

طراحی سیستم‌های تعبیه شده Embedded System Design

فصل ششم - قسمت دوم

اعتبار سنجی

Validation

کاظم فولادی
دانشکده‌ی مهندسی برق و کامپیوتر
دانشگاه تهران

kazim@fouladi.ir



Validation

- Design for testability (DFT) and fault injection -

اعتبارسنجی

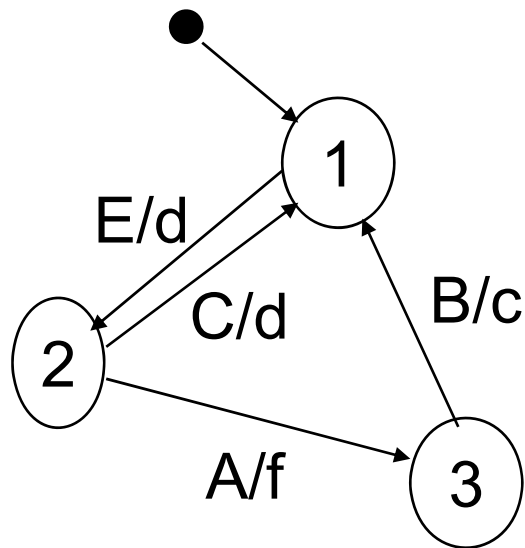
- طراحی برای آزمون پذیری و تزریق نقص -



آزمون ماشین‌های متناهی حالت

Testing finite state machines

Difficult to check states and transitions.

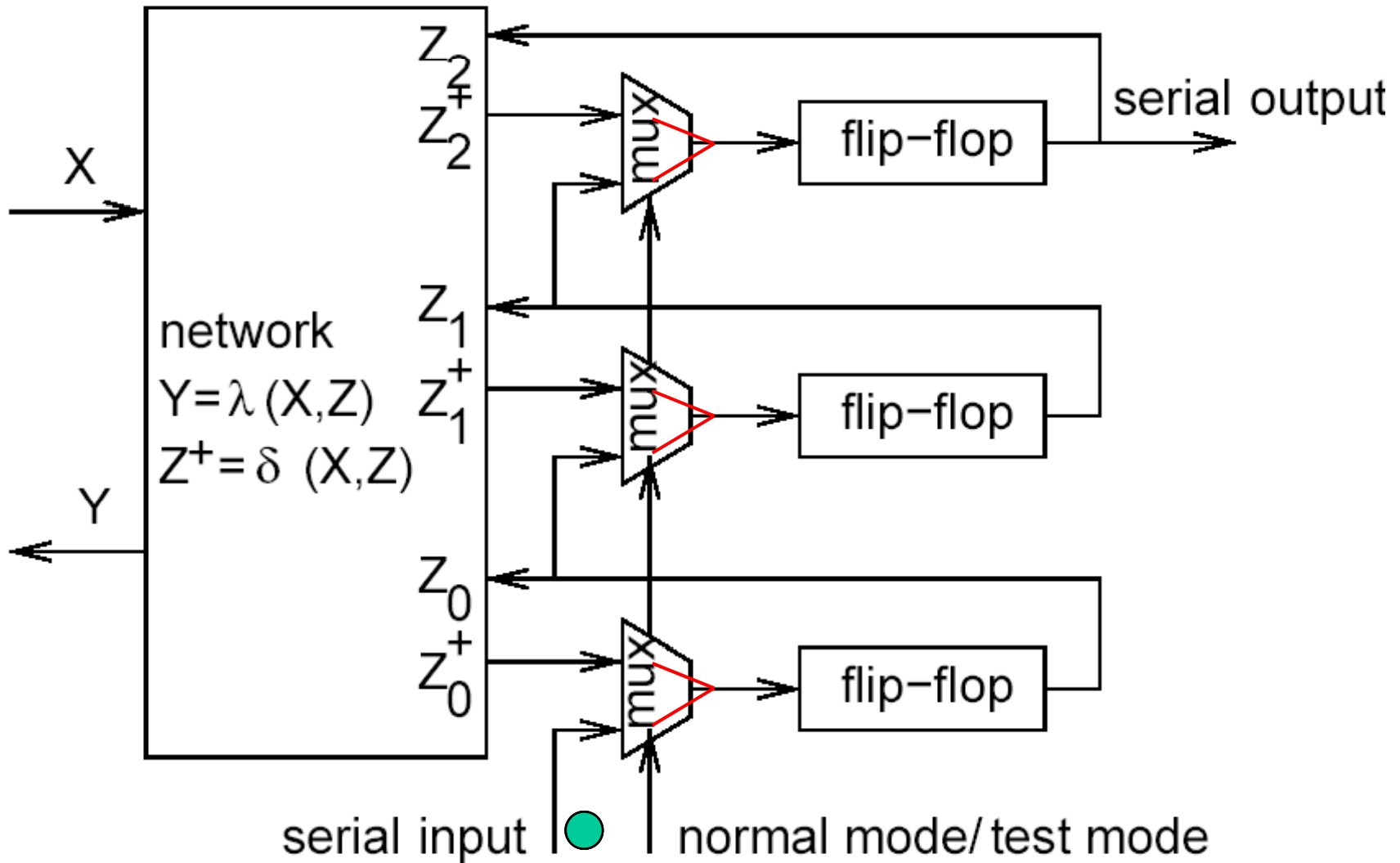


For example, verifying the transition from 2 to 3 requires

- Getting into state 2
- Application of A
- Check if output is f
- Check if we have actually reached 3

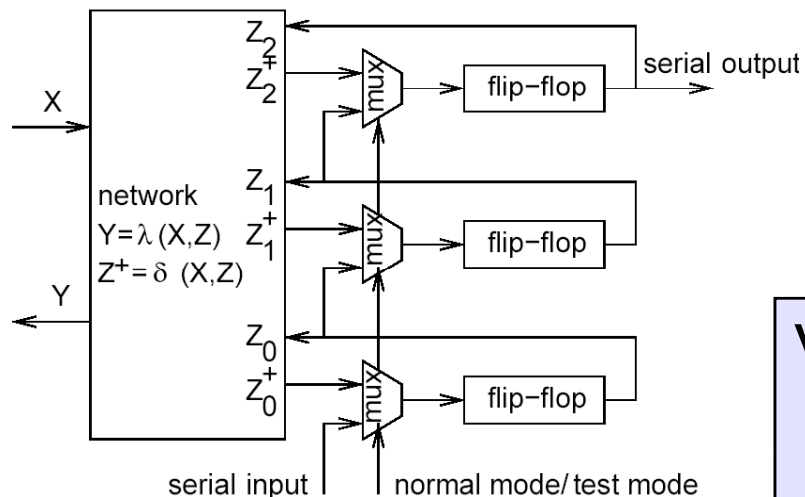
Simplified with scan design:

طراحی پویشی Scan design



طراحی پویشی : مورد استفاده

Scan design: usage



Verifying a transition requires

- Shifting-in the “old state”
- Application of the input pattern
- Checking if output is correct
- Shifting-out the new state and comparing it.

Essentially reduced to testing combinatorial logic



پویش مرزی

JTAG (Boundary scan) (1)

JTAG defines a 4..5-wire serial interface to access complex ICs .. Any compatible IC contains shift registers & FSM to execute the JTAG functions.

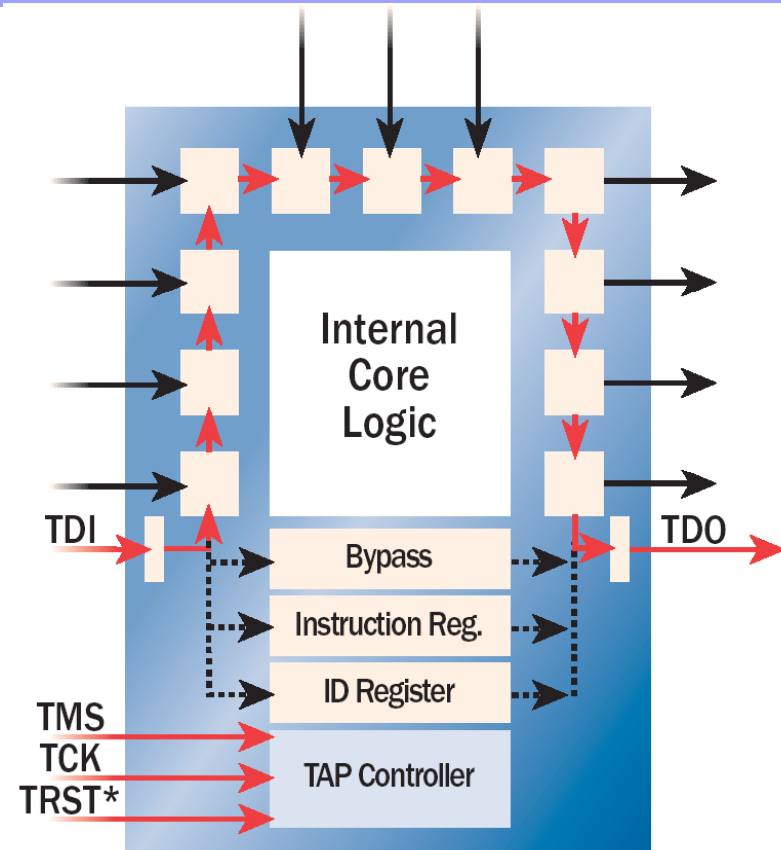
TDI: test data in; stored in instruction register or in one of the data registers.

TDO: test data out

TCK: clock

TMS: controls the state of the test access port (TAP).

Optional **TRST*** is reset signal.



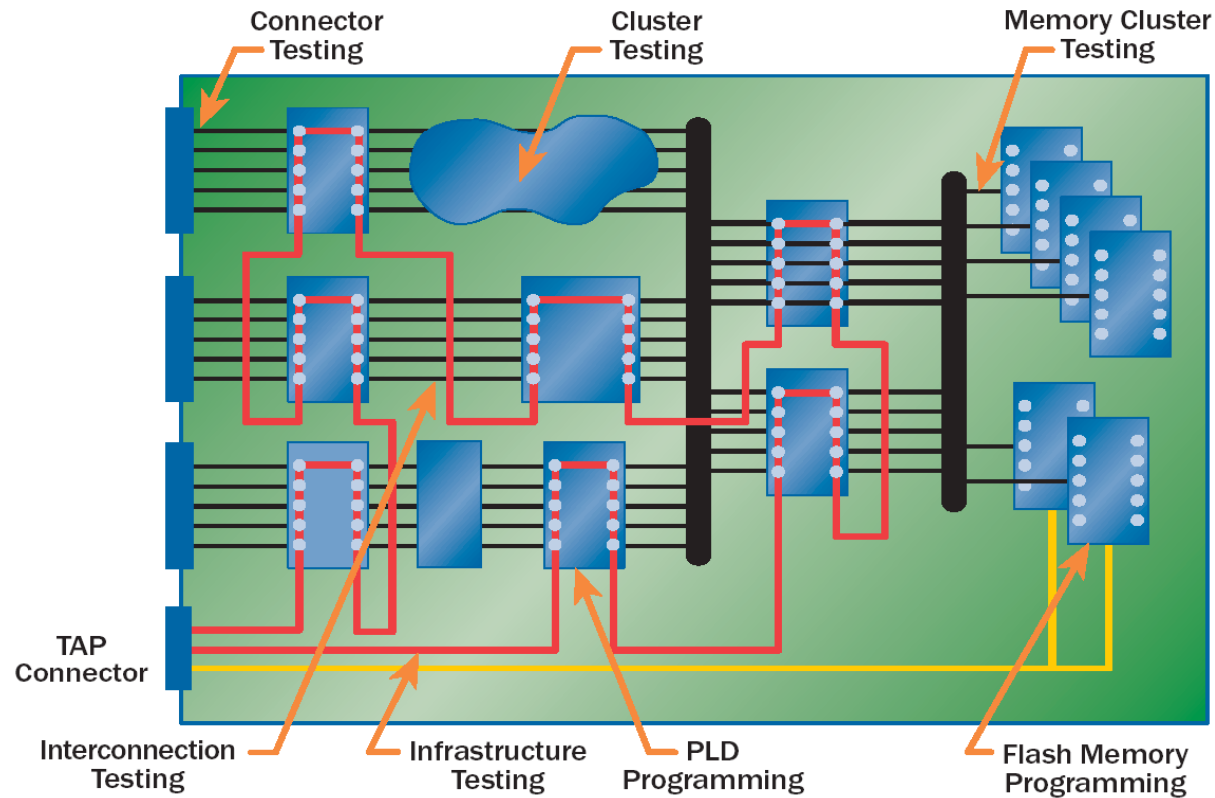
IEEE 1149.1 Device Architecture

Source: <http://www.jtag.com/brochure.php>



JTAG (Boundary scan) (2)

Defines method for setting up a scan chain on a PCB



Applications of Boundary-Scan

Source: <http://www.jtag.com/brochure.php>



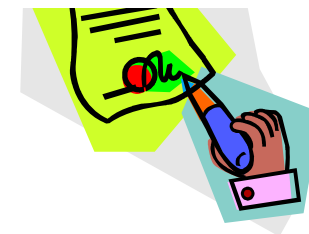
محدودیت‌های زنجیره‌ی پویش متوالی واحد

Limitations of a single serial scan chain

For chips with a large number of flop-flops, serial shifts can take a quite long time.

Hence, it becomes necessary to provide several scan chains.

- Trying to avoid serial shifts by generating test patterns internally and by also storing the results internally.
- Compaction of circuit response in a **signature**.
Shifting the entire result out becomes obsolete, we just shift out the signature.

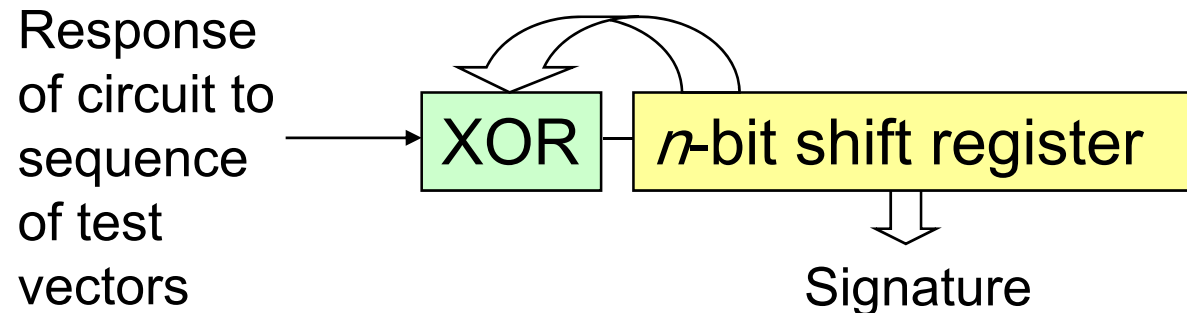


تحليل امضا

Signature analysis

Response of circuit to sequence of test patterns compacted in a signature. Only this signature is compared to the golden reference.

In order to exploit an n -bit signature register as good as possible, we try to use all values possible for that registers. In practice, we use shift-registers with linear feedback:

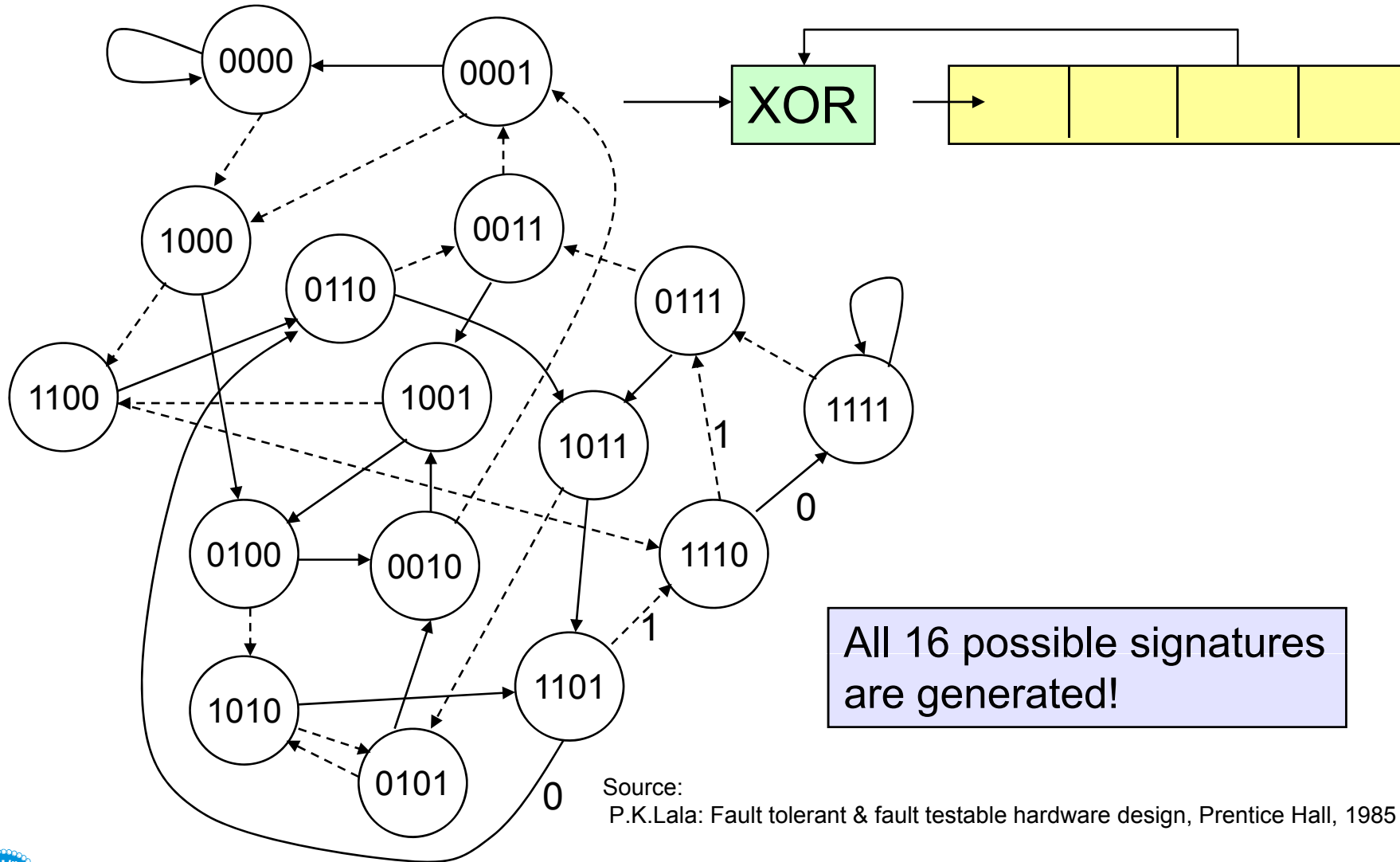


Using proper feedback bits, all values possible for the register can be generated.



مولد امضای ۴ بیتی

Example: 4-bit signature generator



All 16 possible signatures are generated!

Source:
P.K.Lala: Fault tolerant & fault testable hardware design, Prentice Hall, 1985



Aliasing for signatures

Consider aliasing for some current pattern

- An n -bit signature generator can generate 2^n signatures.
- For an m -bit input sequence, the best that we can get is to evenly map $2^{(m-n)}$ patterns to the same signature.
- Hence, there are $2^{(m-n)}-1$ sequences that map to the same signature as the pattern currently considered.
- In total, there are 2^m-1 sequences different from the current one.

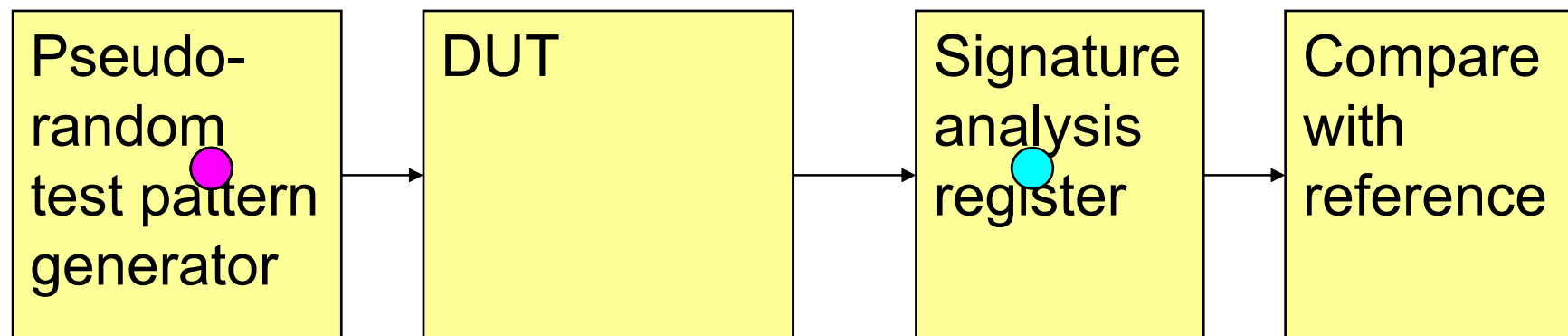
$$P = \text{Probability} \left(\frac{\text{other patterns map to same signature}}{\text{total number of other patterns}} \right) = \frac{2^{(m-n)} - 1}{2^m - 1}$$

$$P = \frac{1}{2^n} \text{ for } m \gg n \text{ provided that we evenly map patterns to signatures}$$



Replacing serially shifted test pattern by pseudo random test patterns

Shifting in test patterns can be avoided if we generate (more or less) all possible test patterns internally with a pseudo-random test pattern generator.

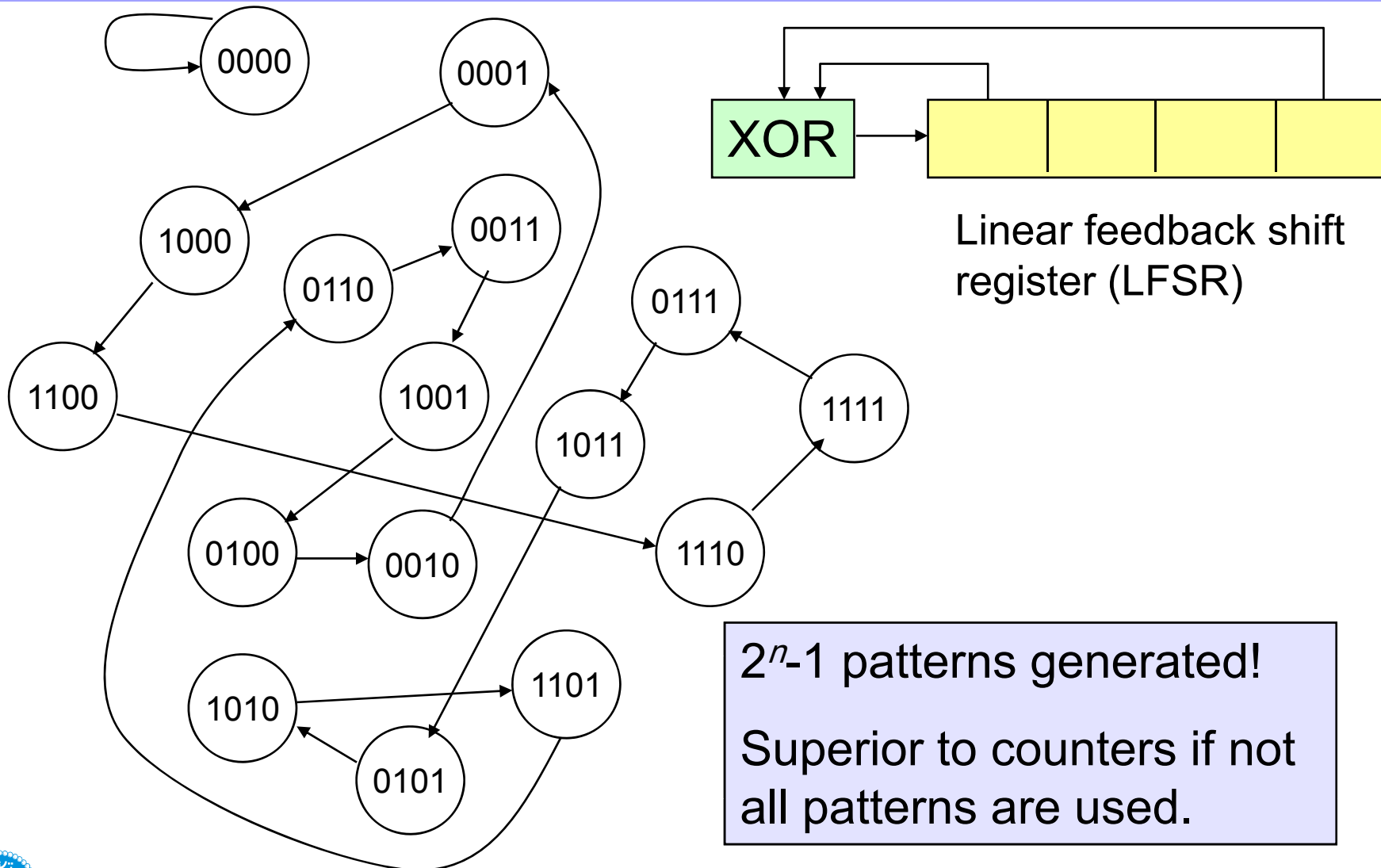


Effect of pseudo random numbers on coverage to be analyzed. Signature analysis register shifted-out at the end of the test.



تولید الگوهای آزمون شبه تصادفی

Pseudo random test pattern generation



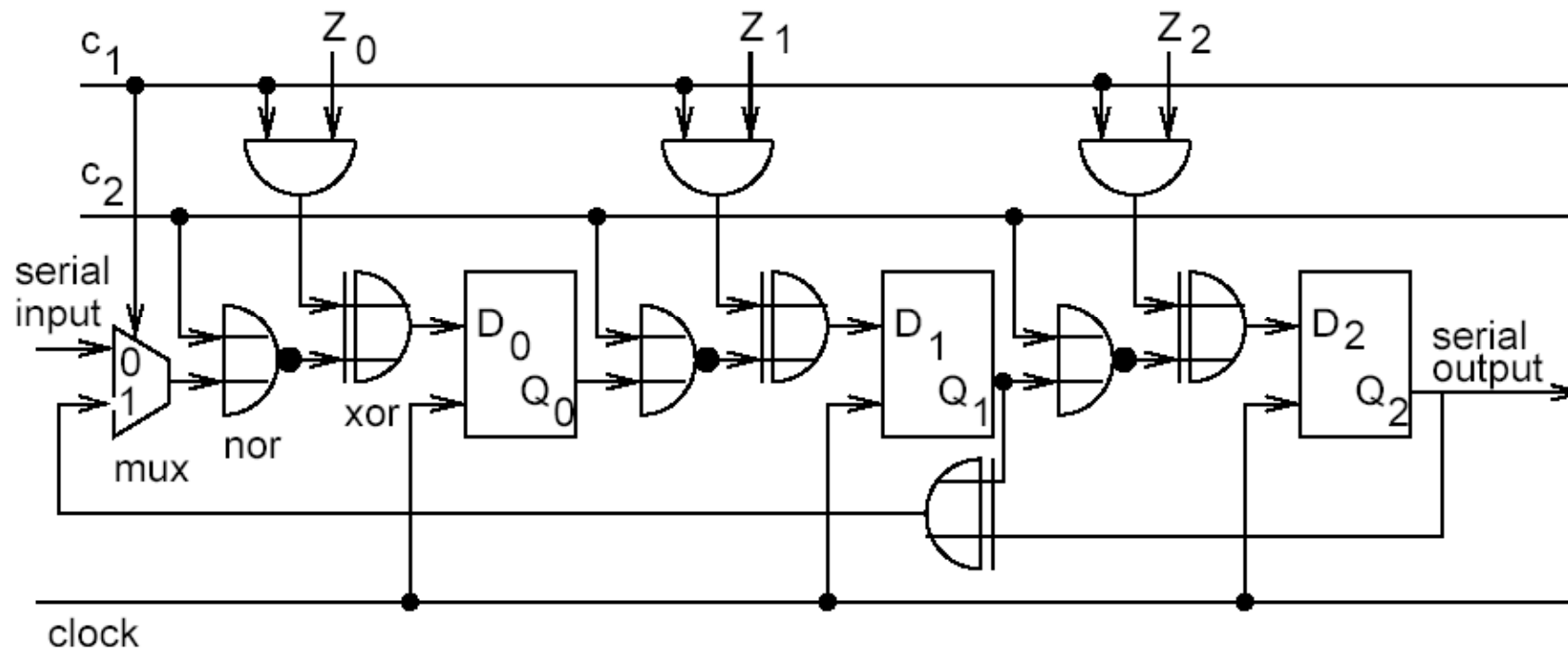
$2^n - 1$ patterns generated!
 Superior to counters if not all patterns are used.



Combining signature analysis with pseudo-random test patterns: Built-in logic block observer (BILBO)

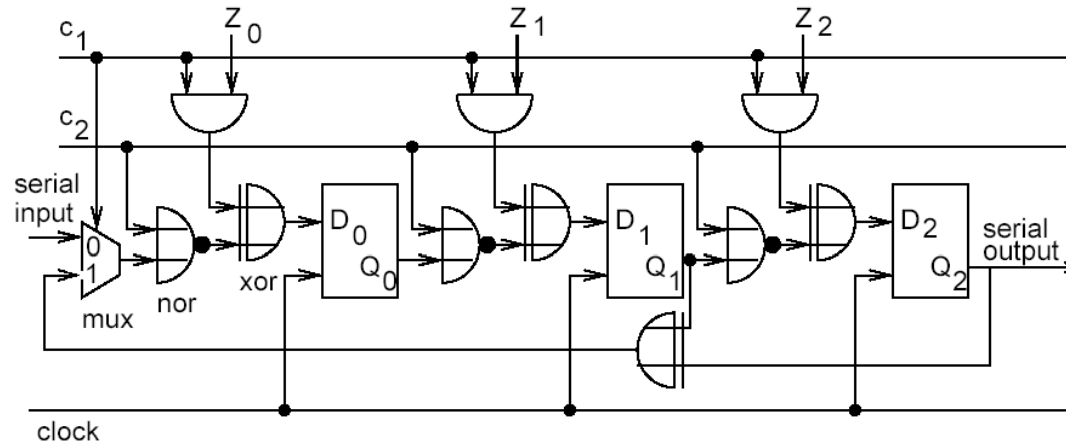
Könemann & Mucha

Uses *parallel* inputs to compress circuit response



Built-in logic block observer (BILBO)

Modes of operation

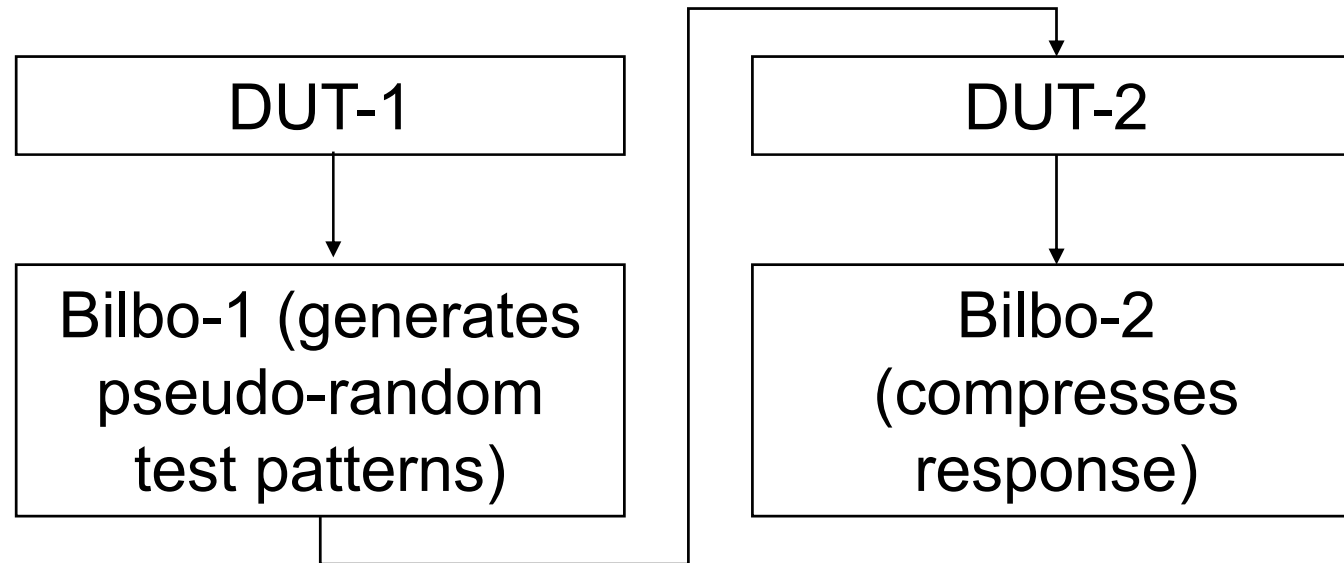


c_1	c_2	D_i	
'0'	'0'	$'0' \oplus \overline{Q_{i-1}} = \overline{Q_{i-1}}$	scan path mode
'0'	'1'	$'0' \oplus '1' = '0'$	reset
'1'	'0'	$Z_i \oplus \overline{Q_{i-1}}$	LFSR mode
'1'	'1'	$Z_i \oplus '1' = Z_i$	normal mode



کاربرد نوعی

Typical application



Compressed response shifted out of Bilbo-2 & compared to known „golden“ reference response.

Roles of Bilbo-1 and 2 swapped for testing DUT-1



تزریق نقص

Fault injection

Fault simulation may be too time-consuming

- ☞ If real systems are available, faults can be injected.

Two types of fault injection:




1. local faults within the system, and
2. faults in the environment (behaviors which do not correspond to the specification).
For example, we can check how the system behaves if it is operated outside the specified temperature or radiation ranges.



تذریق نقص فیزیکی

Physical fault injection

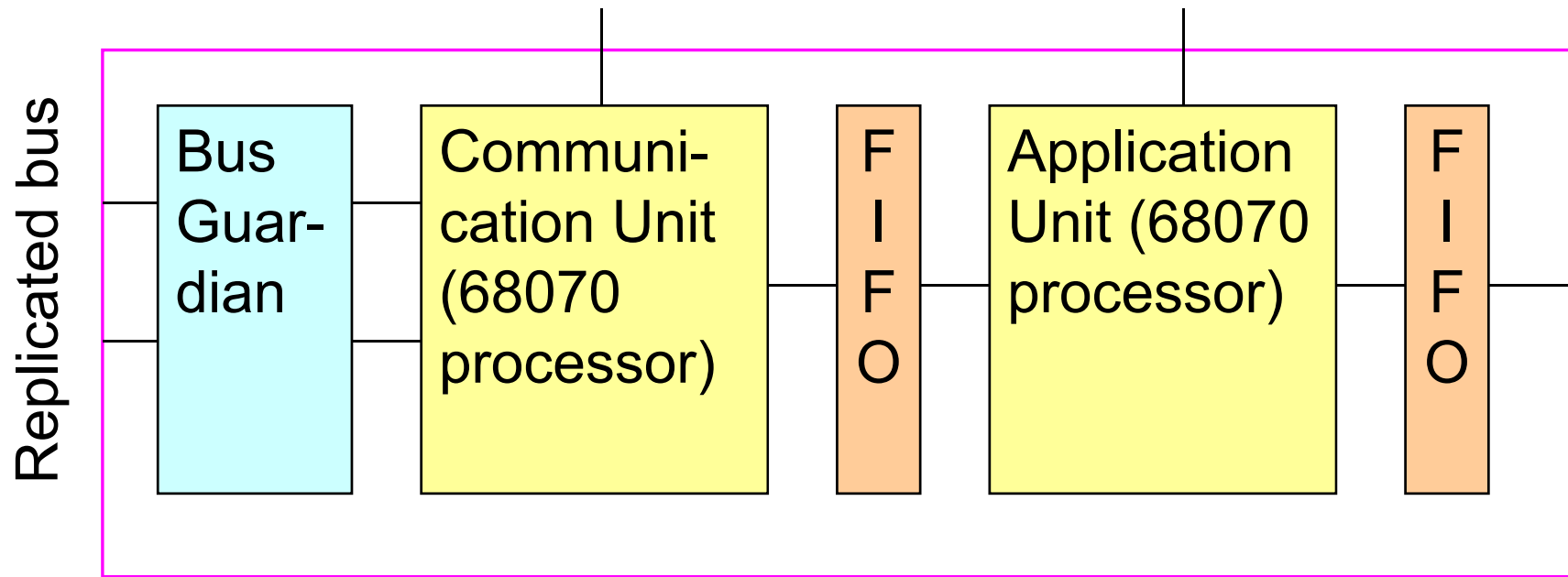
Hardware fault injection requires major effort, but generates precise information about the behavior of the real system.
 3 techniques compared in the PDCS project on the MARS hardware [Kopetz]:

Injection Technique	Heavy-ion 	Pin-level 	EMI 
Controllability, space	Low	High	Low
Controllability, time	None	High/medium	Low
Flexibility	Low	Medium	High
Reproducibility	Medium	High	Low
Physical reachability	High	Medium	Medium
Timing measurement	Medium	high	Low



MARS hardware nodes

2 asynchronous communication ports (RS 232)



Bus guardian protects against "babbling idiots"



پیاده‌سازی مکانیزم‌های آشکارسازی خطا

Implemented error detection mechanisms

1. Hardware:

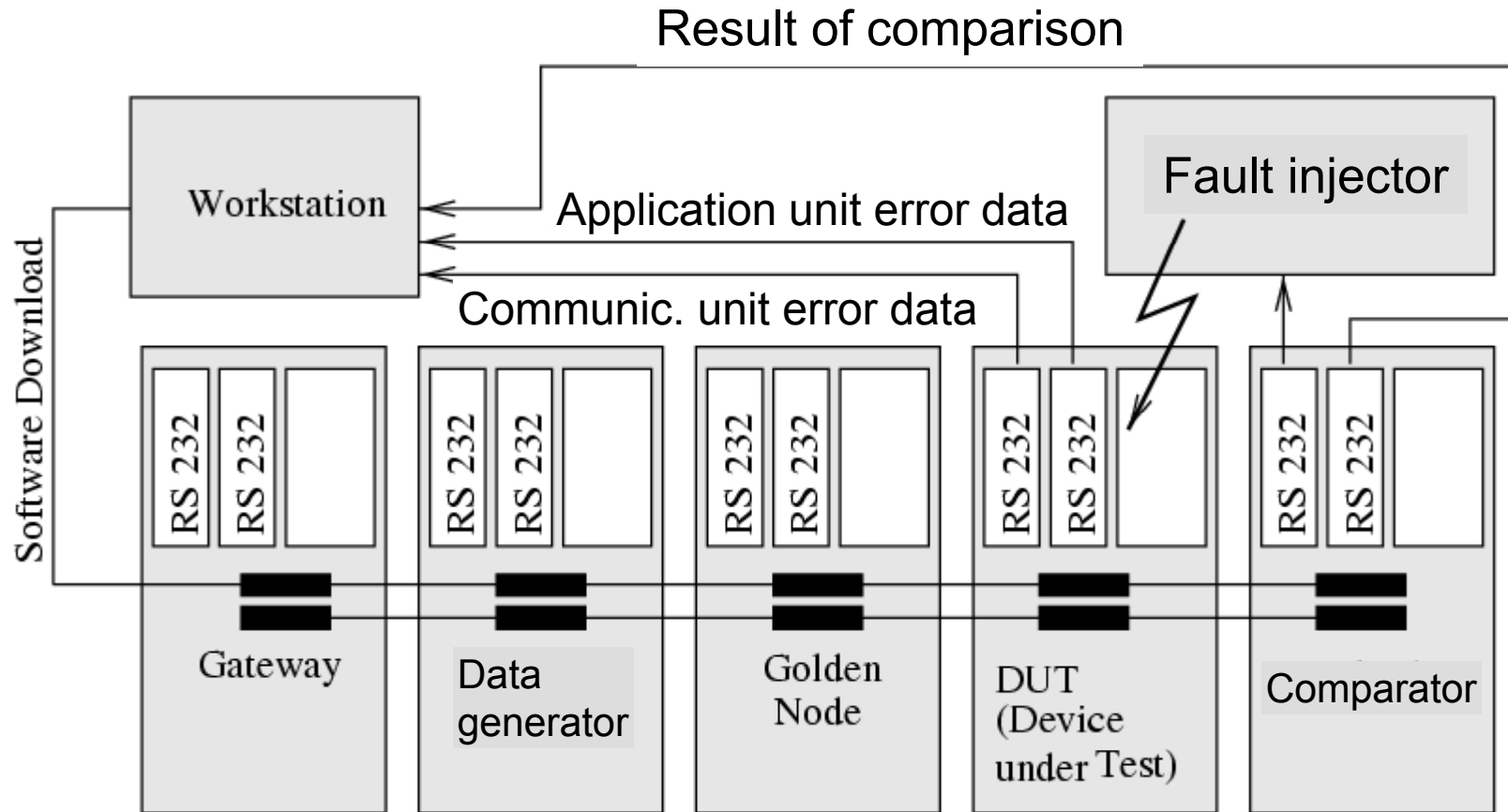
- standard mechanism for 68070 CPU: Illegal instruction & illegal address traps;
- Special mechanisms: bus guardian, FIFO overflow, power supply monitor

2. System software: compiler generated run-time assertions, timing checks to check WCET of RT-tasks

3. Application software: double execution, triple execution (two dynamic, one static in-between), end-to-end CRC



Experimental setup



Software: typical cyclic real-time application



Results

According to experiments reported by Kopetz:

1. With all error-detection mechanisms enabled, no fail-silence violation was observed in any of the experiments.
2. End-to-end error detection mechanisms and double execution of tasks were needed for all 3 fault injection methods if a coverage of $> 99\%$ were to be achieved.
3. For heavy-ion radiation, triple execution was needed.
4. The bus guardian unit was needed in all 3 experiments if a coverage of $> 99\%$ was required.



تزریق نقص نرم‌افزاری

Software fault injection

Errors are injected into the memories.

Advantages:

- **Predictability:** it is possible to reproduce every injected fault in time and space.
- **Reachability:** possible to reach storage locations within chips instead of just pins.
- **Less effort** than physical fault injection: no modified hardware.

Same quality of results?



Results

- Software fault injection with bit-flips in the data is comparable to hardware fault injection
- Application software error detection is higher for software-implemented fault injection. Most hardware-injected faults do not propagate to the application level.
- If application level error detection is turned off, software fault injection generates a higher number of coverage violations than EMI or pin level injections for single task execution.
- Software fault injection comparable to EMI and pin level injections. However, heavy-ion radiation is more stressful.

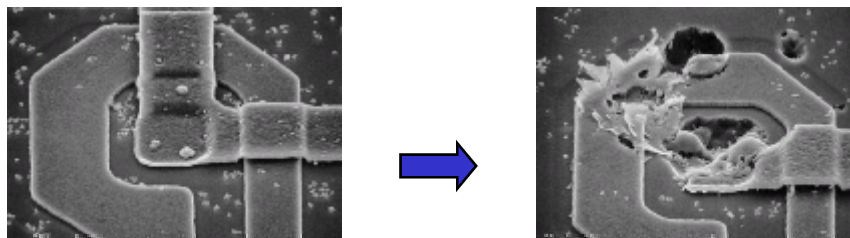


تحليل خطرپذیری و قابلیت اتکا

Risk- and dependability analysis

Example : metal migration @ Pentium 4

www.jrwhipple.com/computer_hangs.html



10^{-9} : For many systems, probability of a catastrophe has to be less than 10^{-9} per hour \equiv one case per 100,000 systems for 10,000 hours.

FIT: failure-in-time unit for failure rate ($=1/\text{MTTF} \approx 1/\text{MTBF}$);

1 FIT: rate of 10^{-9} failures per hour

Damages are resulting from hazards.

For every damage there is a severity and a probability.

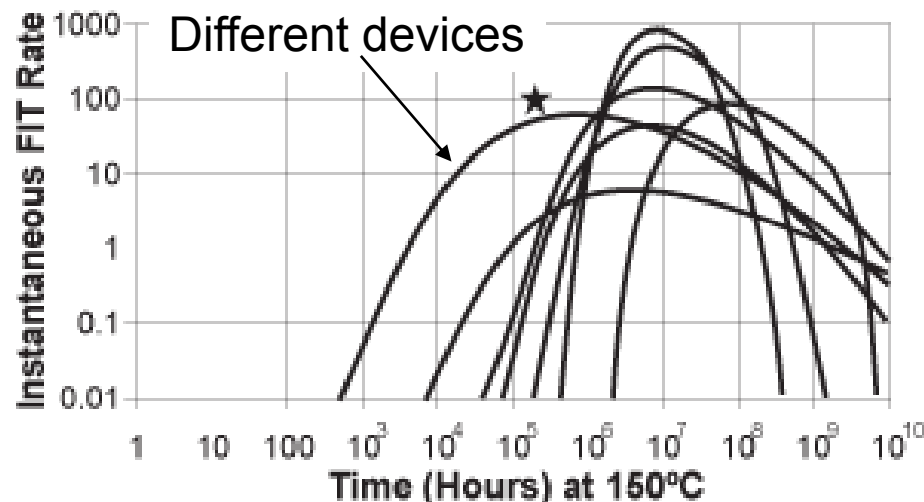
Several techniques for analyzing risks.

نرخ‌های شکست واقعی

Actual failure rates

Example: failure rates less than 100 FIT for the first 20 years of life at 150°C @ TriQuint (GaAs)

[www.triquint.com/company/quality/faqs/faq_11.cfm]



Target: Failures rates of systems ≤ 1 FIT

Reality: Failures rates of circuits ≤ 100 FIT

☞ redundancy is required to make a system more reliable than its components



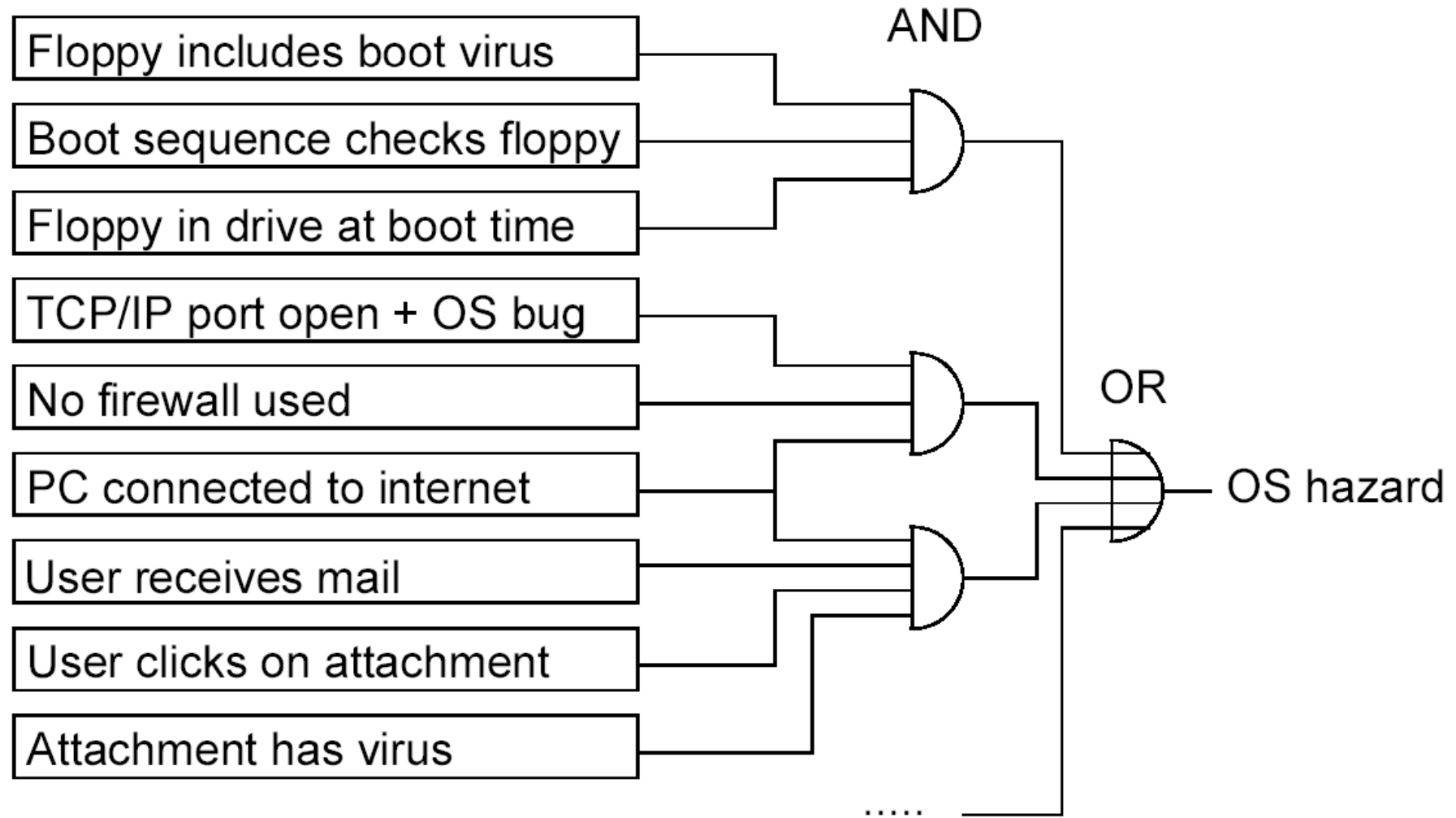
تحليل درخت نقص

Fault tree Analysis (FTA)

- FTA is a top-down method of analyzing risks. Analysis starts with possible damage, tries to come up with possible scenarios that lead to that damage.
- FTA typically uses a graphical representation of possible damages, including symbols for AND- and OR-gates.
- OR-gates are used if a single event could result in a hazard.
- AND-gates are used when several events or conditions are required for that hazard to exist.



مثال Example



محدودیتها Limitations

The simple AND- and OR-gates cannot model all situations. For example, their modeling power is exceeded if shared resources of some limited amount (like energy or storage locations) exist.

Markov models may have to be used to cover such cases.



تحليل حالت نقص و تاثير

Failure mode and effect analysis (FMEA)

- FMEA starts at the components and tries to estimate their reliability. The first step is to create a table containing components, possible faults, probability of faults and consequences on the system behavior.

<i>Component</i>	<i>Failure</i>	<i>Consequences</i>	<i>Probability</i>	<i>Critical?</i>
Processor	metal migration	no service	10^{-6} /h	yes
...

- Using this information, the reliability of the system is computed from the reliability of its parts (corresponding to a bottom-up analysis).



موارد ایمنی Safety cases

Both approaches may be used in “safety cases”. In such cases, an independent authority has to be convinced that certain technical equipment is indeed safe.

One of the commonly requested properties of technical systems is that no single failing component should potentially cause a catastrophe.



خلاصه

Summary

- Design for Test (DFT)
 - Scan path
 - Signature analysis, pseudo random patterns, BILBO
 - Boundary scan
- Fault injection
 - Physical fault injection
 - Software fault injection
- Risk- and dependability analysis
 - FTA
 - FMEA

