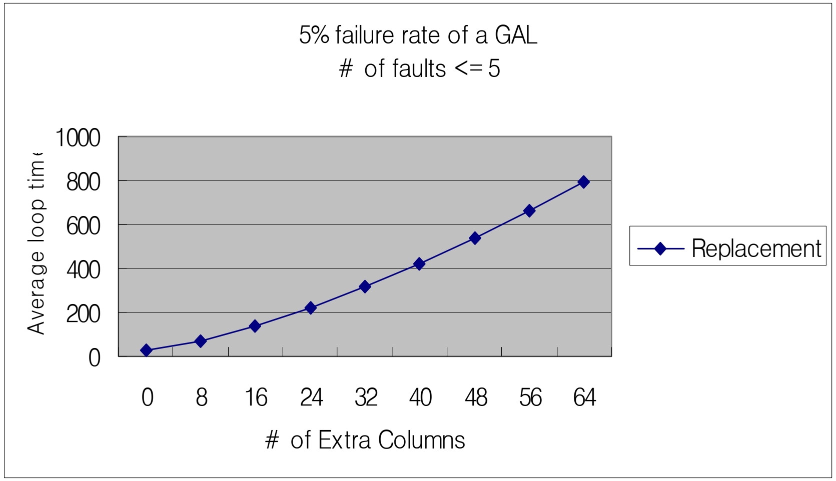
- **The most common measure of reliability** for semiconductors.
- The **FIT (Failure In Time, Long Term Failure Rate)** gives the maximum number of failures that will occur in ten thousand devices operating for ten years as well as the **PPM (Parts Per Million, Early Failure rate)** for one year.
- **The PPM, 0.05% and FIT, 5.00% of EEPROM obtained from National Semiconductor Corporation are adopted as the PPM and the FIT of a GAL in our simulation.**

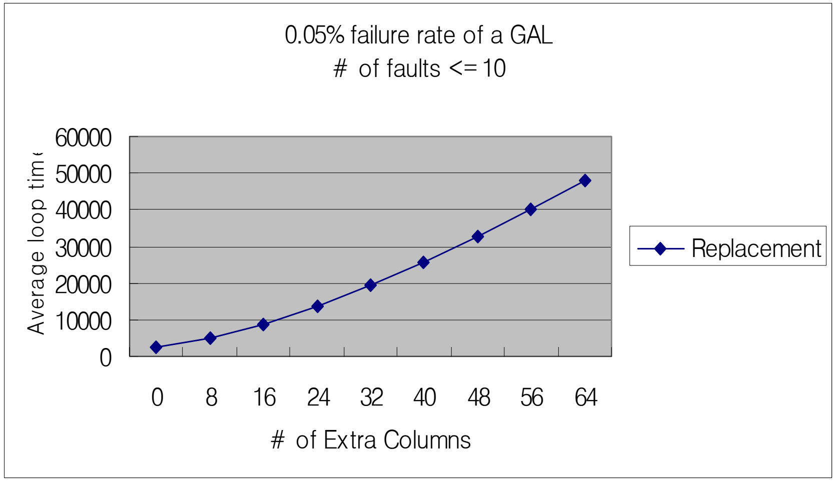
Conditions in Simulating

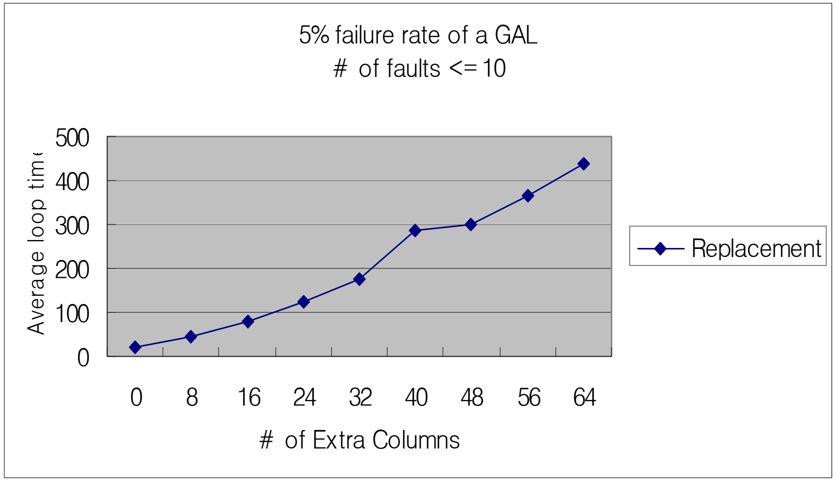
- Using 100% AND array utilization.
- Using a random number generator for all E^2CMOS cells programming.
- Using a random number generator for the faults simulating in an AND array.
- Less or equal to 5, 10, 20, 50, 100, 1000, and 2048 for the maximum number of faults at cross-points at a time.
- Faults on extra columns.
- The extra columns increased in a multiple of 8 because of 8 OLMCs in a GAL.
- The same number of extra columns in each OLMC.

Simulation Results (Correlation of Average Looping Times vs. Numbers of Extra Columns)


0.05% failure rate of a GAL
# of faults <= 5


5% failure rate of a GAL
# of faults <= 5


0.05% failure rate of a GAL
# of faults <= 10


5% failure rate of a GAL
# of faults <= 10

Area Overhead and Ratio

- In the measurement of area overhead, the LSI Logic Corporation's 0.5-micron LCA/LEA500K array-based products are used for synthesis;
  - All area measurements are expressed in cell units, excluding the interconnection wires.

# of Extra Columns	# of Cells	Ratio
0	2410	1
8	3314	1.38
16	3618	1.5
24	3930	1.63
32	4234	1.76
40	4554	1.89
48	4858	2.02
56	5170	2.15
64	5474	2.27

Performance of a Self-Repairable GAL

- The ratio of looping time divided by the ratio of area overhead at each case.

# of Extra Columns	# of faults													
	≤5		≤10		≤20		≤50		≤100		≤1000		≤2048	
	Failure rate of a GAL		Failure rate of a GAL		Failure rate of a GAL		Failure rate of a GAL		Failure rate of a GAL		Failure rate of a GAL		Failure rate of a GAL	
	0.05%	5%	0.05%	5%	0.05%	5%	0.05%	5%	0.05%	5%	0.05%	5%	0.05%	5%
0	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
8	2.04	1.64	1.43	1.45	1.04	1.04	0.83	0.83	0.78	0.76	0.73	0.72	0.72	0.72
16	3.71	3.63	2.43	2.43	1.54	1.53	0.95	0.97	0.79	0.80	0.68	0.67	0.67	0.67
24	5.44	5.40	3.47	3.48	2.11	2.10	1.13	1.13	0.82	0.83	0.63	0.61	0.63	0.61
32	7.22	7.16	4.54	4.57	2.70	2.68	1.37	1.39	0.88	0.88	0.60	0.60	0.58	0.57
40	9.00	8.87	5.60	6.85	3.26	3.28	1.60	1.64	0.96	0.99	0.56	0.56	0.54	0.53
48	10.71	10.63	6.62	6.73	3.83	3.84	1.82	1.88	1.08	1.09	0.53	0.52	0.51	0.52
56	12.42	12.28	7.65	7.76	4.37	4.41	2.05	2.09	1.20	1.21	0.51	0.51	0.49	0.49
64	14.15	13.96	8.86	8.79	4.93	4.95	2.28	2.33	1.31	1.32	0.50	0.48	0.47	0.46

Conclusions and Future Works

- Present the concept of a self-testing and self-repairing digital circuit using an example of GAL.
- Can be applied to FPGAs.
- A design method allows the repair of all multiple faults.
- A self-repairing methodology increases the performance of the device.
- An evaluation methodology proves that the self-repairable GAL will last longer in the field;
    - Present how many extra columns a GAL should have in order to guarantee the reliability and the lifetime of a GAL.

Conclusions and Future Works

- This paper is an initial attempt to present some of the possible approaches to design for self-repair.
- Although we present here only SOP (Sum of Product) type EPLDs, all ESOP (Exclusive OR Sum of Product) type circuits can be also used, possibly leading to even better results.
- Our current work is to design a prototype FPGA that improves on the above approach, and compare the hardware overhead and the reliability for different architectures with the known methods of fault tolerant design.

Questions and Discussion

References

[1] D. L. Ostapko and S. J. Hong, "Fault Analysis and Test Generation for Programmable Logic Array (PLA)," *IEEE Trans. on Computer, Vol. C-28, No. 9*, pp.617-627, September 1979.

[2] K. S. Ramanatha and N. N. Biswas, "An On-Line Algorithm for the Location of Cross Point Faults in Programmable Logic Arrays*,"* *IEEE Trans. on Computer, Vol. C-32, No. 5*, pp.438-444, May 1983.

[3] J. E. Smith, "Detection of Faults in Programmable Logic Arrays*,"* *IEEE Trans. on Computer, Vol. C-28, No. 11*, pp.845-853, November 1979.

[4] H. Fujiwara and K. Kinoshita, "A Design of Programmable Logic Array with Universal Tests," *IEEE Trans. on Computer, Vol. C-30, No. 11*, pp.823-829, November 1981.

[5] W. Daehn and J. Mucha, "A Hardware Approach to Self-Testing of Large Programmable Logic Arrays," *IEEE Trans. on Computer, Vol. C-30, No. 11*, pp.829-833, November 1981.

[6] R. Treuer, H. Fujiwara, and V. K. Agarwal, " Implementing a Built-In Self-Test PLA Design," *IEEE Design and Test*, pp.37-48, April 1985.

[7] K. Seshan, T. J. Maloney, and K. J. Wu, "The Quality and Reliability of Intel's Quarter Micron Process," *Intel Technology 3rd quarter Journal,* http://www.intel.co.uk/technology/itj/q31998/articles/, July 1998.

References

[8] National Semiconductor Corporation, “Quality Network – Failure Rates of Major Processes at National Semiconductor, National Semiconductor Failure Rate Trends, and National Semiconductor Reliability,” *http://207.82.57.10/quality/pages*, May 1999.

[9] D. Sellers, “Quality and Reliability,” Quality and Reliability Hand Book – Space Electronics Inc., *http://www.spaceelectronics.com/Spaceprod/reliability/qr.html*, June 1999.

[10] “PLA and FPGA Devices,” *http://www.elec.uq.oz.au/~e3390/lectures/lect14/sld002.htm*, 1998.

[11] “Sequential Logic Design with PLDs,” *http://www.elec.uq.oz.au/~e3390/lectures/lect14/sld014.htm*, 1998.

[12] Lattice Semiconductor Corporation, “Lattice Semiconductor Data Book 1996,” *Lattice Semiconductor Corporation*, pp.365-392, 1996.

[13] J. P. Hayes, “Fault Modeling,” *IEEE Design & Test of Computers*, pp. 88-95, April 1985.

[14] W. Maly, “Realistic Fault Modeling for VLSI Testing*,” Proceedings of the 24th ACM/IEEE Design Automation Conference*, pp. 173-180, 1987.

[15] Y. K. Malaiya, “A New Fault Model and Testing Technique for CMOS Devices,” *1982 International Test Conference*, PP. 25-34, November 1982.

References

[16] S. C. Ma, "Testing BiCMOS and Dynamic CMOS Logic," *Center for Reliable Computing Technical Report, No. 95-1*, June 1995.

[17] M. Marinescu, "Simple and Efficient Algorithms for Functional RAM Testing," *1982 International Test Conference*, pp. 236-239, November 1982.

[18] R. Nair, S. M. Thatte, and J. A. Abraham, "Efficient Algorithms for Testing Semiconductor Random-Access Memories*," IEEE Transactions on Computers, Vol. C-27, No. 6*, pp. 572-276, June 1978.

[19] F.G. Cockerill, "Quality Control for Production Testing," *1982 International Test Conference*, pp. 308-314, November 1982.

[20] LSI Logic Corporation, "LCA/LEA500K Array-Based Products Databook," *Document DB04-000002-03, Fourth Edition*, May 1997.

Fault Modeling (For Reference)

- The systematic and precise representation of physical faults in a form suitable for simulation and test generation.
- The definition of abstract or logical faults that produce approximately the same erroneous behavior as the actual physical faults.
- Good fault models should be straightforward, accurate, and easy to use.
- The most widely used fault model is the stuck-at fault model, which has been used for fault analysis and test generation in all types of logic circuits.
- The stuck-at fault model provides good coverage of permanent physical faults.

Hardware Overhead (For Reference)

- Separated into two parts; an AND plane and others (OLMCs, Input/Output Buffers/Inverters, a SCR1 (Serial-In-Parallel-Out Shift Register), and a SCR2 (Parallel-In-Serial-Out Shift Register)).
- In measurement of an AND plane size, each cross-point section consumes 1 cell, i.e. 2048 (32 inputs x 64 columns) cell units are measured for an AND plane in case of no extra columns.
- The components used in the library are 2-input EXOR gate, D flip-flop, 1-of-2 MUX, 1-of-3 MUX, 1-of-4 MUX, tri-state buffer, inverter, 2-input OR gate, and 4-input OR gate.

Hardware Overhead (For Reference)

- For total area overhead of a GAL, size of an AND plane is added up OLMCs, Input/Output Buffers/Inverters, a SCR1, and a SCR2.
  - Example: if there are 16 extra columns in an AND plane, the GAL has total number of **3618 cell units**;

    = 2560 cell units (32 x (64 + 16)) for the AND plane

    + 368 cell units (46 x 8) for 8 OLMCs

    + 18 cell units for Input/Output Buffers/Inverters

    + 192 cell units (6 x 32) for 32-bit SCR1

    + 480 cell units (6 x 80) for 80-bit SCR2