Modeling levels

- **Complex architectures decomposed into data and control part**

Abstraction level	control	data
Logic	(binary) logic values	
Register transfer	logic values	words
Instruction	instruction	words
Processor (function)	program	data structures
System	messages	

Modelling of faults

- **Fault: Physical origin of the faulty behavior of the system (fault)**

Design	Specification			• incomplete • inconsistent
	Transformation			• Incorrect
	Realization			• Violation of design rules
Circuit faults	non-permanent	intermittent		• loose contact
		transient		• power spikes • particles hit
	permanent	static manufacturing aging	structure altering	• short
			logic	• component faults • soldering • wires
		dynamic		• Hazard • Slew rate

- ***Error:*** **Faulty state**
- ***Failure*****: Faulty service delivered ()**
- **Criteria of fault modelling**
  - **coherency with UUT modelling**
  - **reality**
  - **mathematical complexity** $\Rightarrow$ e.g. ***Logic fault modelling*****:** ***logic*** **values altered**

Categorization of testing

Criterium	Terminology	Attributes
When is testing performed?	*On-line, concurrent testing*	Simultaneously with the normal operation
	Off-line testing	In separate time
Test evaluation unit source of test inputs	*Self-test*	UUT self
	External testing	External test unit
Fault model	*Verification*	Design fault
	Acceptance test (Burn-in)	Manufacturing fault
	Quality assurance test	Early defects
	Repair	LRU level physical defects
Type of the DUT	*Component test*	IC, PCB
	PCB test	Populated PCB
	System test	System
Order of the tests	*Rigid algorithm*	Independently from the UUT reaction
	Adaptive algorithm	Depending of the UUT reaction
Frequency of the test	*Static testing*	Lower as operational
	AC testing	Normal
	Worst-case testing	Maximal
Evaluation of the response	*Normal*	Entire response
	Compact testing	Compacted response
Measurement points	*Terminal testing*	Functional I/O connector pins, test pins
	In-circuit testing	Bed of nails, measuring probe, ICE,electron ray

- **When is testing performed?**
  - **Concurrent testing: low latency, hardware redundancy**
    e.g. self-checking circuits, memory EDC, master-checker
  - **Off-line testing: greater latency (max. time between two tests), but high fault coverage**
    e.g. memory: off-line exhaustive test; on-line backup of the application data
  - **Frequently a compromise: Quasi-concurrent testing**
- **Evaluation / generation of the test**
  - **Self-test: utilization of the built-in intelligence**
  - **Which one has priority, evaluation or test generation?: fault masking!**

Generation of the test stimuli and reference

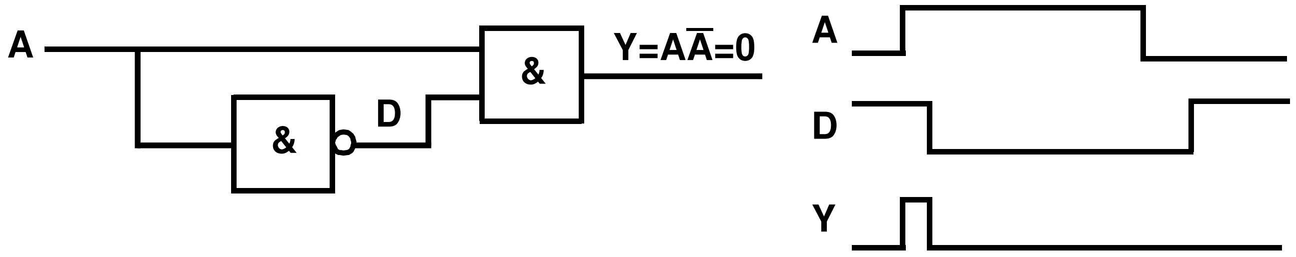
- **Estimation of the test stimuli ("theoretical" problem)**
  - ***Algorithmic -*** **Algorithm (Network model, fault model) ⇒ test stimuli**
  - ***Exhaustive*** **(e.g. combinatorial network: all input patterns)**
  - ***Pseudoexhaustiv*****e (all "k out-of n" combinations)**
  - ***Statistic*** **(random/ pseudorandom)**
- **Generation of the test patterns ("technical" problem)**
  - ***Stored, pre-calculated patterns***
  - ***Stored algorithm*** **(test program in an ATE)**
  - ***On-line HW pattern generator*** **for simple algorithmic patterns (e.g. memory test)**
- **Estimation of the reference**
  - ***In time***
    - **Simultaneously (on-line)**
    - **Stored**
  - ***Source of reference***
    - **Model based ("Software")**
    - **Hardware based**
- **Evaluation of the response**
  - ***fault detection: comparison with the reference***
  - ***fault diagnosis:*** **comparison with all models included into the fault model (adaptive)**

Fault models

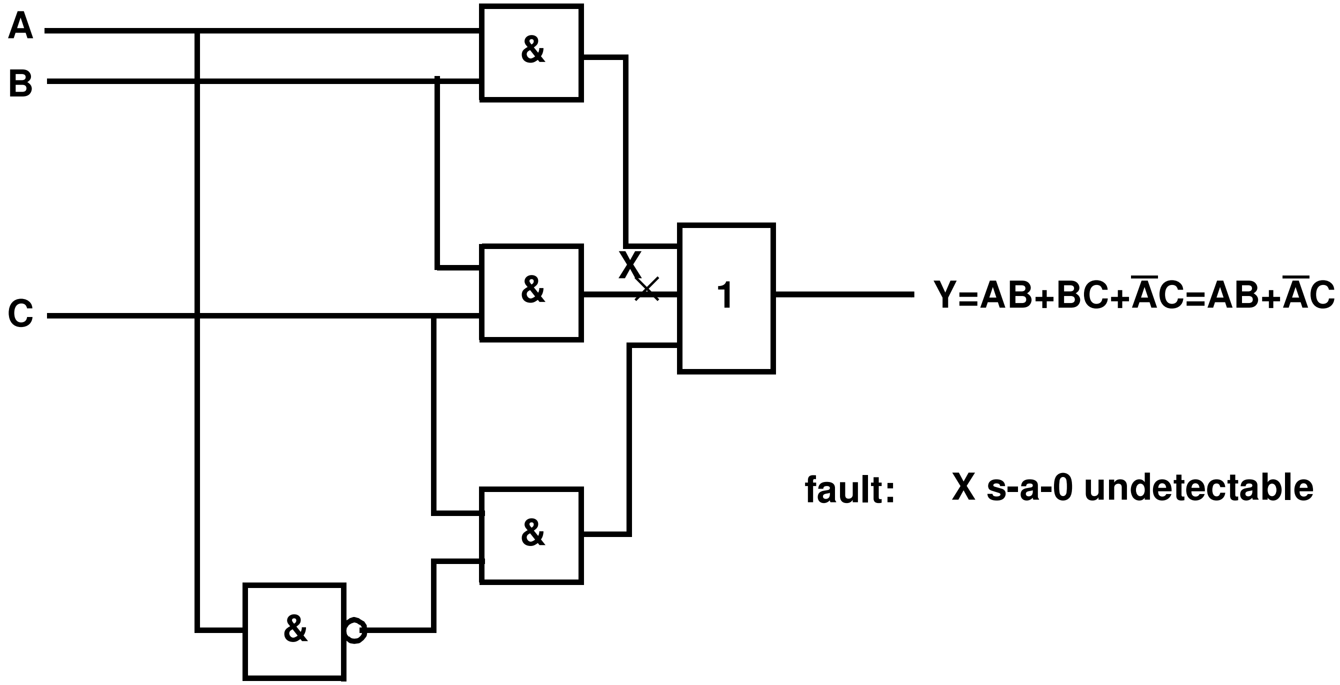
- **Definition of the set of anticipated faults by**
  - **Enumeration:** ***explicit***
  - **Attributes:** ***implicit***
- **Description of the UUT by**
  - **Structure**
  - **Function**
- **Fault detection**
  - **Observability of the faulty behavior on the output ⇔ detectability of the fault**
    - **e.g. combinatorial circuit:** $f(\underline{x}) \neq f_{fault}(\underline{x})$
    - **single output:** $f(\underline{x}) \oplus f_{fault}(\underline{x}) = 1$
    - **by redundancy**

Redundancy and undetectable faults

- If a fault has no effect on output ⇒ redundancy in the circuit
- E.g. undetectable fault of the type s-a-0: a constant zero can be applied at the fault location
- Frequently: intentional redundancy
  - fault masking: e.g modular redundancy (TMR)
  - functional redundancy, but dynamic function: signal delay

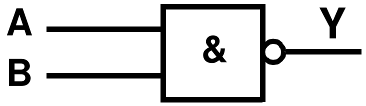


- **Hazards**



- **The detectability of a fault is not a portable attribute, if multiple faults can appear**

Fault equivalence



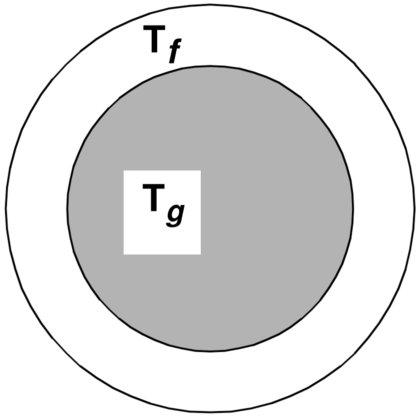
- f and g are equivalent faults, if $Y_f(t)=Y_g(t)$ for all test input sequences
- Failure matrix of a NAND gate

A	B	Y_{normal}	f_1	f_2	f_3	f_4	f_5	f_6
			$Y_{A\ s\text{-}a\text{-}0}$	$Y_{A\ s\text{-}a\text{-}1}$	$Y_{B\ s\text{-}a\text{-}0}$	$Y_{B\ s\text{-}a\text{-}1}$	$Y_{Y\ s\text{-}a\text{-}0}$	$Y_{Y\ s\text{-}a\text{-}1}$
0	0	1	1	1	1	1	0	1
0	1	1	1	0	1	1	0	1
1	0	1	1	1	1	0	0	1
1	1	0	1	0	1	0	0	1

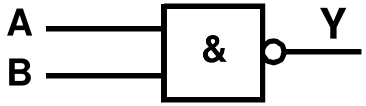
f_1, f_3 and f_6 are equivalent $\Rightarrow$ aggregation into a single equivalence class

Fault dominance

- ***f dominates g,*** **if for all test of *g* they are functionally equivalent**



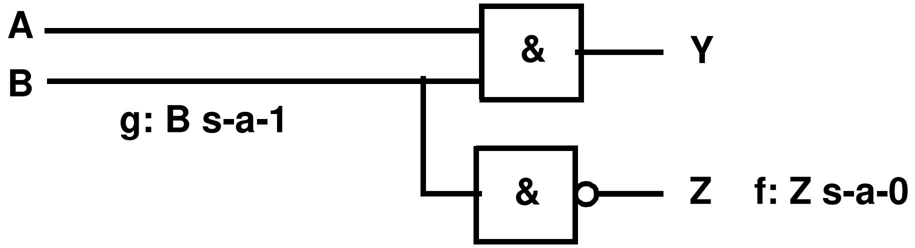
- **Example:**



: f_5 dominates f_2

A	B	Y_{normal}	f_1	f_2	f_3	f_4	f_5	f_6
			$Y_{A\ s\text{-}a\text{-}0}$	$Y_{A\ s\text{-}a\text{-}1}$	$Y_{B\ s\text{-}a\text{-}0}$	$Y_{B\ s\text{-}a\text{-}1}$	$Y_{Y\ s\text{-}a\text{-}0}$	$Y_{Y\ s\text{-}a\text{-}1}$
0	0	1	1	1	1	1	0	1
0	1	1	1	0	1	1	0	1
1	0	1	1	1	1	0	0	1
1	1	0	1	0	1	0	0	1

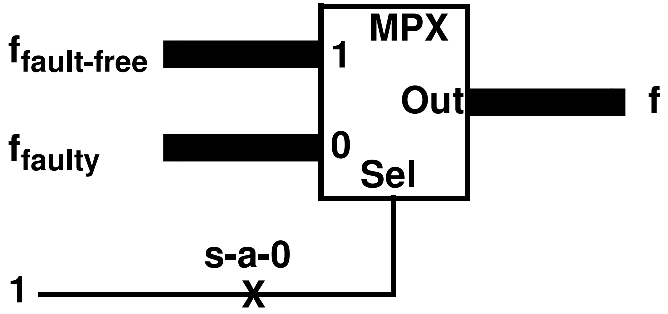
$T_g \subseteq T_f$ can be still valid without any dominance



T_g={A=1, B=0}

Single stuck-at faults

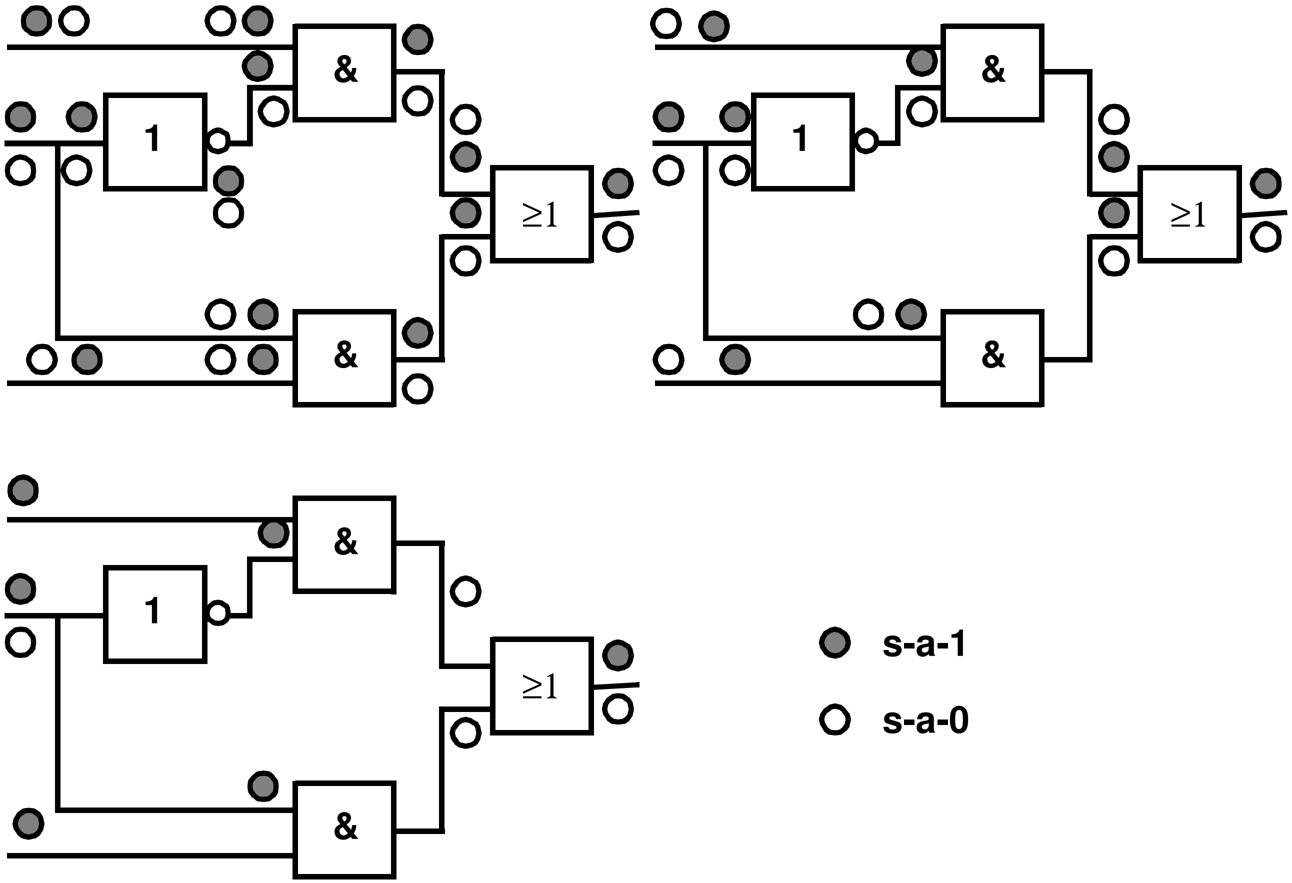
- **Validity**
  - non-universal,
  - many physical faults described correctly
  - independent of the technology
  - experience: many non-classic faults detected too
  - relative moderate number of faults, reduced further by ***fault-collapsing***
  - capable for the modelling of other faults as well



Number of the faults:

- **All outputs and primary inputs of a fanout f_i increases the number of faults of the models by k_i, where**
  $k_i=1$ if $f_i=1$
  $k_i=1+f_i$ if $f_i>1$
  - **Total number of the faults:** $n = \sum_i k_i$
  - **Is q_i defined as $q_i=1$ if $f_i=1$ and $q_i=0$ if $f_i>1$, then** $n = \sum_i (\quad)$
  - **Average number of fanout:** $f = \frac{\sum_i f_i}{G + I}$ **, where G is the number of gates, I: number of the inputs**
  - **Proportion of the fanout-free gates:** $q = \frac{\sum_i q_i}{G + I}$
  - **Accordingly** $n = (\quad) \times (\quad)$
- **Reduction by *fault collapsing*? decision on functional equivalency is NP-complete**
- **Structural equivalency**

Local analysis



Effectivity of the elimination by fault-collapsing

- **At the beginning: 2n faults**
- **If a gate of g_j inputs eliminated, then m_j faults eliminated, where**
  $m_j=g_j$ if $g_j>1$
  $m_j=2$ if $g_j=1$
- **Total number of the faults eliminated:** $m = \sum_i m_i$
- **Is p_i defined as $p_i=1$ if $g_i=1$ and $p_i=0$if $g_i>1$ then** $m = \sum_i (\quad)$
- **Average number of the inputs:** $g = \frac{\sum_i g_i}{G}$
- **Proportion of the single input gates:** $p = \frac{\sum_i p_i}{G}$ **, then** $m = G \times (\quad)$
- **In the practice: ~ 50%**

- **Sufficiency of a test sequence: If an input sequence detects all single s-a-x faults at primary inputs and fanout nodes**
  then ALL single stuck-at faults are detected as well