Reliability

Embedded Systems

Lirida Alves de Barros-Naviner
Master Program

Outline

Outline

Outline

Dependability

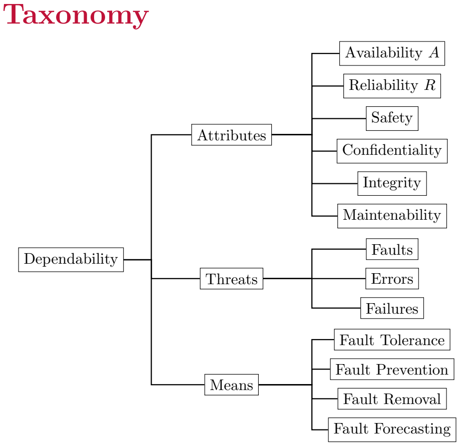
Definition

Dependability is the ability of a system to deliver service that can *justifiably* be trusted.

Definition

Dependability is the ability of a system to avoid *service failures* that are *more frequent or more severe* than is *acceptable*.

Taxonomy



Dependability Attributes

- **Availability:** readiness for correct service.
- **Reliability**: continuity of correct service.
- **Safety**: absence of catastrophic consequences on the user(s) and the environment.
- **Confidentiality (security)**: absence of unauthorized disclosure of information.
- **Integrity (security)**: absence of improper system alterations.
- **Maintainability**: ability to undergo modifications and repairs.

Dependability Threats

- **Fault**: an *unexpected (incorrect) condition* that can lead the system to achieve *abnormal states*. A fault can lead to an error.
- **Error**: an *undesired (incorrect) state* of the system. An error can lead to an incorrect response of the system.
- **Failure**: an *incorrect response* of the system. It means the service provided by the system differs from the expected one.

Means to Ensure Dependability

- **Fault prevention**: avoid things go wrong!
- **Fault tolerance**: deal with, when things go wrong!
- **Fault removal**: make it right, if things went wrong!
- **Fault forecasting**: be aware of how wrong things can go

Failure Rate

Definition

The **failure rate** λ is the expected number of failures per unit time.

- For a system with n components λ can be estimated as:

n independent components

$$\lambda = \sum_{i=1}^{n} \lambda_i$$

Mean Time to Failure

Definition

The **Mean Time to Failure (MTTF)** of a system is the expected time of the occurrence of the first system failure.

n components

$$\text{MTTF} = \frac{1}{n}\sum_{i=1}^{n} t_i$$

Failures In Time

$$\text{FIT} = \frac{10^9}{MTTF}$$

Mean Time to Repair

Definition

The **Mean Time to Repair (MTTR)** of a system is the average time required to repair the system.

- MTTR is often given in terms of the repair rate μ, which is the expected number of repairs per unit of time

$$\text{MTTR} = \frac{1}{\mu}$$

Availability

Definition

Instantaneous availability $A(t)$ is the probability the system operates at time t.

- **Interval availability** stands for the average of $A(t)$ over a mission period:

$$A(T) = \frac{1}{T}\int_0^T A(t)dt$$

- **Steady-state availability** applies when availability is time independent:

$$A(\infty) = \lim_{T\to\infty} A(T) = \frac{n \times MTTF}{n \times MTTF + n \times MTTR} = \frac{\mu}{\mu+\lambda}$$

  - Supposes n failures during lifetime

Mean Time Between Failures

Definition

The **Mean Time Between Failures (MTBF)** is the average time between failures of the system.

$$MTBF = MTTF + MTTR$$

Assuming repair makes the item perfect

$$MTBF = \frac{MTTF}{A(\infty)}$$

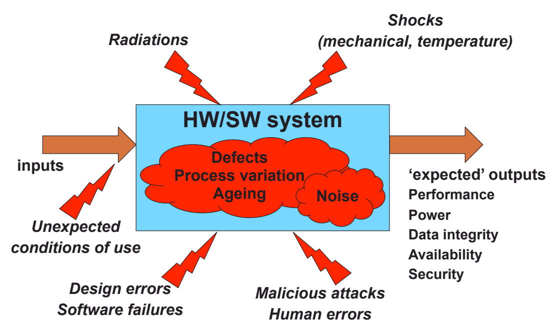
Fault Coverage

Definition

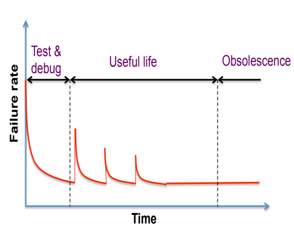
The **Fault Coverage** FC is the conditional probability related to expected actions when faults occurs.

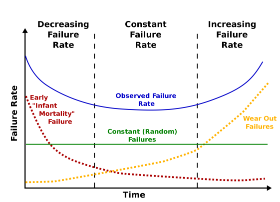
- FC= P(detected faults | existent faults)
- FC= P(located faults | existent faults)
- FC= P(recovered faults | existent faults)
- FC= P(contained faults | existent faults

What About Embedded Systems?

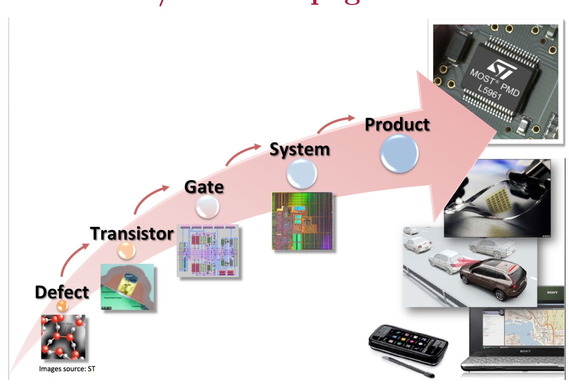


SW and HW Faults

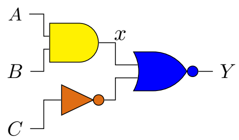




Default/Fault Propagation

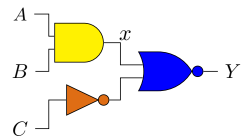


Fault Models: Bit-flip and Stuck-at



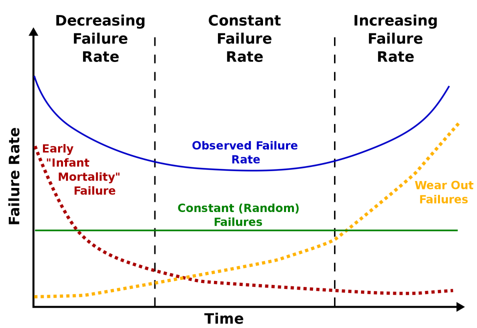
A	B	C	x	Y
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	0

Fault Models: Bit-flip and Stuck-at

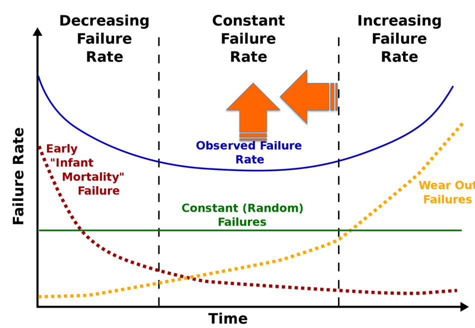


A	B	C	x	Y
0	0	0	1	0
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	1	0
1	0	1	1	0
1	1	0	1	0
1	1	1	1	0

CMOS Scaling and Reliability



CMOS Scaling and Reliability



Outline

Dependability
- Introduction
- Deterministic models
- Probabilistic models

Fault and defect tolerance improvement

Fault tolerance assessment

Prior to Beginning

- We focus on system modeling
- We consider the system consists of several components: $c_1, c_2, \cdots, c_n$
- We look for a function that enables reliability analysis

Deterministic Model

Definition

The **state of a component** c_i is defined as

$$x_i = \begin{cases} 0 & \text{if the component } c_i \text{ is not fonctionning} \\ 1 & \text{if the component } c_i \text{ is functionning} \end{cases}$$

Definition

The **state set** is defined as the vector composed by the components states

$$\mathbf{x} = (x_1 x_2 \cdots x_n)$$

Deterministic Model (cont.)

Definition

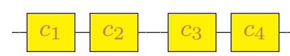
The **system state** is defined as

$$\xi(\mathbf{x}) = \begin{cases} 0 & \text{if the system is not fonctionning with state set } \mathbf{x} \\ 1 & \text{if the system is functionning with state set } \mathbf{x} \end{cases}$$

Reliability Block Diagram

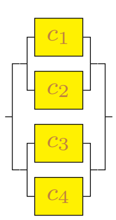
- Static representation (no reference to time)
- Each component represented by a block
- Based on logic (Boolean algebra)
- Independence of components failures
- Behavior facing faults represented by the connections between blocks

Series System



$$\begin{aligned} \xi(\mathbf{x}) &= \begin{cases} 0 & \text{if there exists an } i \text{ such that } x_i = 0 \\ 1 & \text{if } x_i = 1 \text{ for all } i \in [1; n] \end{cases} \\ &= \prod_{i=1}^{n} x_i \end{aligned}$$

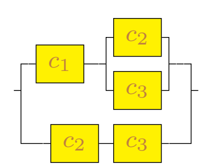
Parallel System



$$\begin{aligned} \xi(\mathbf{x}) &= \begin{cases} 0 & \text{if } x_i = 0 \text{ for all } i \in [1; n] \\ 1 & \text{if there exists an } i \text{ such that } x_i = 1 \end{cases} \\ &= 1 - \prod_{i=1}^{n} (1 - x_i) \end{aligned}$$

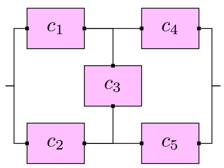
Combined Series-Parallel System

Example: 2 out of 3 structure



$$\xi(\mathbf{x}) = \begin{cases} 0 & \text{if } \sum_{i=1}^{n} x_i < k \\ 1 & \text{if } \sum_{i=1}^{n} x_i \geq k \end{cases}$$

Non Series-Parallel System



Coherent System

Definition

A system of n components is **coherent** if its function $\xi(\mathbf{x})$ is nondecreasing in $\mathbf{x}$ and there are no irrelevant components.

Definition

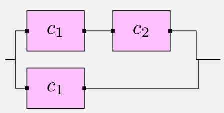
A function $\xi(\mathbf{x})$ is **nondecreasing** in $\mathbf{x}$ if $\xi(x_1 \cdots x_{i-1}\mathbf{0}x_{i+1} \cdots x_n) \leq \xi(x_1 \cdots x_{i-1}\mathbf{1}x_{i+1} \cdots x_n)$.

Definition

A component c_i is **irrelevant** if its state x_i has no impact on the function $\xi(\mathbf{x})$.

Non coherent System

Example



Structural Importance

Definition

The **structural importance** of a component c_i in a coherent system of n components is

$$I_\xi(i) = \frac{1}{2^{n-1}} \sum [\xi(1_i, \mathbf{x}) - \xi(0_i, \mathbf{x})]$$

Path Vector

Definition

A **path vector** for a coherent system is a vector $\mathbf{x}$ such as $\xi(\mathbf{x}) = 1$.

Definition

A **minimal path** for a coherent system is a path vector $\mathbf{x}$ such as $\xi(\mathbf{y}) = 0$ for all $\mathbf{y} < \mathbf{x}$.

Definition

Given two vectors $\mathbf{x}$ and $\mathbf{y}$, $\mathbf{x} < \mathbf{y}$ if and only if $x_i \leq y_i$ for $i = 1, 2, \cdots, n$ and $x_i < y_i$ for some i.

Definition

A **minimal path set** P_j for a coherent system is a set with all components associated to a given minimal path vector.

Cut Vector

Definition

A **cut vector** for a coherent system is a vector $\mathbf{x}$ such as $\xi(\mathbf{x}) = 0$.

Definition

A **minimal cut vector** for a coherent system is a cut vector $\mathbf{x}$ such as $\xi(\mathbf{y}) = 1$ for all $\mathbf{y} > \mathbf{x}$.

Definition

A **minimal cut set** C_j for a coherent system is a set with all components associated to a given minimal cut vector.

Outline

Probabilistic Model

Definition

The **random state of a component** c_i is defined as

$$X_i = \begin{cases} 0 & \text{if the component } i \text{ has failed} \\ 1 & \text{if the component } i \text{ is functionning} \end{cases}$$

Definition

The **random state of the set of components** in a system is defined as

$$\mathbf{X} = (X_1 X_2 \cdots X_n)$$

Component and System Reliability

Definition

The **reliability of a component** c_i is defined as the *probability* that component c_i is functionning [at prescribed time]

$$R_i = P\{X_i = 1\} = q_i$$

Definition

The **reliability of a coherent system** is defined by

$$R = P\{\xi(\mathbf{X}) = 1\}$$

Lifetime Models

Definition

Reliability is the ability of an item to perform its *required functions* under *stated conditions* and for a *specified period of time* (IEEE definition).

- A *item* or a *component* may mean a simple (i.e logic gate) or a complex system.
- The definition suggests *behaviour item evolution*.

Lifetime Representations

- We denote T a continuos nonnegative random variable that represents the **lifetime** of a item.
  - Note that *time* may stand to hours but also to number of flips, number of km, etc.
- We consider functions that define the distribution of T, representing the **failure time** of a item.

Probability Density Function

Definition

The **probability density function** (PDF) is defined as

$$f(t) = \lim_{\Delta t \to 0} \frac{P\{t \leq T \leq t + \Delta t\}}{\Delta t}$$

$$f(t) = 0 \text{ for } t < 0 \quad f(t) \geq 0 \text{ for } t \geq 0 \quad \int_0^1 f(t)dt = 1$$

- The PDF indicates the likelihood of failure for any t

Reliability (or Survivor) Function

Definition

The **reliability function** $R(t)$ is defined as

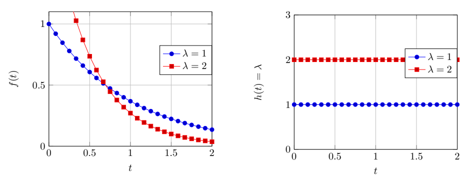
$$R(t) = R(\mathbf{q}, t) = \quad P\{T \geq t\} \quad \forall t \geq 0$$

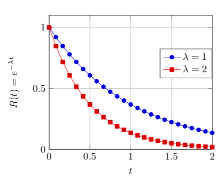
$R(t)$ must be nonincreasing and respect $R(0) = 1$, $\lim_{t\to\infty} R(t) = 0$

Lifetime Distributions

	Exponential	Weibull	Gamma
$R(t)$	$e^{-\lambda t}$	$e^{-(\lambda t)^\kappa}$	$1 - I(\kappa, \lambda t)$
$f(t)$	$\lambda e^{-\lambda t}$	$\kappa\lambda^\kappa t^{\kappa-1} e^{-(\lambda t)^\kappa}$	$\frac{\lambda}{\Gamma(\kappa)}(\lambda t)^{\kappa-1} e^{-\lambda t}$
$h(t)$	λ	$\kappa\lambda^\kappa t^{\kappa-1}$	$\frac{f(t)}{R(t)}$

Exponential Distribution





Applies for useful life zone in bathtub curve

Markov Chain

State	Time
Discrete	Discrete
Discrete	Continuous
Continuous	Discrete
Continuous	Continuous

Continuous Time Markov Chains (CTMC)

- Memoryless system
- Discrete space
- Exponential distribution (events at constant rates)

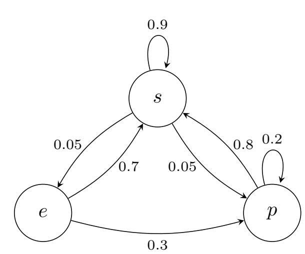
Markov Chain

A lazy, gourmand, and lovely hamster

- When Doudou sleeps, there are 9 chances out of 10 that it will be lying in bed the next minute. When it wakes up, it climbs to its happiness, so there is 1 chance out of 2 that it will be playing and 1 chance out 2 it will be eating.
- Its meals last for one minute and then it starts to play (3 chances out of 10) or it goes to sleep (7 chances out of 10).
- Doudou gets tired quickly. Frequently it goes back to sleep (8 chances out of 10) but, as it loves its spinning wheel, sometimes it continues to play.

- Knowing that Doudou is sleeping now, what will it likely be doing in three minutes?

Markov Chain & Simulation Matrix



$$S = \begin{bmatrix} 0.9 & 0.05 & 0.05 \\ 0.7 & 0 & 0.3 \\ 0.8 & 0 & 0.2 \end{bmatrix}$$

- There are three states: sleep (s), eat (e) and play (p)
- Each element $s_{i,j} \in S$ gives the probability of next state being j given that actual state is i

Simulation Matrix & Behavior

- $P(t) = \begin{bmatrix} P_s(t) & P_e(t) & P_p(t) \end{bmatrix}$ gives the probability of each state for a given time t
- Hypothesis: initial state is s, then
  - $P(0) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$
- Probability of next states are:
  - $P(1) = P(0).S = \begin{bmatrix} 0.9 & 0.05 & 0.05 \end{bmatrix}$
  - $P(2) = P(1).S = \begin{bmatrix} 0.885 & 0.045 & 0.07 \end{bmatrix}$
  - $P(3) = P(2).S = \begin{bmatrix} 0.884 & 0.04425 & 0.07175 \end{bmatrix}$
- Probability at time n: $\boxed{\text{P(n)=P(n-1).S=P(0)S}^n}$

Markov Chain & Transition Matrix

$$P_i(t+dt) = P_i(t)\left[1 - \sum_{j \neq i} s_{i,j}(t)dt\right] + \sum_{j \neq i} P_j(t)s_{j,i}dt$$

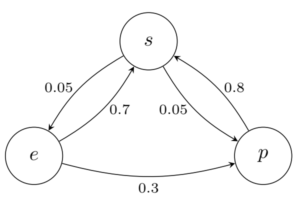
$$\frac{P_i(t+dt) - P_i(t)}{dt} = -P_i(t)\sum_{j \neq i} s_{i,j}(t)dt + \sum_{j \neq i} P_j(t)s_{j,i}dt$$

$$\frac{dP(t)}{dt} = M \cdot P(t)$$

- M is the transition matrix. Each $m_{i,j} \in M$ gives the rate with system passes from state i (at time t) to state j (at time $t + dt$)
  - $m_{i,j,i \neq j} = s_{i,j}$ and $m_{i,i} = -\sum_{j \neq i} s_{j,i}$

Markov chain & Transition Matrix

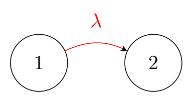
- The hamster



$$\text{M=}\begin{bmatrix} -0.1 & 0.05 & 0.05 \\ 0.7 & -1.0 & 0.3 \\ 0.8 & 0 & -0.8 \end{bmatrix}$$

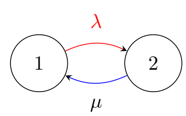
Markov chain & Transition Matrix

- One component without repair



$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} = \begin{bmatrix} -\lambda & \lambda \\ 0 & 0 \end{bmatrix}$$

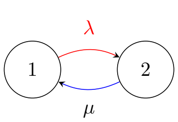
- One component with repair



$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} = \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}$$

State Transition Equations (STE)

- One component with repair



$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} = \begin{bmatrix} -\lambda & \lambda \\ \mu & -\mu \end{bmatrix}$$

$$\begin{aligned} -\lambda P_1 + \mu P_2 &= 0 \\ \lambda P_1 - \mu P_2 &= 0 \\ P_1 + P_2 &= 1 \end{aligned}$$

$$\boxed{P_1 = \frac{\mu}{\lambda + \mu} \text{ and } P_2 = \frac{\lambda}{\lambda + \mu}}$$

Reliability and STE

$$\boxed{\text{R(t)=}\textstyle\sum_{i \in \mathcal{T}} P_i(t)} \quad \boxed{\text{R(t)=1-}\textstyle\sum_{i \in \mathcal{F}} P_i(t)}$$

Assuming repair makes the item perfect, $\mathcal{T}$ is the set of fonctionning states, $\mathcal{F}$ is the set of failing states

Outline

Dependability

Introduction

Deterministic models

Probabilistic models

Fault and defect tolerance improvement

Fault tolerance assessment

Digital Design for Reliability

How to design reliable processors on unreliable devices?

- Defect tolerance
- Fault tolerance

- Prevention
- Masking
- Detection
- Correction

Defect tolerance

Based on hardening the devices

- More strict design rules at mask-level
- More expensive manufacturing (area)

Based on programmability

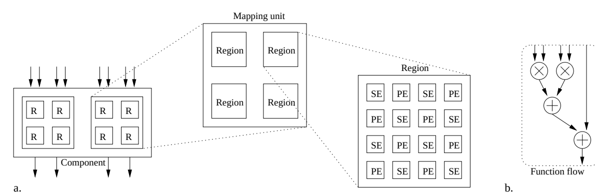
- Defectuous parts of the circuit are isolated
- Defect-free parts are used to implement the target function

Based on coding

- Very popular for memories
- Information redundancy

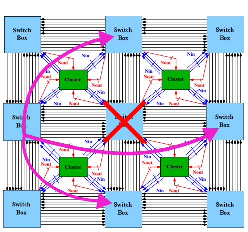
Hierarchic Nanofabric

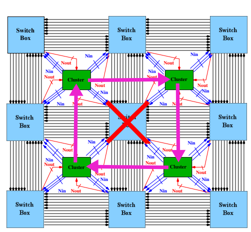
- Reconfiguration at a coarser grain
  - PEs can perform 8-bit arithmetic and logic operations
  - Tries to minimize time-consuming task of testing an mapping all of the nanofabric resources



SE= switch element, PE=processing element

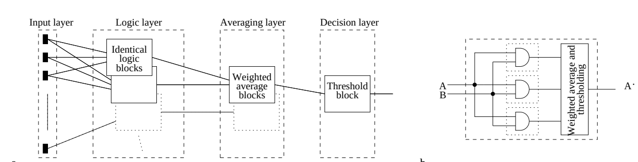
FPGA: additional connections



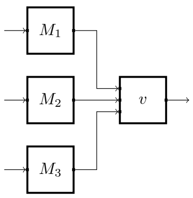


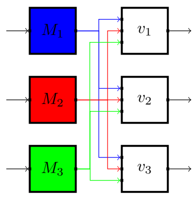
Multiple-valued Logic Approach

- Use of bit stream operators

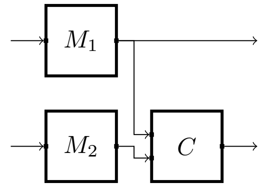


Triple Modular Redundancy

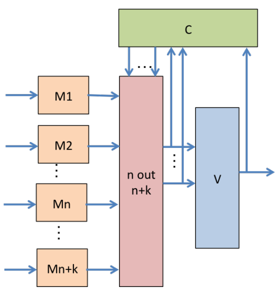




Duplication With Comparison



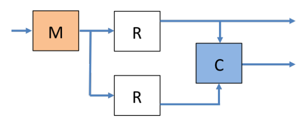
NMR with Spare



- n blocks are active at a time.
- The block **C** detects faulty blocks. It controls the action of the n-out-$n+k$ selector
- A faulty block is replaced by a spare one.

Time Redundancy & Recomputing

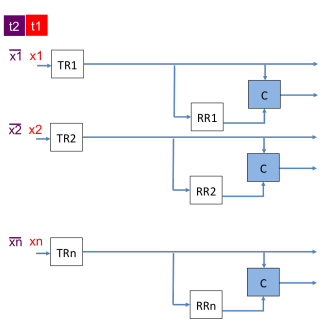
Transient faults: Duplication



Fault detection

Recomputing: Alternating Logic

Useful for permanent faults

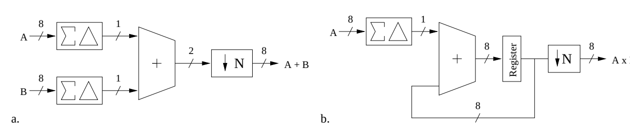


Permanent fault detection

Time Redundancy

Sigma Delta Modulation Approach

- Use of bit stream operators



Information Redundancy

- Add redundant bits to the information representation
- Mainly used for memories (data storage)
  - Error correcting coding (ECC)
- Hardware/time redundancy can be viewed as information redundancy

Parity Coding

- We consider a $n-$bits code
  - $k = n-1$ bits of information
  - 1 check bit
- Example:
  - Even parity: $\vec{x} = 1010 \Rightarrow \vec{c} = 10100$
  - Odd parity: $\vec{x} = 1010 \Rightarrow \vec{c} = 10101$

x_2	x_1	x_0	c_3	c_2	c_1	c_0
0	0	0	0	0	0	0
0	0	1	0	0	1	1
0	1	0	0	1	0	1
0	1	1	0	1	1	0
1	0	0	1	0	0	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	1	1	1	1

Even parity coding

.

(n, k) Linear Code

Matrix Representation

$$\vec{c} = \vec{x} \cdot \mathbf{G}$$

- $\vec{c}$ is the codeword
- $\vec{x}$ is the information word
- $\mathbf{G}$ generator matrix
  - The rows of $\mathbf{G}$ are k vectors which are a basis of C.
  - $\mathbf{G}$ has n columns.

Hamming Code

$$\vec{c} = \vec{x}\mathbf{G} = (1010)\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = (1011010)$$

Code (7,4): generation matrix $\mathbf{G}$, syndrome matrix $\mathbf{H}$

No error $\Rightarrow \vec{s} = \vec{c}\mathbf{H^T} = \vec{0}$

$$\vec{s} = \vec{c}\mathbf{H^T} = (1011010)\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} = (000)$$

H is in a lexicographic form

Error Vector	Syndrome $\mathbf{s} = \mathbf{cH^T}$
1000000	100
0100000	010
0010000	001
0001000	110
0000100	101
0000010	011
0000001	111

Hamming Code

$$\vec{c} = \vec{x}\mathbf{G} = (1010)\begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = (1011010)$$

Code (7,4): generation matrix $\mathbf{G}$, syndrome matrix $\mathbf{H}$

No error $\Rightarrow \vec{s} = \vec{c}\mathbf{H^T} = \vec{0}$

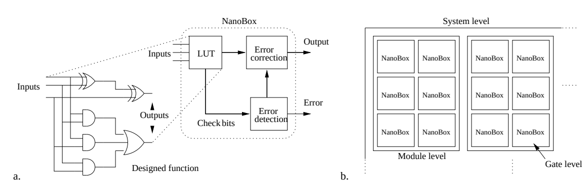
$$\vec{s} = \vec{c}\mathbf{H^T} = (1011110)\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} = (101)$$

H is in a lexicographic form

Error Vector	Syndrome $\mathbf{s} = \mathbf{cH^T}$
1000000	100
0100000	010
0010000	001
0001000	110
0000100	101
0000010	011
0000001	111

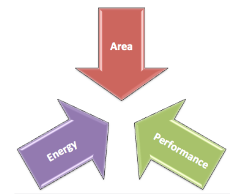
NanoBox Approach

- Dense regular structures with reconfigurable capabilities



Outline

Optimized Design of Processors



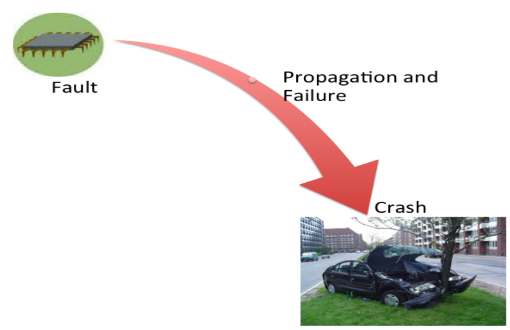
Optimized Design of Processors

How to get **optimized** design of **reliable** processors based on **unreliable** devices?

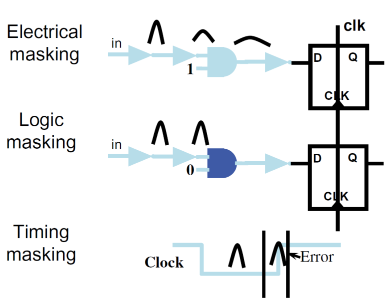
- Risk minimization
- More (than) Moore
- Fabless generalization

Reliability improvement
⇒ penalties !

Fault Propagation

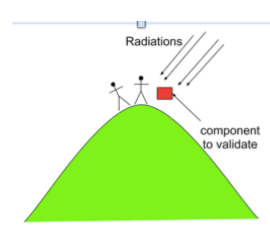


Masking Effects



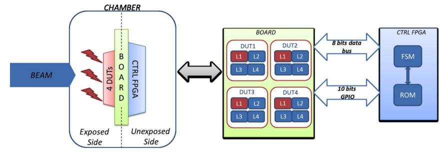
⇒ Reliability assessment !

Hardware Fault Injection

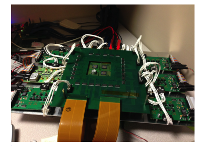


Laser Fault Injection

Heavy Ions Test on a μProc

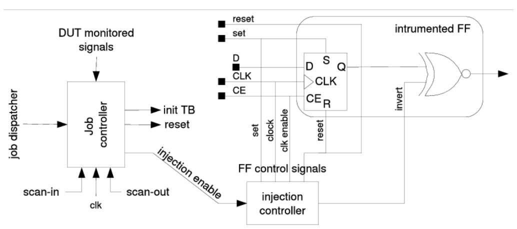


Test Board and Experimental Setup



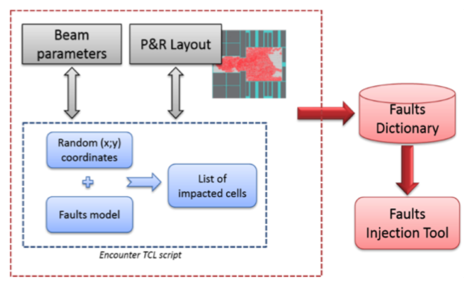
Alpha-particles irradiation for setup validation

Fault Injection: HW Emulation



Saboteur. Setup for SEU Fault Injection

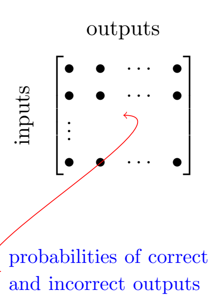
Fault Injection: Simulation



Platform General Structure

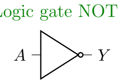
Probabilistic Transfer Matrix (PTM)

- A matrix that models the probability of output values according to the occurrence of input values and gate reliability.
- A fault-free gate is modeled by ideal transfer matrix (ITM).



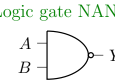
Examples

Logic gate NOT



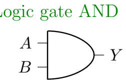
$$PTM = \begin{bmatrix} p_0 & q_0 \\ q_1 & p_1 \end{bmatrix} \quad ITM = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Logic gate NAND



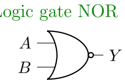
$$PTM = \begin{bmatrix} p_{00} & q_{00} \\ p_{01} & q_{01} \\ p_{10} & q_{10} \\ q_{11} & p_{11} \end{bmatrix} \quad ITM = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Logic gate AND



$$PTM = \begin{bmatrix} q_{00} & p_{00} \\ q_{01} & p_{01} \\ q_{10} & p_{10} \\ p_{11} & q_{11} \end{bmatrix} \quad ITM \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Logic gate NOR



$$PTM = \begin{bmatrix} p_{00} & q_{00} \\ q_{01} & p_{01} \\ q_{10} & p_{10} \\ q_{11} & p_{11} \end{bmatrix} \quad ITM \begin{bmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$$

Reliability ↔ PTM

Definition

$$R = \sum_{(i,j)|ITM(i,j)=1} p(j|i)p(i) = \sum_{(i,j)|ITM(i,j)=1} PTM(i,j)p(i) \quad (1)$$

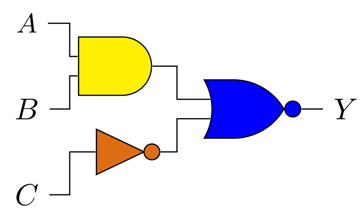
Logic gate NAND

$$R = P\{00,1\} + P\{01,1\} + P\{10,1\} + P\{11,0\}$$

$$P_{AB} = \begin{bmatrix} a_0b_0 \\ a_0b_1 \\ a_1b_0 \\ a_1b_1 \end{bmatrix} \quad PTM_{NAND} = \begin{bmatrix} p_{00} & q_{00} \\ p_{01} & q_{01} \\ p_{10} & q_{10} \\ q_{11} & p_{11} \end{bmatrix} \quad ITM_{NAND} \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$R = a_0b_0q_{00} + a_0b_1q_{01} + a_1b_0q_{10} + a_1b_1q_{11}$$

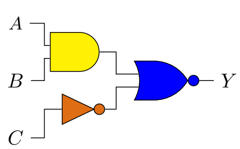
Example



$p_{ij} = p$ and $q_{ij} = q$ for all (i,j)

Parallel blocks: Kronecker Product

$$M_1 = \begin{bmatrix} q_{00} & p_{00} \\ q_{01} & p_{01} \\ q_{10} & p_{10} \\ p_{11} & q_{11} \end{bmatrix} \quad M_2 = \begin{bmatrix} p_0 & q_0 \\ q_1 & p_1 \end{bmatrix} \quad M_{L_1} = M_1 \otimes M_2$$

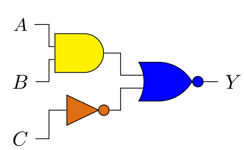


$$M_{L_1} = \begin{bmatrix} pq & q^2 & p^2 & pq \\ q^2 & pq & pq & p^2 \\ pq & q^2 & p^2 & pq \\ q^2 & pq & pq & p^2 \\ pq & q^2 & p^2 & pq \\ q^2 & pq & pq & p^2 \\ p^2 & pq & pq & q^2 \\ pq & p^2 & q^2 & pq \end{bmatrix}$$

$p_{ij} = p$ and $q_{ij} = q \forall (i,j)$

Series blocks: Scalar Product

$$M_3 = \begin{bmatrix} p_{00} & q_{00} \\ q_{01} & p_{01} \\ q_{10} & p_{10} \\ q_{11} & p_{11} \end{bmatrix} \quad PTM_{circ} = M_{circ} = M_{L1}.M_3$$



$$\begin{bmatrix} 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ p^2q + 3pq^2 & p^3 + 2p^2q + q^3 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ p^2q + 3pq^2 & p^3 + 2p^2q + q^3 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ p^2q + 3pq^2 & p^3 + 2p^2q + q^3 \\ p^3 + 2pq^2 + q^3 & 3p^2q + pq^2 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2 \end{bmatrix}$$

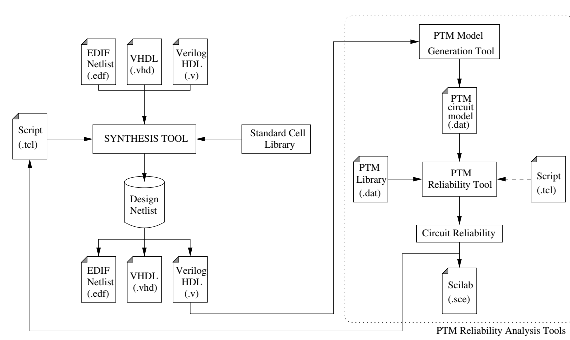
$p_{ij} = p$ and $q_{ij} = q \forall (i,j)$

Reliability Calculation

$$M_{circ} = \begin{bmatrix} 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ p^2q + 3pq^2 & p^3 + 2p^2q + q^3 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ p^2q + 3pq^2 & p^3 + 2p^2q + q^3 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \\ p^2q + 3pq^2 & p^3 + 2p^2q + q^3 \\ p^3 + 2pq^2 + q^3 & 3p^2q + pq^2 \\ 2p^2q + pq^2 + q^3 & p^3 + p^2q + 2pq^2 \end{bmatrix} \quad ITM_{circ} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$$

$$\begin{aligned} R \;=\; & a_0b_0c_0\left(2p^2q + pq^2 + q^3\right) + a_0b_0c_1\left(p^3 + 2p^2q + q^3\right) + \\ & a_0b_1c_0\left(2p^2q + pq^2 + q^3\right) + a_0b_1c_1\left(p^3 + 2p^2q + q^3\right) + \\ & a_1b_0c_0\left(2p^2q + pq^2 + q^3\right) + a_1b_0c_1\left(p^3 + 2p^2q + q^3\right) + \\ & a_1b_1c_0\left(p^3 + 2pq^2 + q^3\right) + a_1b_1c_1\left(2p^2q + pq^2 + q^3\right) \end{aligned}$$

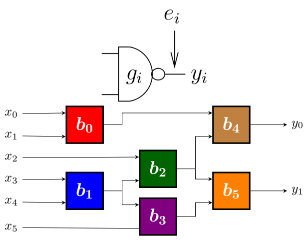
PTM Computing Flow



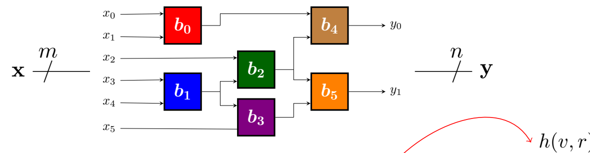
Probabilistic Binomial Reliability (PBR)

Fault Modelling

- $f_i = 1$ inverts expected g_i output
- Remark: a component c_i can be a simple gate g_i or a block of gates $b_i = 1$



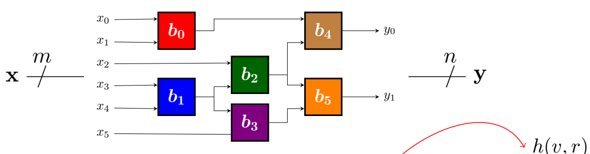
Reliability Calculation with PBR



$$R = \sum_{r=0}^{2^n-1} P\{\mathbf{f}_r\} \sum_{v=0}^{2^m-1} P\{\mathbf{x}_v\} \overbrace{(\mathbf{y}(\mathbf{x}_v, \mathbf{f}_0) = \mathbf{y}(\mathbf{x}_v, \mathbf{f}_r)}$$

- Inputs relevance
- Technology & fault relevance
- Logical masking

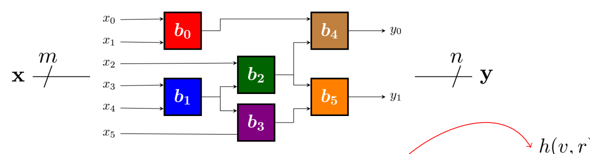
Reliability Calculation with PBR



$$R = \sum_{r=0}^{2^n-1} P\{\mathbf{f}_r\} \sum_{v=0}^{2^m-1} P\{\mathbf{x}_v\} \overbrace{(\mathbf{y}(\mathbf{x}_v, \mathbf{f}_0) = \mathbf{y}(\mathbf{x}_v, \mathbf{f}_r)}$$

- Inputs relevance
- Technology & fault relevance
- Logical masking

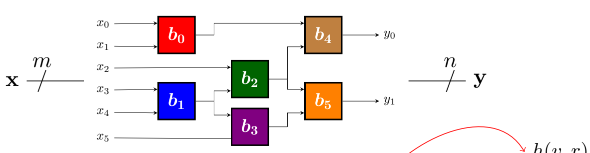
Reliability Calculation with PBR



$$R = \sum_{r=0}^{2^n-1} P\{\mathbf{f}_r\} \sum_{v=0}^{2^m-1} P\{\mathbf{x}_v\} \overbrace{(\mathbf{y}(\mathbf{x}_v, \mathbf{f}_0) = \mathbf{y}(\mathbf{x}_v, \mathbf{f}_r)}$$

- Inputs relevance
- Technology & fault relevance
- Logical masking

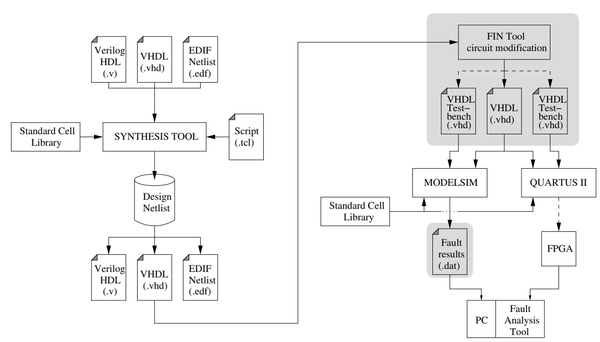
Reliability Calculation with PBR



$$R = \sum_{r=0}^{2^n-1} P\{\mathbf{f}_r\} \sum_{v=0}^{2^m-1} P\{\mathbf{x}_v\} \overbrace{(\mathbf{y}(\mathbf{x}_v, \mathbf{f}_0) = \mathbf{y}(\mathbf{x}_v, \mathbf{f}_r)}$$

- Inputs relevance
- Technology & fault relevance
- Logical masking

PBR Computing Flow



Simulation or FPGA emulation!

Signal Probability Analysis (SPR)

- Fault prone signals s are in one of four different states
- The possible states and respective probabilities are represented in two 2 by 2 matrices

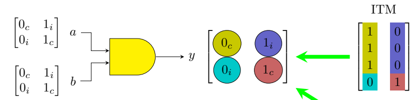
$$s = \begin{bmatrix} 0_c & 1_i \\ 0_i & 1_c \end{bmatrix} \text{ and } P\{s\} = \begin{bmatrix} P\{0_c\} & P\{1_i\} \\ P\{0_i\} & P\{1_c\} \end{bmatrix} \quad (2)$$

$$P\{0_c\} + P\{0_i\} + P\{1_i\} + P\{1_c\} = 1 \quad (3)$$

- The probability matrix $P(s)$ embeds the reliability information

$$R_s = P\{0_c\} + P\{1_c\} \quad (4)$$

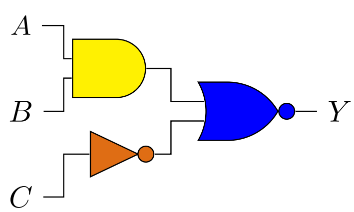
Propagation of Signal Probabilities



$$\begin{bmatrix} 0_c0_c & 0_c1_i & 1_i0_c & 1_i1_i \\ 0_c0_i & 0_c1_c & 1_i0_i & 1_i1_c \\ 0_i0_c & 0_i1_i & 1_c0_c & 1_c1_i \\ 0_i0_i & 0_i1_c & 1_c0_i & 1_c1_c \end{bmatrix} \cdot \begin{bmatrix} q & p \\ q & p \\ q & p \\ p & q \end{bmatrix} = \begin{bmatrix} 0_c0_cq + 0_c1_iq + 1_i0_cq + 1_i1_ip & 0_c0_cp + 0_c1_ip + 1_i0_cp + 1_i1_iq \\ 0_c0_iq + 0_c1_cq + 1_i0_iq + 1_i1_cp & 0_c0_ip + 0_c1_cp + 1_i0_ip + 1_i1_cq \\ 0_i0_cq + 0_i1_iq + 1_c0_cq + 1_c1_ip & 0_i0_cp + 0_i1_ip + 1_c0_cp + 1_c1_iq \\ 0_i0_iq + 0_i1_cq + 1_c0_iq + 1_c1_cp & 0_i0_ip + 0_i1_cp + 1_c0_ip + 1_c1_cq \end{bmatrix}$$

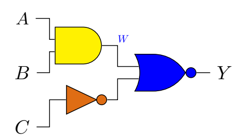
Input signal matrix PTM

Example



$p_{ij} = p$ and $q_{ij} = q$ for all (i, j)

L1: Output of Gate NAND



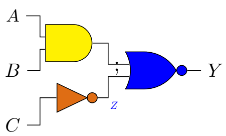
$$AB = \begin{pmatrix} a_{0c}b_{0c} & a_{0c}b_{1i} & a_{1i}b_{0c} & a_{1i}b_{1i} \\ a_{0c}b_{0i} & a_{0c}b_{1c} & a_{1i}b_{0i} & a_{1i}b_{1c} \\ a_{0i}b_{0c} & a_{0i}b_{1i} & a_{1c}b_{0c} & a_{1c}b_{1i} \\ a_{0i}b_{0i} & a_{0i}b_{1c} & a_{1c}b_{0i} & a_{1c}b_{1c} \end{pmatrix}$$

$$PTM = \begin{pmatrix} a_{0c}b_{0c}q + a_{0c}b_{1i}q + a_{1i}b_{0c}q + a_{1i}b_{1i}p & a_{0c}b_{0c}p + a_{0c}b_{1i}p + a_{1i}b_{0c}p + a_{1i}b_{1i}q \\ a_{0c}b_{0i}q + a_{0c}b_{1c}q + a_{1i}b_{0i}q + a_{1i}b_{1c}p & a_{0c}b_{0i}p + a_{0c}b_{1c}p + a_{1i}b_{0i}p + a_{1i}b_{1c}q \\ a_{0i}b_{0c}q + a_{0i}b_{1i}q + a_{1c}b_{0c}q + a_{1c}b_{1i}p & a_{0i}b_{0c}p + a_{0i}b_{1i}p + a_{1c}b_{0c}p + a_{1c}b_{1i}q \\ a_{0i}b_{0i}q + a_{0i}b_{1c}q + a_{1c}b_{0i}q + a_{1c}b_{1c}p & a_{0i}b_{0i}p + a_{0i}b_{1c}p + a_{1c}b_{0i}p + a_{1c}b_{1c}q \end{pmatrix}$$

$$\begin{aligned} w_{0c} &= a_{0c}b_{0c}q + a_{0c}b_{0i}q + a_{0c}b_{1c}q + a_{0c}b_{1i}q + a_{0i}b_{0c}q + a_{0i}b_{1i}q + \\ &\quad a_{1c}b_{0c}q + a_{1c}b_{1i}p + a_{1i}b_{0c}q + a_{1i}b_{0i}q + a_{1i}b_{1c}p + a_{1i}b_{1i}p \\ w_{0i} &= a_{0i}b_{0i}q + a_{0i}b_{1c}q + a_{1c}b_{0i}q + a_{1c}b_{1c}p \\ w_{1c} &= a_{0i}b_{0i}p + a_{0i}b_{1c}p + a_{1c}b_{0i}p + a_{1c}b_{1c}q \\ w_{1i} &= a_{0c}b_{0c}p + a_{0c}b_{0i}p + a_{0c}b_{1c}p + a_{0c}b_{1i}p + a_{0i}b_{0c}p + a_{0i}b_{1i}p + \\ &\quad a_{1c}b_{0c}p + a_{1c}b_{1i}q + a_{1i}b_{0c}p + a_{1i}b_{0i}p + a_{1i}b_{1c}q + a_{1i}b_{1i}q \end{aligned}$$

$$W = \begin{pmatrix} w_{0c} & w_{1i} \\ w_{0i} & w_{1c} \end{pmatrix}$$

L1: Output of Gate NOT



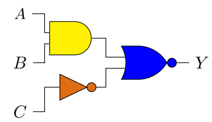
$$C = \begin{pmatrix} c_{0c} & c_{1i} \\ c_{0i} & c_{1c} \end{pmatrix}$$

$$PTM = \begin{pmatrix} c_{0c}p + c_{1i}q & c_{0c}q + c_{1i}p \\ c_{0i}p + c_{1c}q & c_{0i}q + c_{1c}p \end{pmatrix}$$

$$\begin{aligned} z_{0c} &= c_{0i}p + c_{1c}q \\ z_{0i} &= c_{0c}p + c_{1i}q \\ z_{1c} &= c_{0c}q + c_{1i}p \\ z_{1i} &= c_{0i}q + c_{1c}p \end{aligned}$$

$$Z = \begin{pmatrix} c_{0i}p + c_{1c}q & c_{0i}q + c_{1c}p \\ c_{0c}p + c_{1i}q & c_{0c}q + c_{1i}p \end{pmatrix}$$

L2: Output of Gate NOR



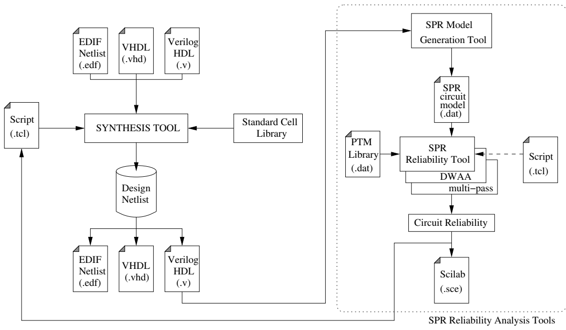
$$WZ = \begin{pmatrix} 3q^2/8 & 3q(-q+1)/8 & 3q(-q+1)/8 & 3(-q+1)^2/8 \\ 3q(-q+1)/8 & 3q^2/8 & 3(-q+1)^2/8 & 3q(-q+1)/8 \\ q(-q+1)/8 & (-q+1)^2/8 & q^2/8 & q(-q+1)/8 \\ (-q+1)^2/8 & q(-q+1)/8 & q(-q+1)/8 & q^2/8 \end{pmatrix}$$

$$PTM = \begin{pmatrix} \frac{3p}{8}q^2 + \frac{3q^2}{4}(-q+1) + \frac{3q}{8}(-q+1)^2 & \frac{3p}{4}q(-q+1) + \frac{3p}{8}(-q+1)^2 + \frac{3q^3}{8} \\ \frac{3p}{8}q(-q+1) + \frac{3q^3}{8} + \frac{3q^2}{8}(-q+1) + \frac{3q}{8}(-q+1)^2 & \frac{3p}{8}q^2 + \frac{3p}{8}q(-q+1) + \frac{3p}{8}(-q+1)^2 + \frac{3q^2}{8}(-q+1) \\ \frac{pq}{8}(-q+1) + \frac{q^3}{8} + \frac{q^2}{8}(-q+1) + \frac{q}{8}(-q+1)^2 & \frac{pq^2}{8} + \frac{pq}{8}(-q+1) + \frac{p}{8}(-q+1)^2 + \frac{q^2}{8}(-q+1) \\ \frac{p}{8}(-q+1)^2 + \frac{q^3}{8} + \frac{q^2}{4}(-q+1) & \frac{pq^2}{8} + \frac{pq}{4}(-q+1) + \frac{q}{8}(-q+1)^2 \end{pmatrix}$$

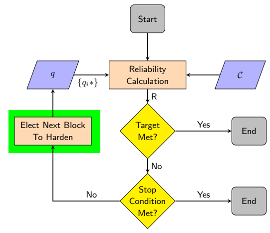
$$\begin{aligned} y_{0c} &= \frac{5q^3}{8} + \frac{3q^2}{4}(-q+1) + q(-q+1)^2 + \frac{1}{8}(-q+1)^3 \\ y_{0i} &= \frac{9q^2}{8}(-q+1) + \frac{3q}{8}(-q+1)^2 \\ y_{1c} &= \frac{3q^3}{8} + \frac{3q}{4}(-q+1)^2 + \frac{3}{8}(-q+1)^3 \\ y_{1i} &= \frac{9q^2}{8}(-q+1) + \frac{7q}{8}(-q+1)^2 + \frac{1}{2}(-q+1)^3 \end{aligned}$$

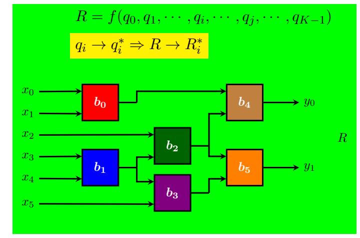
$$Y = \begin{pmatrix} y_{0c} & y_{1i} \\ y_{0i} & y_{1c} \end{pmatrix}$$

SPR Computing Flow



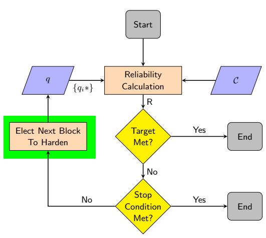
Selective Hardening

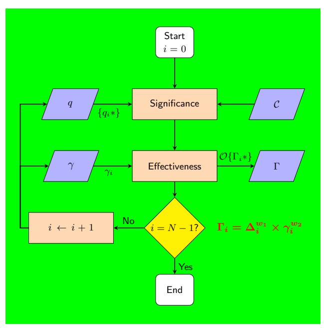




- Sensitivity to reliability change: $\sigma_i = \partial R / \partial q_i$
- Impact of reliability improvement $q_i \rightarrow q_i^*$: $\Delta_i = R_i^* - R$
- Easiness to harden: $\gamma = f(A, P, T)$

Selective Hardening





- New partial ordering is needed after hardening a block

Conclusions

- Reliability issues and challenges
- Need of cost-effective fault tolerant architectures
- Need of efficient assessment approaches

- Main issues of reliability assessment approaches
  - Scalability: accuracy, complexity
  - Design flow integration