Dependability and Fault Tolerance
Zuverlässigkeit und Fehlertoleranz

Chapter 7: Repair Technologies

H. T. Vierhaus

Winter Semester 2018/ 2019

Repair Chips- what for ??

- **Produce a higher production yield:**

  Regular structures on ICs kann be manufactured with some redundant elements. In case of defects, redundant elements are activated, usíng, for example, laser-based cutting of lines.

  Application: Memories with spare lines / columns.

- **Repair of circuits / systems after failures „in the field“.**

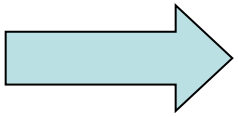
  Repair technologies can enhance long-term dependability significantly.

  Application: High-reliability computer systems in banks !

Can Silicon-Chips be „repaired“?

- Cut metal lines by lasers
  deposit (metal) on a surface by using fast ions to eject particles of it from a target.
  coat (a surface
- Selective sputtering of metal patches by ion beams
  (in specific vacum reactor vessels)
  ... only on very small areas, which are not covered by other materials.

- Active components (e. g. transistors with gate shorts) are not „repairable“ !

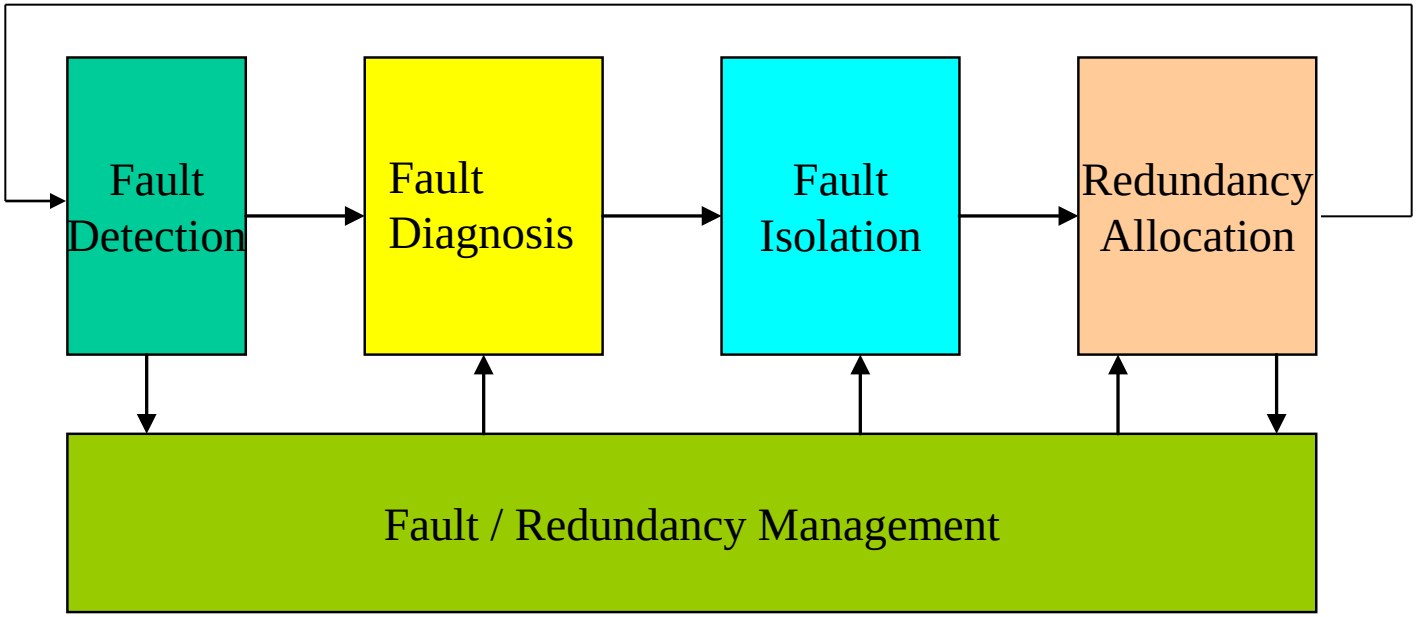


Repair „in the field“ cannot use a physical repair process, but can only activate spare devices by „normal“ switching functions !

Why Repair Technologies ?

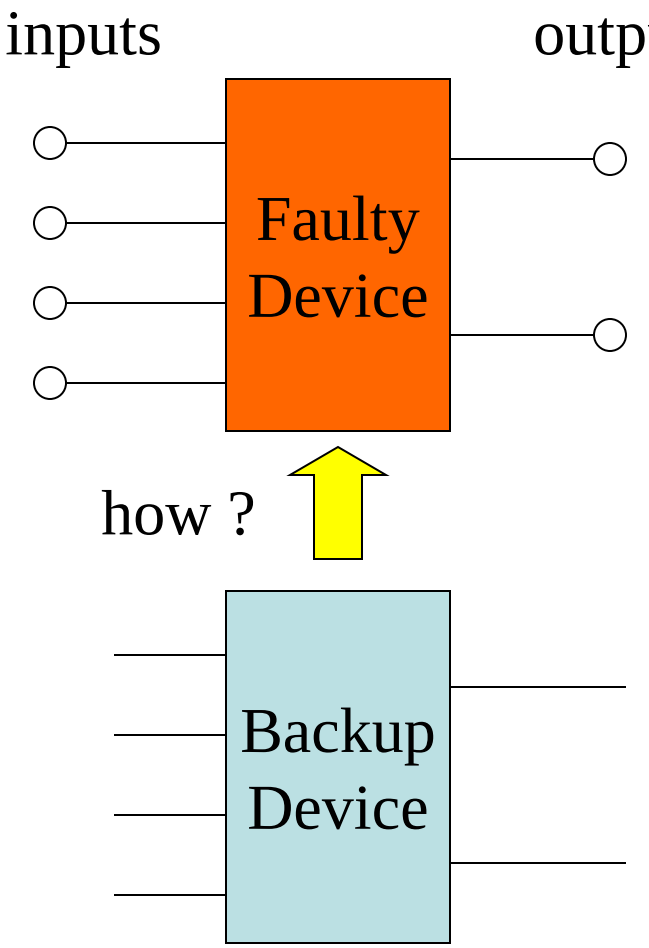
- Compensation for Early-Life-Failures
- Selective compensation of failures by (e. g. from early wear-out):
  - -- Metal Migration
  - -- Hot-Carrier-Injection (HCI)
  - -- Negative Bias Thermal Instability (NBTI)
  - -- Gate-Oxide und Field-Oxide-Shorts.

Repair Process

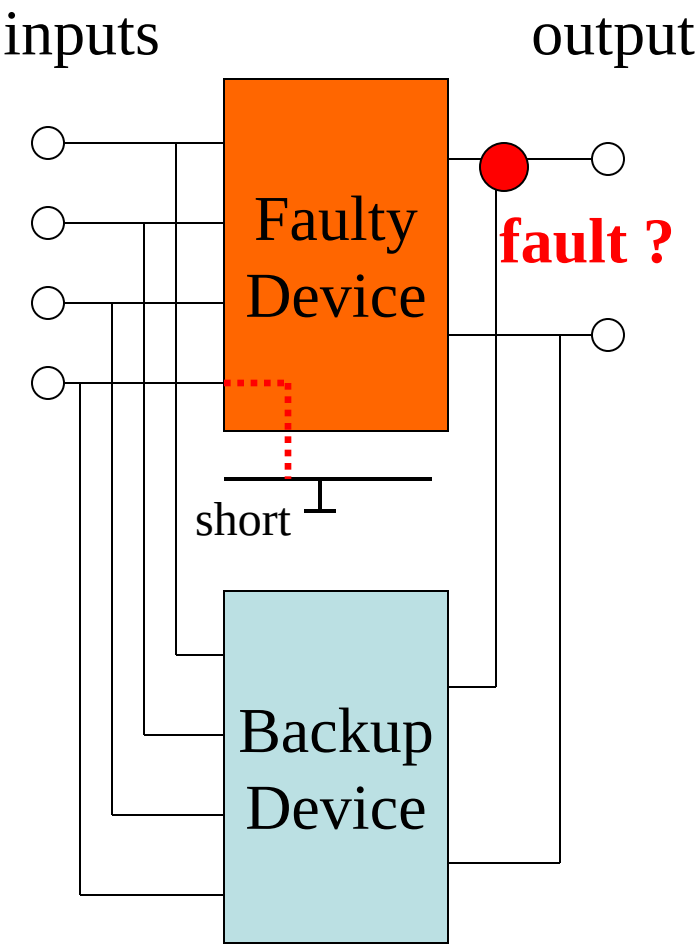


Typically, repair procedures are relatively complex and lengthy in time. They may last for many milli-seconds up to seconds or even minutes. To be executed off-line, while the serviced function is not needed.
Error detection on-line and error correction have to be done by one of the methods discussed in earlier chapters.

Redundancy



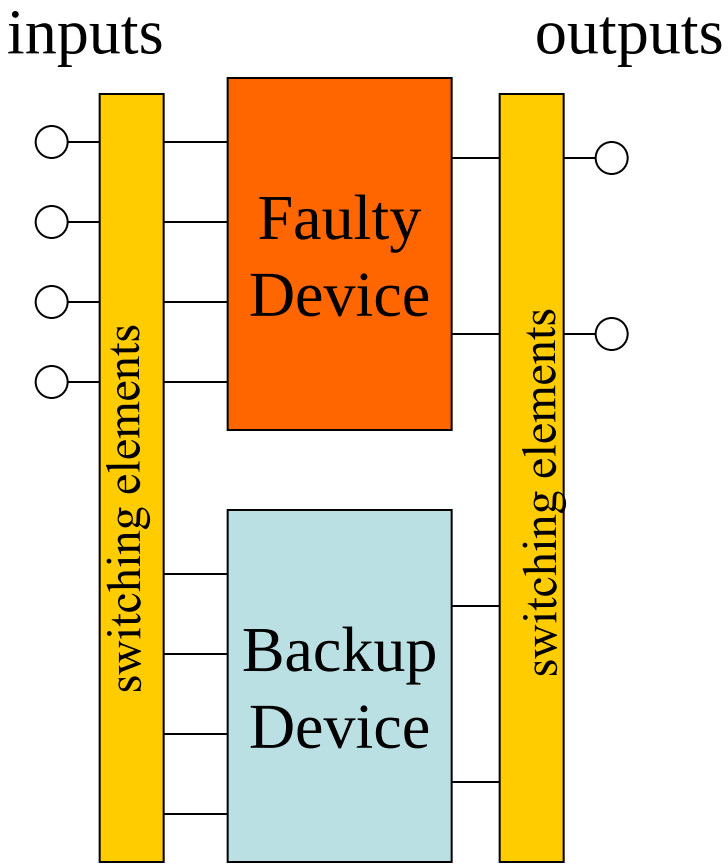
Active Redundancy



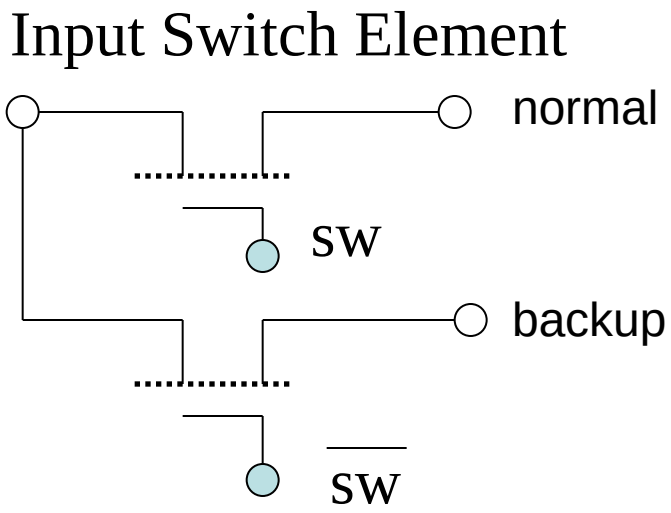
Problem:

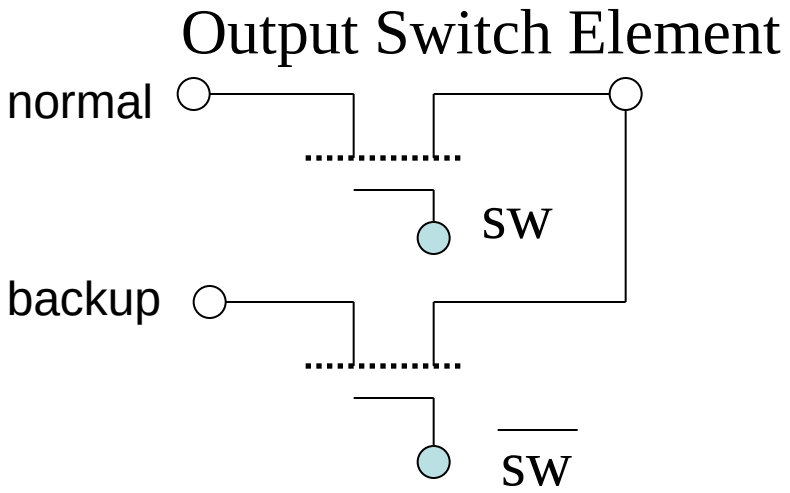
Typically we need not only the activation of the backup device, but also the **isolation of the faulty unit** !

Fault Isolation

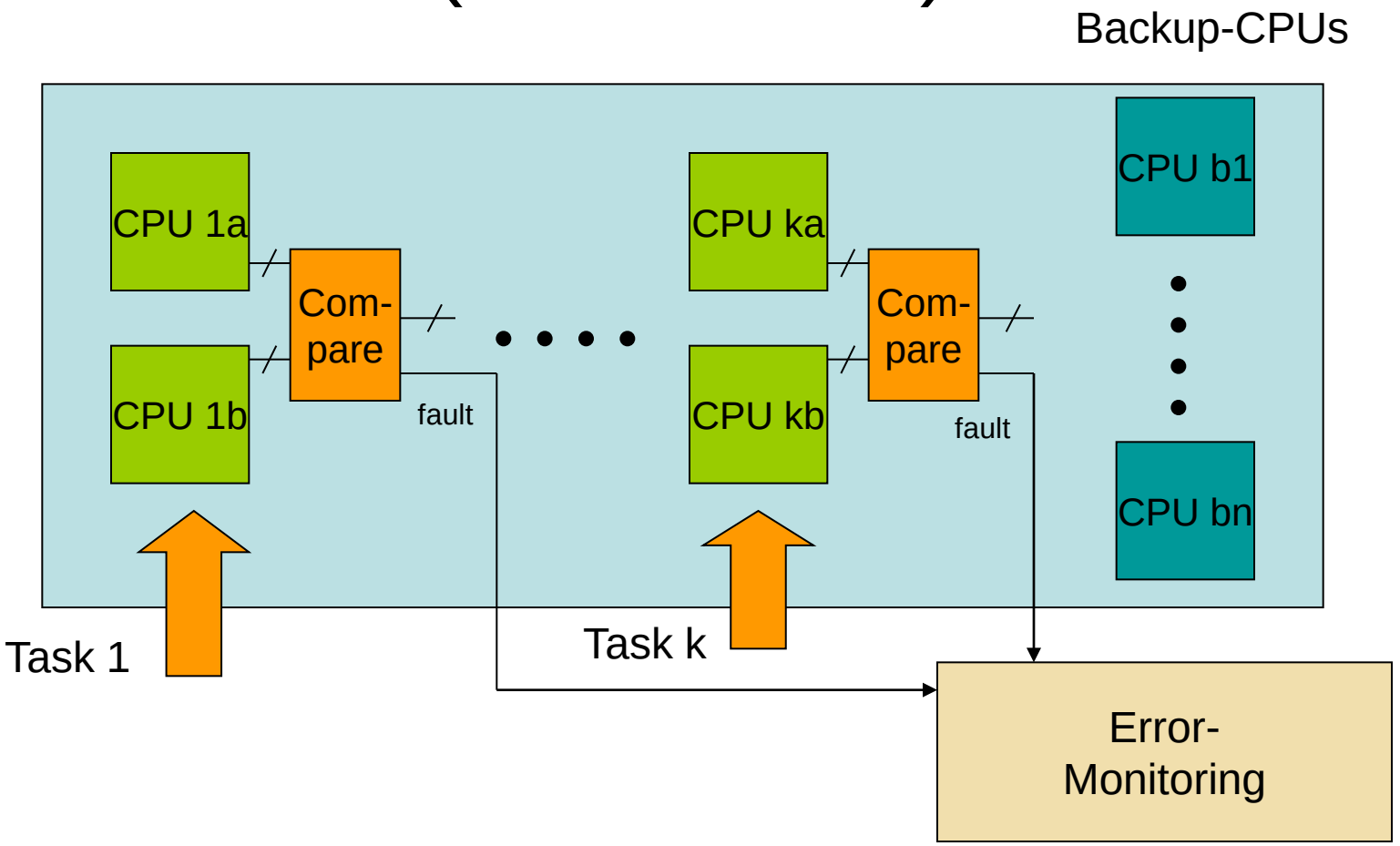


..needs switching elements !!

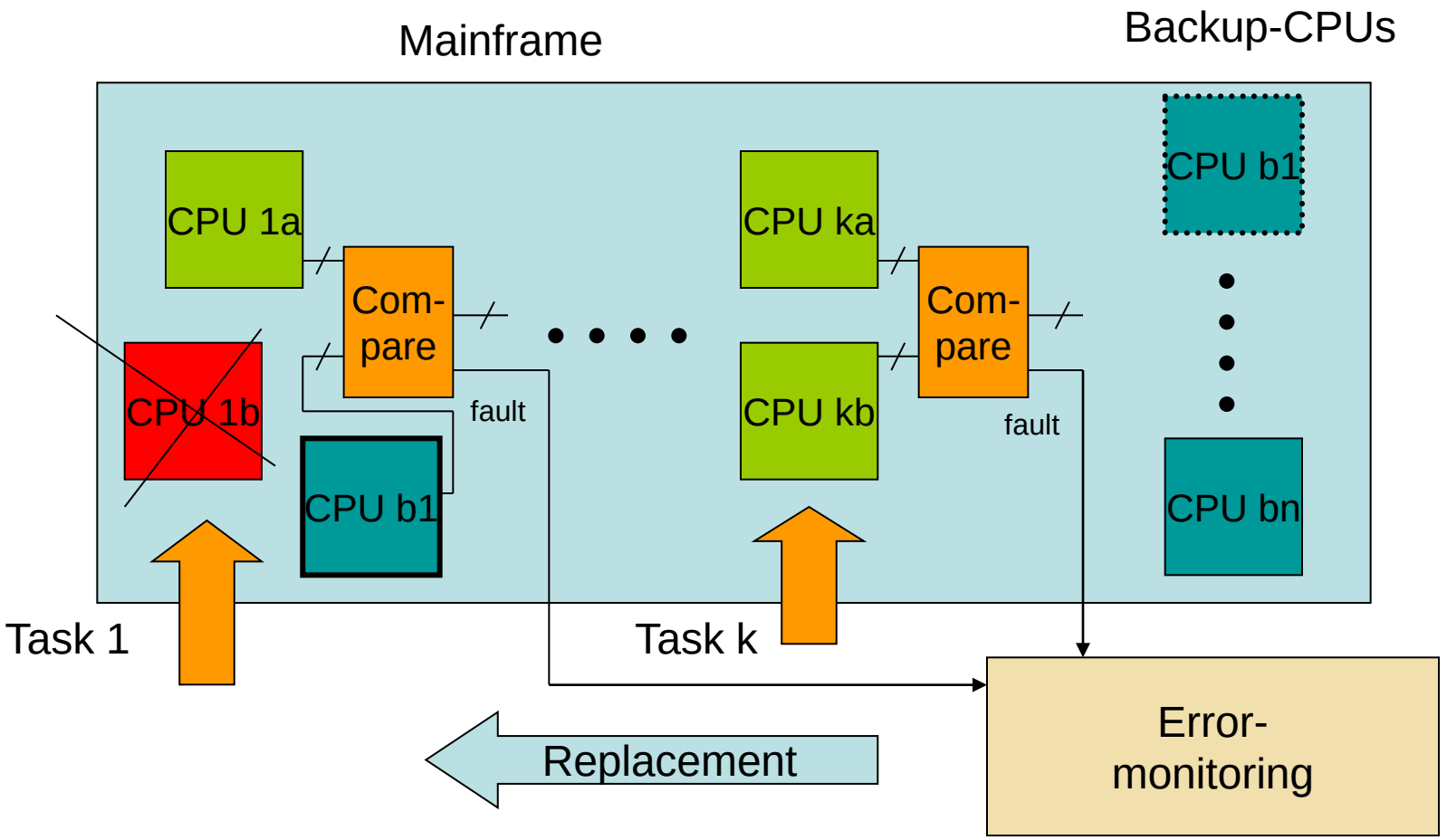




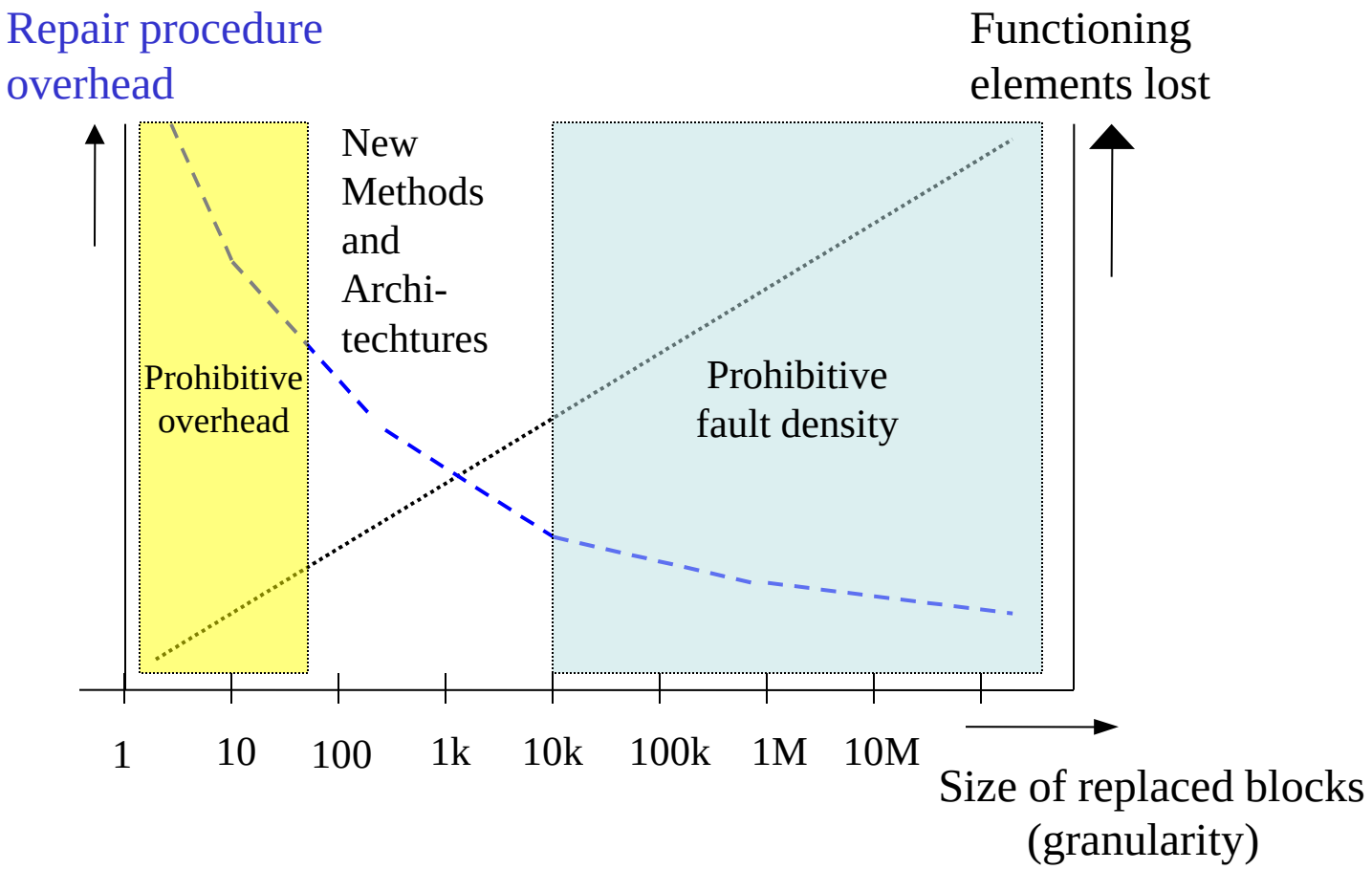
Large-Scale Self Repair in Computers (Mainframes)



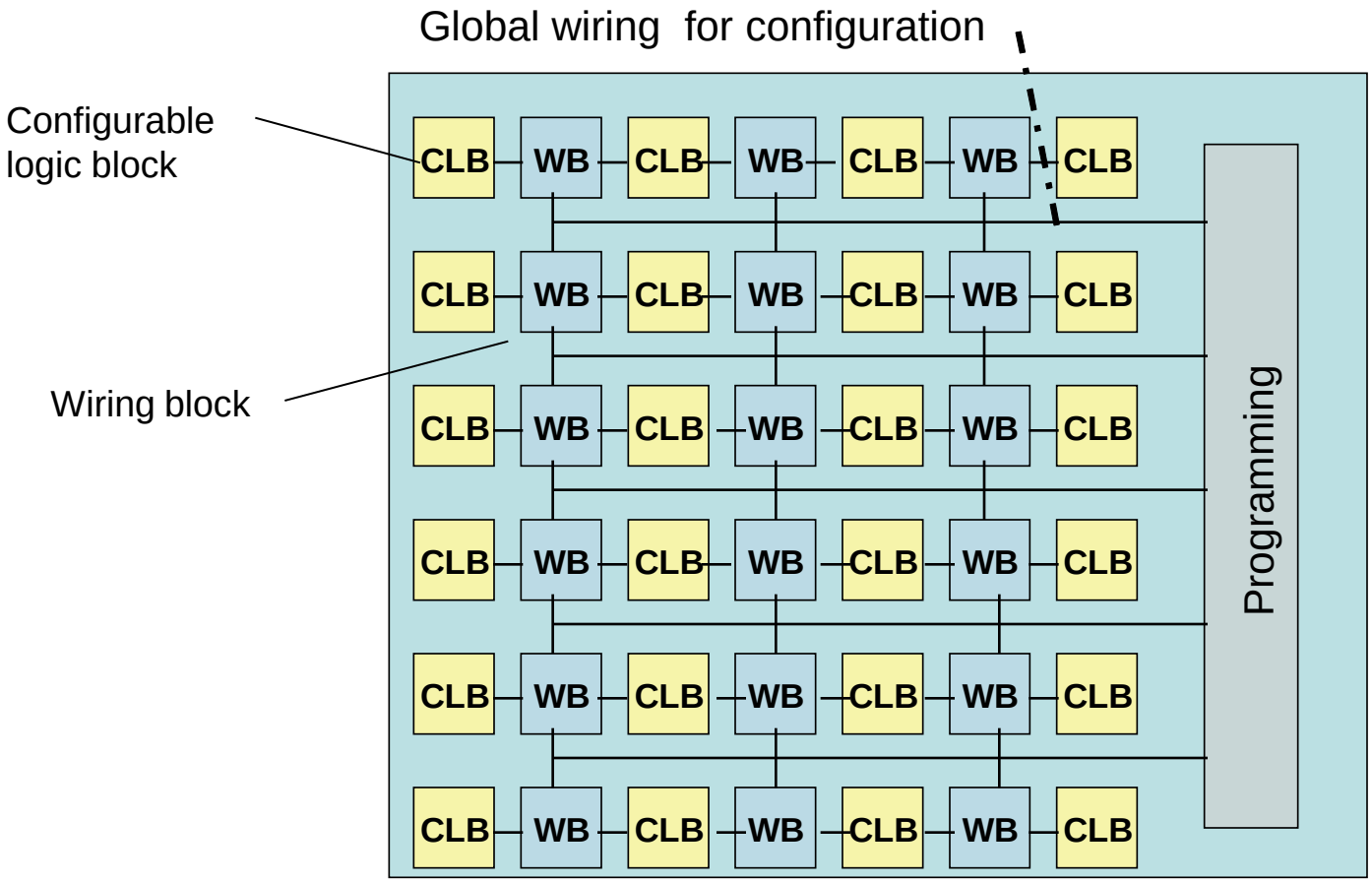
CPU-Replacement



Granularity of Repair



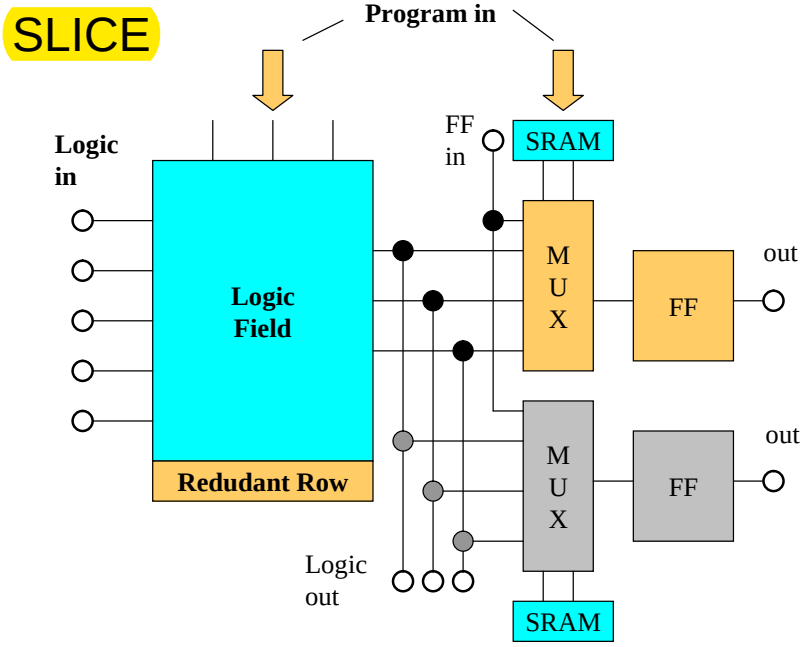
FPGA-Structure

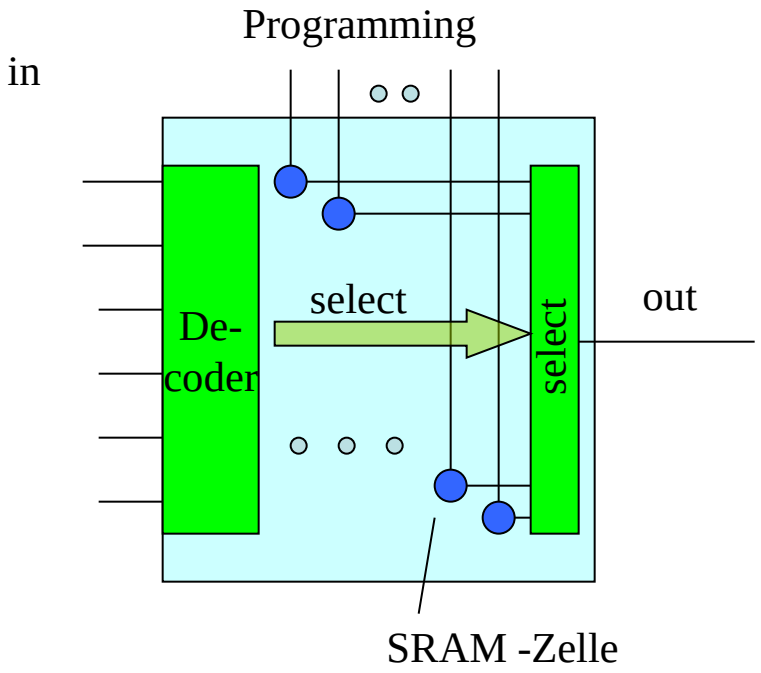


Structure of FPGAs

- Configurable Logic-Blocks (CLBs) with
  - Lookup-Tables
  - Flip-Flops
  - Multiplexers

Typically one Lookup-Table with Muxers and Flip-Flops makes a „Slice“.

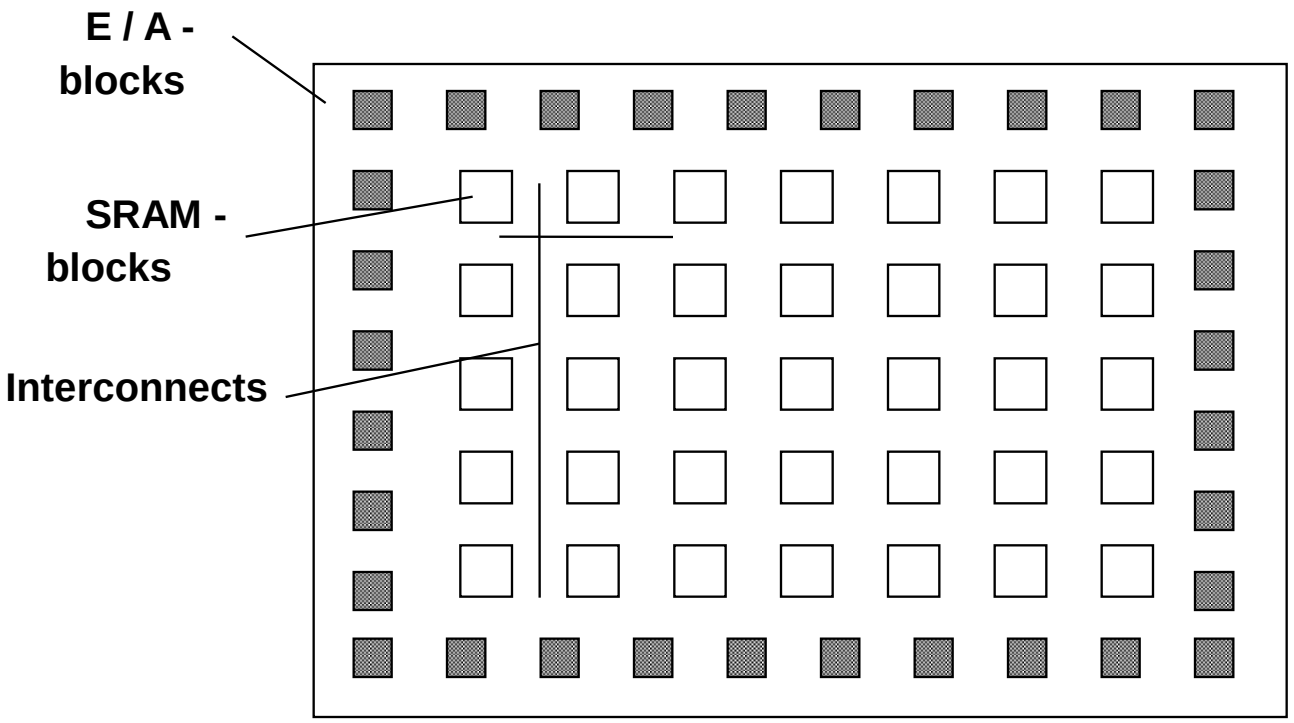




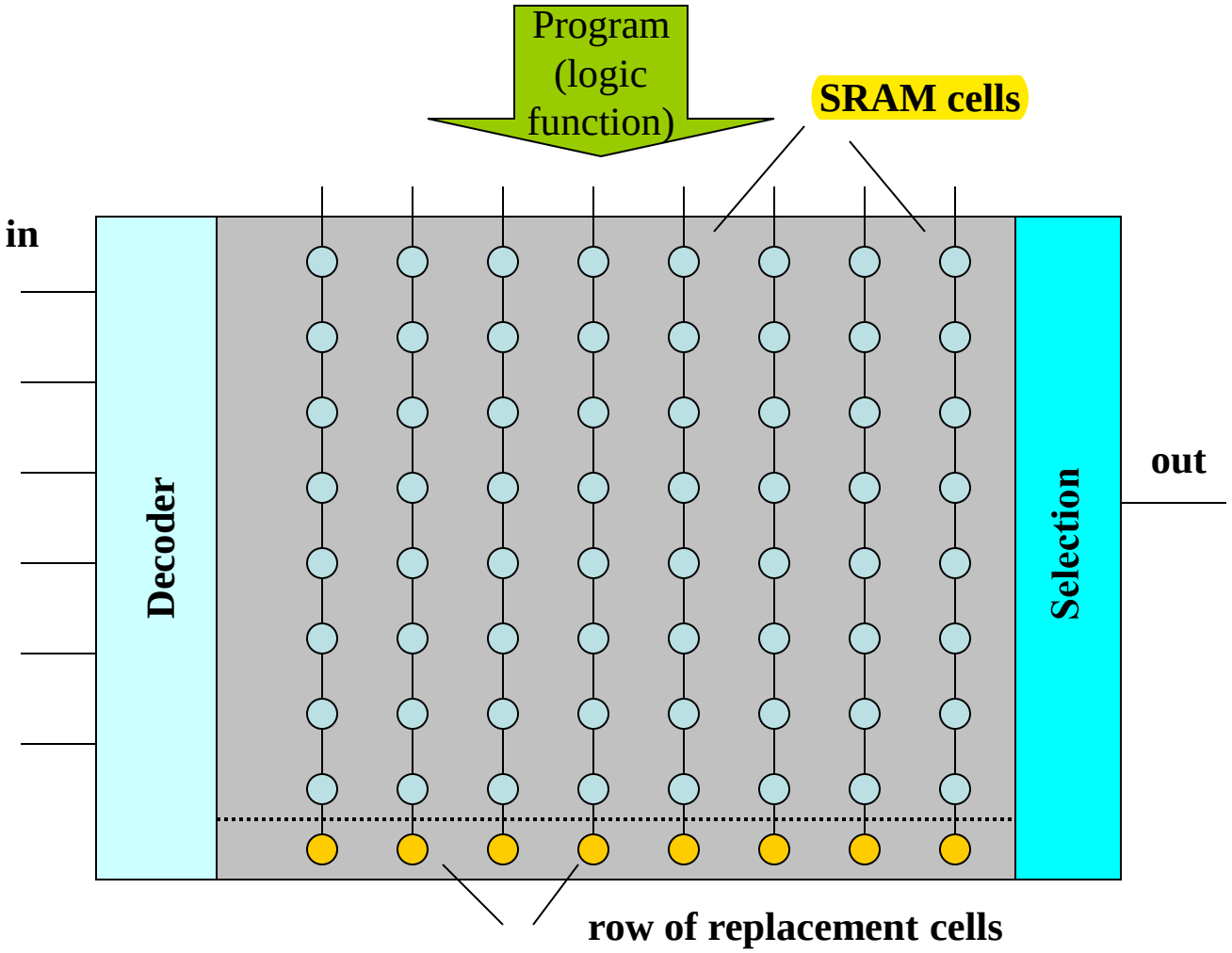
- Configrable nodes for interconnects

Mostly 2 slices per CLB.

Struktures of FPGAs (2)



Lookup-Table with Spare Cells



Re-Programming of FPGAs

the combination of ideas to form a theory or system

- Configuration data for FPGAs are computed off-line in a complex synthesis process and stored in a non-volatile memory device (EEPROM, flash).

  Electrically Erasable Programmable Read-Only Memory

- An FPGA can be re-programmed „in the field“ from such a memory, in total or in parts.

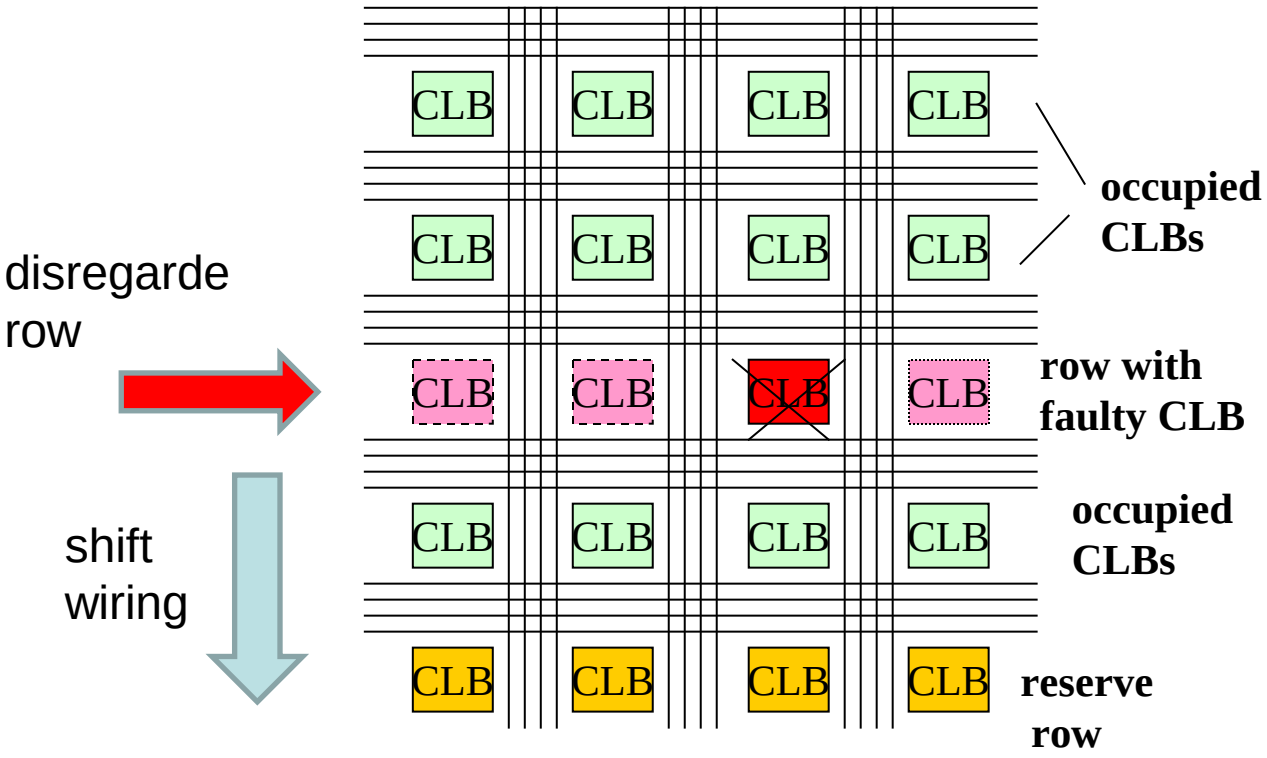
- If an FPGAs need to be adapted to changing tasks, there must be pre-computed sets of (re-)configuration data, typically 2-5 sets.

- It is even possible that parts of an FPGA act as a processor and control the re-configuration og other parts. Alternatively, many FPGAs have „hard“ processor cores on-chip.

- Memory cells (RAM or EEPROM) are needed to set the configuration of an FPGA. These memory cells are volatile by themselves !

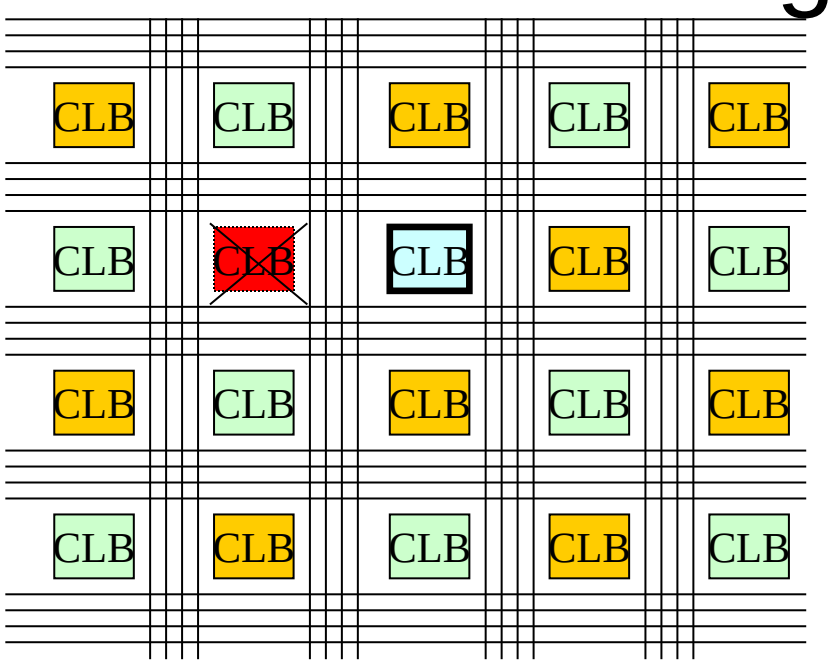
Regular FPGA Re-Structuring for Repair

A permanent fault in one of the CLBs.
Re-structuring the FPGA around the faulty CLB is necessary



Regular re-structuring by using an extra row of CLBs is a regular solution (as in memories), but may waste a number of useful CLBs.

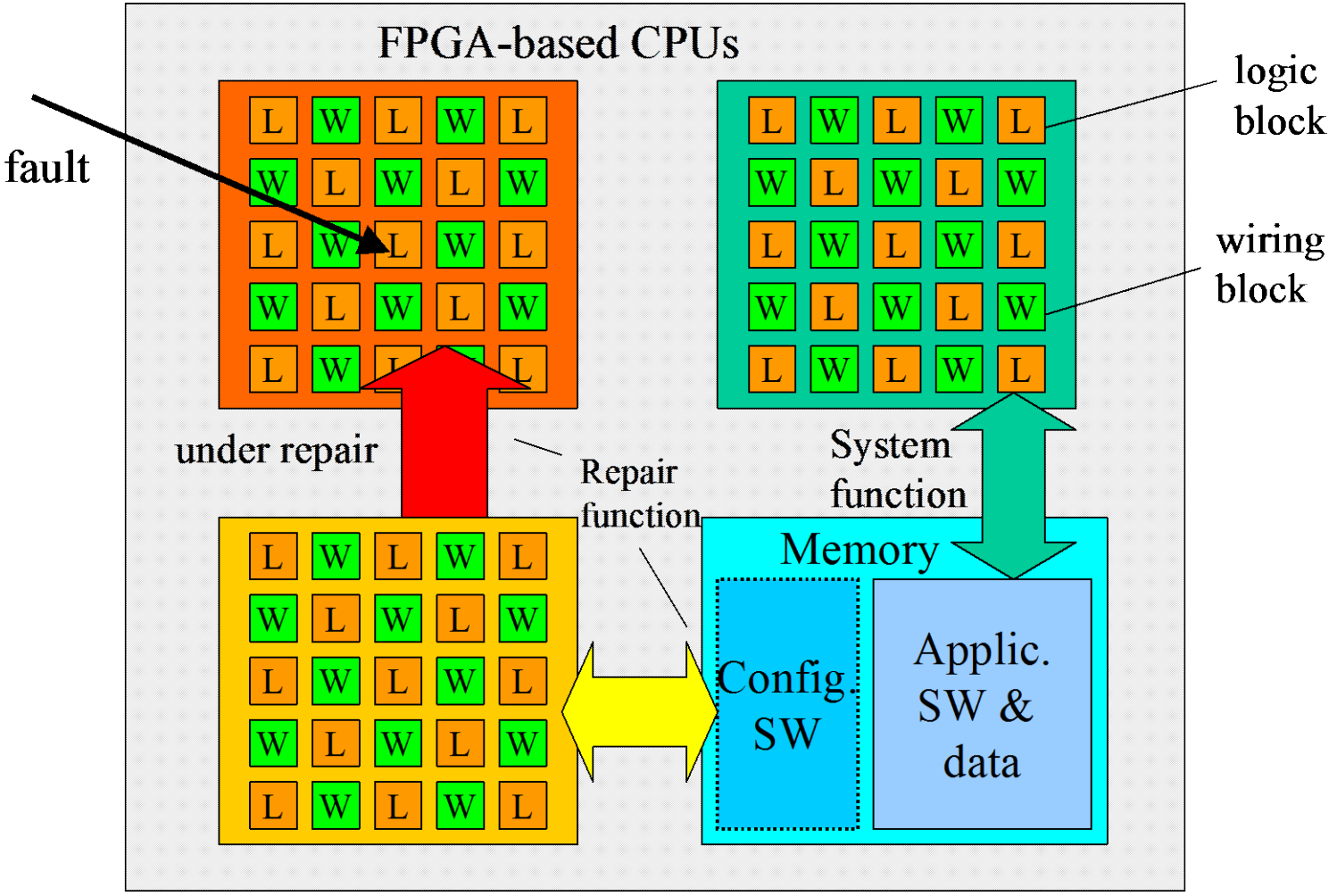
Irregular FPGA Re-Configuration



Saves on discarded functional CLBs, but requires an irregular re-synthesis with modifications on local and global wiring !

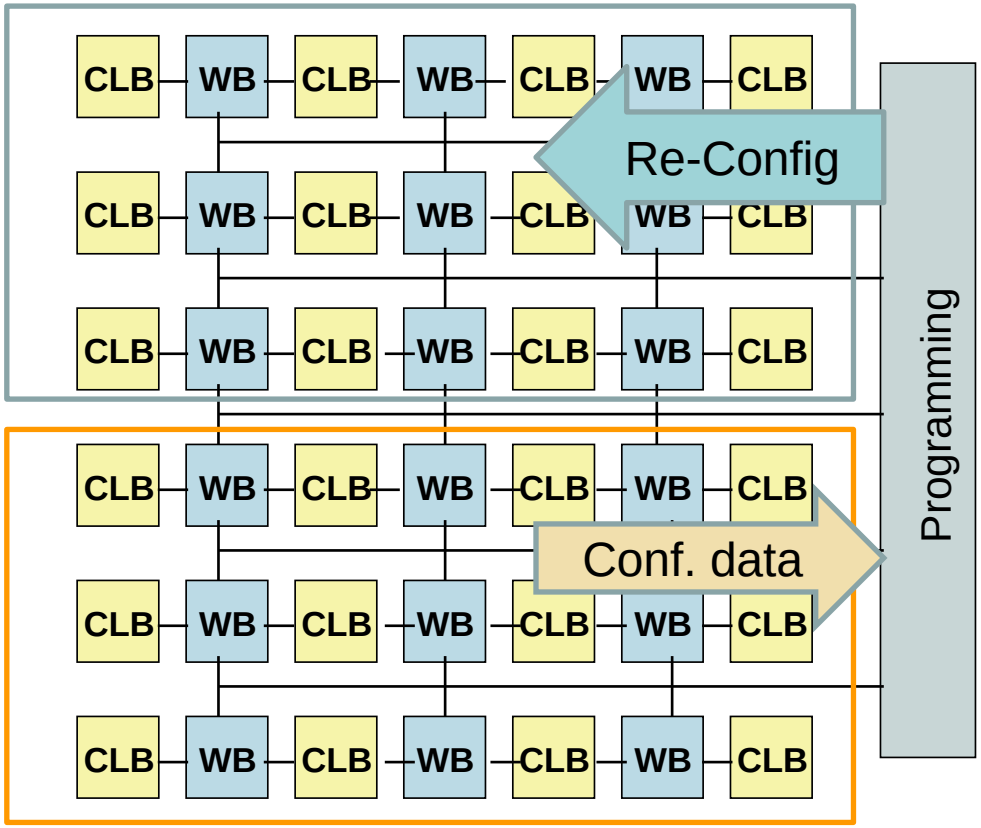
the combination of components or elements to form a connected whole.

Self Repair Function on FPGAs



FPGA with „Soft“-Processor

FPGA-section to be re-configured



FPGA-section working as a processor

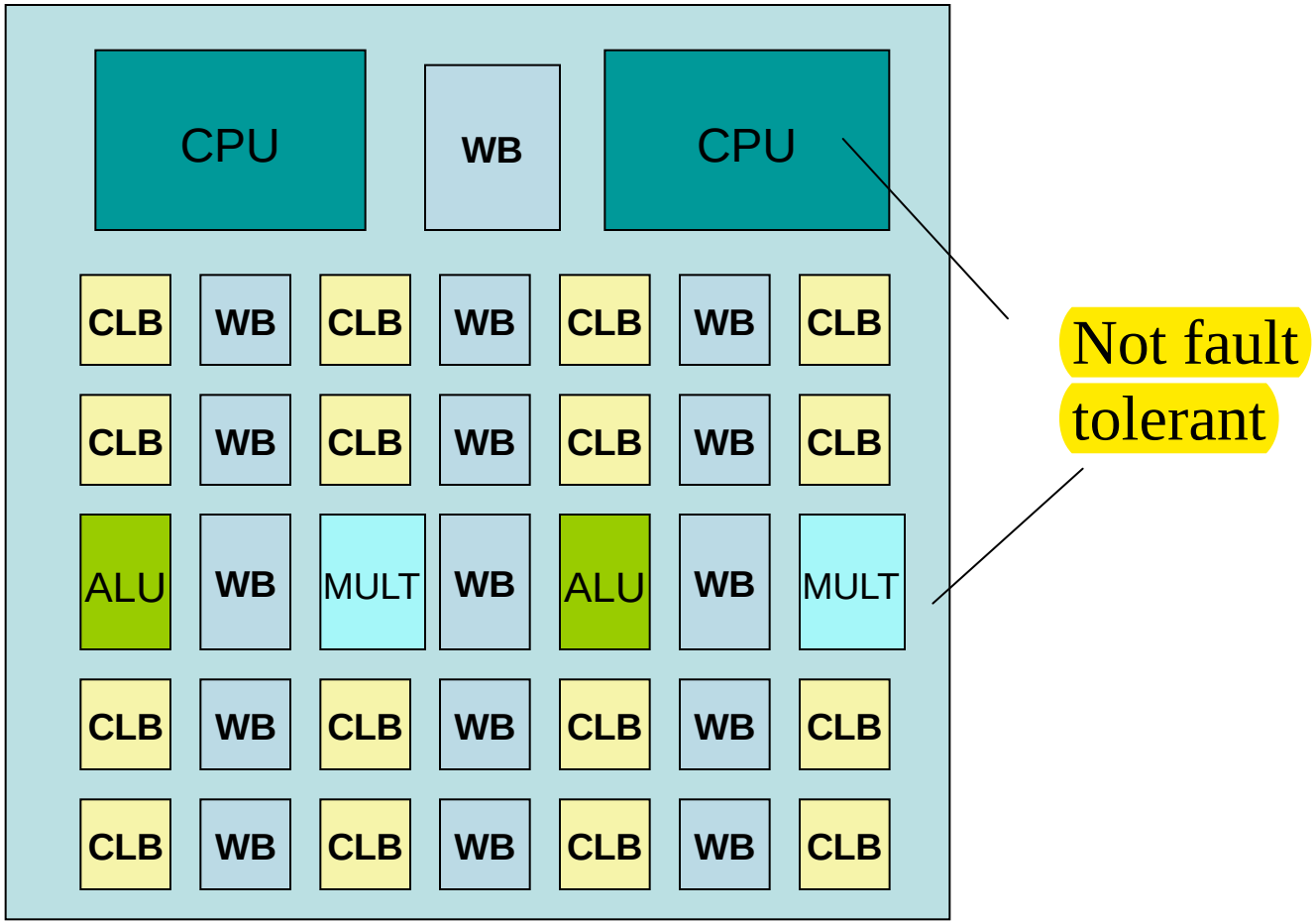
Extended FPGAs

- Prozessor-Cores (16, 32 Bit, RISC) in optimized CMOS-Logic
- Arithmetic Units (ALUs, Multipliers) in CMOS-Logic

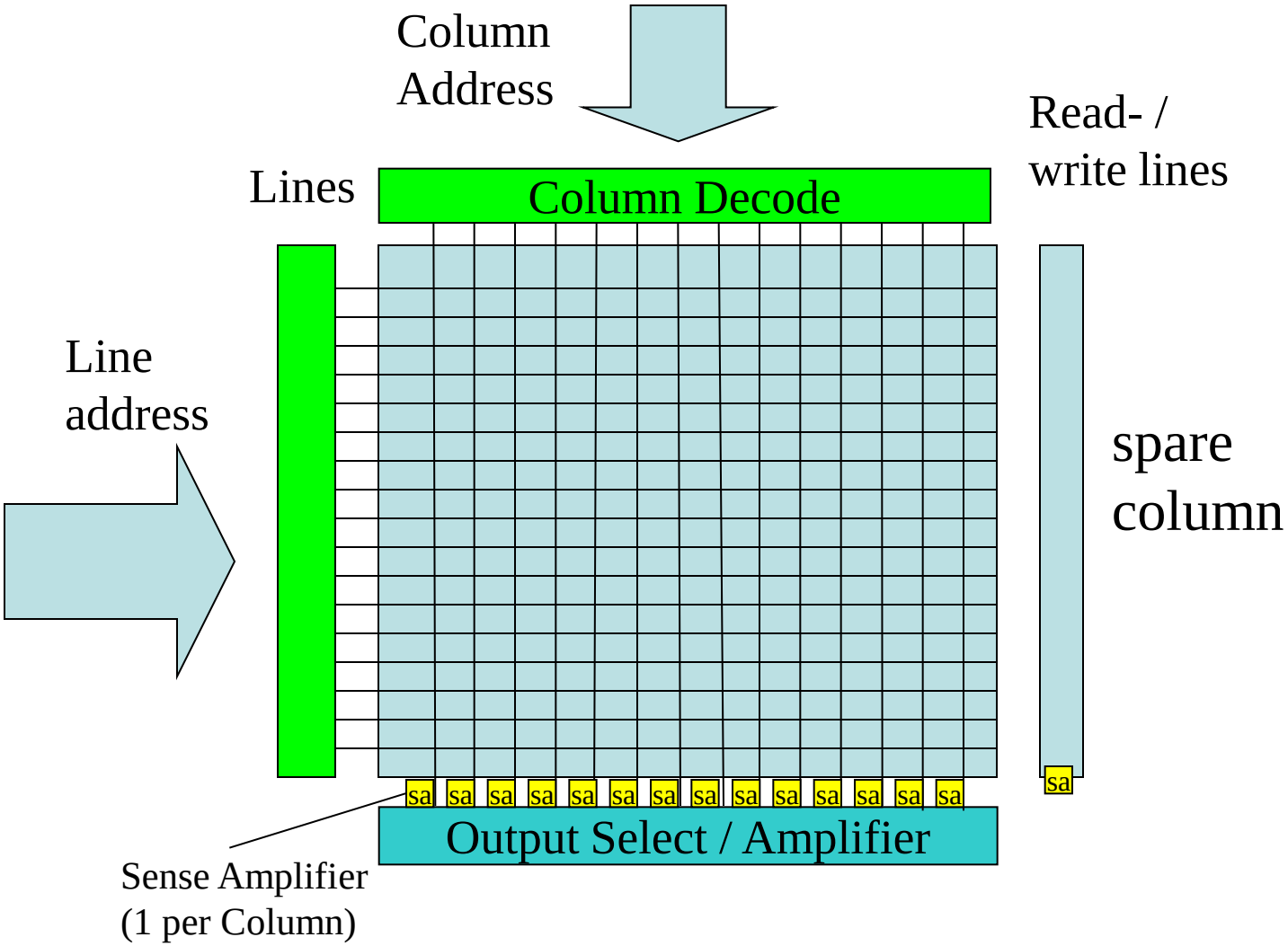
⇒ Single- and multiple processor units on FPGAs get much more powerful !

But: Few if any FPGAs in newest technologies that are specifically „radiation hard“ !

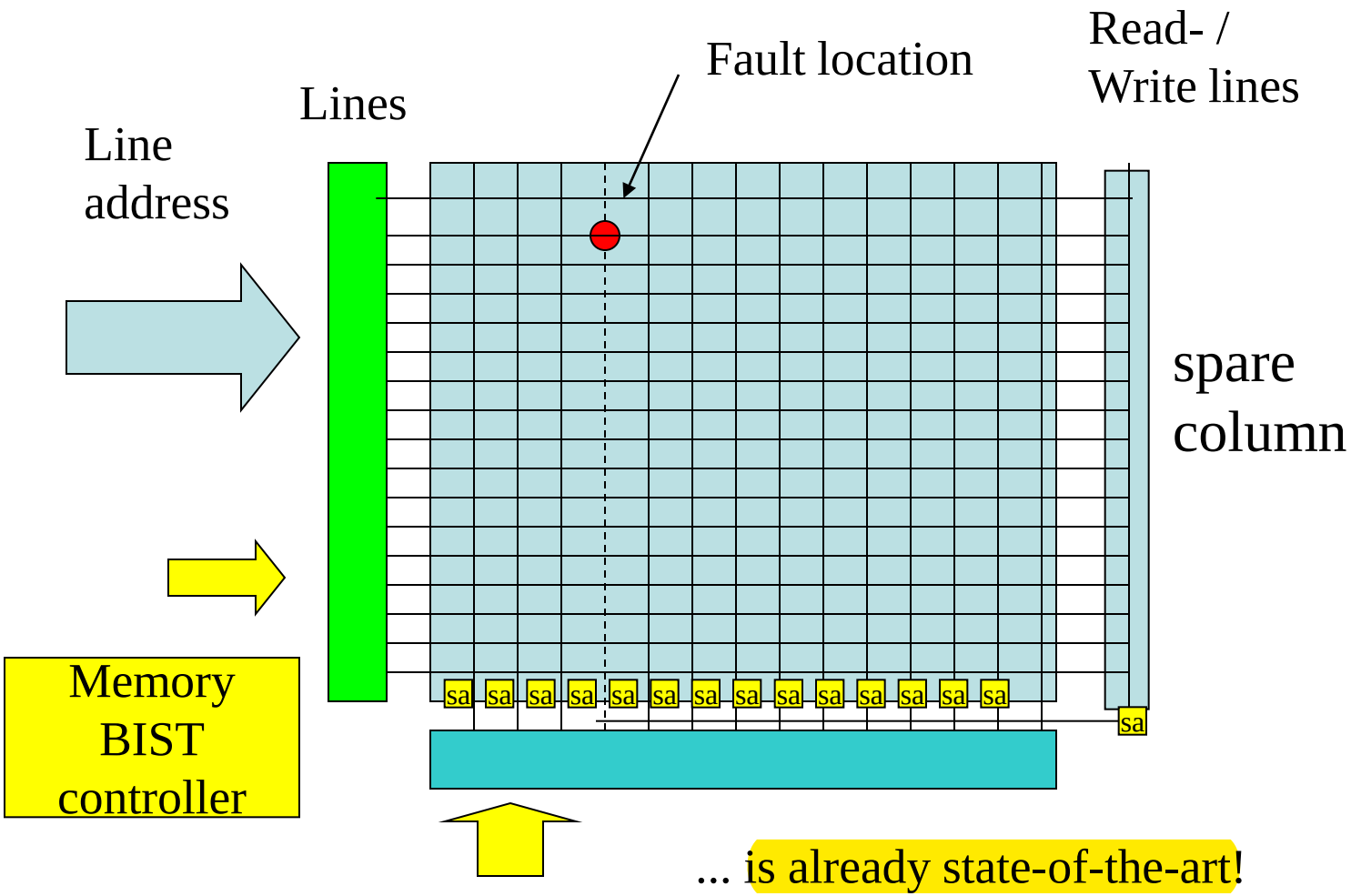
FPGA with CPUs and Functional Blocks



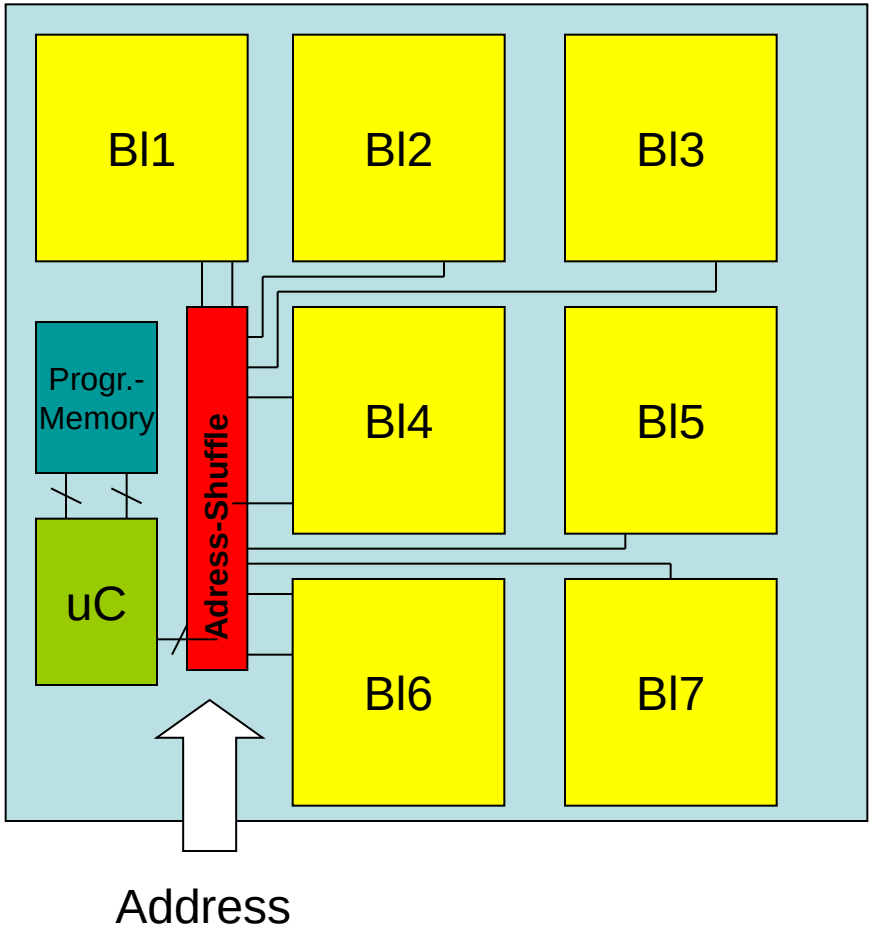
Memory Structure with Redundancy



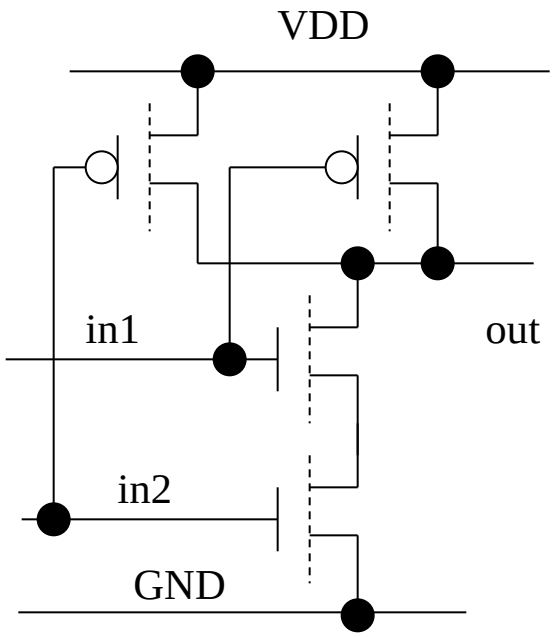
Repair by Spare Column



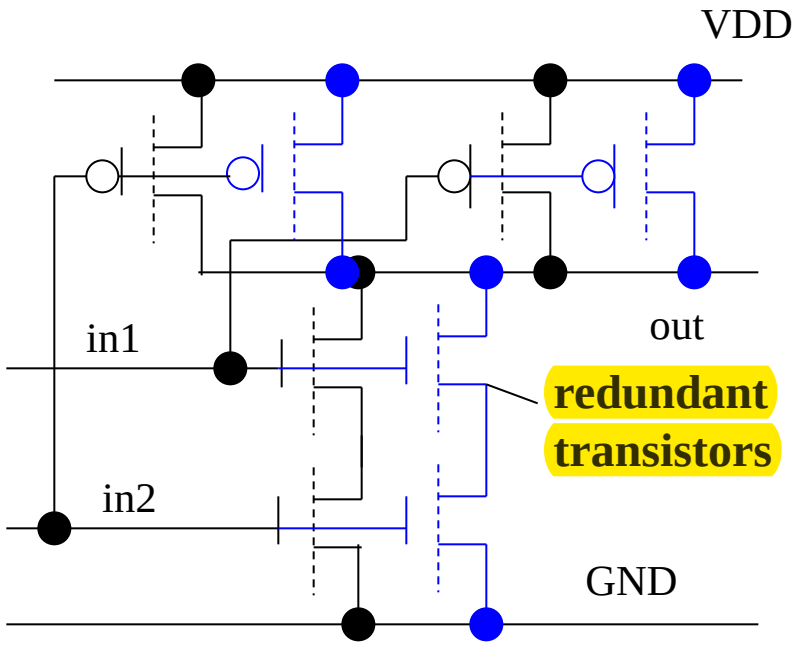
Flash-Memory with Coordinator



Transistor Circuit with Redundancy

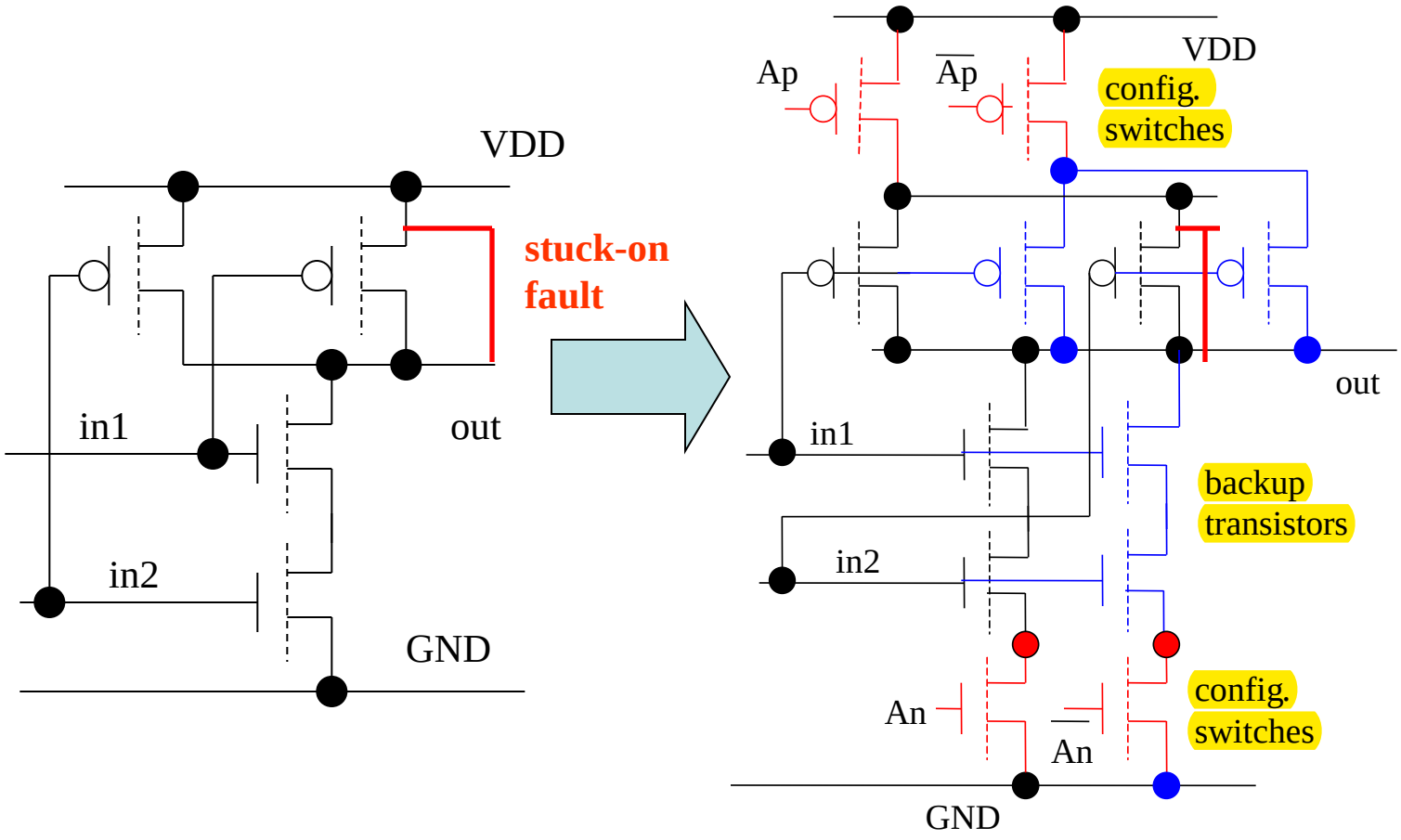


Basic gate

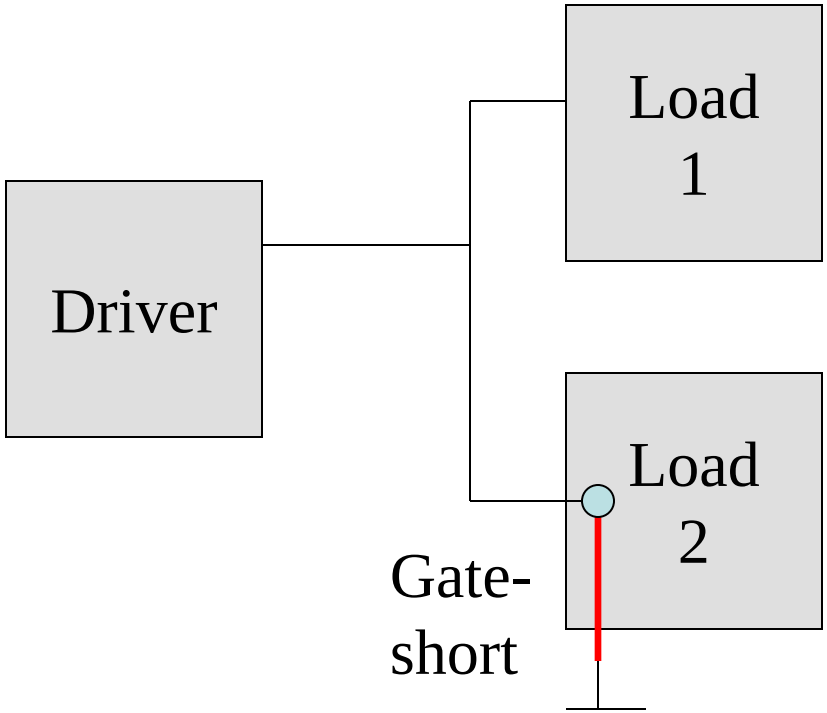


Gate with redundant transistors

Extensions for Fault Isolation

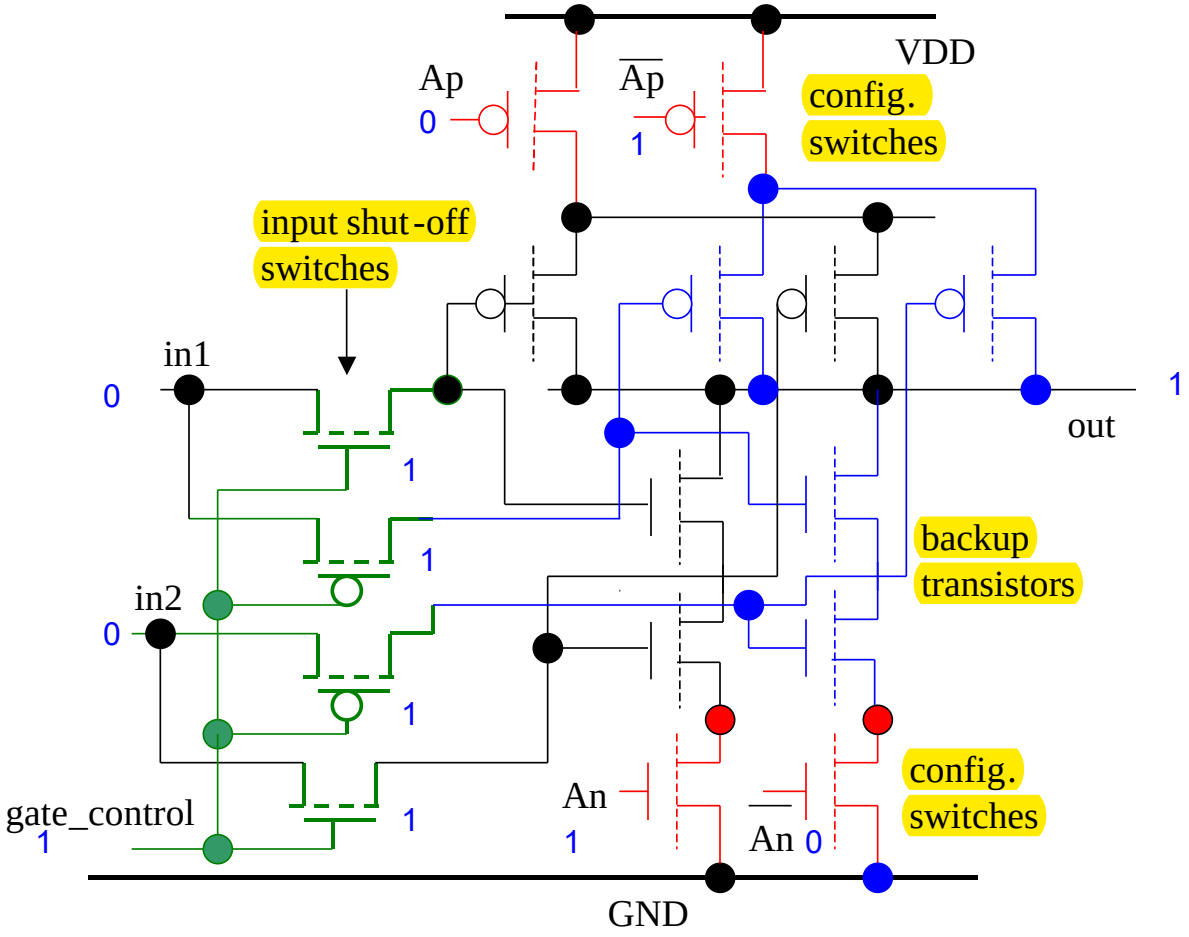


Fault Isolation as a Problem



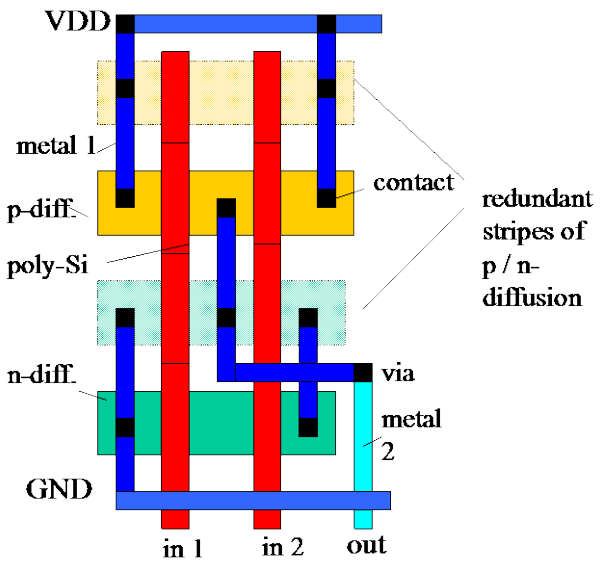
GND-shorts of input gates affect the whole fan-in network and make redundancy obsolete!!

Transistor Circuit (2-NAND) with full Fault Isolation

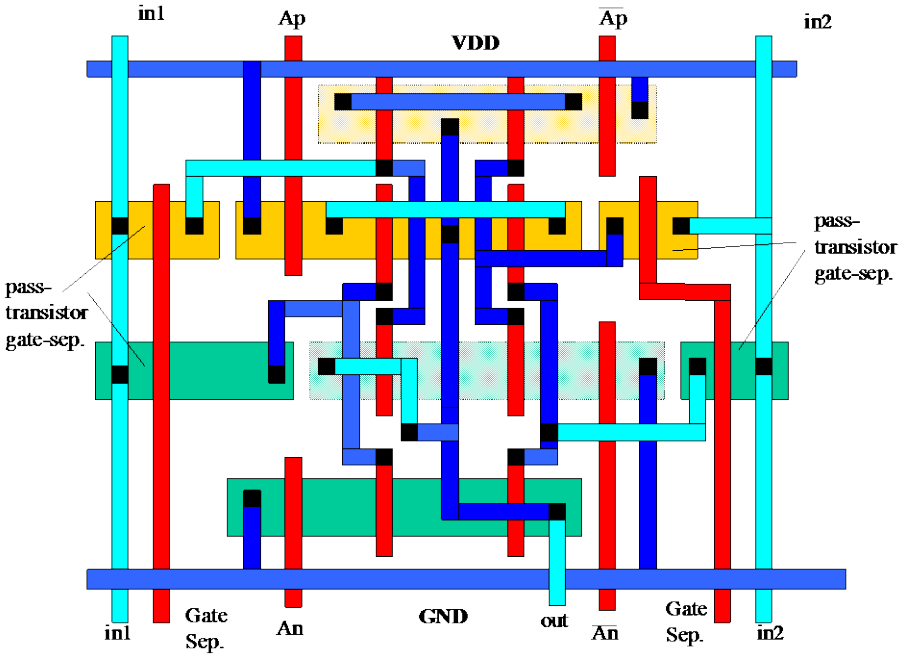


Number of transistors multiplies by 4, not counting extra control signals !

2-NAND, Schematic Layout

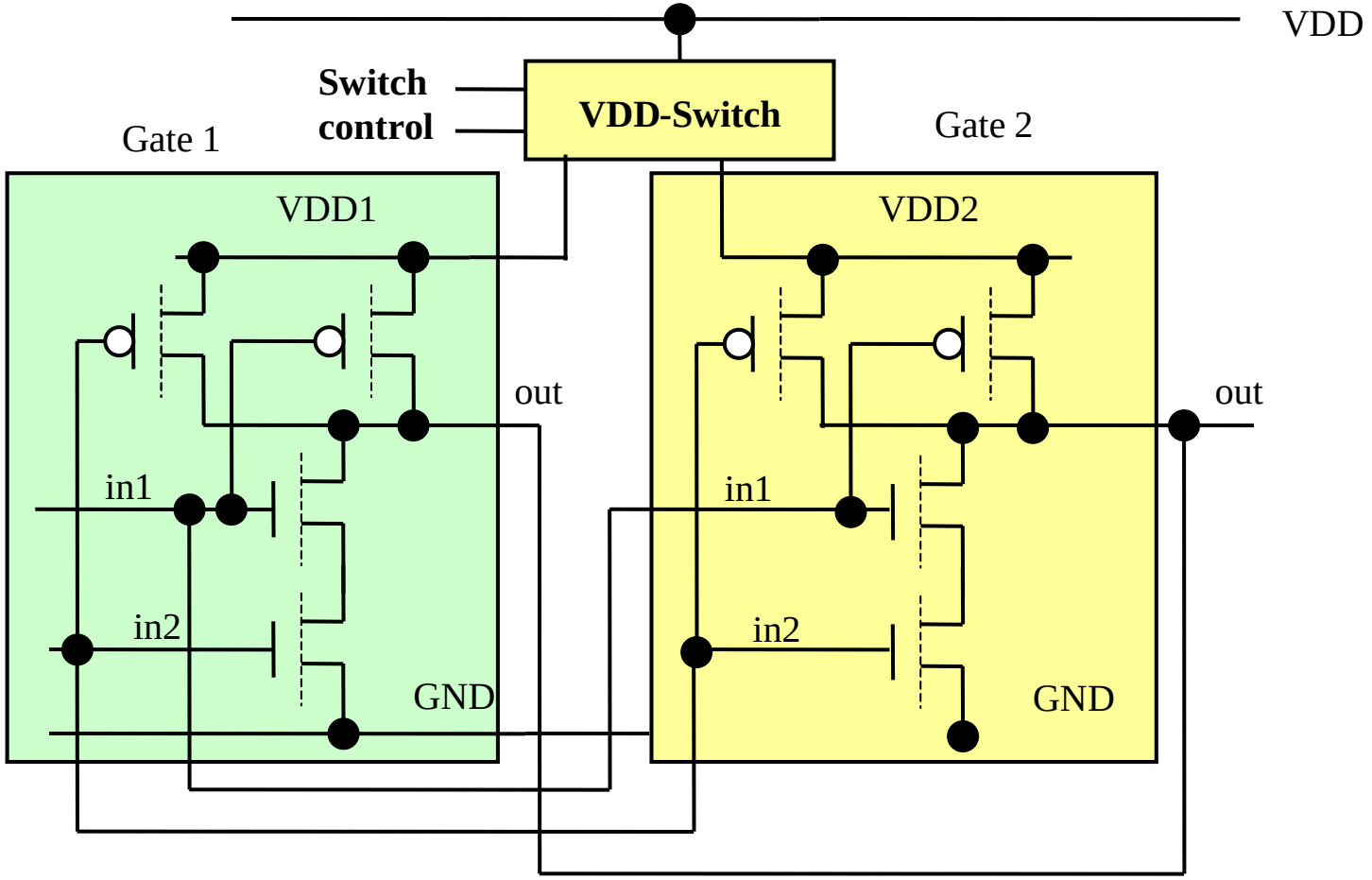


Gate with parallel redundancy

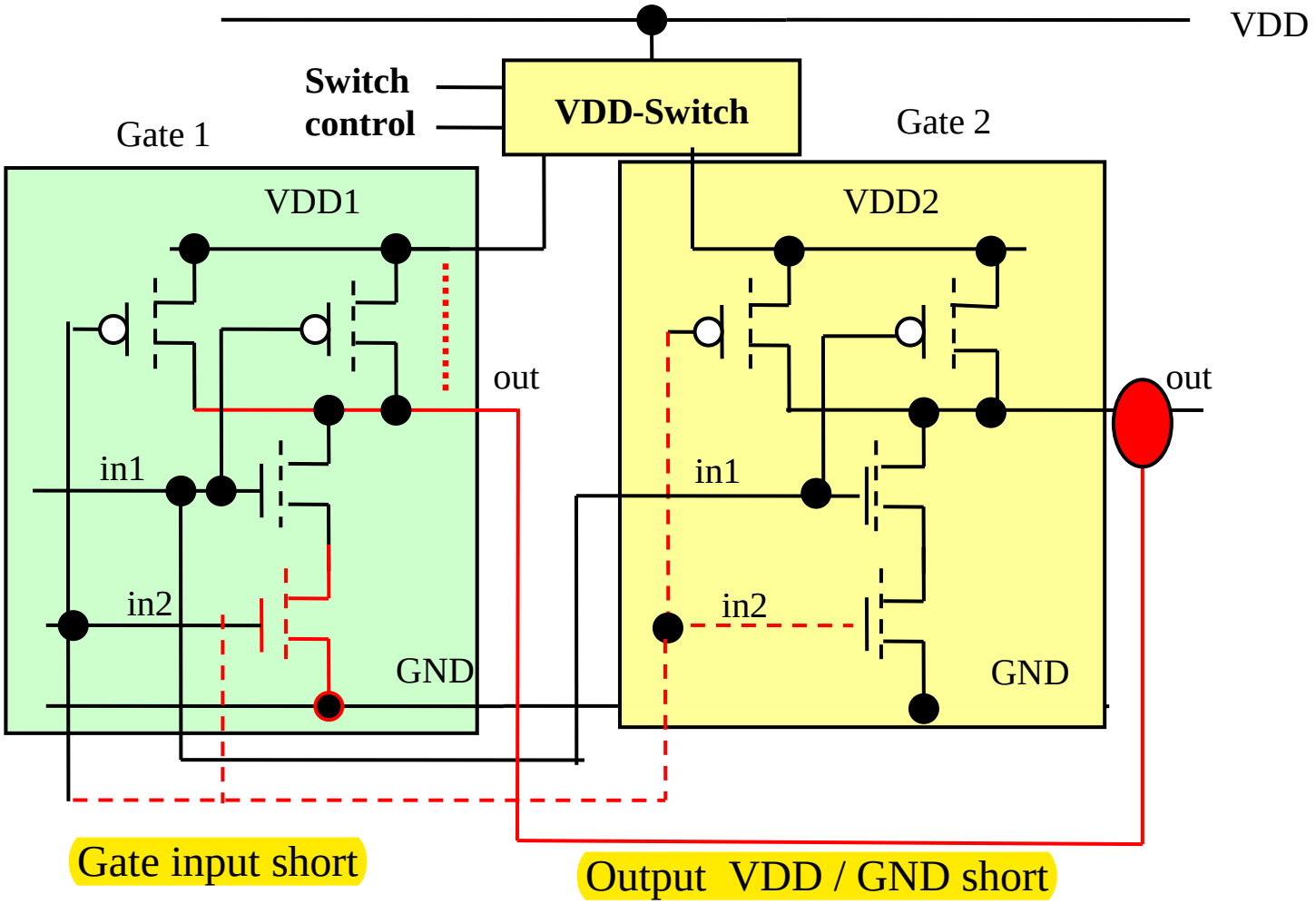


Gate with parallel redundancy and fault isolation

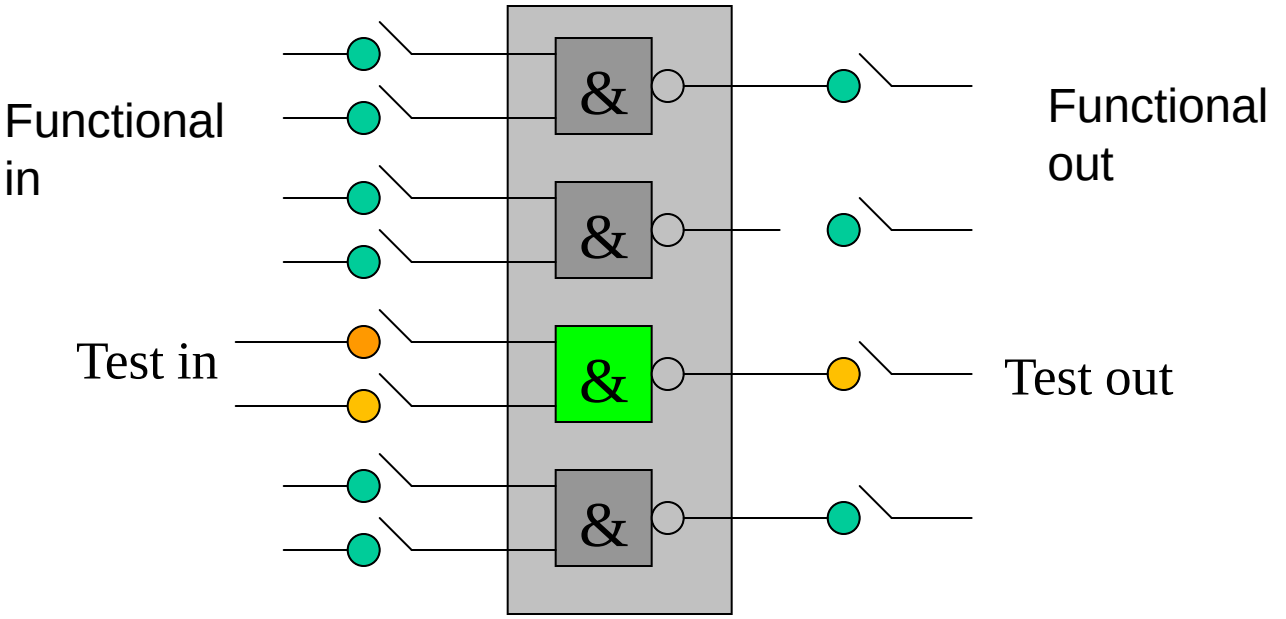
Duplication with Shut-Down



Again: Fault Isolation

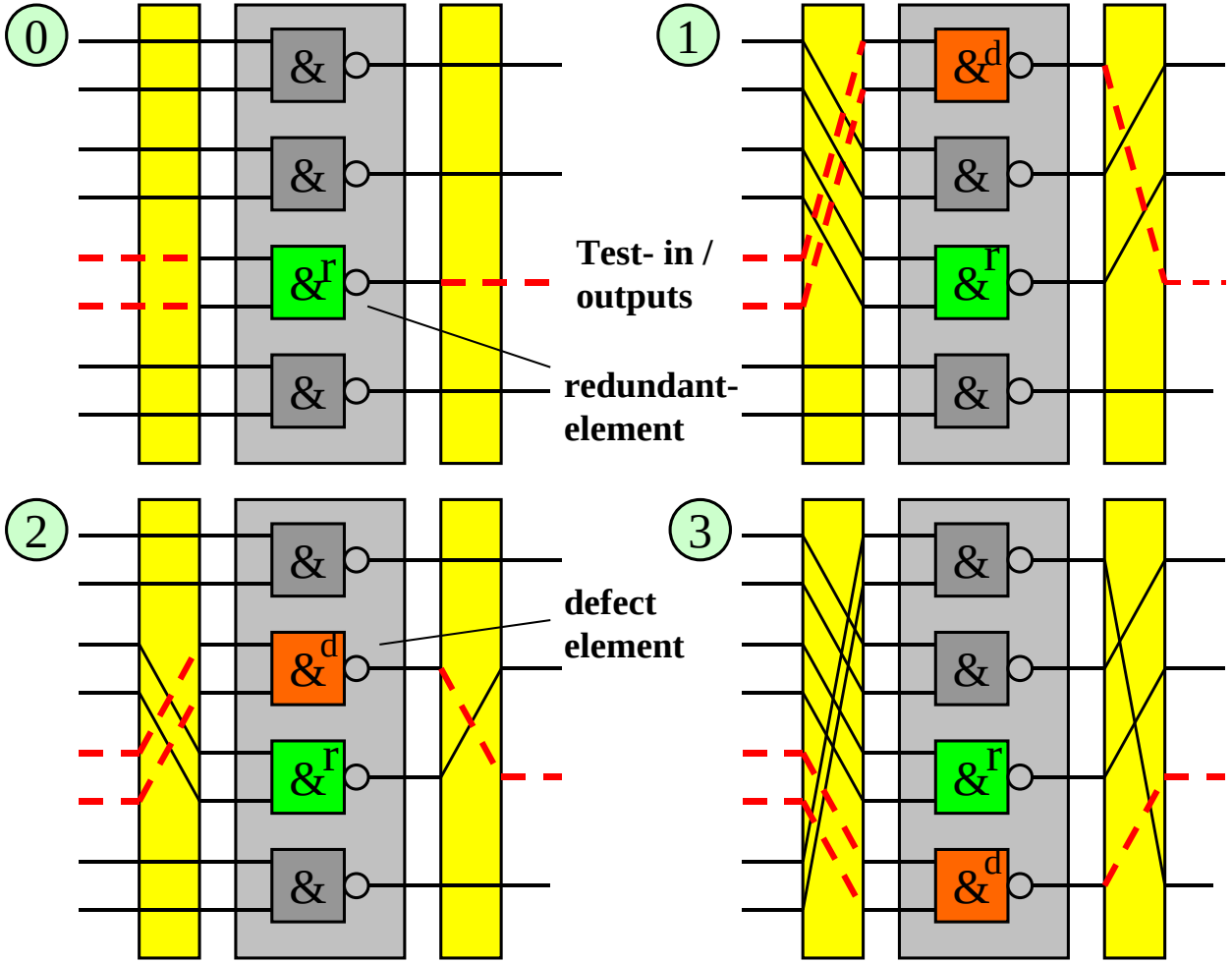


M out of N Redundancy

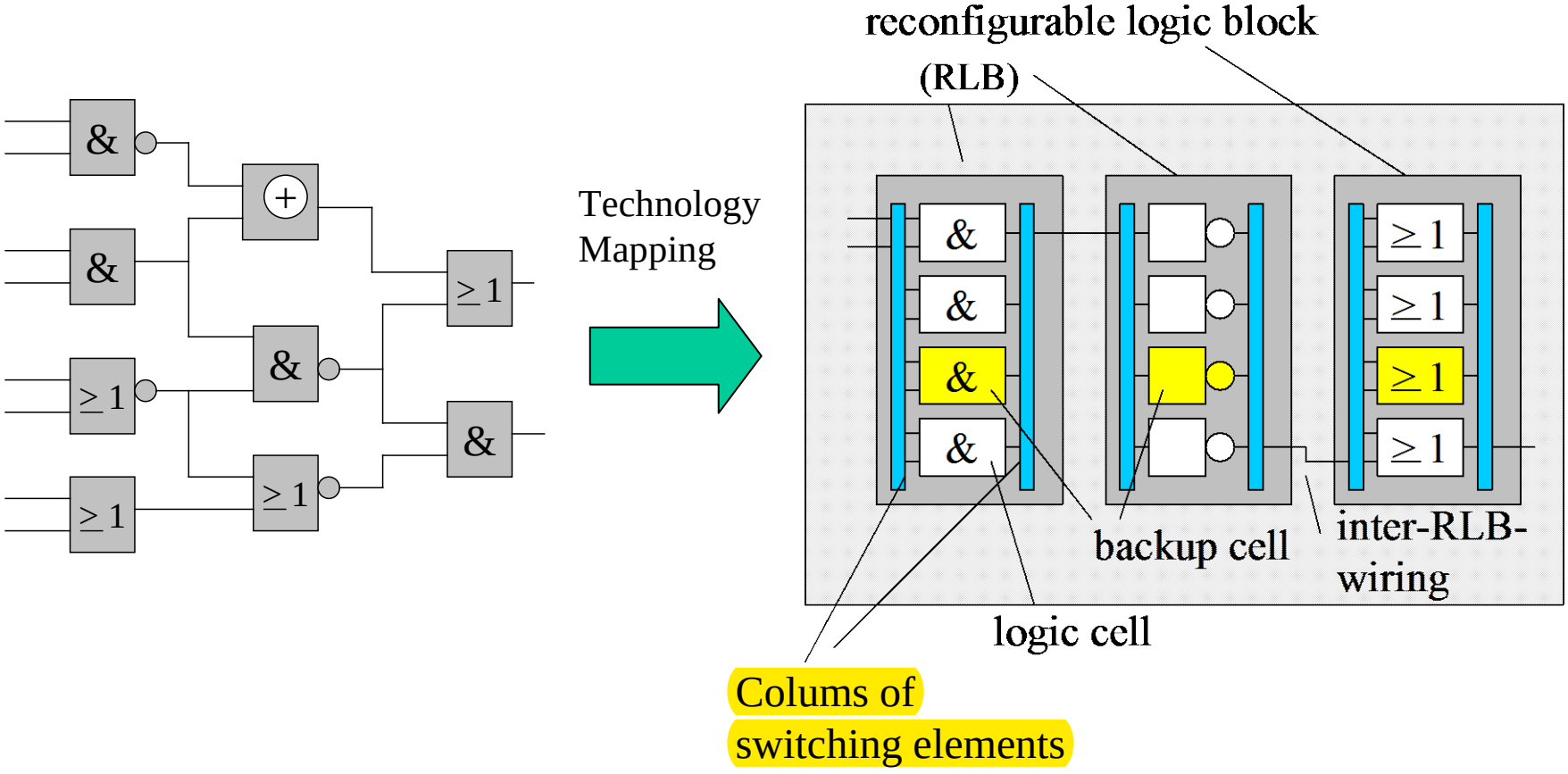


- Re-configurable blocks with n identical units, m used in operation.
- (n-m) units as spares for repairs, columns of switches at inputs / outputs.
- Switches provide fault isolation even for gate-shorts
- Minimum overhead is two pass transistors per I / O

Extended Concept with Test Ports



Logic-Mapping Re-Configurability



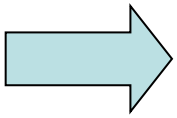
..is possibly not useful due to high switch / control overhead at gate level !!

Note: replacing single gates would possibly not cure defects on interconnects !

Relative Overhead

Basis-Gruppe	Basis-Trans.	mit Backup	Schalt-Trans.	Overhead/ %
2-in Gatter	4	8	6	250 %
Verdoppl.	6	12	6	200 %
2- NAND*	12	16	18	180 %
2-AND*	18	24	18	130 %
Halbadd.*	36	48	24	100 %
Volladd.*	90	120	30	66 %

*Gruppen mit 4 Gattern, davon 1 Ersatz



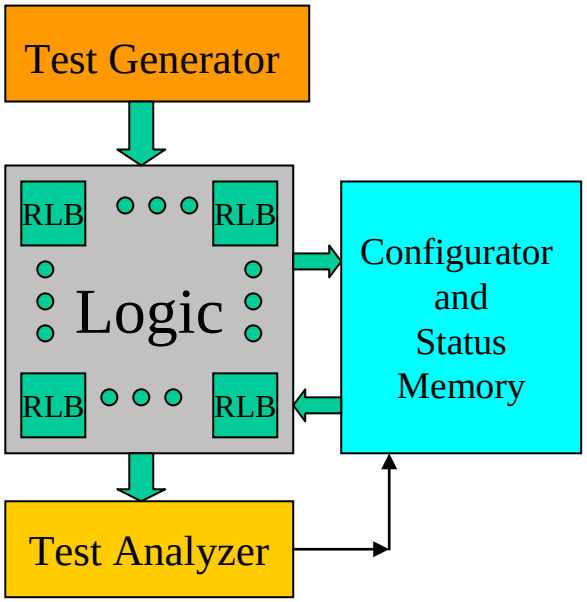
Overhead is dominated by switches !!

Overhead with Test Function

Transistors per RLB (3 functional units)

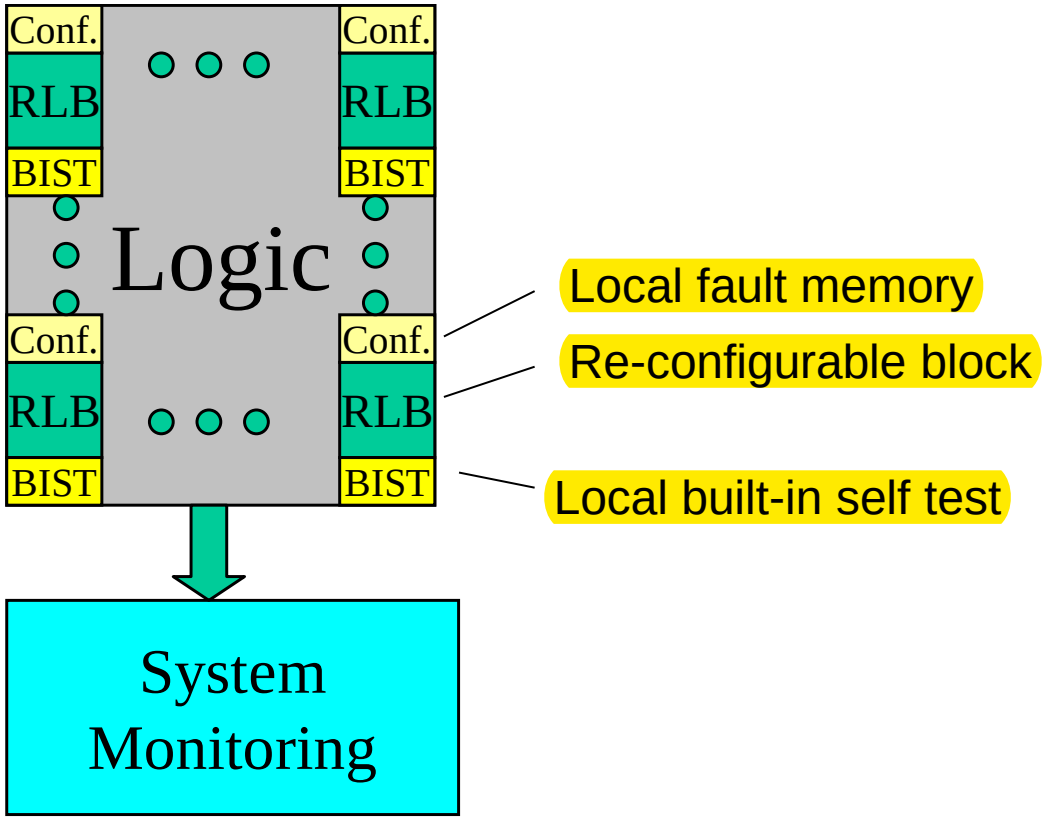
Basic Block	functional	backup	Switches min. / ext.	Overhead
2- NAND	12	4	18 /24	230 %
2- AND	18	6	18 /24	160 %
XOR	18	6	18 /24	160 %
Half Adder	36	12	24 /30	116 %
Full Adder	90	30	30 /36	73 %
8-bit ALU	4500	1500	168 / 224	38 %

Centrally organized (Re-) Configuration



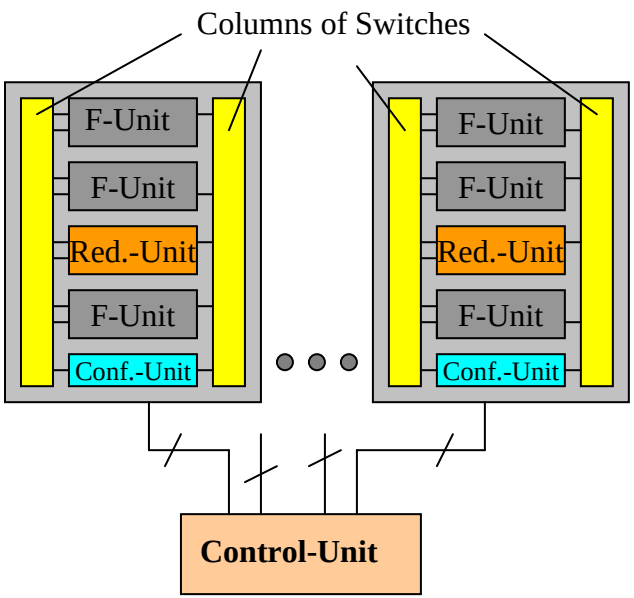
Problem: There must be a permanent non-volatile memory for recording of permanent fault condition.

De-centralized (Re)- Configuration Control

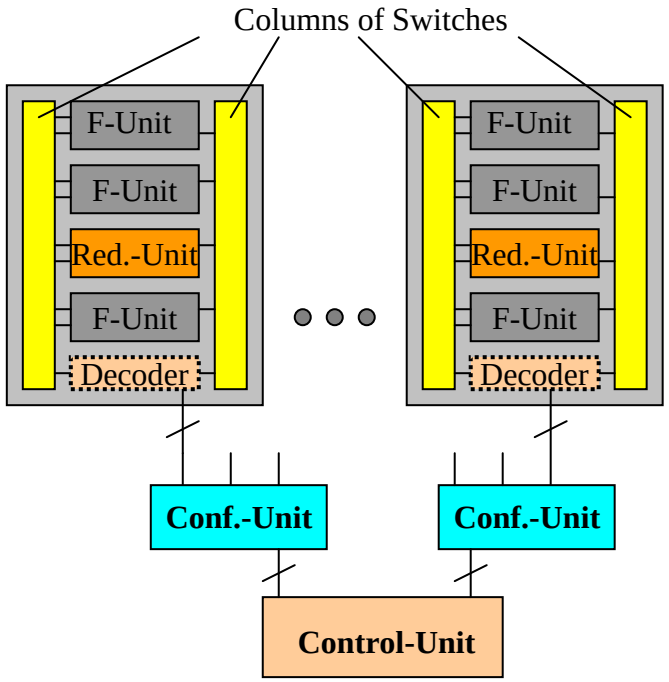


Separate and Common (Re-) Configuration

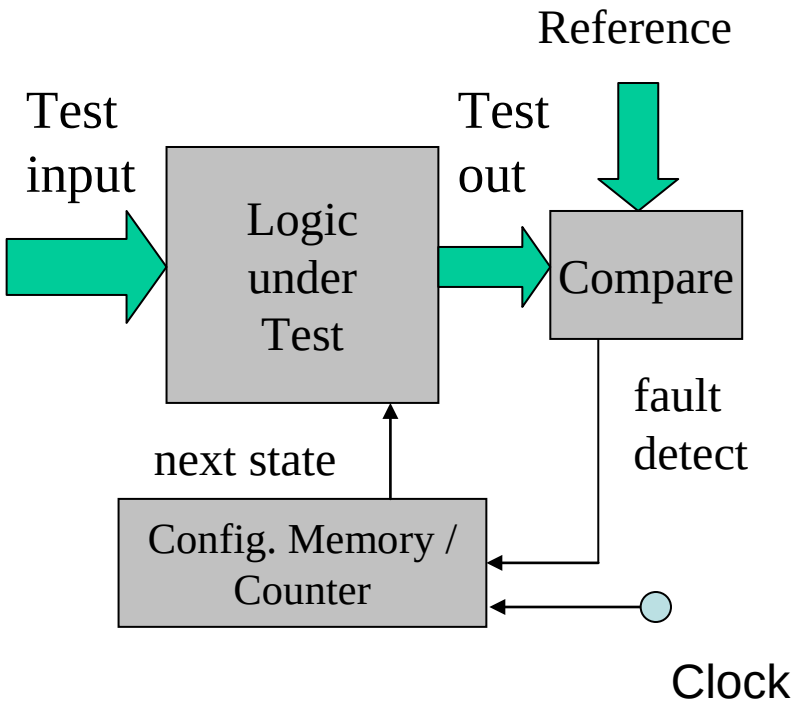
Local (re-) configuration



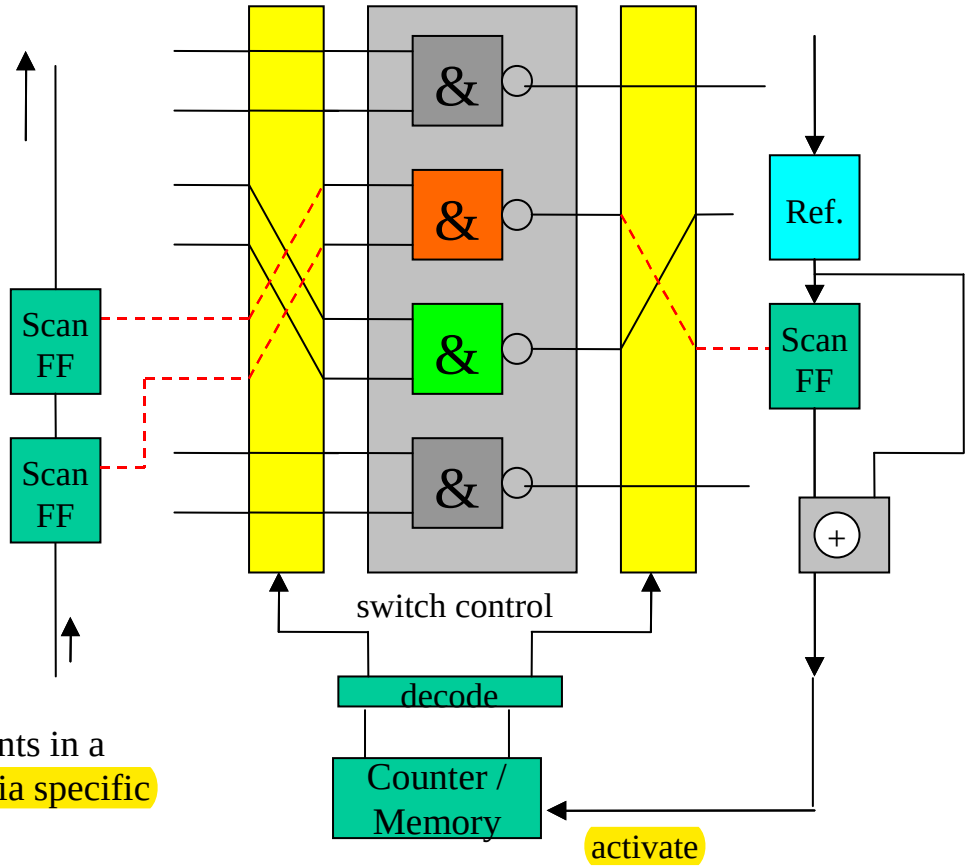
Remote (re-) configuration



Test and Re-Configuration

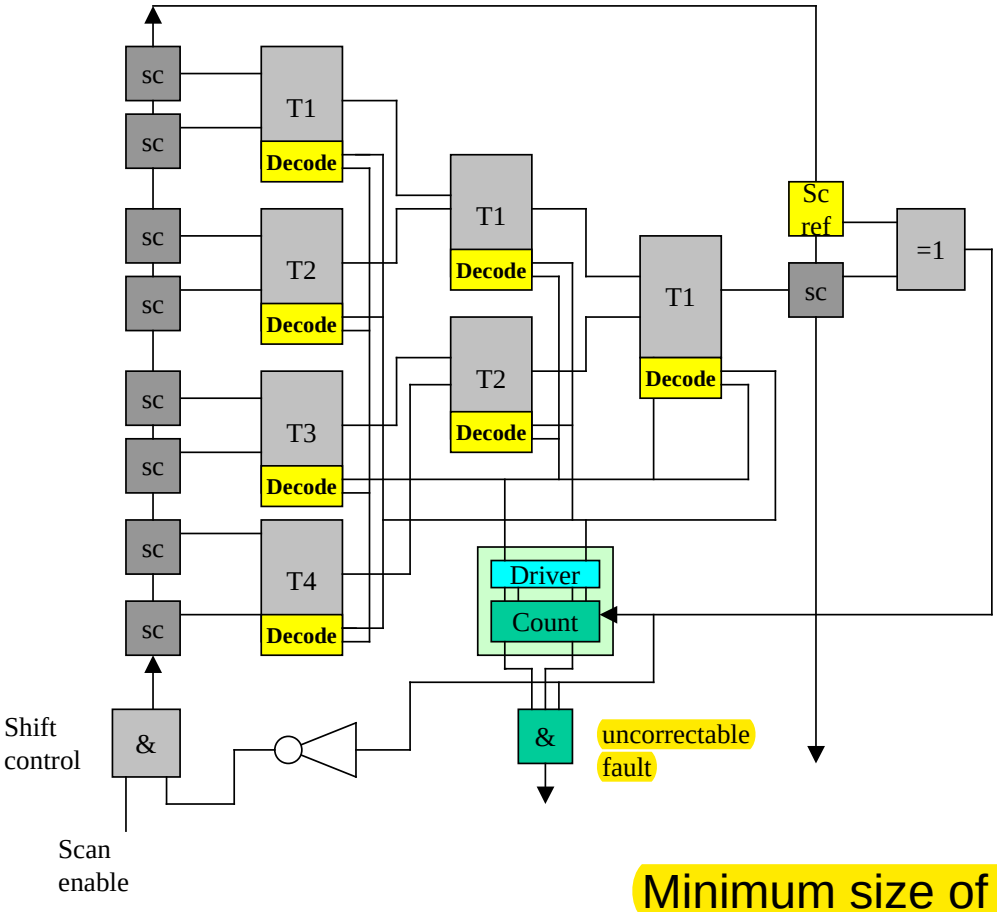


Test and Re-Configuration



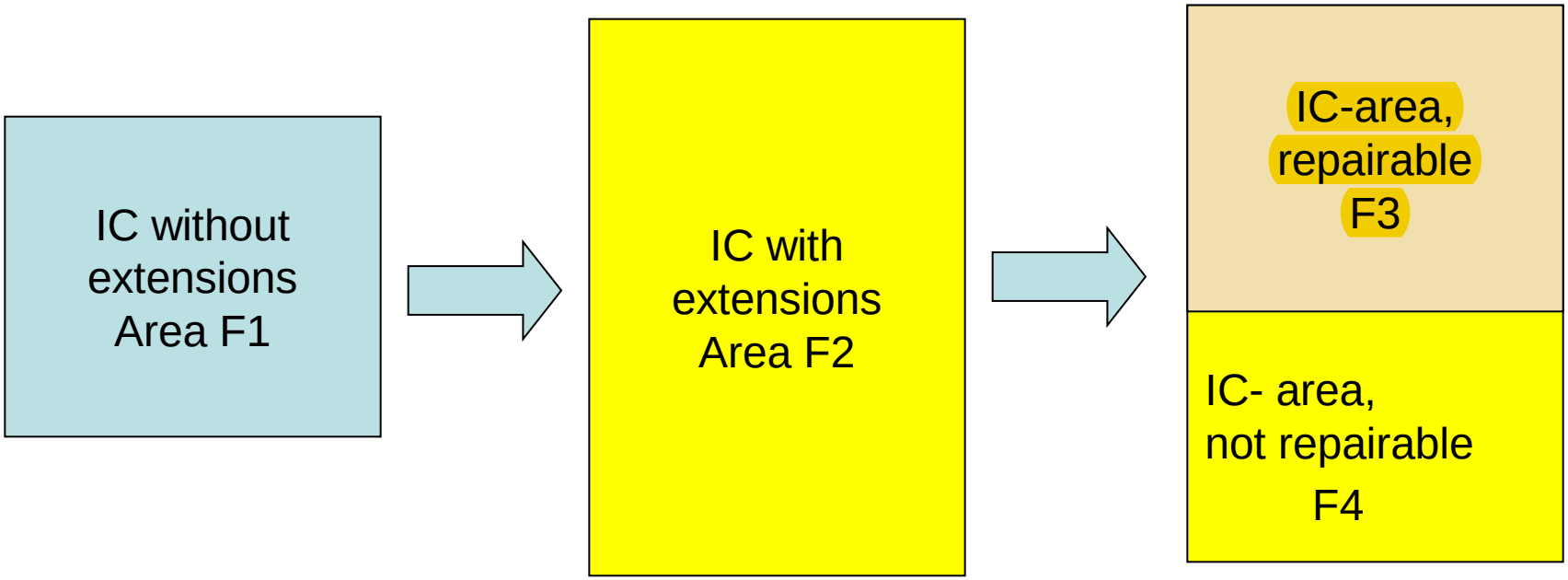
- Each of the elements in a block is testable via specific test inputs.
- Test is done by comparison with reference outputs.

Common (Re-)Configuration



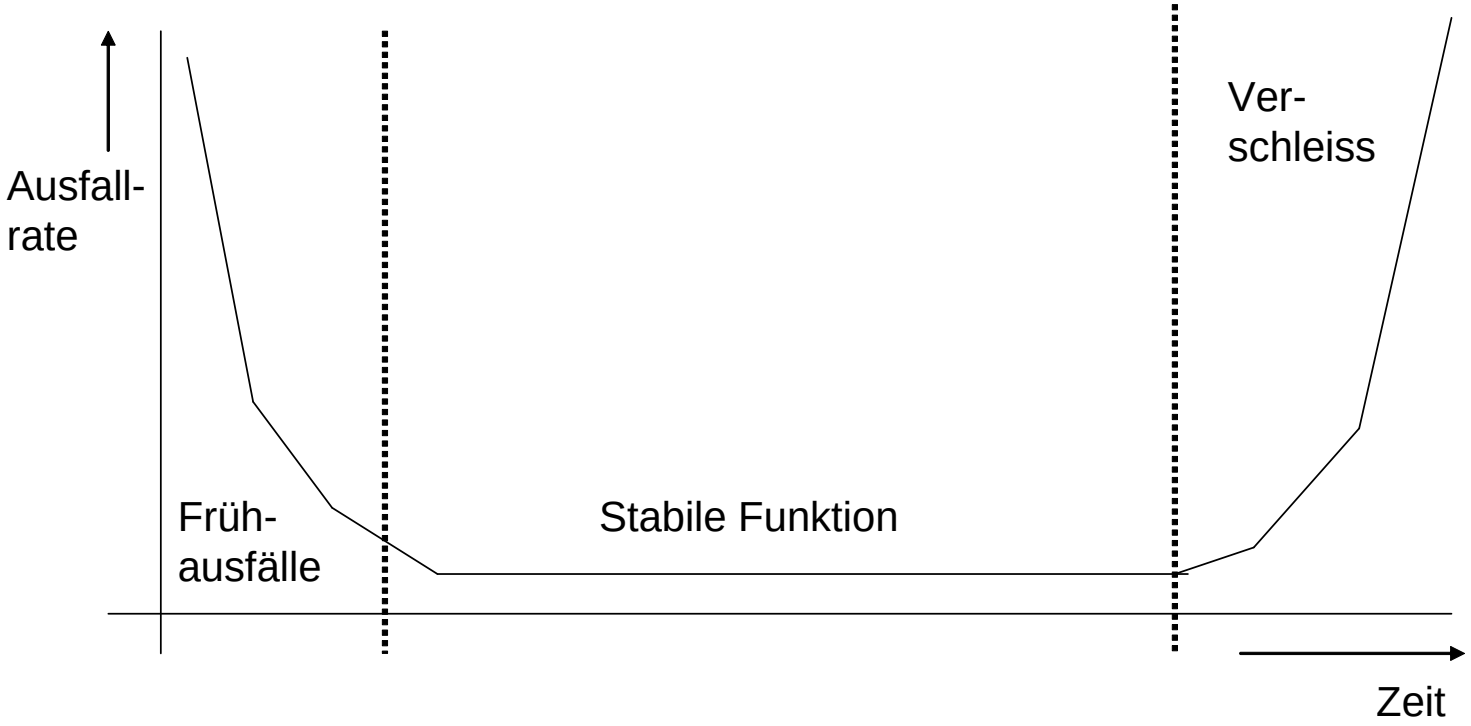
Minimum size of control logic for a block: about 60 to 90 transistors!

Overhead and Reliability



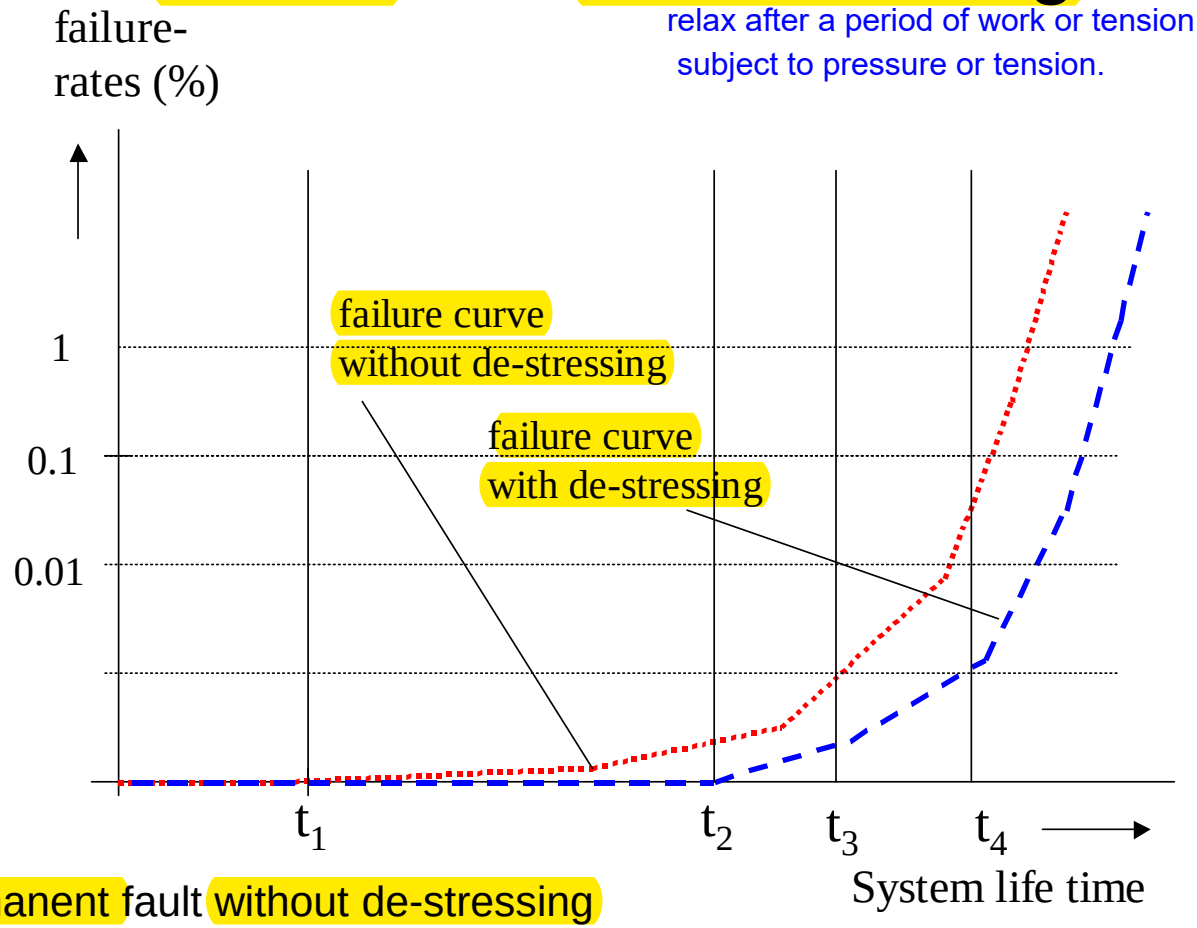
If F4 becomes larger than F1, the whole scheme does not work !

The Bath Tub Curve



Faults and De-Stressing

relax after a period of work or tension.
subject to pressure or tension.



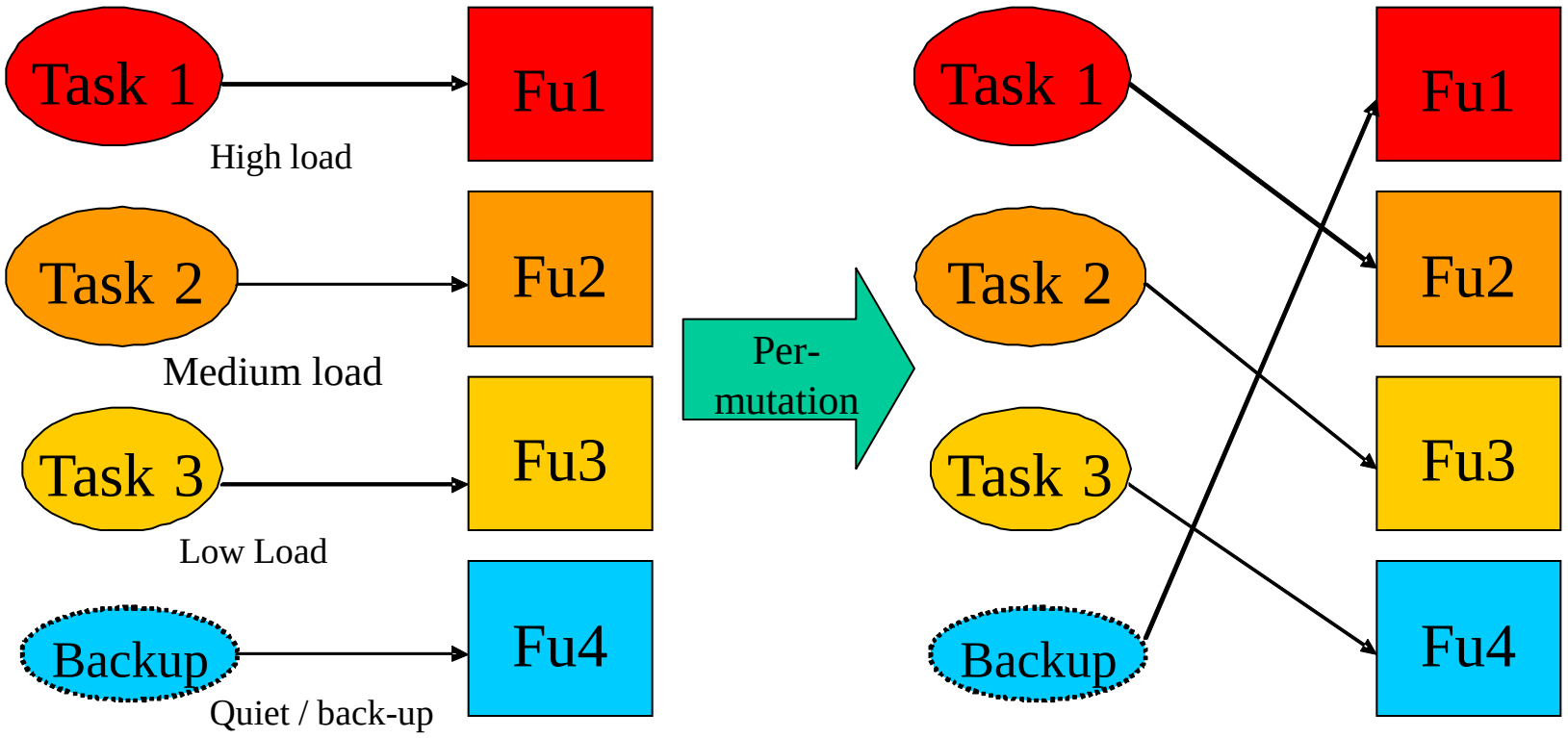
t_1: first permanent fault without de-stressing

t_2: first permanent fault with de-stressing

t_3: failure after repair resources exhausted, no de-stressing

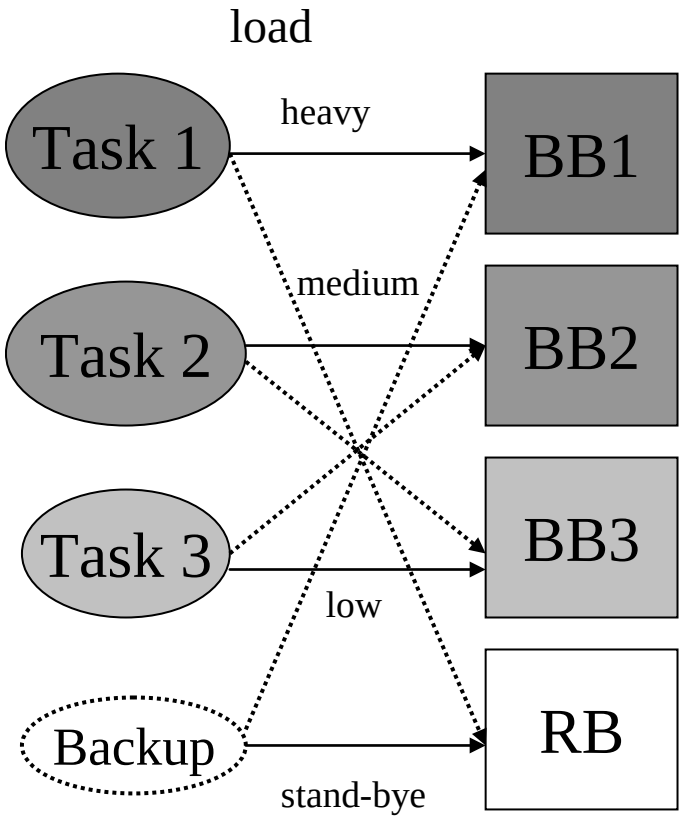
t_4: failure after repair resources exhausted, with de-stressing

Tasks, Functional Units and Stress



Is not an optimal scheme, since units may heavy load !

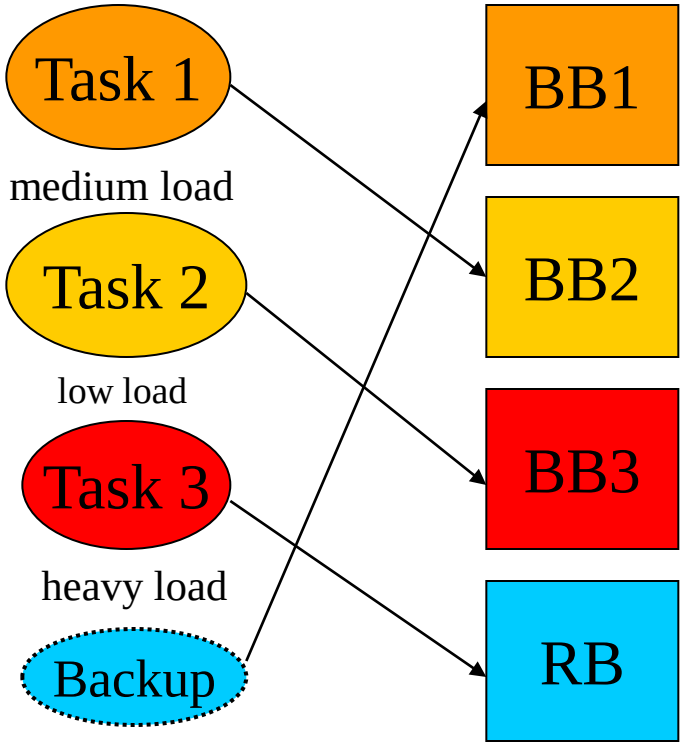
Optimized De-Stressing



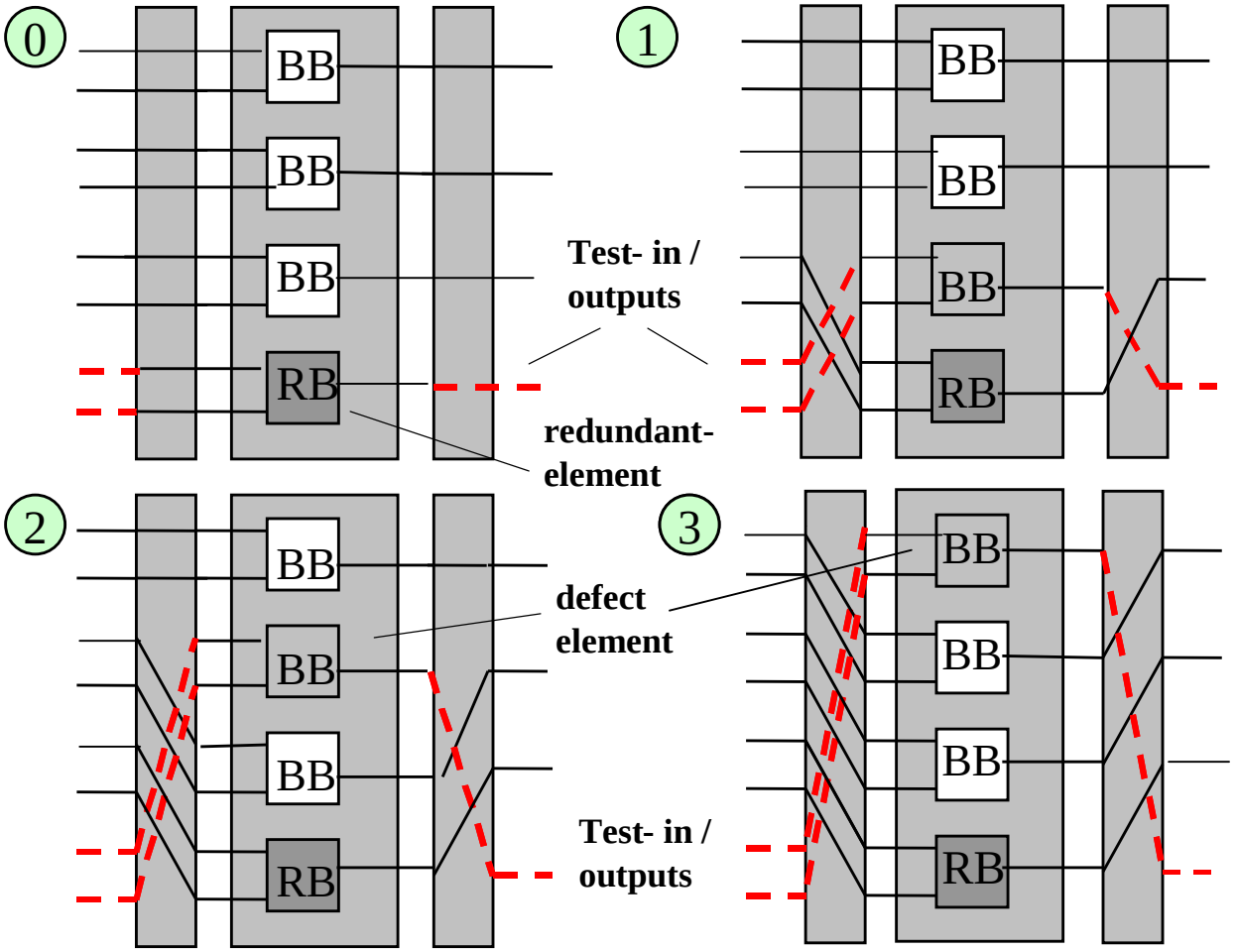
Problem:

In case of pair-wize exchange, not all units can be replaced for repair!

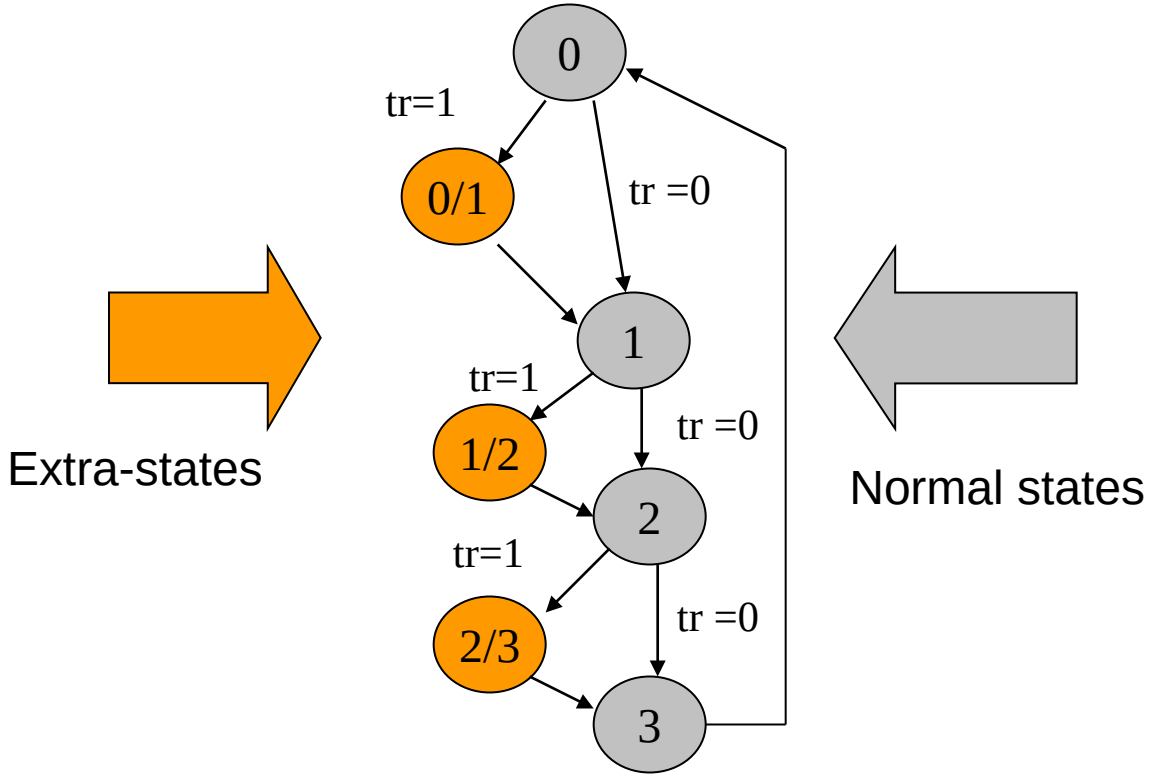
Repair Compatible De-Stressing



States for De-Stressing and Repair

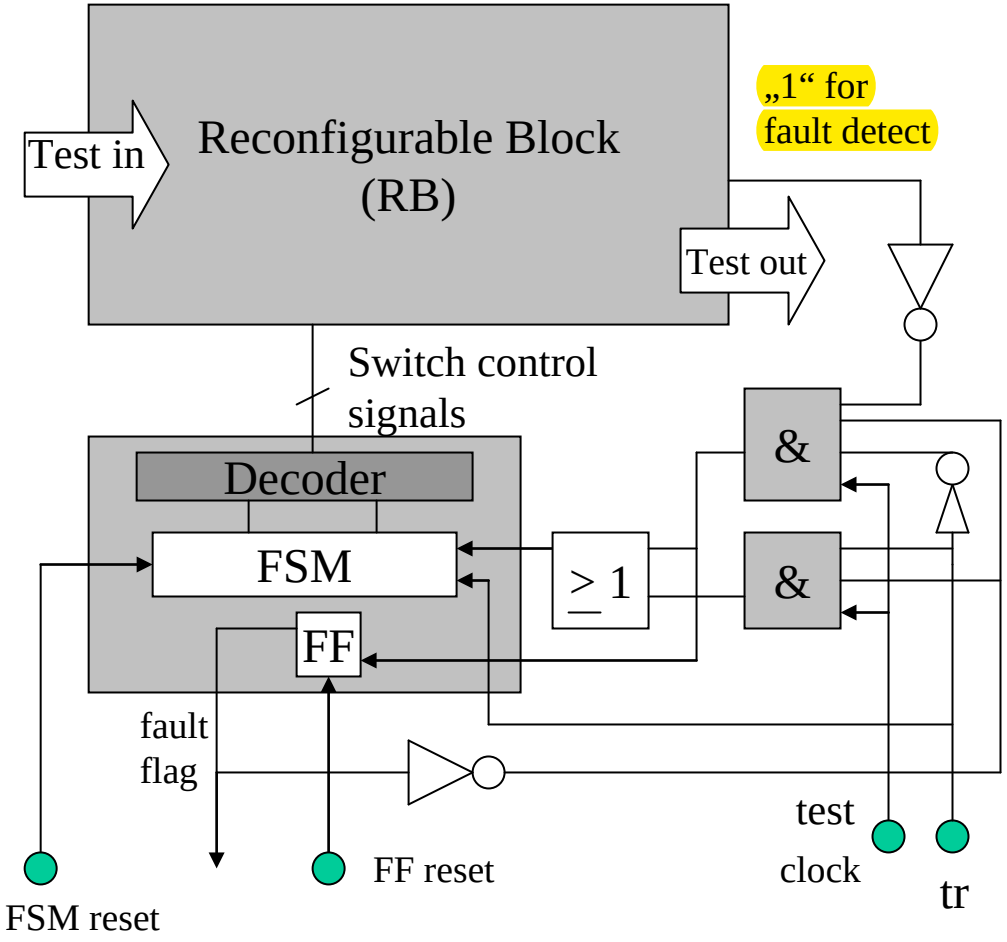


System States for De-Stressing and Repair



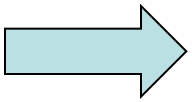
For changing units in de-stressing, we can keep two units running fully in parallel for the transition, as long as none of them is faulty !

Extended Control Logic



Overhead and RLB-Size

Basic-Elements	Basic-Trans.	With Backup	Switch-Trans.	Overhead/ %	Control Logic		Overhead / %	
					basic	de-stress	basic	de-stress
Half Adder	36	48	30	116	112	228	427	750
Full Adder	90	120	36	73	112	228	197	326
8-bit ALU	4500	6000	224	38	112	228	40.8	43.3



The local control logic has a significant impact on the overhead, but it grows sub-linearly with circuit size.

Minimum RLB size: ca. 500 Transistors !!

RLB-Zahlen und Overhead

Basic Block	function	red.+sw.	cntl.	ext.cntl.	Overhead factor
2-NAND	4 /12	28	112	228	11.6 / 21.3
2- AND	6 /18	30	112	228	7.9 / 14.3
H-Adder	12 /36	42	112	228	4.3 / 7.6
F-Adder	30 /90	66	112	228	2.0 / 3.3
Macro	500/1500	700	112	228	0.54 / 0.62
ALU	1500 /4500	1724	112	228	0.40 / 0.43

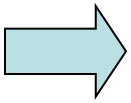
Single Points of Failure

- Clock networks
- Transistor switches for re-configuration
- Control-logic for (re-)configuraton
- Memory for control information

.. can hardly be designed to be fail-safe themselves !

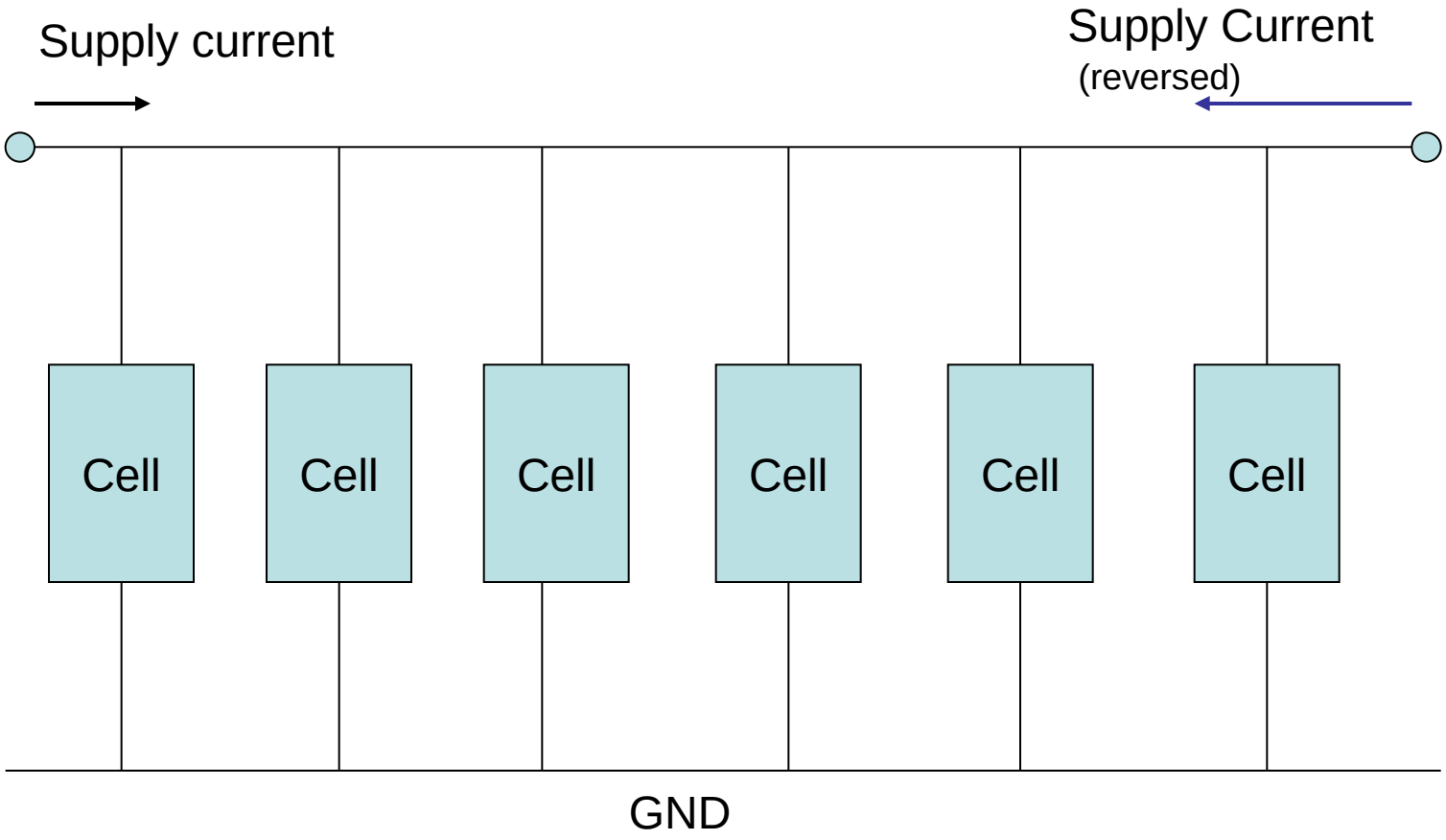
A fail-safe in engineering is a design feature or practice that in the event of a specific type of failure, inherently responds in a way that will cause no or minimal harm to other equipment, the environment or to people.

But: Configuration logic is not active most of the time. Therefore it can be assumed to have less wear-out effects than „functional“ logic. But if it is biased and inactive, it will suffer from NBTI / PBTI !!



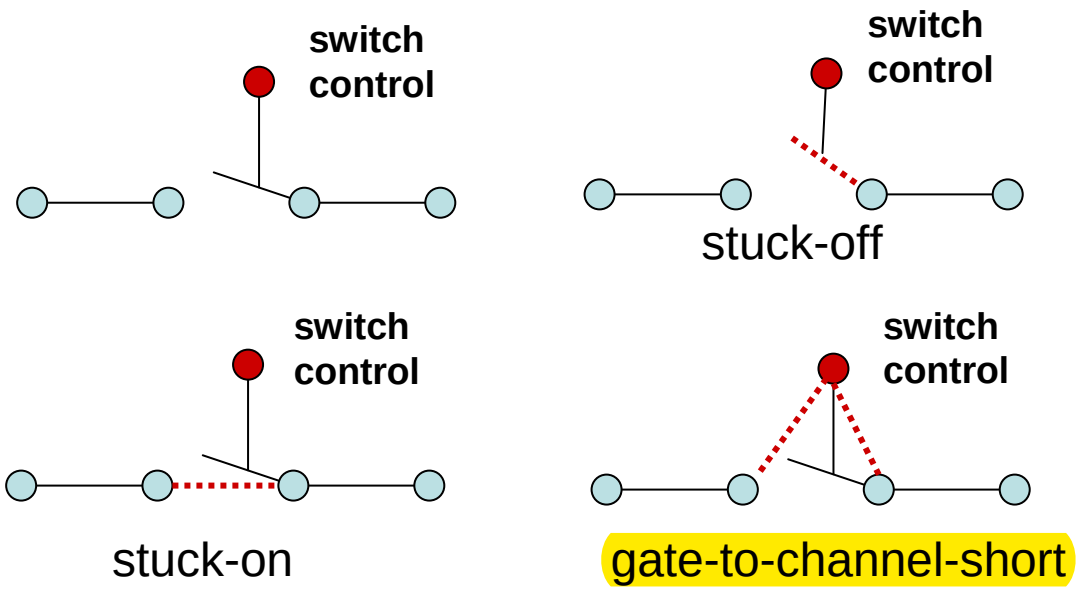
Self repair is mostly useful for the compensation of (pre-mature) wear-out induced defects !!

De-Stressing of VDD-Lines



Effects of electro-migration can partly be compensated by reversing the direction of IDD current flow !

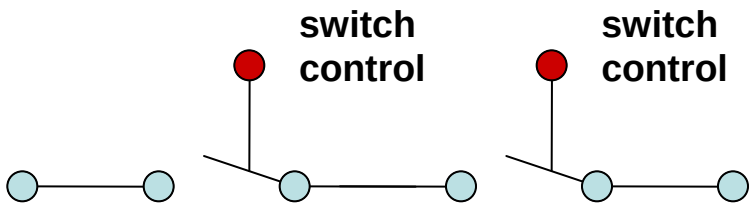
Transistor Switches and Fault Effects



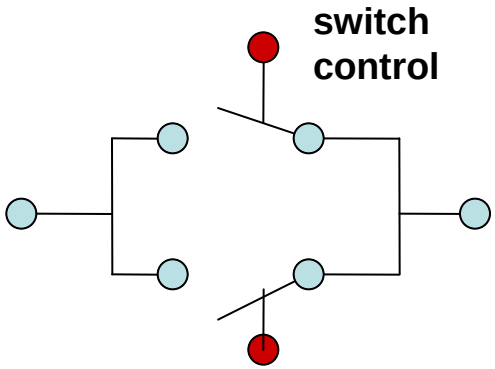
Iddq testing is a method for testing CMOS integrated circuits for the presence of manufacturing faults. It relies on measuring the supply current (Idd) in the quiescent state (when the circuit is not switching and inputs are held at static values). The current consumed in the state is commonly called Iddq for Idd (quiescent) and hence the name.

Redundant Switching-Structures

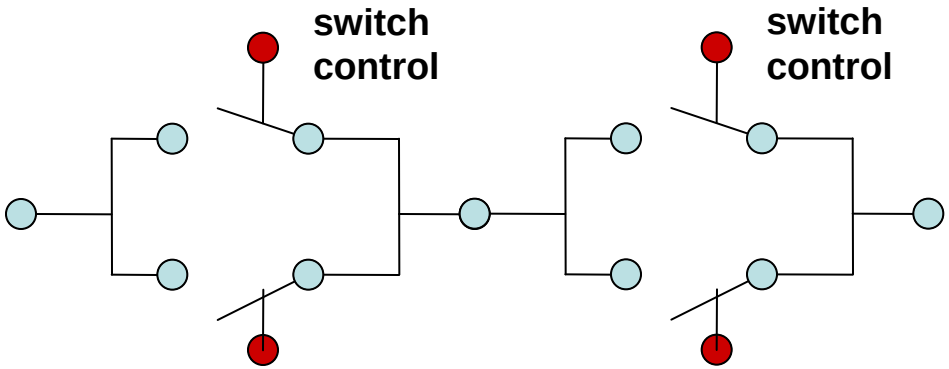
Compenationable Faults



Transistor stuck-on



Transistor stuck-off

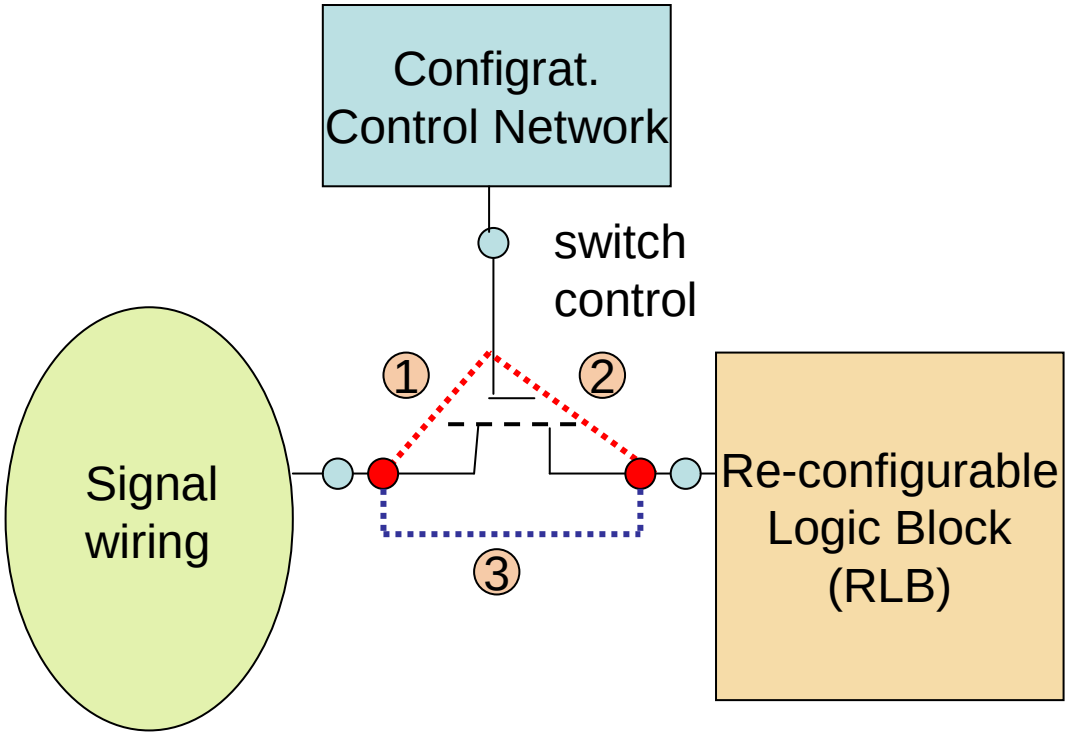


Transistor stuck-on /
stuck-off

Defects in Switches

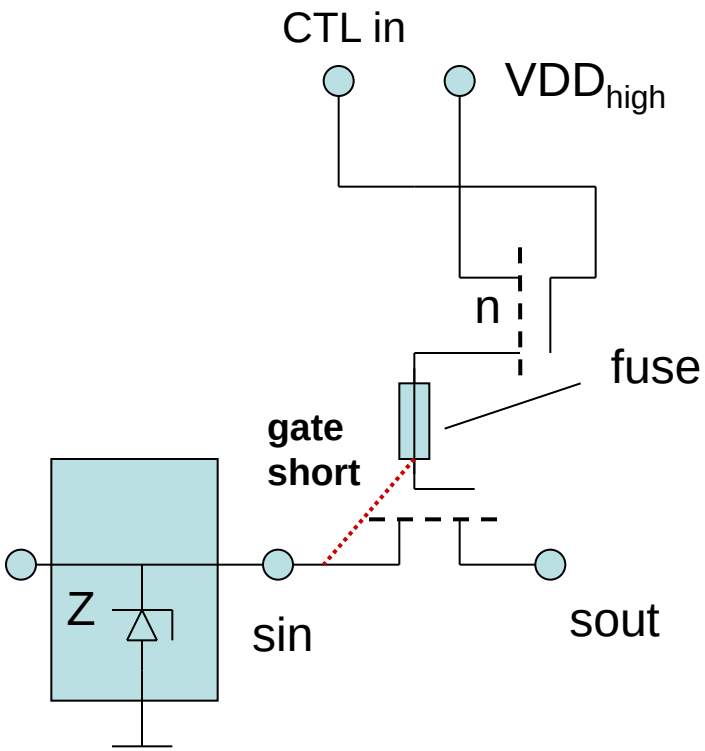
- In case a configuration switch is „alsowasy on“ or „always off, we can still find a configuration that can use this state….
- … unless we get multiple faults in switches of in logic.
- In the re-configurable logic, we have a network for the normal signal flow and another network to administrate the repair process. Bridges that make connections between these networks would cause problems !

Gate-Channel-Shorts

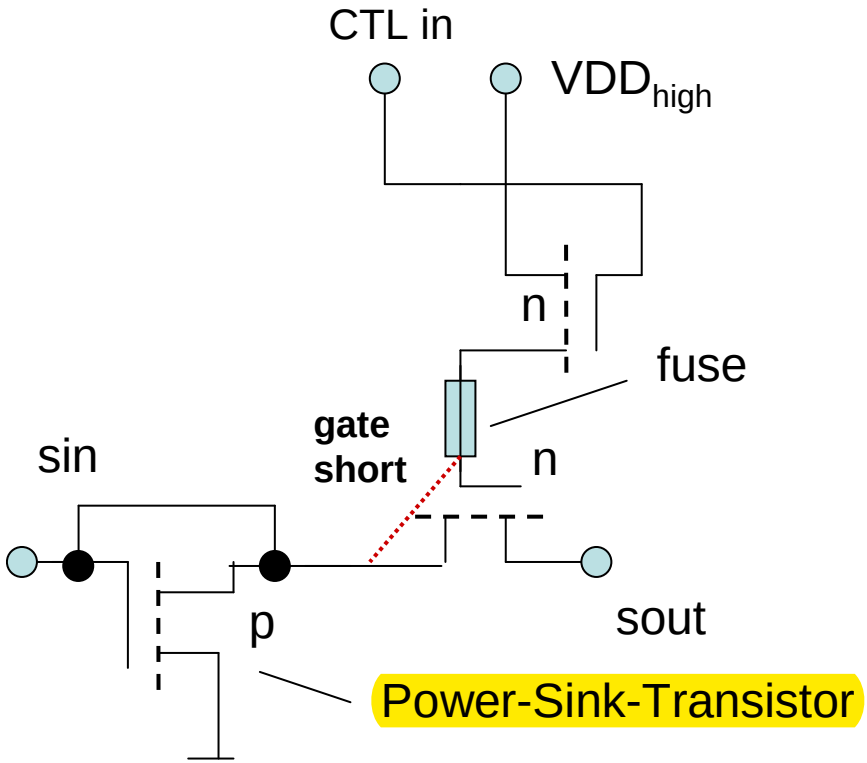


.. actually makes a bridge between the signal network and repair control ! !

Switches with „Fuse“

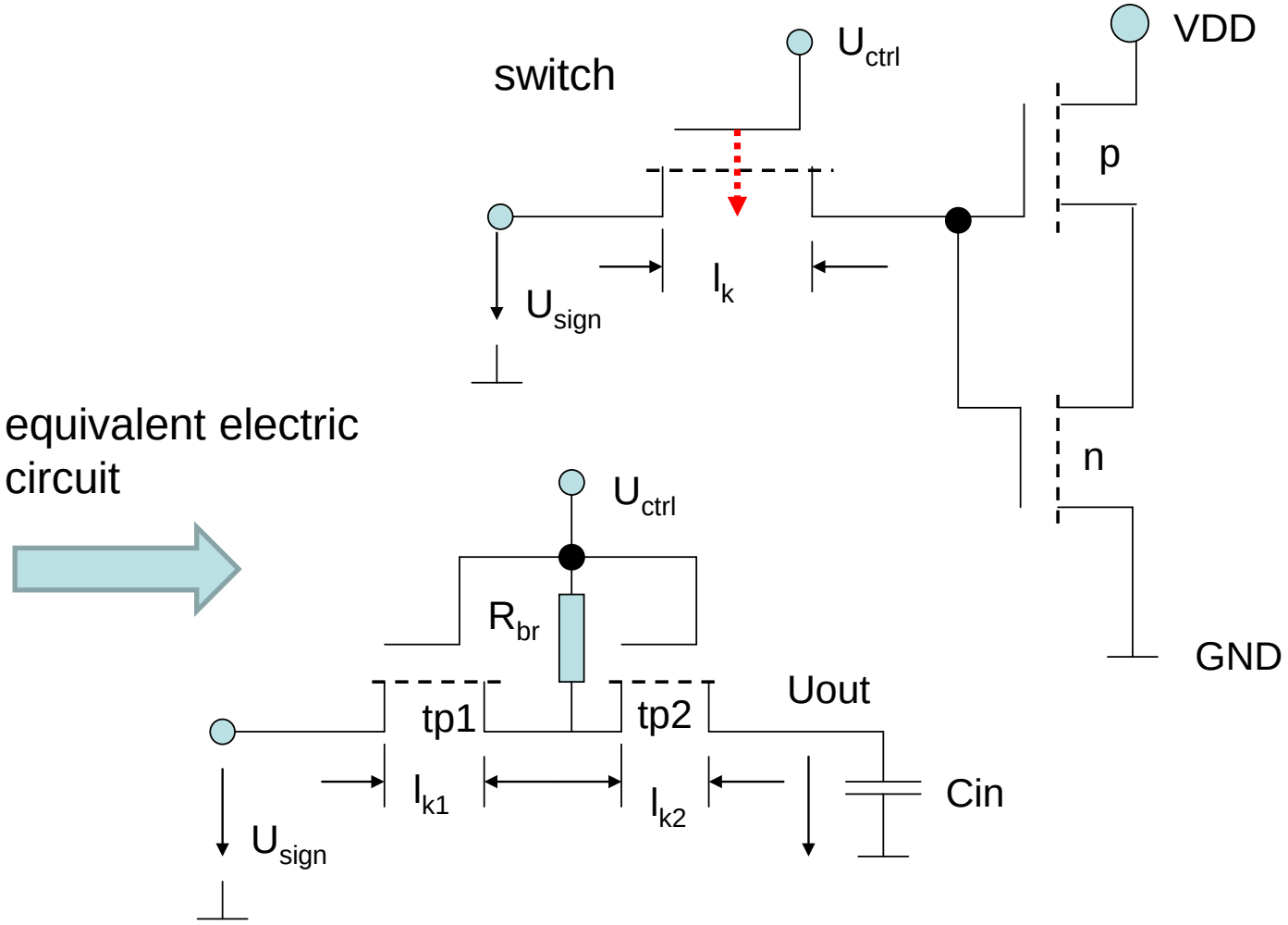


„Fuse“ with currect drainage by Z-diode



Drainage by „Power Sink“-transistor

Gate-Breakdown



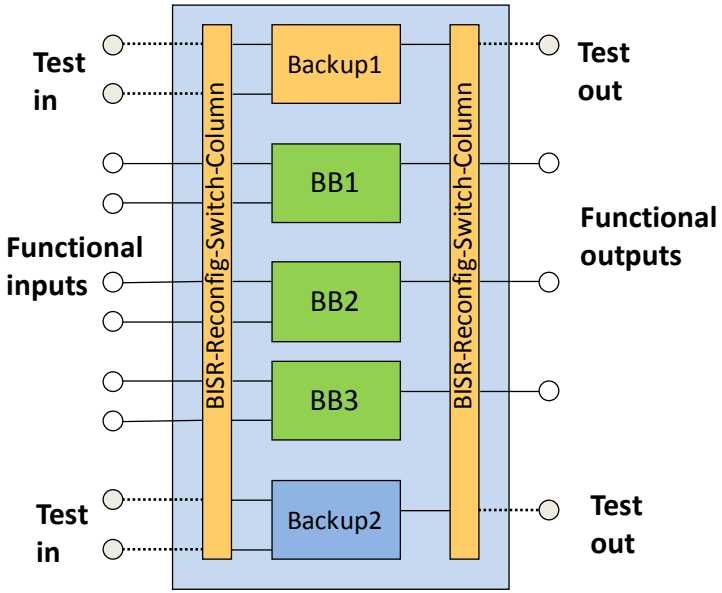
Secondary Effects after Gate Oxide Breakdown

U_{sign}	U_{in}	U_{ctrl}	U_{int}	U_{out}
1	1	Hel	Hel	Hel
1	?	0	0	0
0	0	0	0	0
0	?	Hel	He	Hel

Next channel short likely !

Hel = elevated 1- voltage level

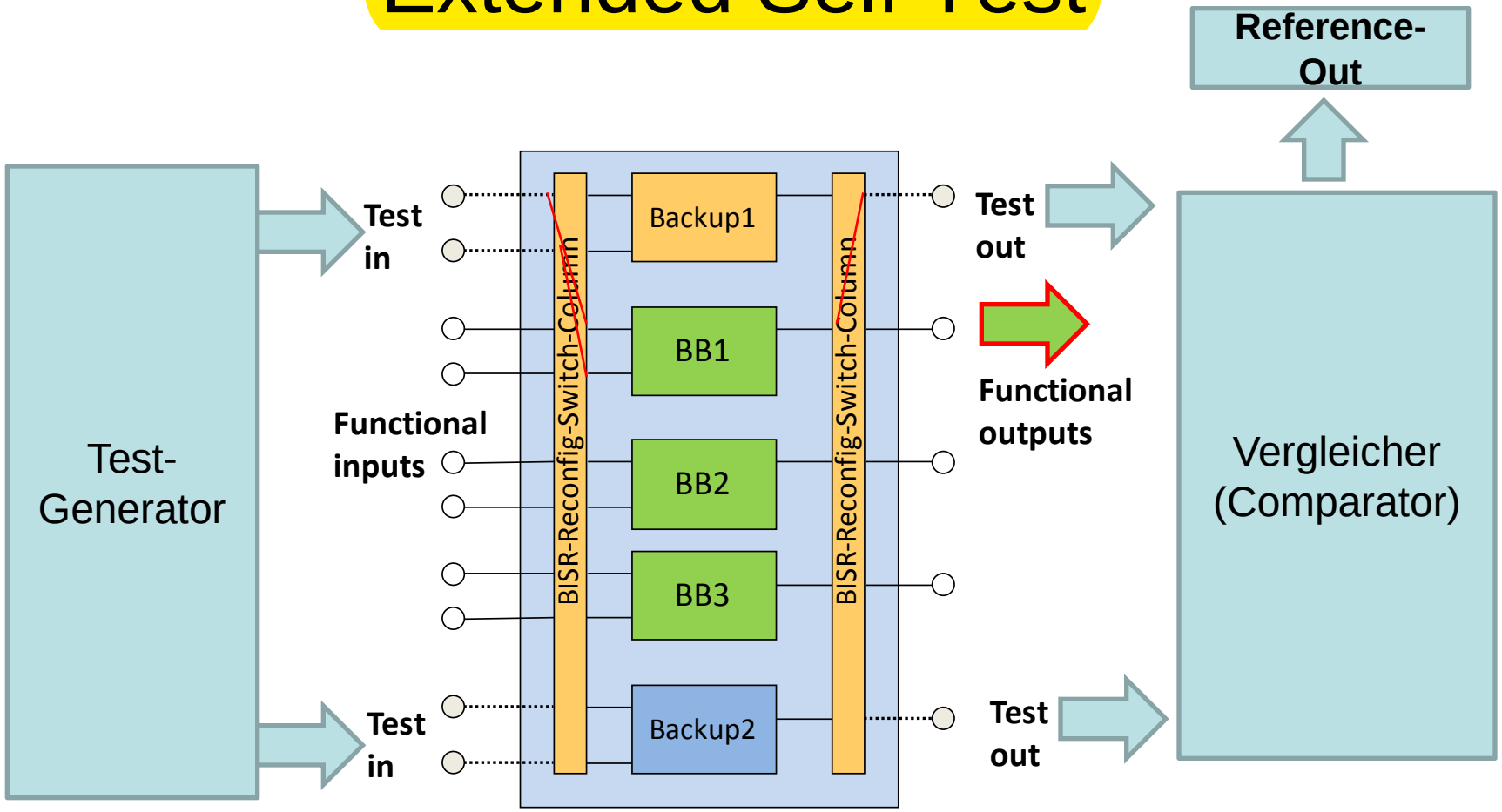
Re-Configurable Block with (n+2)- Architecture



Characteristics: - 2 backup units
- 3 switches at either input / output for „straight“, „up“, „down“.

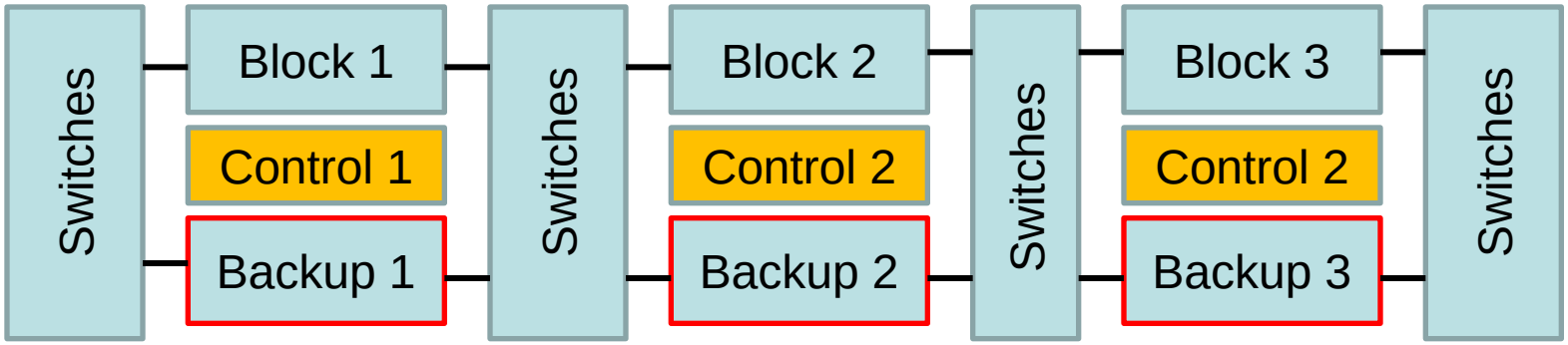
Any 2 out of n blocks can be replaced !

Extended Self Test



With 2 backup units, the system can do a functional self test at start-up !

Self Repair in Real Time ?



Asynchronous pipeline with duplicate units for local back-up !

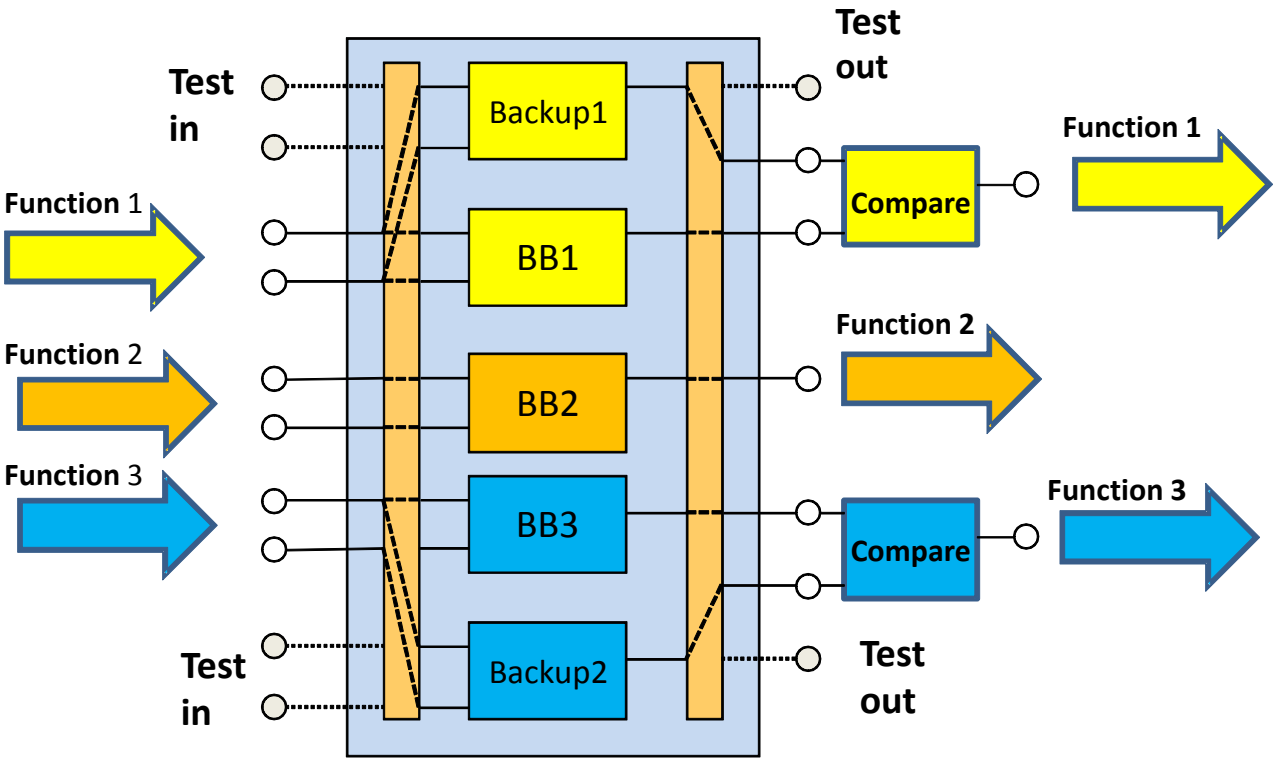
Function:

The controller monitors the signals that are needed to synchronize A-synchronous communication. If certain signal do not turn up, the respective local block is assumed to be faulty. It is then replaced by the backup unit.

Pro: The (sub-)system is self-timed and takes ist time for the repair !

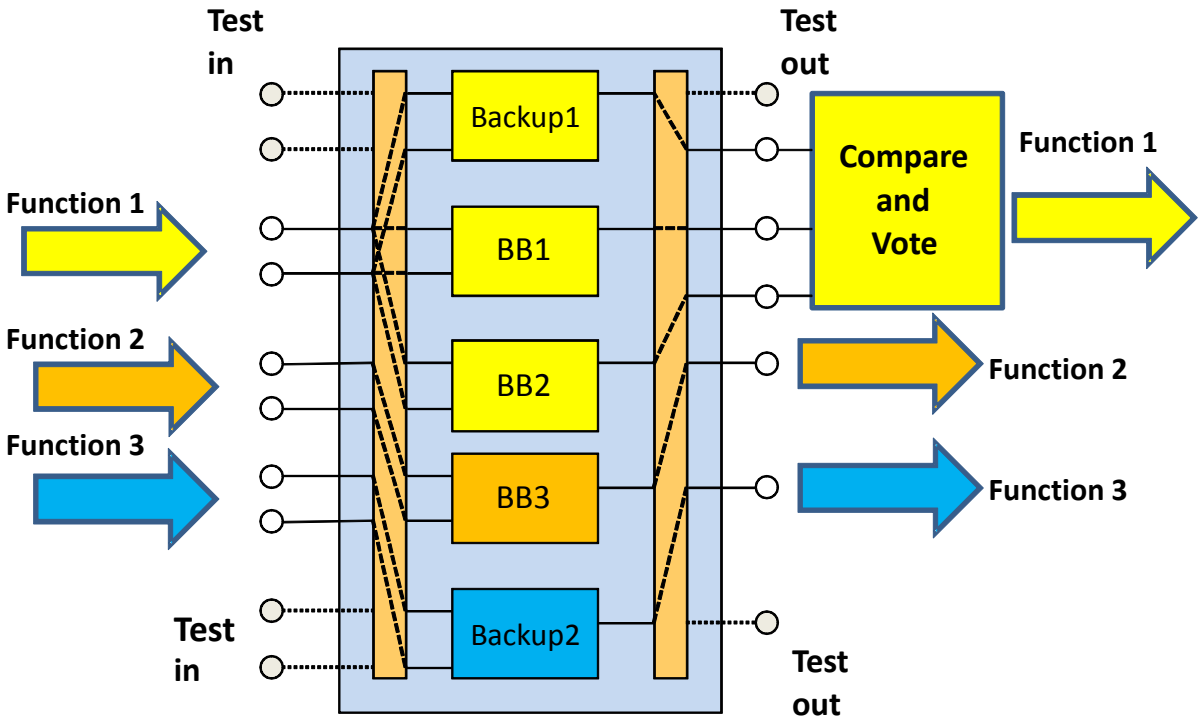
Con: High overhead, over TMR, and the repair function is untestable !

(n+2)-Architecture with On-Line-Error Detection



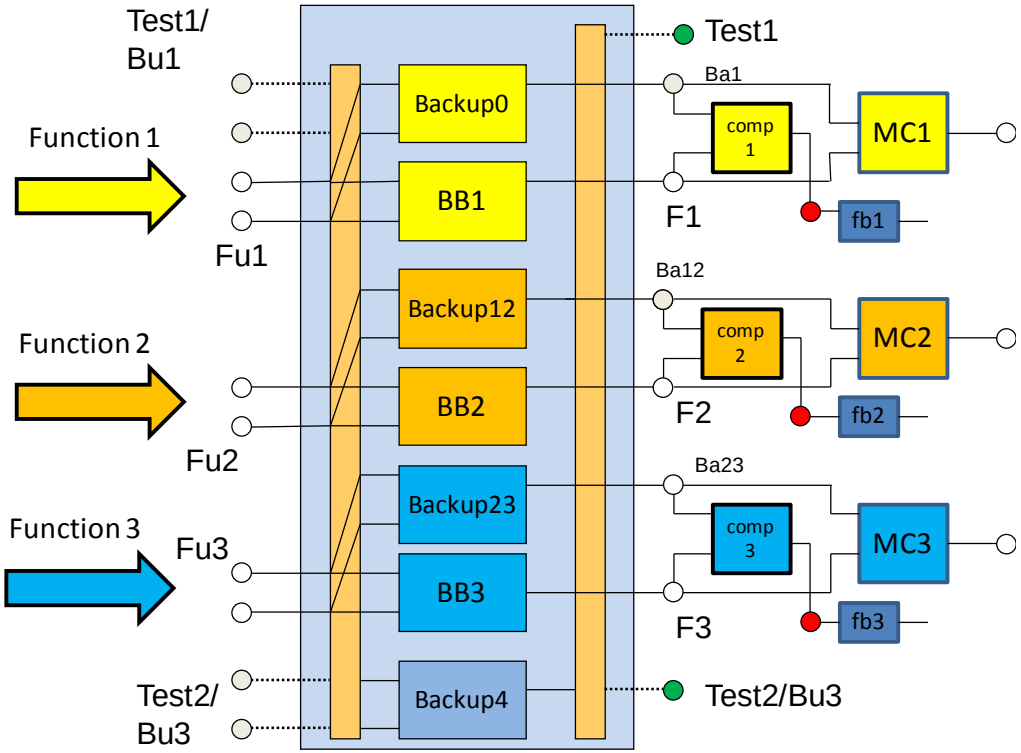
Existing redundany may be used hot / on-line for error detection !

(n+2)-Architecture with selective TMR



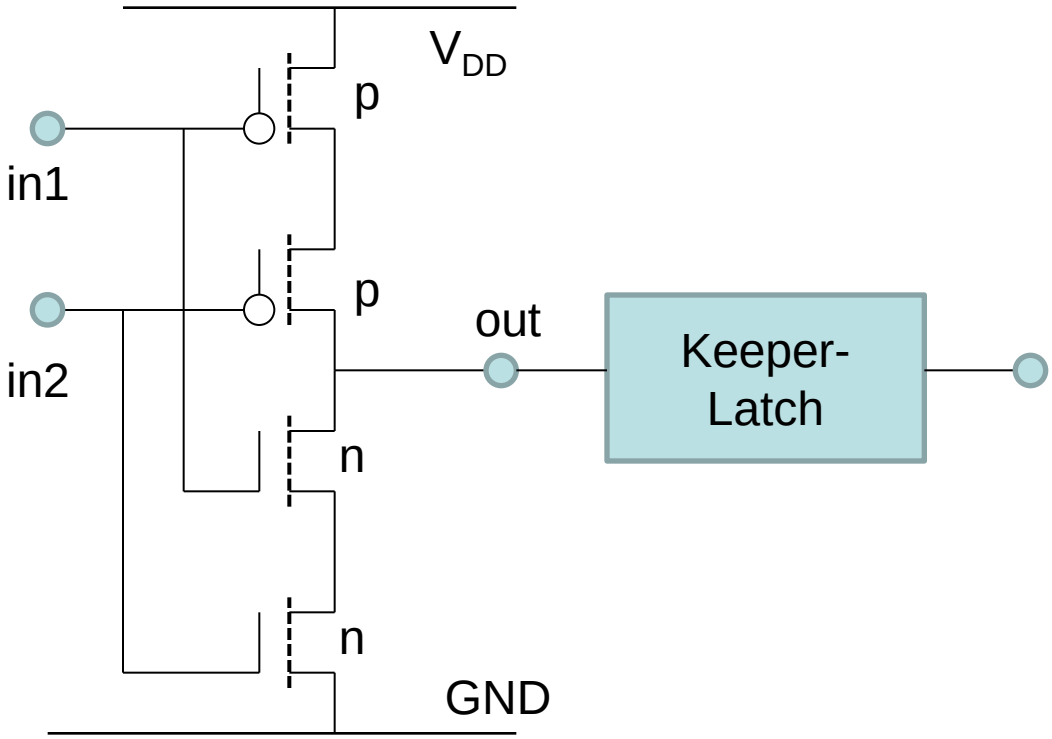
One out of 3 functions can tempararily be operated under „triple modular redundancy“ by borrowing a neighboring unit. Extra storage and voting units are needed !

(2n+1)-Architecture with Permanent Error Monitoring

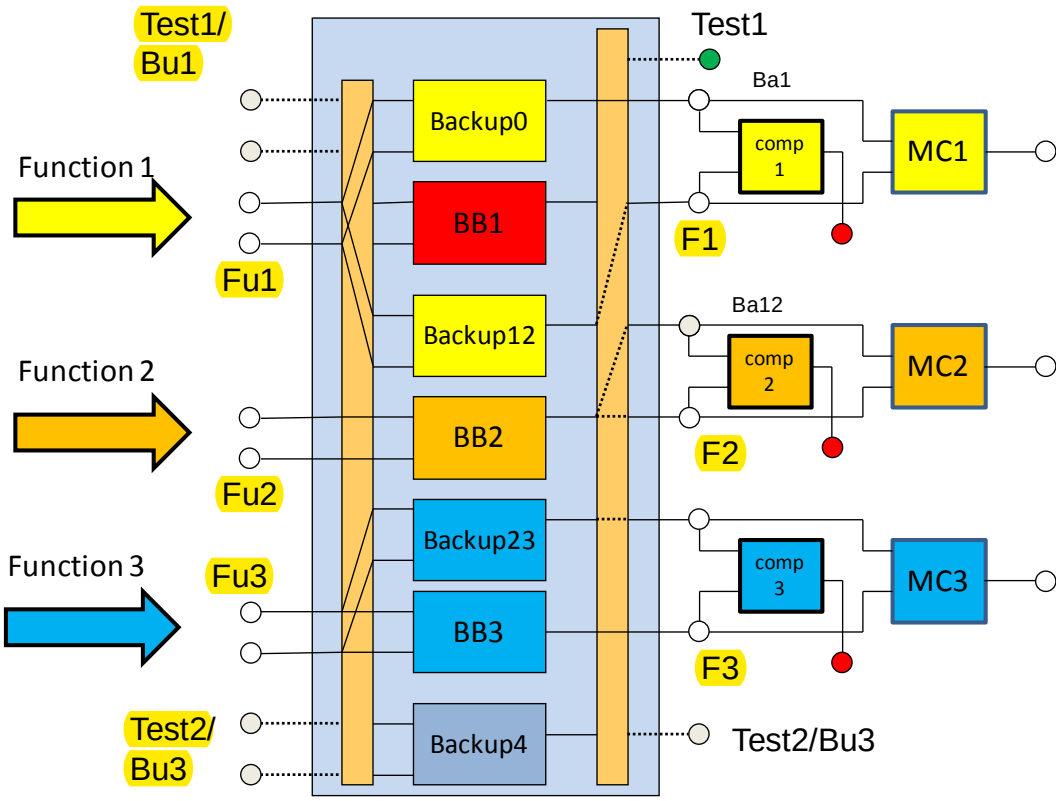


All units are pernamently monitored for errors by duplication.

Muller-C-Element

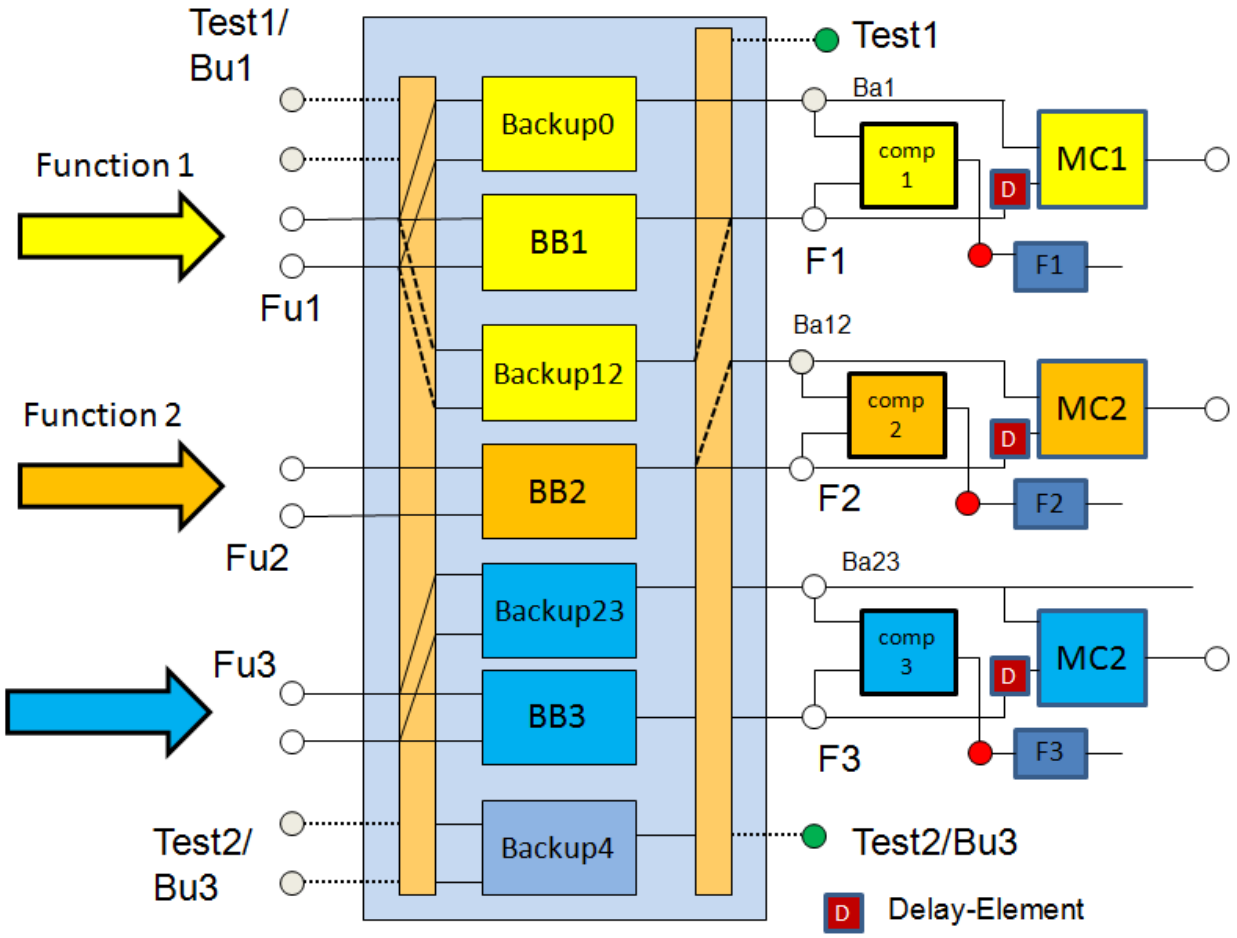


Selective TMR



Alternatives : a) parallel triplication and compare only after „error detect“,
b) permanent comparison in 3 successive clock cycles

TMR-Capable Architecture with Extensions to Detect / Filter Short-Term fault Events/ Glitches



Selective TMR

Pre-Condition: All functional blocks are duplicated and in active operation.

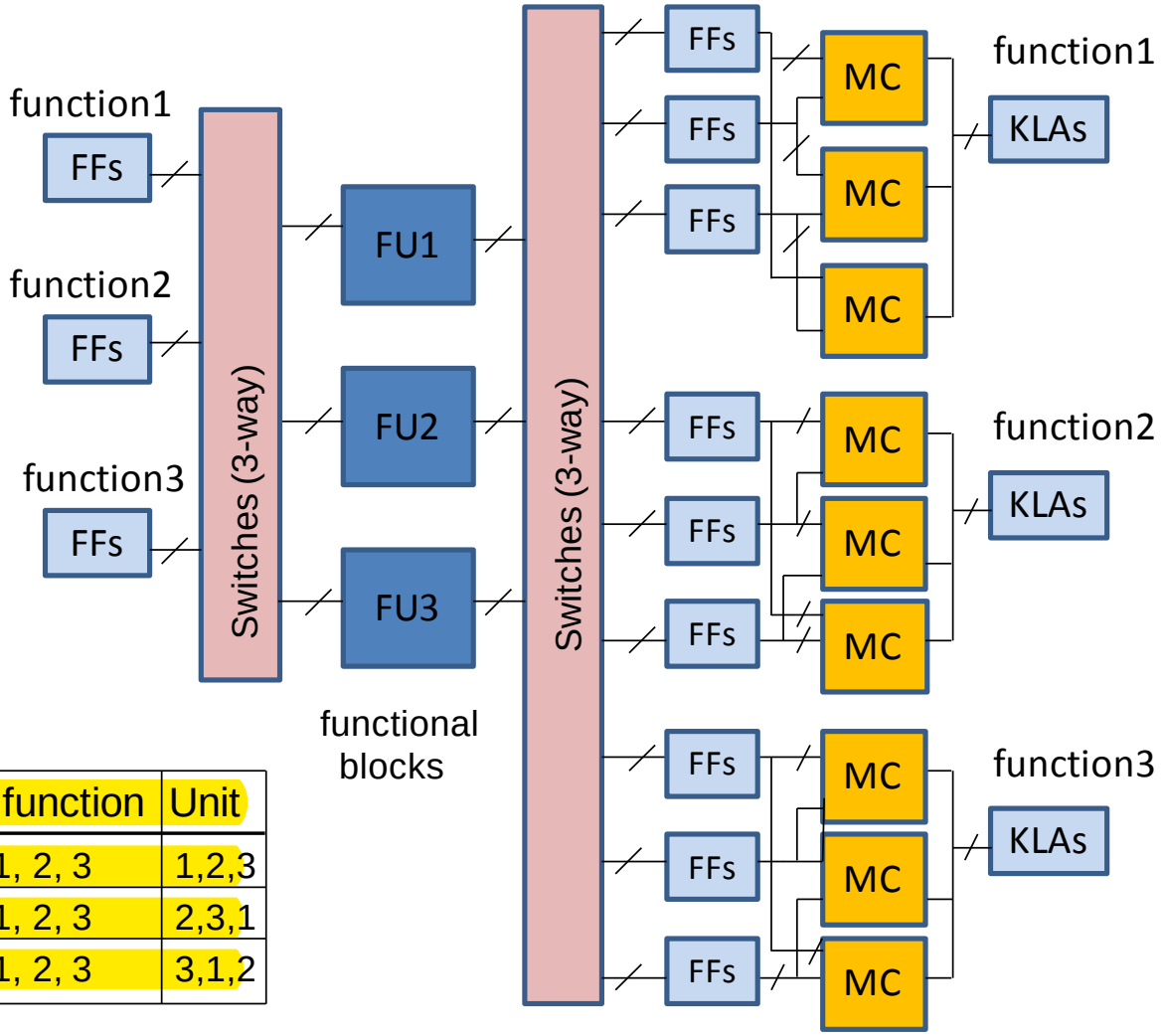
Muller-C-elements at outputs are used to pass only „equal“ outputs from both feeding functional blocks.

Action: The output comparator of any function can detect errors and perform switching to acquire a third unit for majority vote from the neighbors. Meanwhile, the neighboring unit that has donated its backup unit is without protection.

Pro: The method can detect and compensate transient and permanent fault effect withon 2-3 extra clock cycles.

Con : Sudden irregular delays in case of an error, double power, needs complex control if already „faulty“ units shall not be re-used..

Time-Distributed TMR



Clock	function	Unit
1	1, 2, 3	1,2,3
2	1, 2, 3	2,3,1
3	1, 2, 3	3,1,2

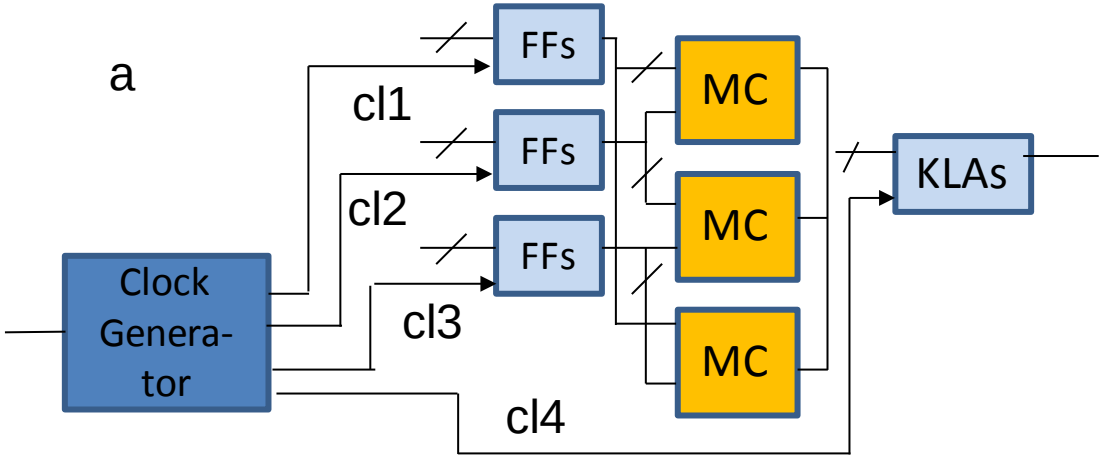
Time-Distributed TMR

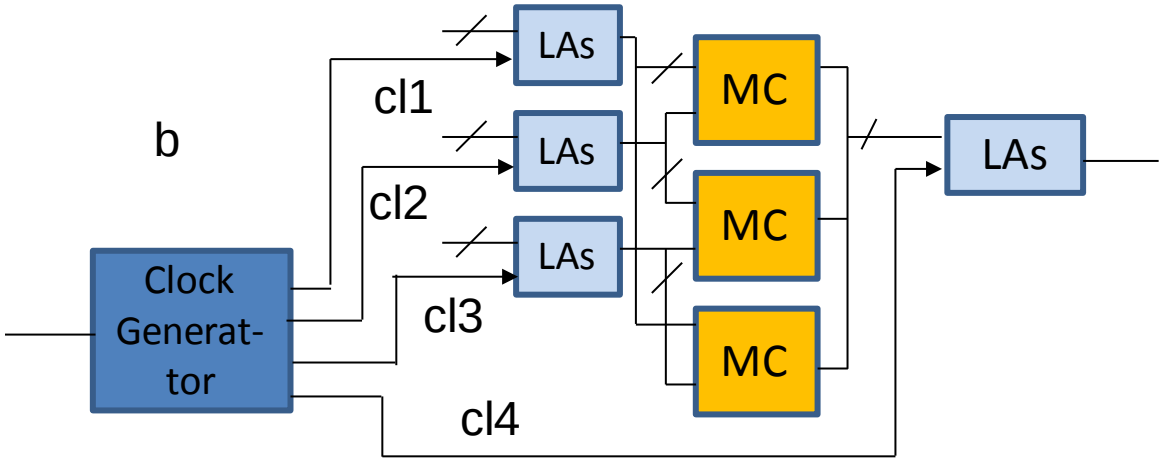
Scheme: We need at least 3 identical functional blocks, which are use to execute 3 parallel schemes flows of operations. Then each operation is executed 3 times, shifted between 3 functional blocks. Results are stored, compared, and a majority vote is taken.

Pro: Scheme can detect and eliminate all types of faults. No real triplicaton of hardware for fault tolerance. Regular timing scheme !

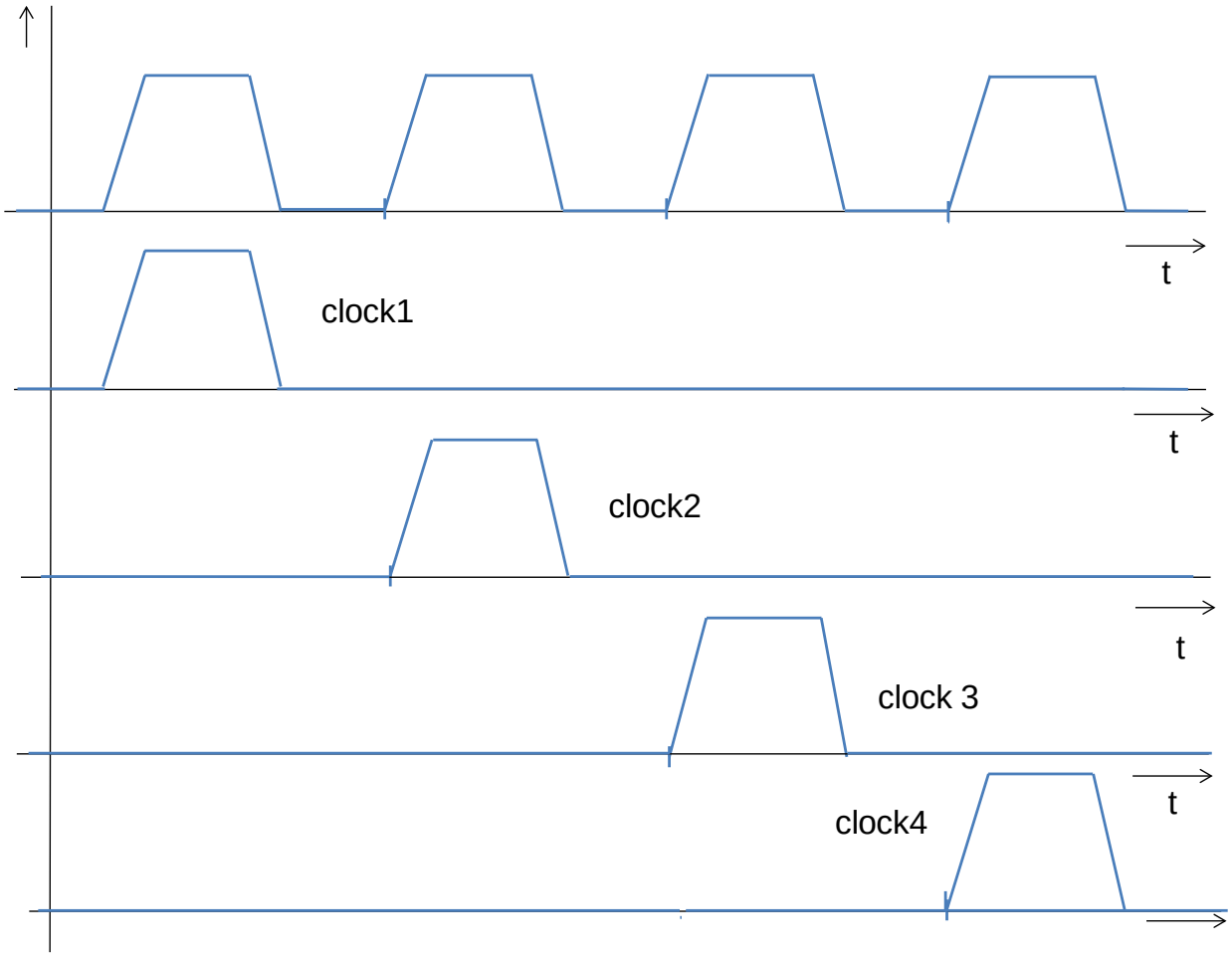
Con: Reduction in throughput by a factor of 3 versus normal TMR. Not triple power, but triple energy. Switching elements are now always active, which may limit their life time.

MC-Voter with Flip-Flops / Latches

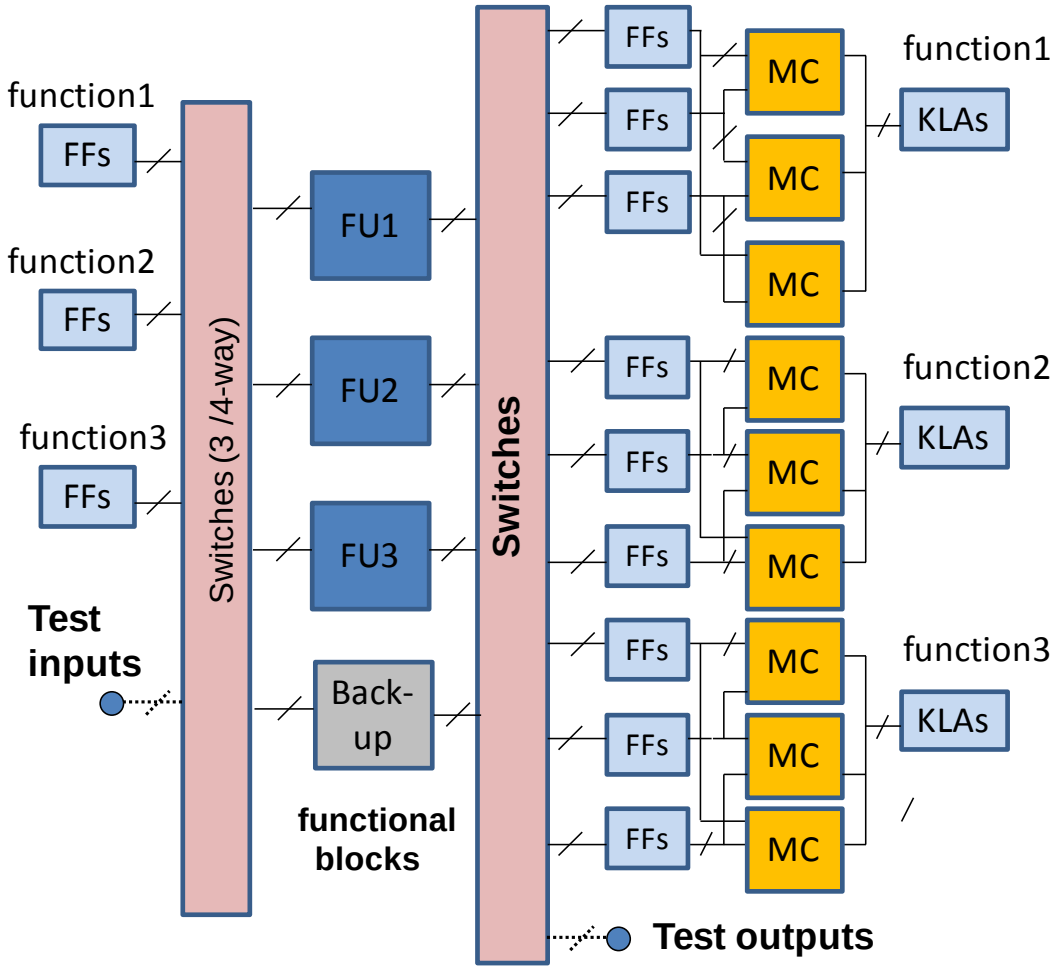




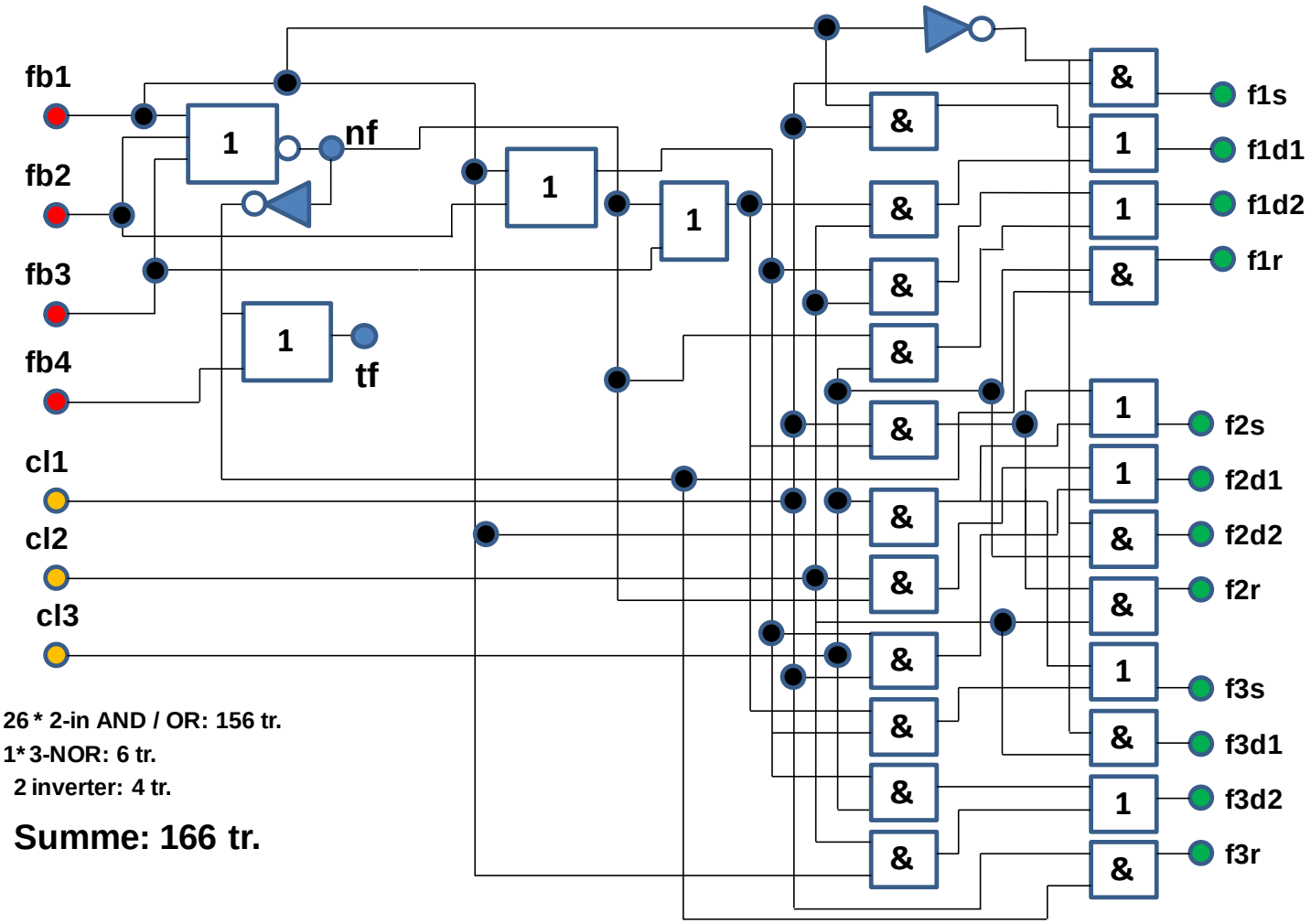
Clocking Scheme



Time-Distributed TMR with Repair Function



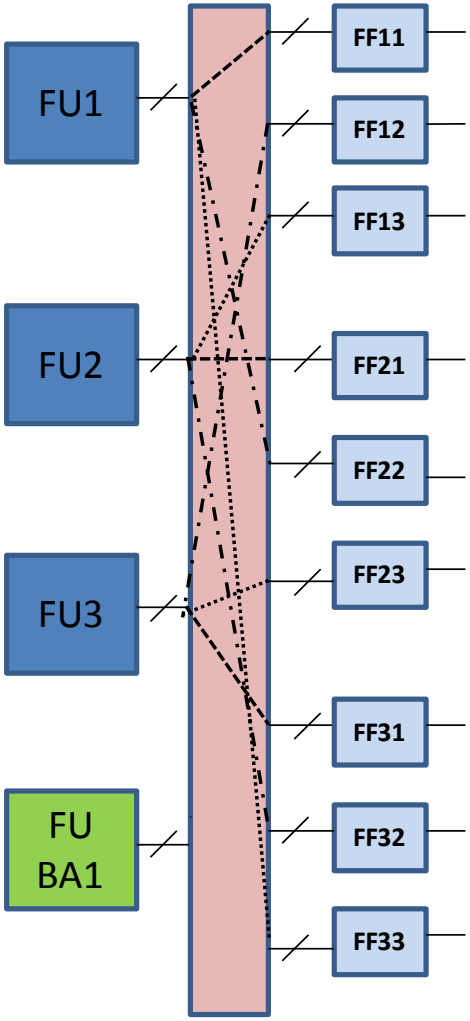
Control-Logic for Switches (Inputs)

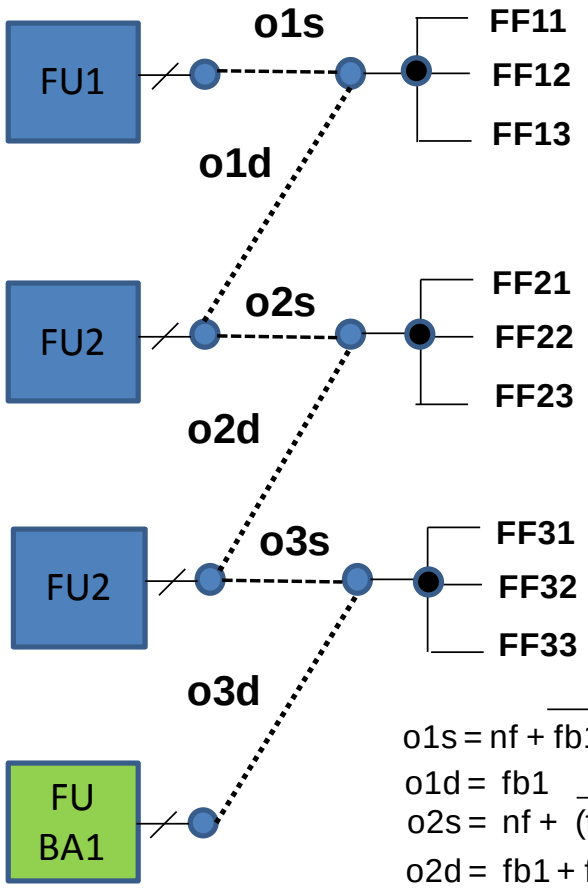


26 * 2-in AND / OR: 156 tr.
1* 3-NOR: 6 tr.
2 inverter: 4 tr.

Summe: 166 tr.

Output Switches





$o1s = \overline{nf} + fb1$

$o1d = fb1$

$o2s = nf + \overline{(fb1 * fb2)}$

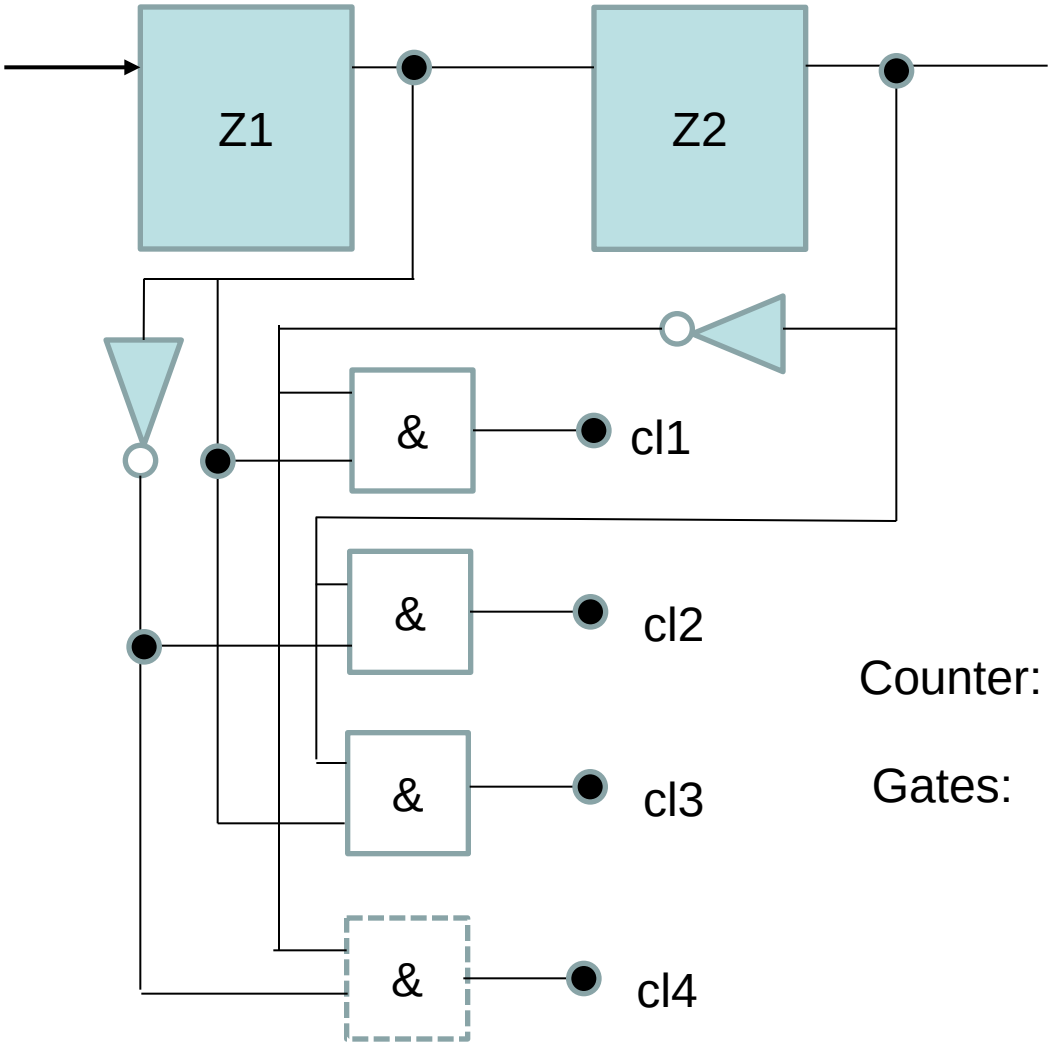
$o2d = fb1 + fb2$

$o3s = nf$

$o3d = fb1 + fb2 + fb3$

Transistoren: 32

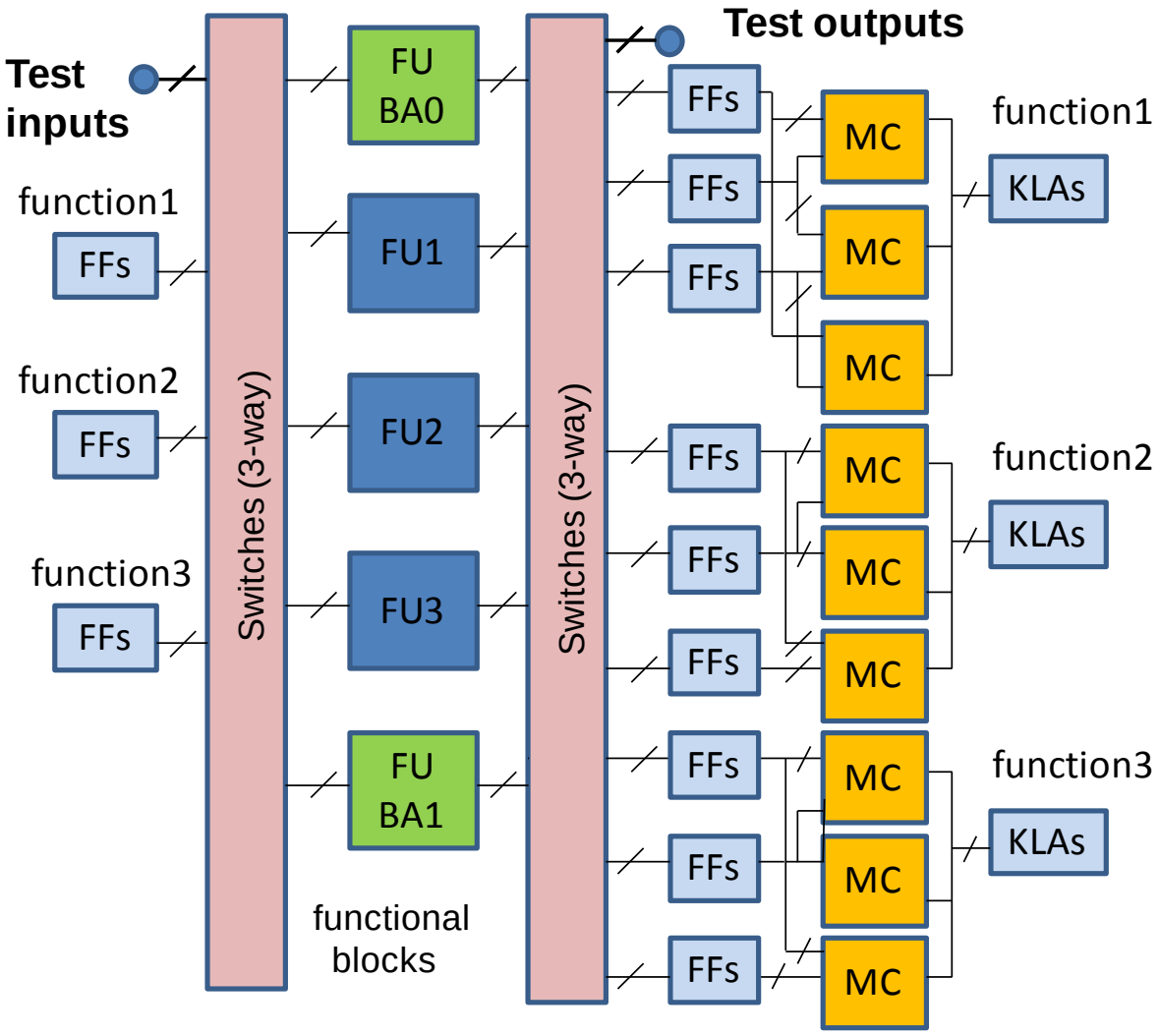
Time Control



Counter: 20 transistors

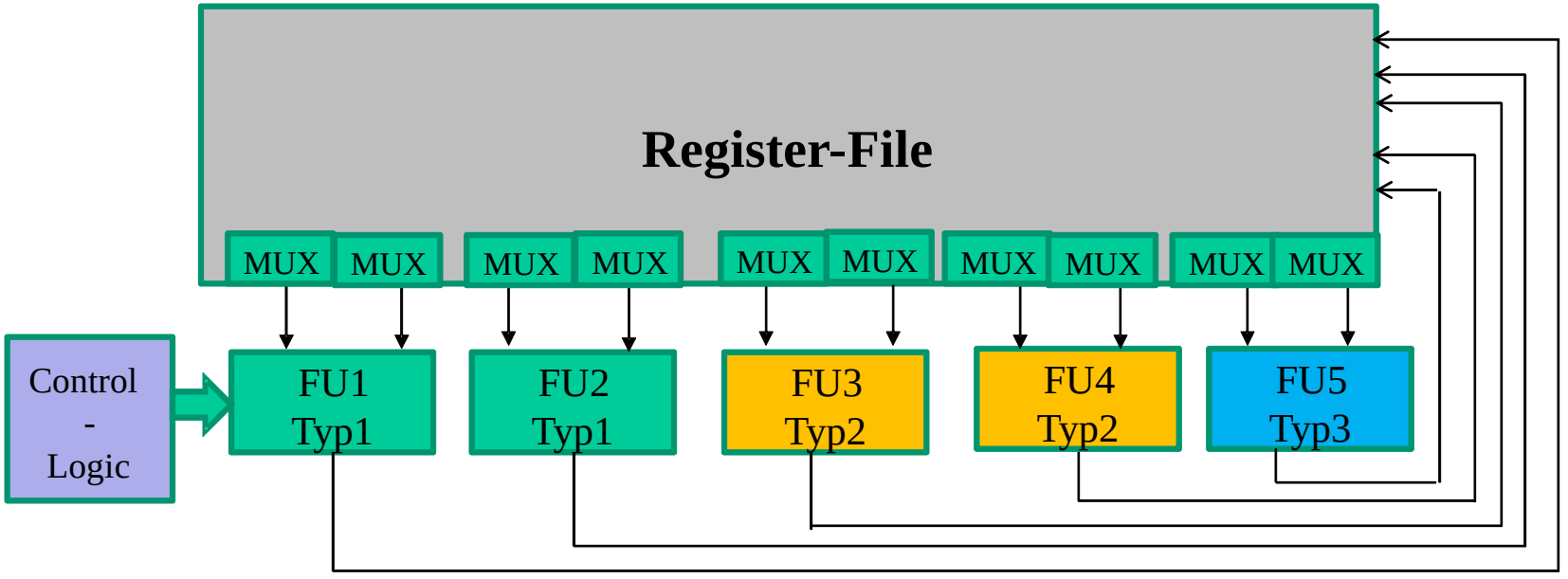
Gates: 28 transistors

BIST-capable Version



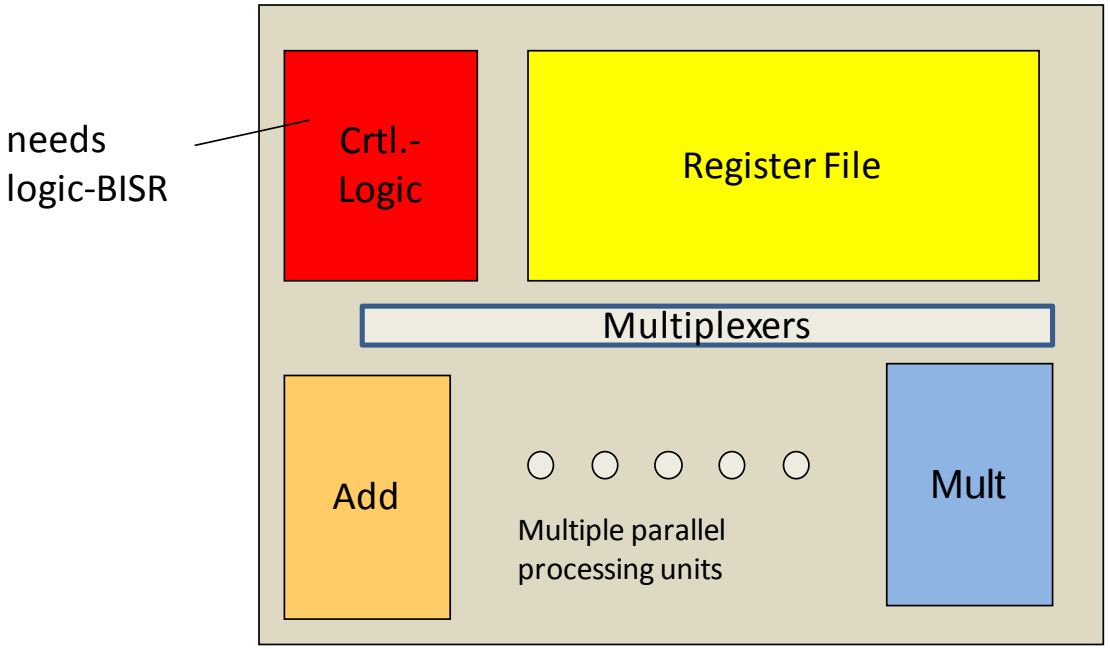
Hardware / Software- based Repair Functions

Requires a regularly structured processor architecture with 2 or more functional blocks (add, shift, sub, mult) of any type.



Regular VLIW-Processors

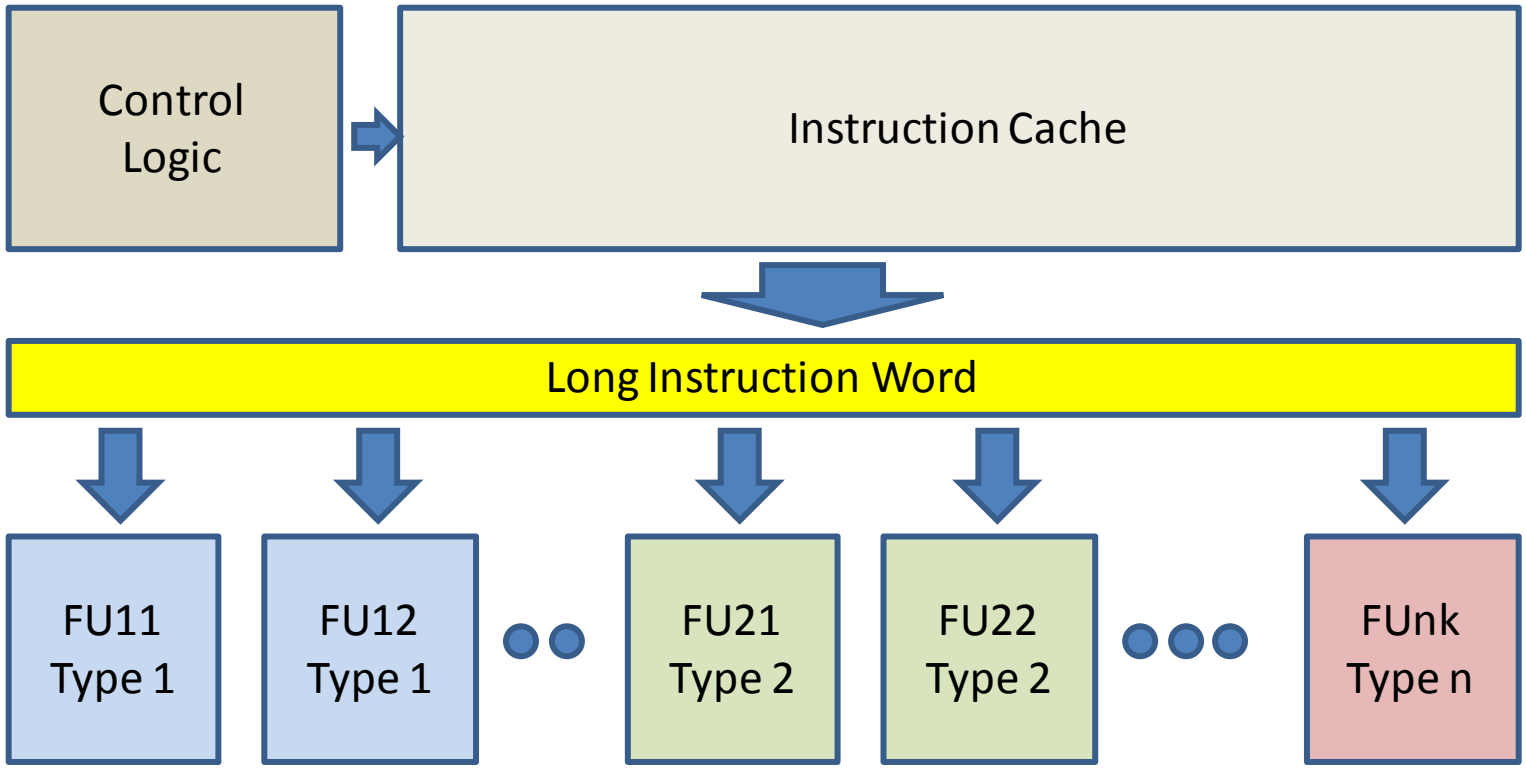
Structure of a regular multi-slot-processor
(e. g. Very Long Instruction Word Processor, VLIW)



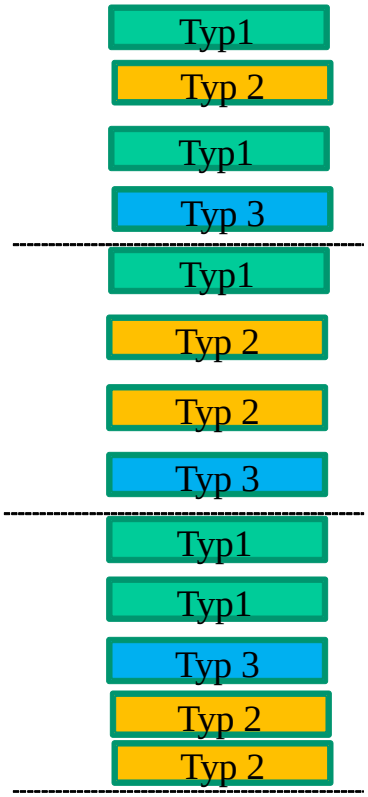
- Small control logic, execution of instructions scheduled by compiler
- Multiple (2 or more) identical execution units of several types accessible via multiplexers

VLIW Regularity

They contain several building blocks (slots) of equal type and structure such as adders, shifters, registers, etc.



VLIW-Prozessor with fixed „Schedule“



Schedule:

Typ1	Typ1	Typ 2	Typ 2	Typ 3
Typ1	Typ 1	Typ 2	Typ 2	Typ 3
Typ1	Typ1	Typ 2	Typ 2	Typ 3

The scheduling of operations on multiple parallel execution units is done by the compiler off- line and in advance.
It cannot be modified at run-time.

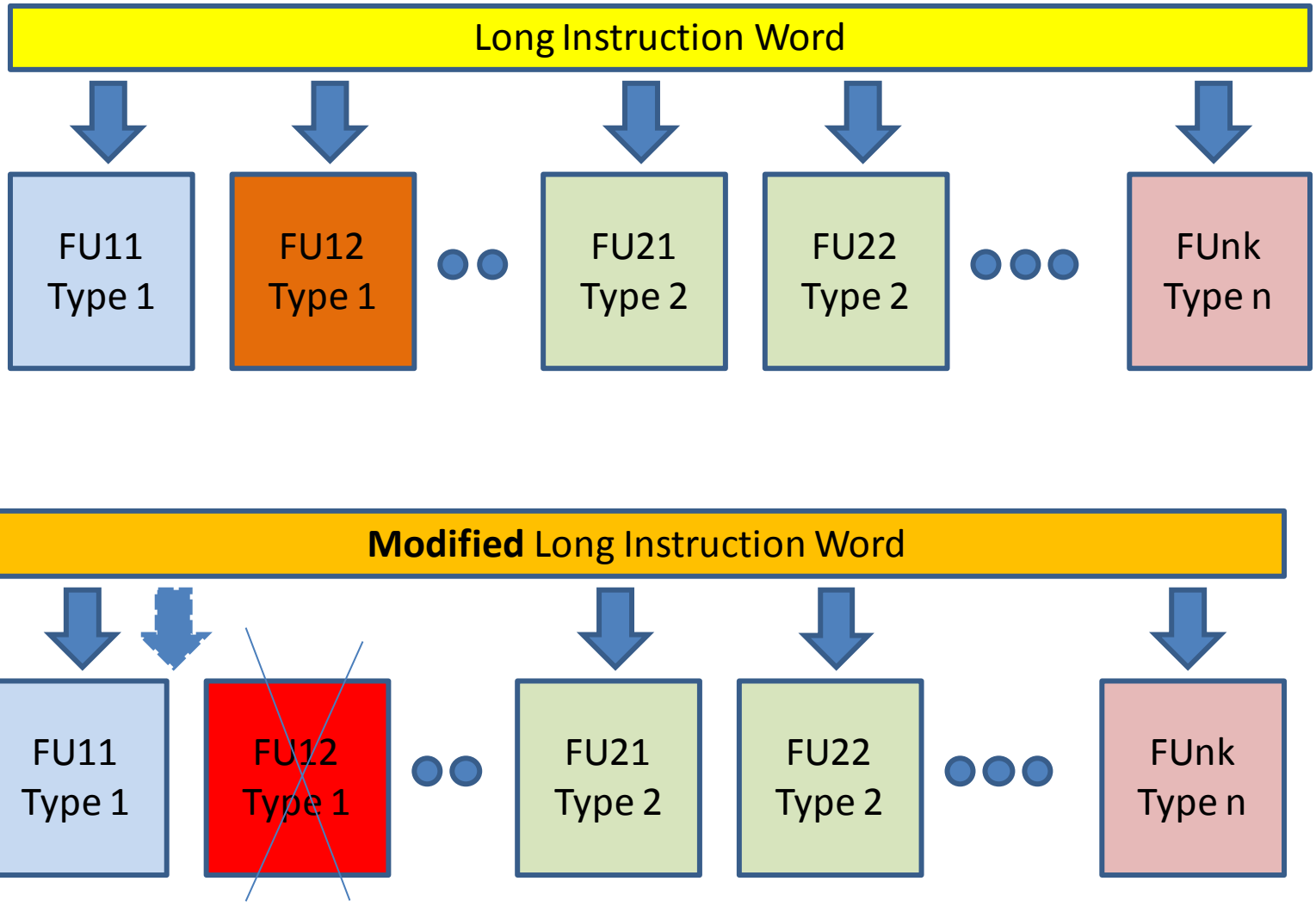
Repair in VLIW-Processors

1. Error detection (code, duplication, etc.)
2. (Self) -Test with fault diagnosis (off-line)
3. Computation of a new schedule which excludes the faulty unit(s).
4. Modified user code is stored for execution

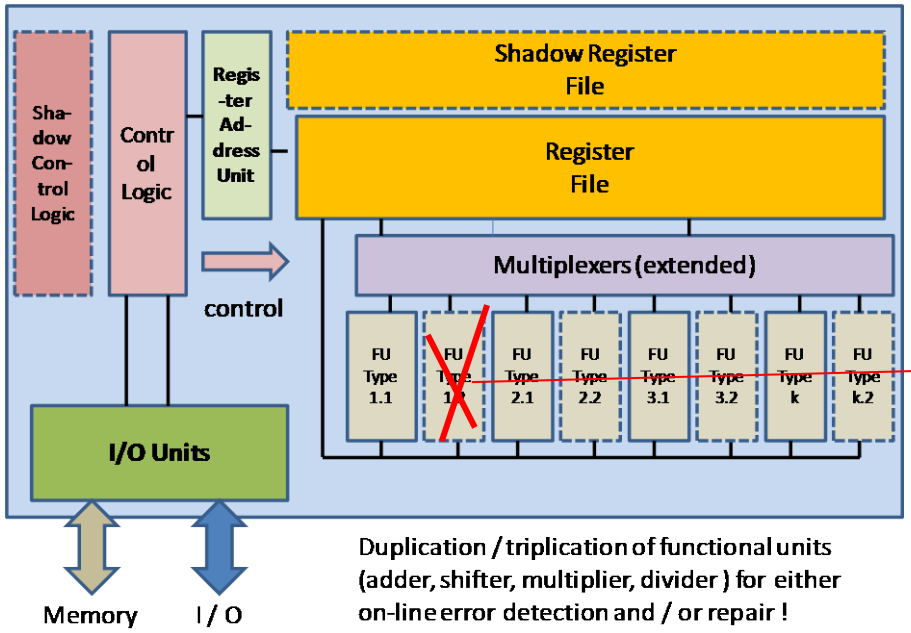
Problems: Who perfoms test and diagnosis ?
Who will compute the new schedule ?
Who controls the re-organisation process?
AND: This wouold be self-modifying code !

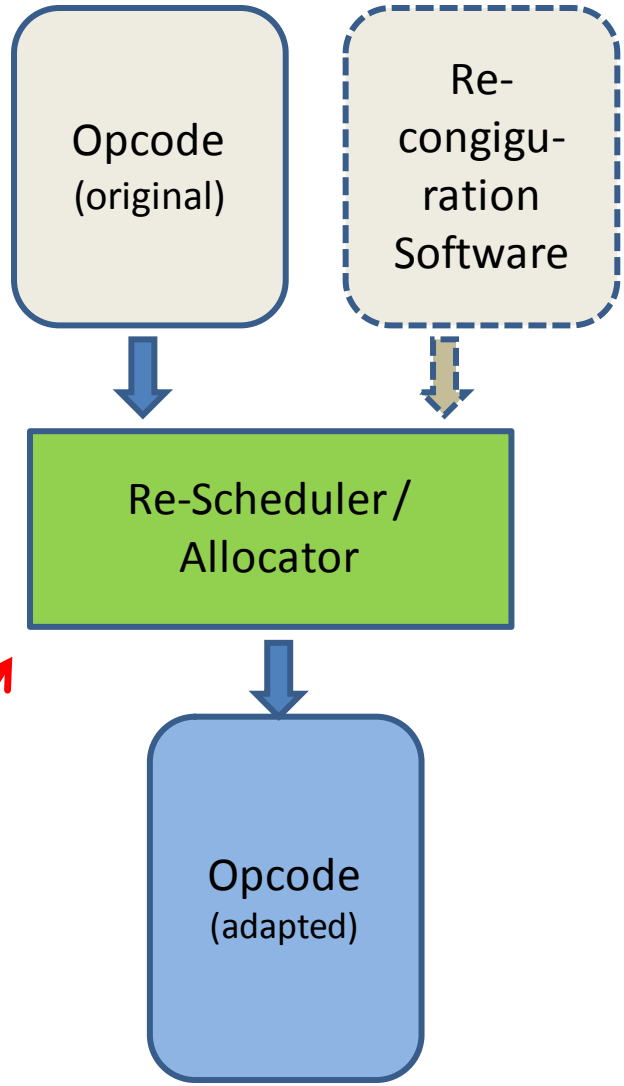
Pros : - Graceful degrdation of performance instead of total failure
- Potentially low hardware overhead !

Repair by Re-Configuration



Repair-Procedure





Re-scheduler / allocator is software. It exists in several versions, each around a known functional FU. Or it runs on an external „auxiliary“ processor.

Repair-Process

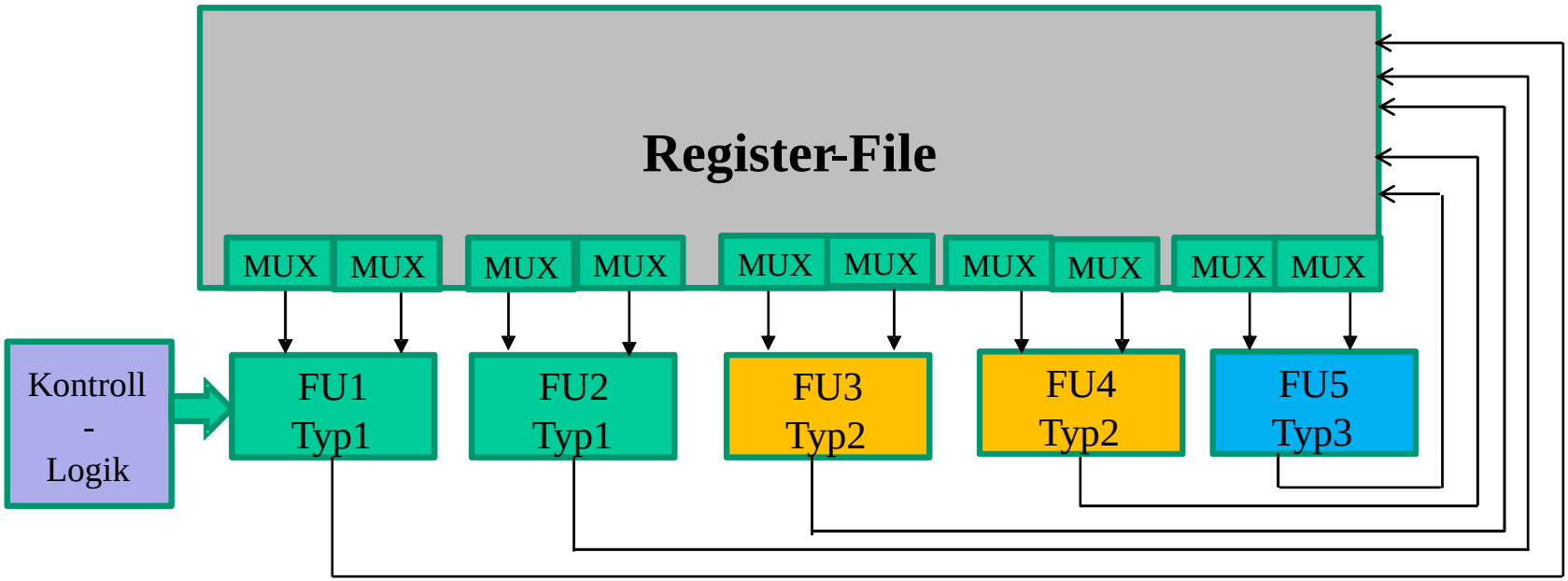
1. Diagnostic (software-based) self test with identification of faulty:
   - registers,
   - functional units of partial functions of FUs,
   - multiplexors,
   - forwarding units.

2. The faulty processor itself, using only the ramaining functional units, or an external (auxiliary) processor runs a program of re-scheduling and re-allocation to create the sequence of long instruction words that may run on the „crippled“ processor.

3. The user program can be re-executed, typically with a loss of 10-30 % in performance.

- Relatively small overhead in hardware (10-15%) for self repair
- Re-organisation only off-line, may take minutes !
- Multiple parallel FUs may also be used to operate in a mode of parallel execution for on-line test !

Fine-granular Fault Diagnosis

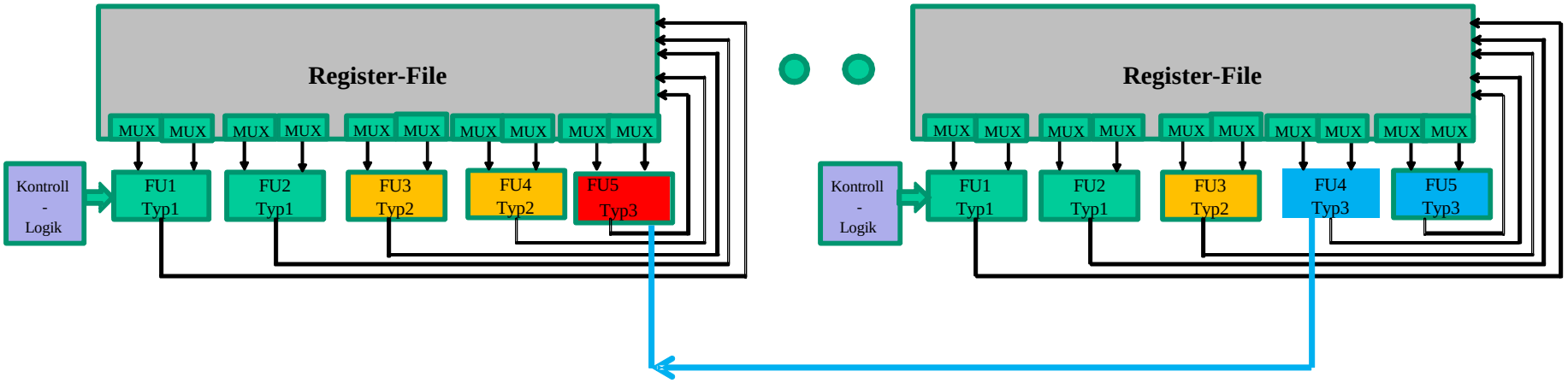


Reparatur-functions are more effective and economical, if faulty functional units can be used with remaining functionality.

- Defect register
- Defect paths on Multipolexors
- Defect partial functions on FUs

High-resolution software-based self test is needed !

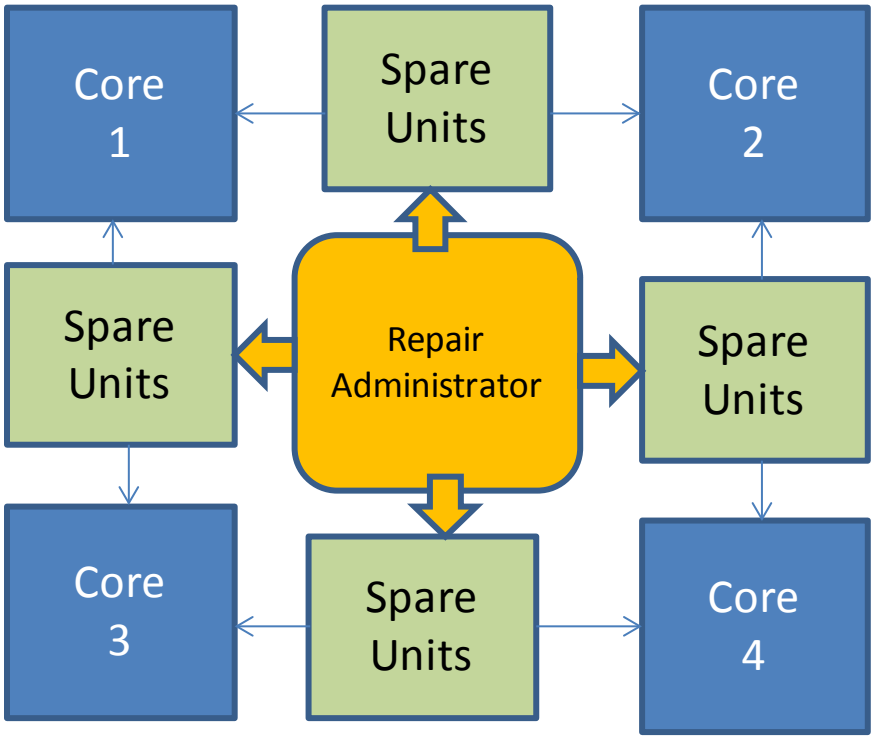
Multi Processor Systems



The system contains several parallel CPUs, each built with the same type of functional units, but possibly different numbers of FUs for each CPU.

Now we can establish a scheme where FUs may be re-distributed upon demand between several CPUs. Also a „common backup store“ that can serve several CPUs alternatively is possible.

Multi-Core Self Repair



Spare units for self-repair can be allocated to different processor units in a processor cluster „upon demand“.