

# **Dependability and Fault Tolerance**

in micro- and nano-electronic circuits and systems

## **Introduction**

H. T. Vierhaus

BTU Cottbus-Senftenberg

Computer Engineering

Winter Semester 2018 / 2019

# Outline

1. Everyday Experiences
2. Microelectronics Evolution
3. Errors and Dependability
4. Fault Tolerance and Self Repair
5. Re-configurable Systems
6. Outlook

# 1. Everyday Experiences

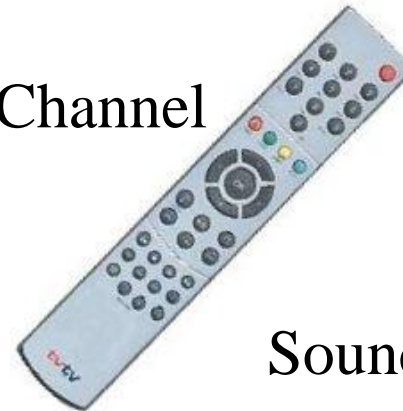
# How to Control a TV Set



Channel



Sound



Image



# Cars



No electronics,  
but operationable!



With up to about 120 embedded  
computers in control units a car  
of today is a multi-processor network !

.. but without electronics there is no  
emission control, no ABS (break control),  
no navigation system.



# Automotive Electronics



....in Audi- Advertisement

Not possible without electronic traction control !!

# Text Processing



Typewriter  
(ca. 1970).

*mechanical,  
no SW*

Muti-Media- PC (2010)



PC-System  
(1985)

100 000 Trans.  
10 MByte SW



10 000 000 Trans.  
10 000 MByte SW

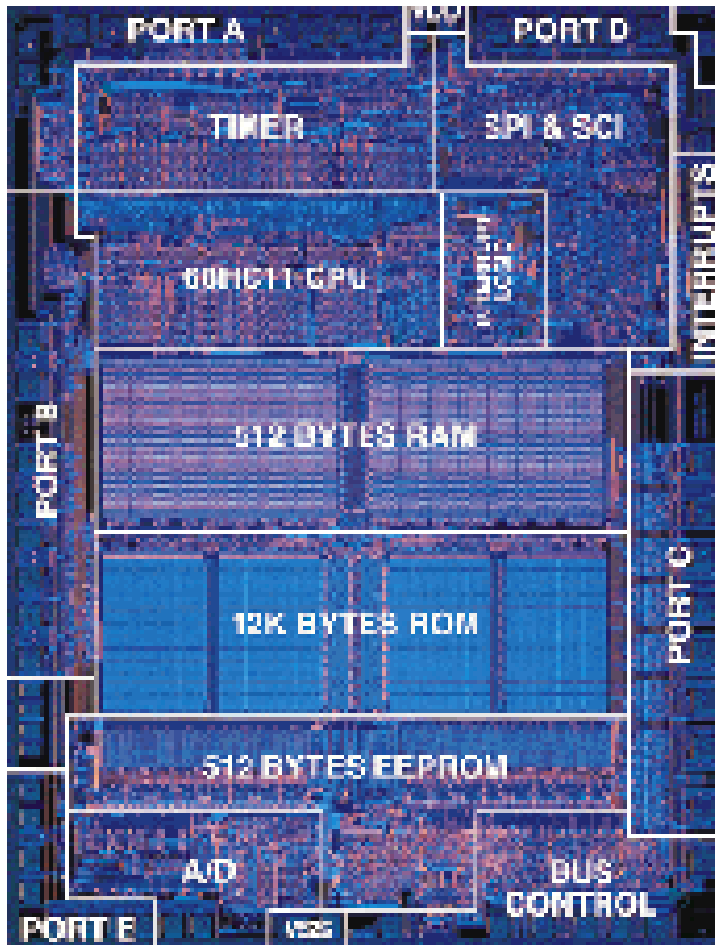
# General Tendency

- For a large variety of technical systems, complex functions are not possible without embedded micro-electronic devices, processors, memory, **Software**.
- Tools, appliances, devices of everyday use are becoming increasingly more complex with respect to:
  - structure
  - handling
  - maintenance / service
  - reliability in operation.

**Where does complexity come from?? For what purpose??**



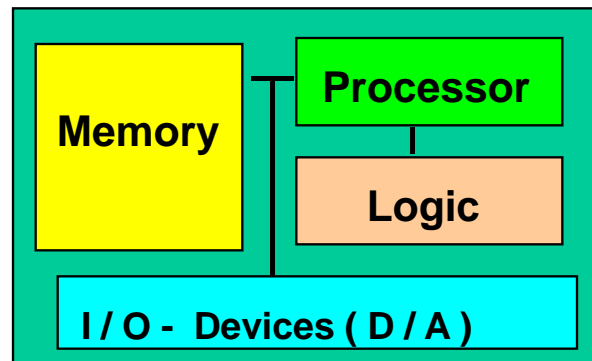
## 2. Microelectronics as the Driver



Large-scale integrated circuits with embedded processors make „Systems on a Chip“ (SoCs)



# Embedded Electronics



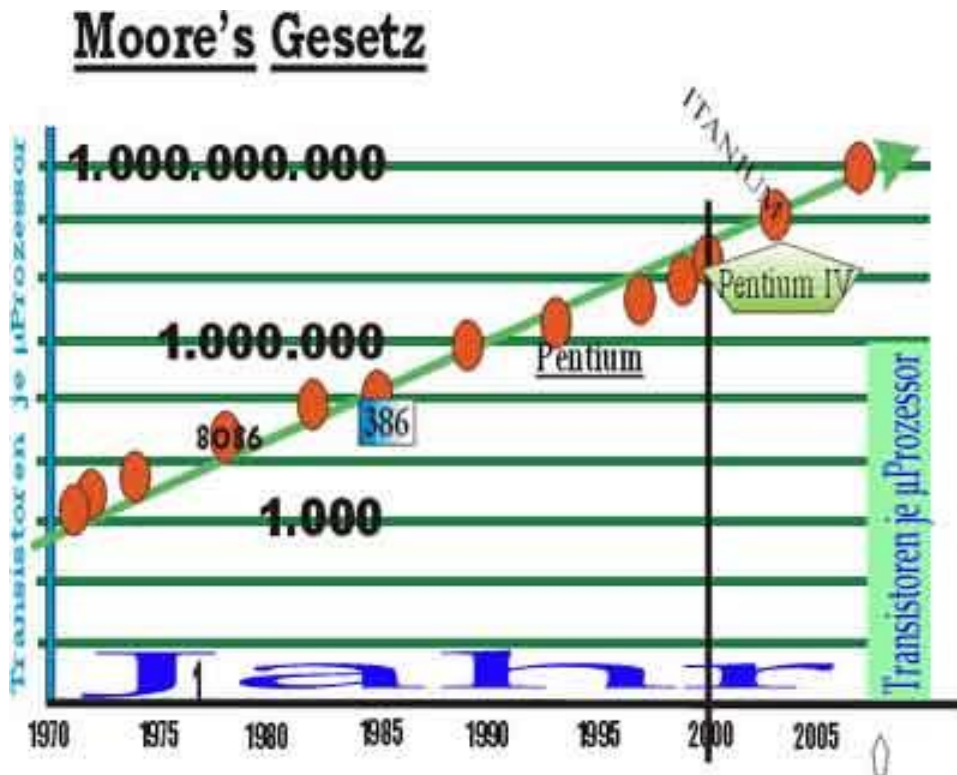
An embedded system consists of one or more processors, memory blocks (RAM, ROM), I / O devices. Typically it operates on fixed pre-defined software, typically

- in cars (motor control, break control, traction control, navigation)
- in PCs (keyboard, display, printer)
- in household appliances (washing machine, dish washer)

# Evolution of Microelectronics

Year	Min. Struct. in nm	Trans. per cm <sup>2</sup>	Clock frequ. MHz	Pins	Metal- Layers
1970	20 000	1000	1	16	1
1975	10 000	10 000	5	28	1
1980	5 000	20 000	10	50	1
1985	1 000	100 000	20	68	2
1990	500	1 000 000	50	200	3
1995	300	5 000 000	200	350	4
2000	130	50 000 000	1000	500	6
2005	65	200 000 000	3000	800	8
2010	30	1 000 000 000	4000	1000	10
2015	14	10 000 000 000	4000	15000	12

# Complexity in Microelectronics



The number of transistors on the same chip area duplicates about every 18 months !

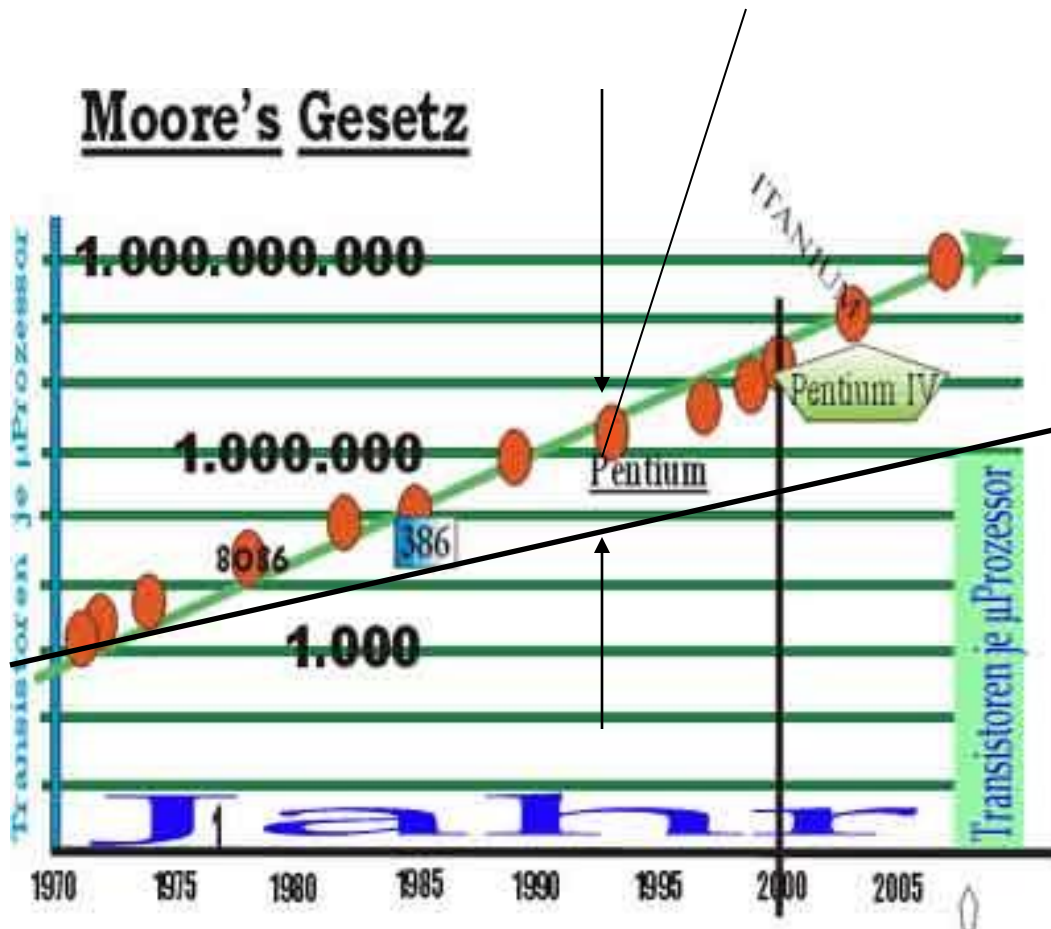
... and there must be somebody to use it and pay for it !!



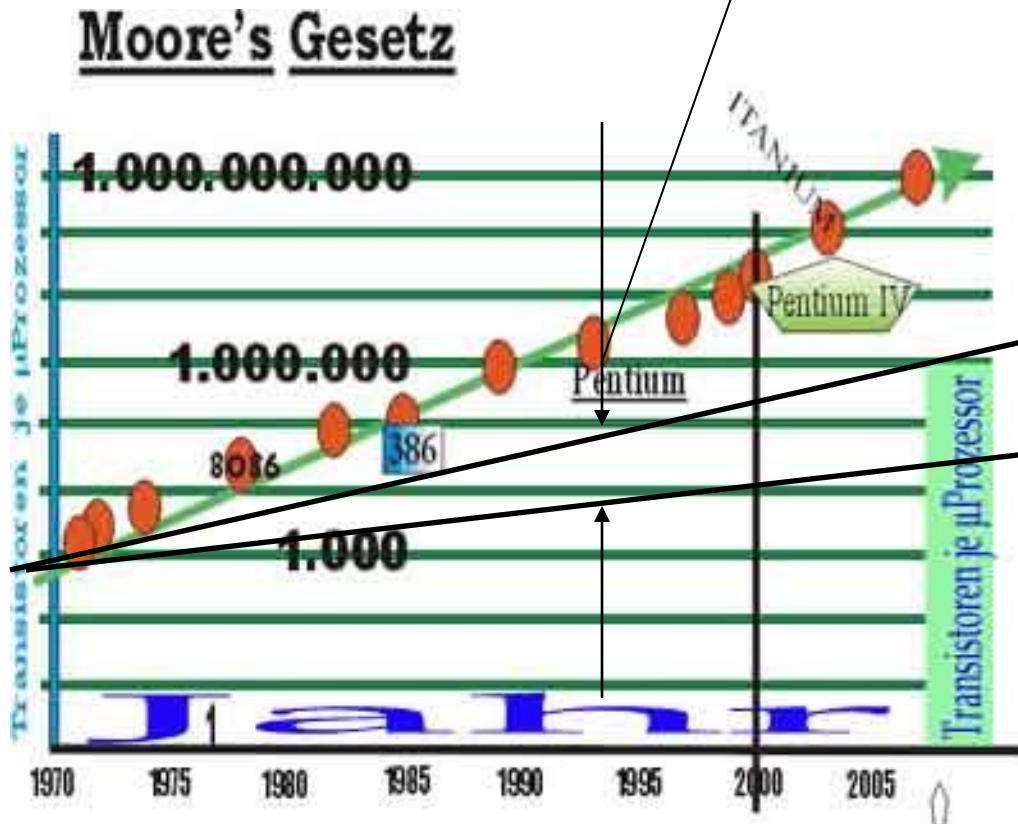
# Fundamental Wisdom About Microelectronics

- Many complex functions in cars, tools, airplanes, manufacturing etc. are not possible without microelectronics and software.
- Complex ICs can economically be produced only in large volumes. For nano-circuits, the volume is in multi-millions. „Old“ technologies may allow production volumes in hundreds of thousands.
- Mobile phones are the „technology driver“. Manufacturers compete for the lowest price per transistor.
- Aspects of reliability and operational life time are not a strong issue in mobile phones !! They become dominating if nano-technologies have to be used for long-living systems in safety-critical applications !

# The „Design Gap“



# The Validation Gap



The number of transistors whose correct design a person can validate per day is even much lower !

## Was Out :

Software?? **No!!**

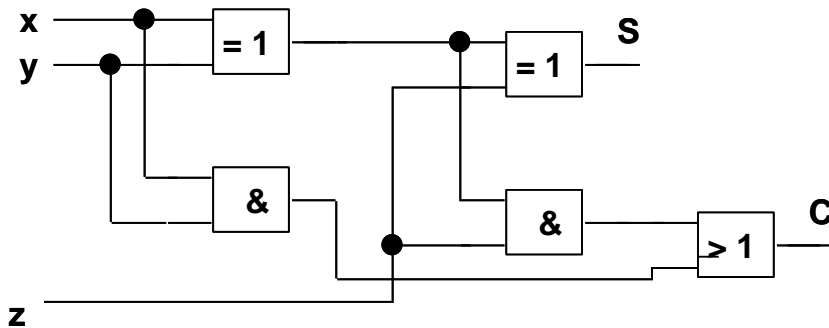
*Digital hardware people can partly „prove“ design correctness, SW people can not !*

# Microelectronics

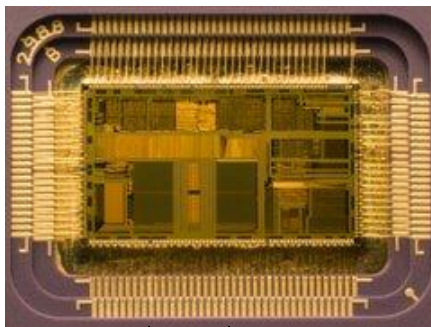
- Microelectronics is the driver of complexity.
- Complex functions are impossible to implement without embedded processors and software.
- The transition from hardware to software-based functions typically means a jump in complexity by factor of 100 and more..
- Formal proof a software correctness is not possible for real-life systems.
- But for test and error handling we need to go to hardware details !



# How to Design and Build an Adder



From pure Hardware:  
ca. 100 – 1 000 Transistors



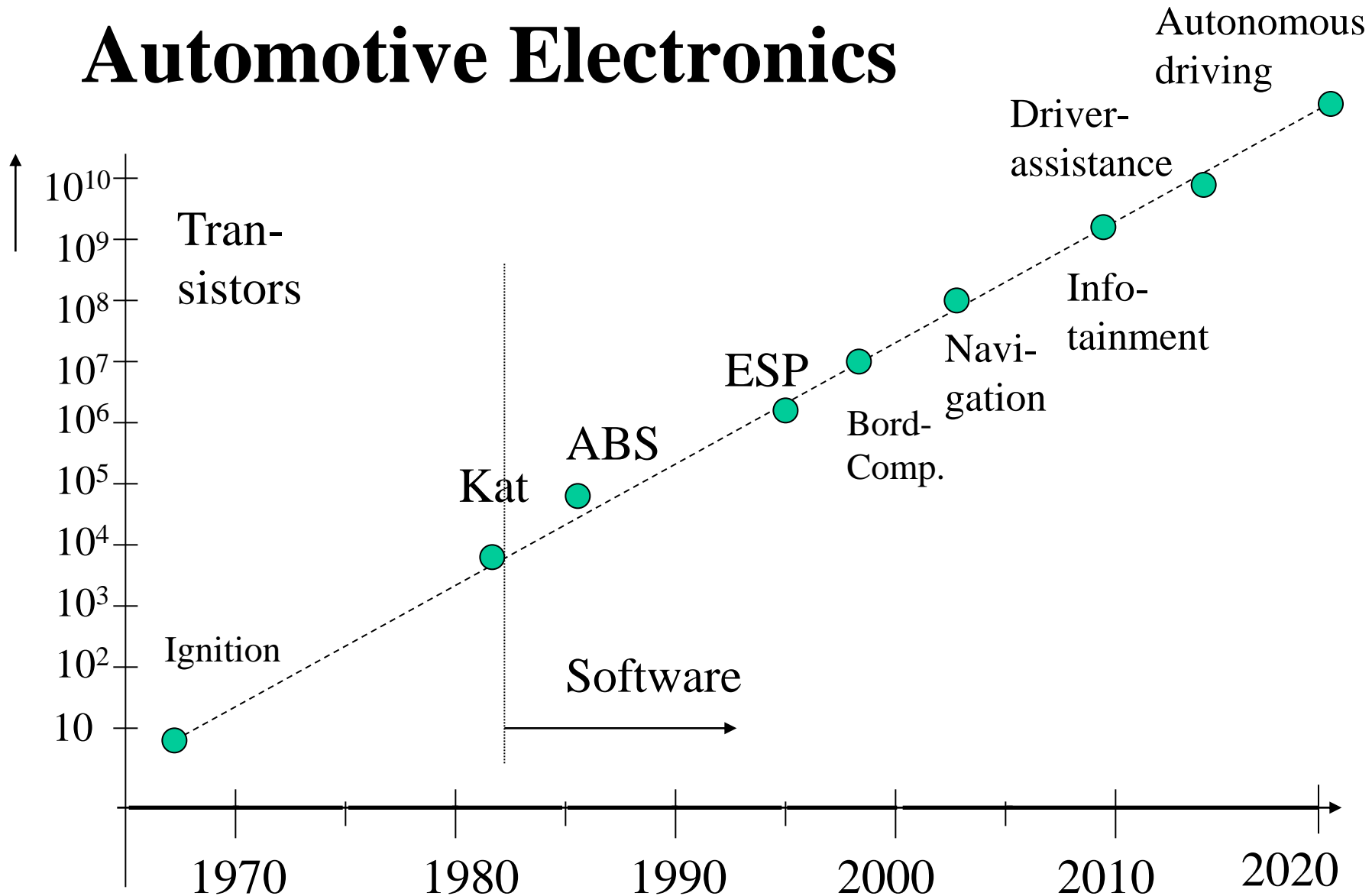
Memory

From Processor + Software:  
ca. 30 000 bis 100 000  
Transistors!

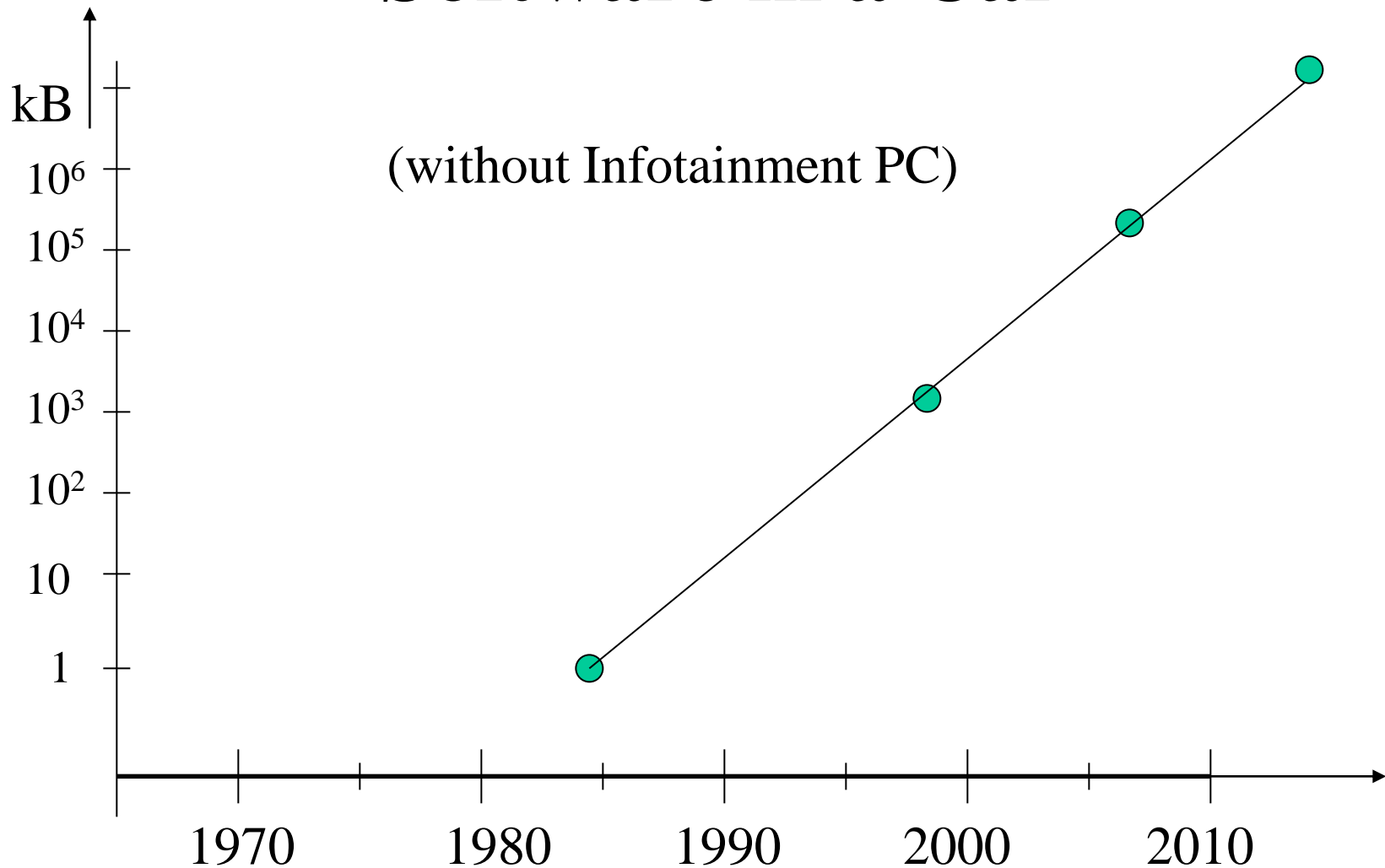


Typically the transition of a function from  
hardware to software means \*100 in complexity!

# Automotive Electronics



# Software in a Car



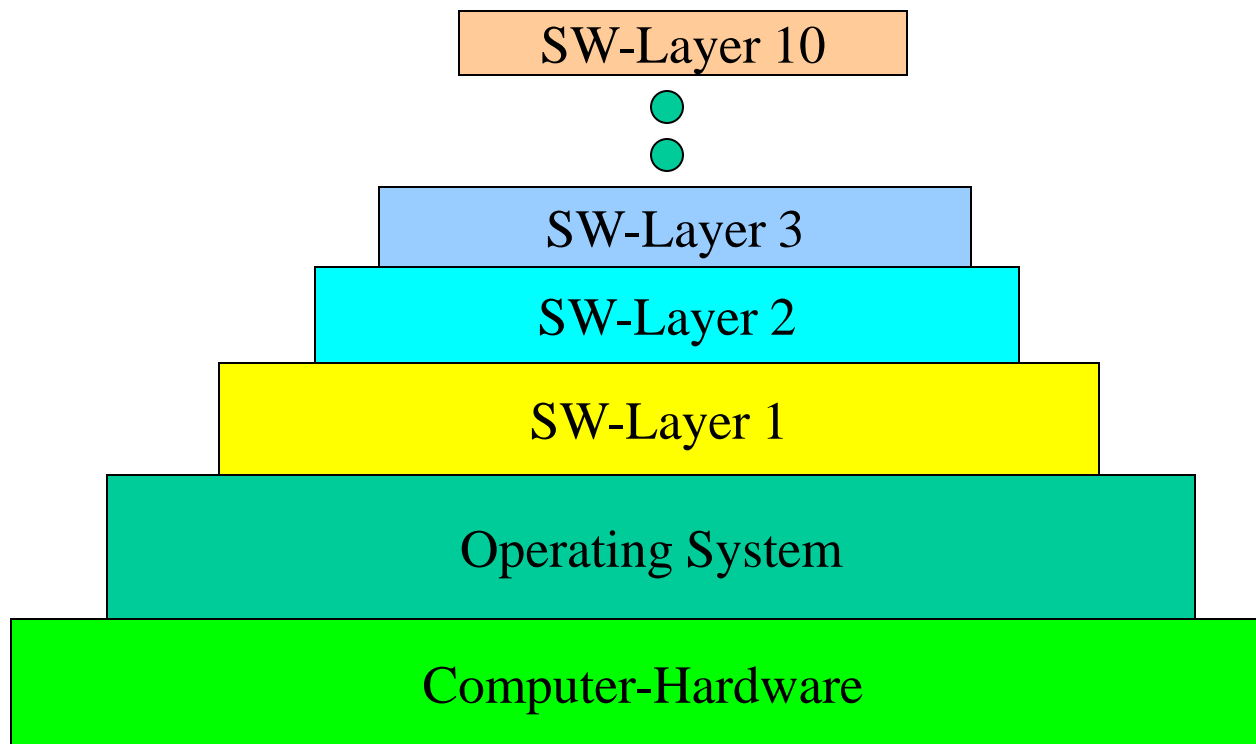
# Audi again....



Computation of tractive control may need several processors and a lot of software !



# Software Complexity Organisation



... and every „higher“ level assumes that the lower levels work correctly !!

# Migrating to Software

- As ICs in advanced technologies can be produced economically only in volumes of millions, few types of ICs and functional implementation by software becomes the dominating tendency.
- System design is often based on specification, automatic software generation and final compilation.
- As system design tries to avoid problems of hardware design, it inherits all types of software !! **SECURITY** !!
- Some application areas such as automotive electronics and avionics have strict standards and regulations which cannot be met by „smart phone“-oriented nano-technologies.

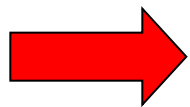
# 3. Faults, Errors and Reliability

Complexity as such is not a problem, as long as you can control it.

But complexity also results in new sources of errors ....

# Defects / Faults /Errors in Cars

- until 1975: Mechanics, hydraulics, electrics (battery!!)
- since 1975: Electronics, sensors
- since 1990: Software
- since ca. 2000: ICs, Interconnects, networks
- since ca. 2014: Security on automotive networks



There has been a dramatic increase in possible sources of faults and errors due to networking of heterogeneous computing units !

*Computer scientists have problems enough with parallel computing on regular and homogenous networks !*

(diverse in character or content)



# Faults /Errors in Electronic Systems

## Level of Verifiability

Faulty Specification

Faulty Design (Software)

Faulty Design (Hardware)

Manufacturing Defects

Errors in Operation  
(transient HW errors)

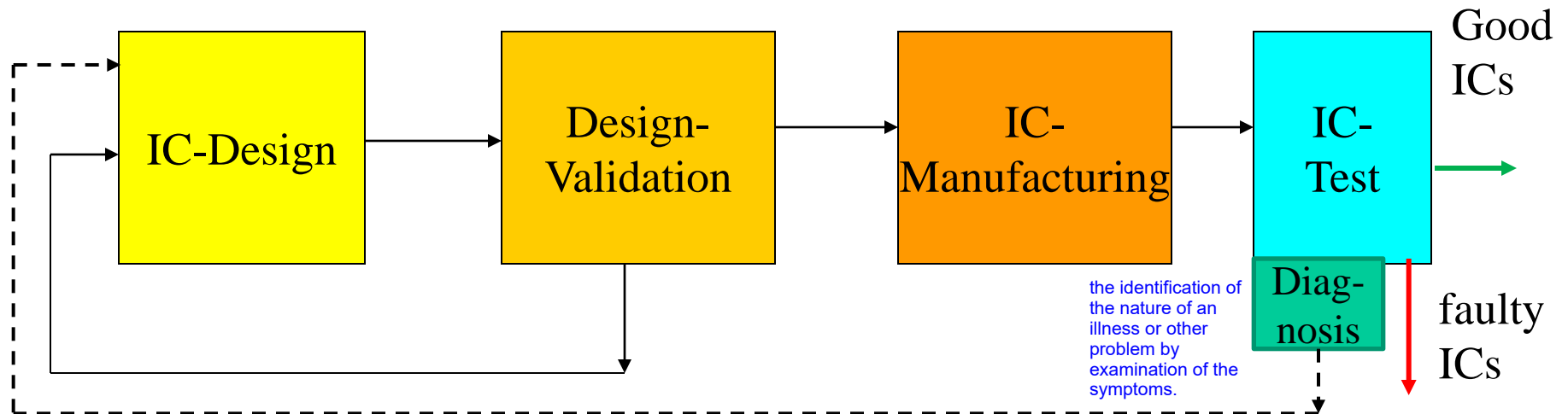
(lasting only for a short time; impermanent-not permanent)

Faults / errors due to  
aging and over-stress

handable	limited	very poor
		●
		●
	●	
	● *	
● *		
		●

\* May cost a lot !

# The Test Gap



Manufacturing Yield: 
$$\frac{\text{Fault-free Chips}}{\text{All Chips}}$$

# The Test Gap

The ratio of of transistors on a single chip that can be:

**manufactured** over

**validated** (Hardware) over

**tested** is about:

**100 : 10 : 1      !!!**

.. unless digital circuits and systems are designed fo testability !

# Can Electronics be Fault-Free ??

- Typically, specifications are not complete in a formal sense and may even contain implicit contradictions.
- Neither hardware nor software design is error-free. But, for example, formal verification of processor designs is possible.
- According to general experience, software design is ( likely or liable to suffer from, do, or experience something unpleasant or regrettable. ) error-prone. Debugging is done by testing, which is not complete in a formal sense. Typically, software bugs seem to be in rarely-used functions.
- Integrated circuits may suffer from manufacturing defects. They have to be found by production testing. Repair is sometimes done, mainly for memories.
- ● Transient hardware errors of otherwise correctly working circuits in normal operation become more likely with smaller features and lower operating voltage.

# 4. Fault Tolerance and Self Repair

Faults are everywhere. You can fight them.

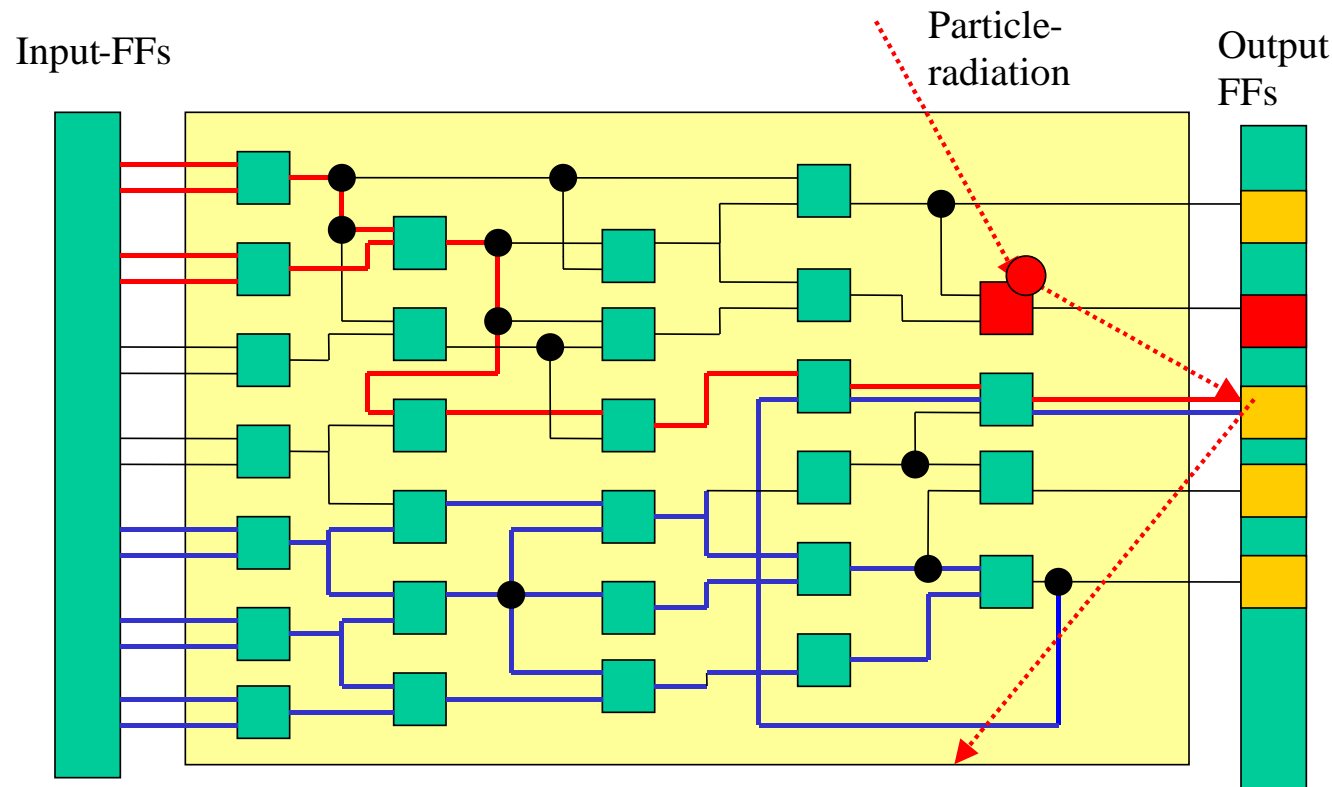
But then you have to know first where and when they occur ...



# Mikroelectronics for Cars

Year	min. feature in nm	Trans. pro cm <sup>2</sup>	Clock frequency MHz	Pins	Metall- layers	VDD/ V
1970	20 000	1000	1	16	1	10
1975	10 000	10 000	5	28	1	10
1980	5 000	20 000	10	50	1	5
1985	1 000	100 000	20	68	2	5
1990	500	1 000 000	50	200	3	3
1995	300	5 000 000	200	350	4	2
2000	130	50 000 000	1000	500	6	1
2005	65 250	200 000 000	3000	800	8	1
2010	30	1 000 000 000	4000	1000	10	1
2015	65	-automotive electronics trails behind !				

# Transient Hardware Faults



Particles from cosmic radiation and from atomic decay may hit electronic devices and trigger (mainly) transient and (much less often) permanent faults.

# Some Recent Data\*

**Cray XT5, Oak Ridge, Tennessee, „Jaguar“**

Memory size: 360 terabytes, all ECC-protected

(Error-correcting code memory)

ECC error rate: 350 per minute (single bit errors)

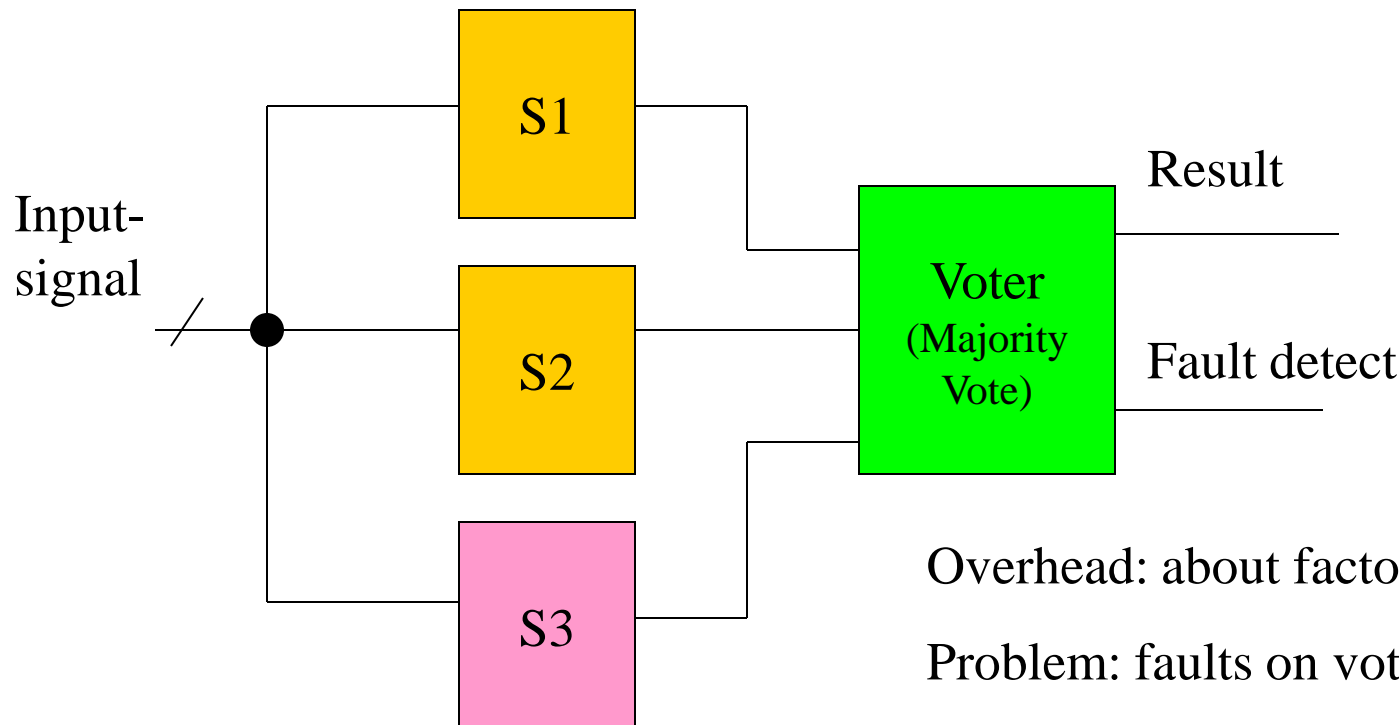
1 per 24 hours (double bit errors)

*This corresponds to one single bit error on a 1 GByte-memory  
about every 16 hours !*

Radiation- induced „hits“ causing permanent damages were also measured, but  
much less frequently !

\*IEEE Spectrum, March 2016, pp. 28-31

# Fault Detection by Active Redundancy



Overhead: about factor 3-4

Problem: faults on voter



„Self-checking Checker“

# Dependable Hardware

- Error detection and – compensation for transient errors in running operation. Often using code-based methods or re-execution.

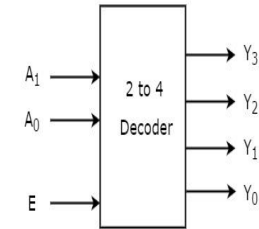
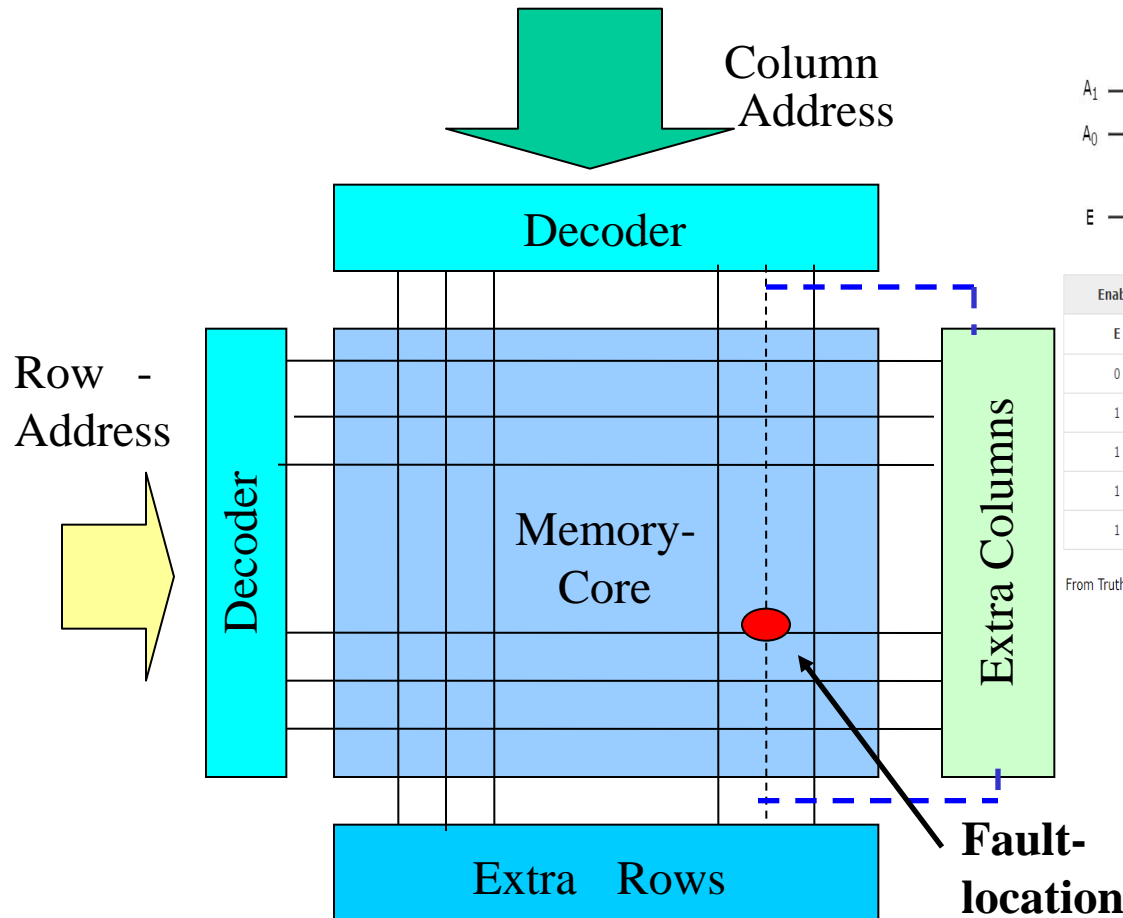
*Has been the focus of related research for decades. Relatively well investigated. Requires extra hardware, often time, and always extra energy.*

- Error detection and correction of permanent faults in running operation. Some strategies like re-execution fail. Requires „Built-in Self Repair“- BISR.

*BISR seems to work nicely for regular structures like memory blocks, Is much more difficult to apply to irregular logic.*



# Self Repair for Memory Blocks



Enable	Inputs		Outputs			
E	$A_1$	$A_0$	$Y_3$	$Y_2$	$Y_1$	$Y_0$
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

From Truth table, we can write the **Boolean functions** for each output as

$$Y_3 = E \cdot A_1 \cdot A_0$$

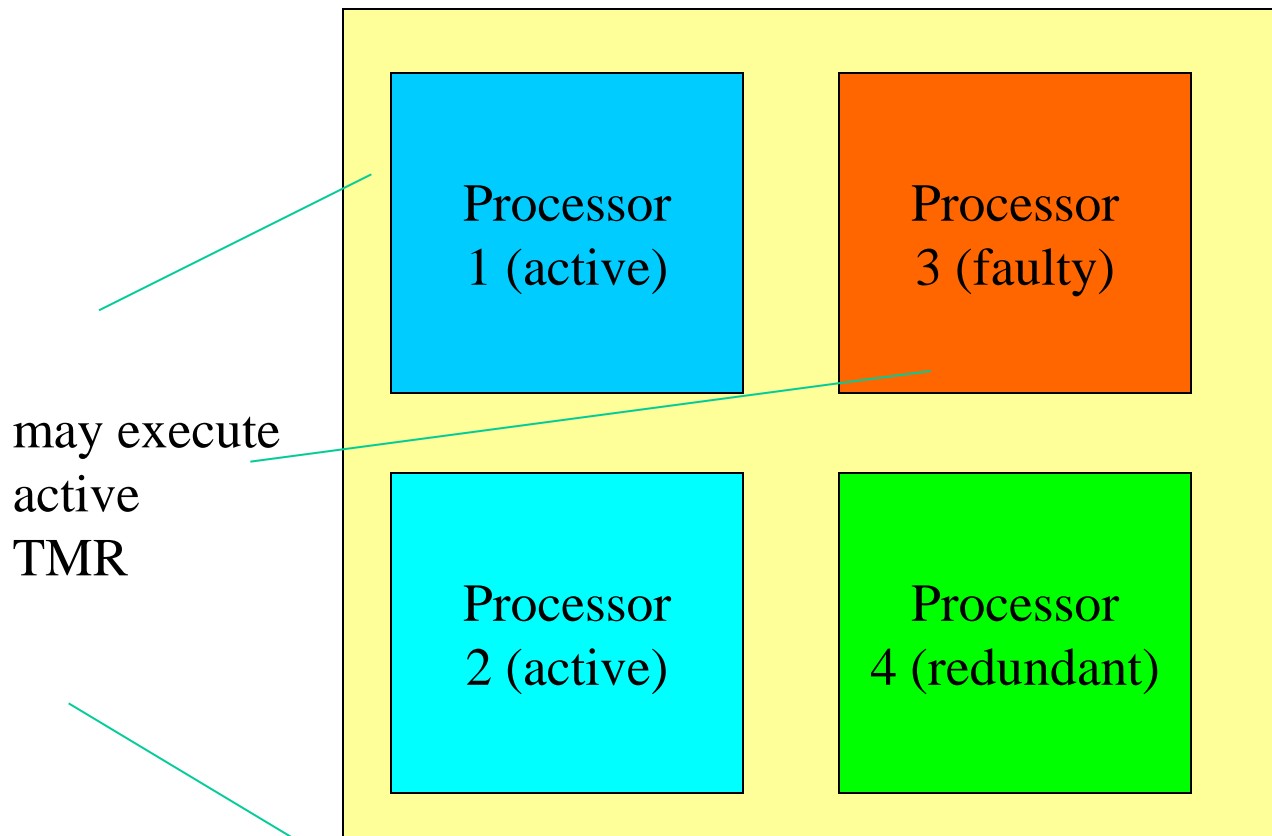
$$Y_2 = E \cdot A_1 \cdot A_0'$$

$$Y_1 = E \cdot A_1' \cdot A_0$$

$$Y_0 = E \cdot A_1' \cdot A_0'$$

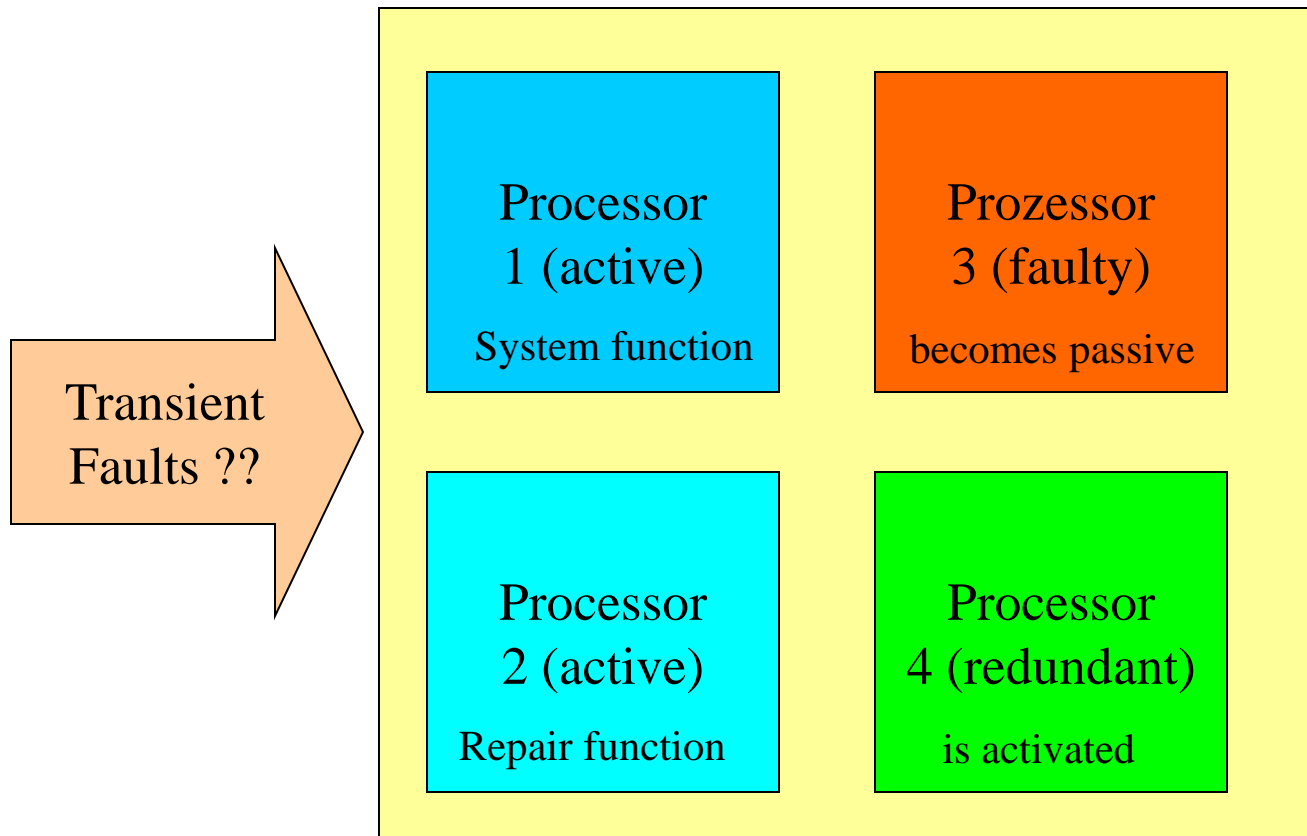
... seems to work reasonably well and is in practical use (e. g. for cashes).

# Self Repair by Redundancy



( In computing, triple modular redundancy, sometimes called triple-mode redundancy,[1] (TMR) is a fault-tolerant form of N-modular redundancy, in which three systems perform a process and that result is processed by a majority-voting system to produce a single output. If any one of the three systems fails, the other two systems can correct and mask the fault.)

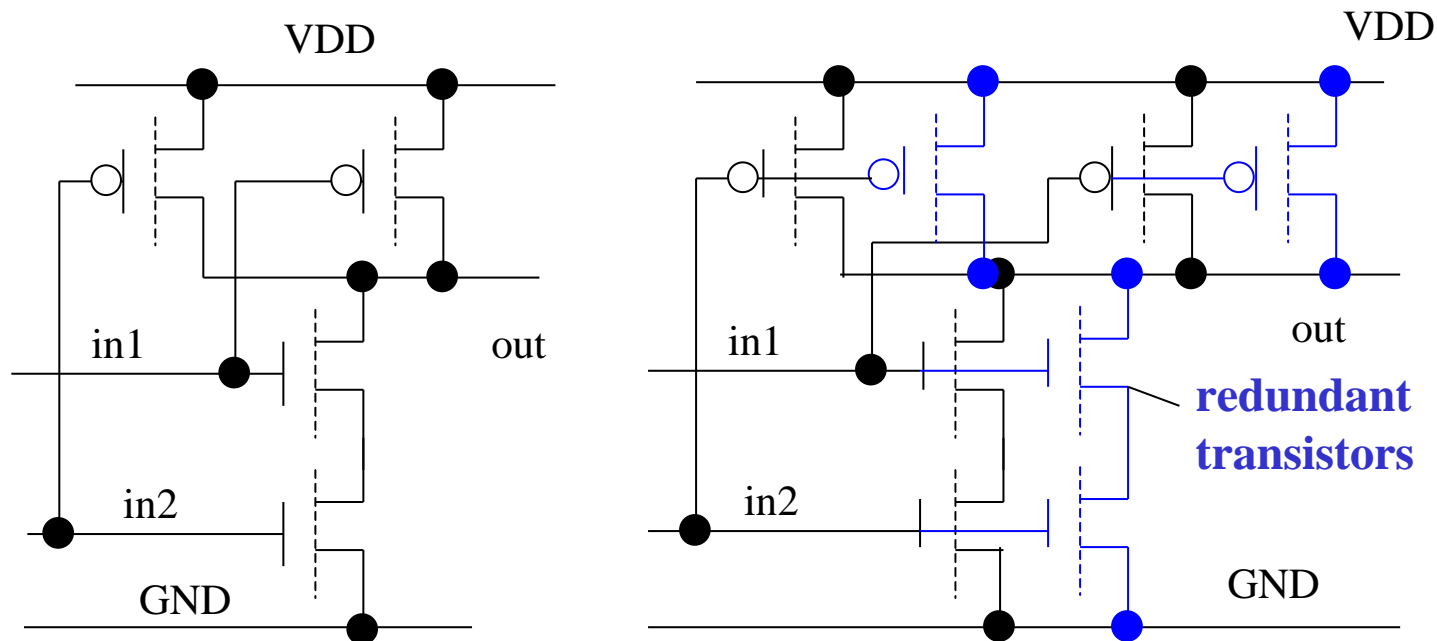
# Coarse-Grain Self Repair



(very tired.)

... works only as long as the supply of spare parts is not exhausted !

# Fine-Grain Self Repair



Repair at the transistor level is not feasible due to the large overhead necessary to organize the repair process.

# Dependable Hardware ??

- Microelectronics circuits and devices show a remarkably high level of reliability, even over a long life time (about 15 years in cars).
- Microelectronics circuits and devices based on nano-technologies with a feature size below about 30 nm may need extra care.
- Control of hardware- related faults and errors in digital circuits and systems is relatively well understood. Sensors seem to make major problems.

*Hardware: P8 of 1906,  
up to 60 years in service.*

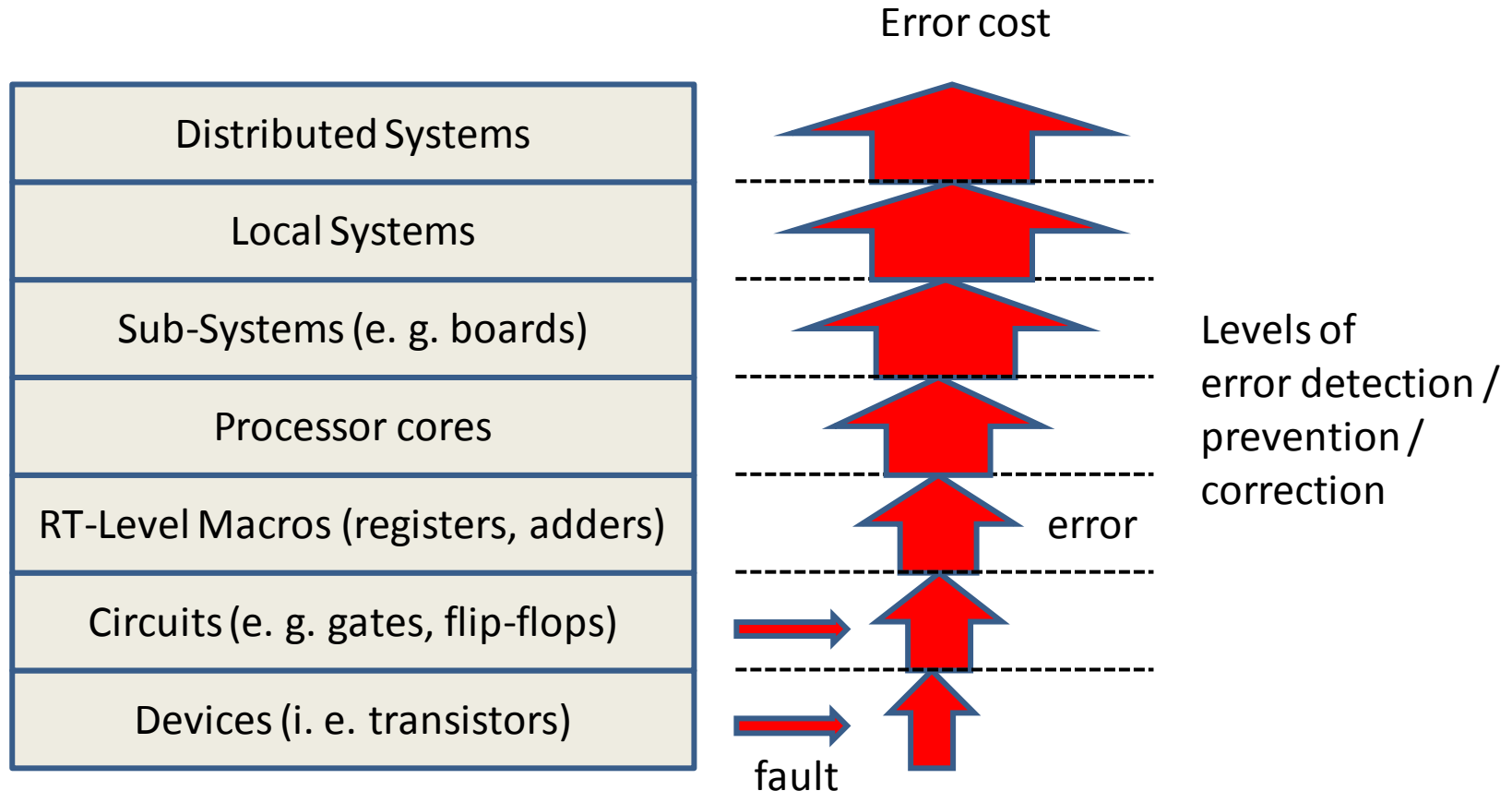


ICE ??





# Error Resilience



The real objective is to achieve „**error resilience**“ by keeping low- level / local errors from propagating to higher levels. Systems should be „fail-safe“ even under local error conditions ! *This does not necessary mean total error correction !*

# Dependable Software??



Software gets „mature“ while being in use over time (Banana-Effect).

That is why parts of the software, which are rarely used (such as software for error management) are most likely to be and remain faulty!!!

At least there are new ways that show interactions in large-scale software systems (Software-Tomography).



# 5. Reconfigurable Systems

(Symbiosis is a relationship between two or more organisms that live closely together.)

Hardware and Software make some sort of a symbiosis by (re-) programmable hardware structures such as FPGAs (field-programmable gate arrays).

(Field Programmable Gate Arrays (FPGAs) are semiconductor devices that are based around a matrix of configurable logic blocks (CLBs) connected via programmable interconnects. FPGAs can be reprogrammed to desired application or functionality requirements after manufacturing. This feature distinguishes FPGAs from Application Specific Integrated Circuits (ASICs), which are custom manufactured for specific design tasks. Although one-time programmable (OTP) FPGAs are available, the dominant types are SRAM based which can be reprogrammed as the design evolves.

This gives new opportunities on system design, but there are also new risks.

The FPGA-synthesis-software may contain bugs or Trojans !

# How Softies see the World of Hardware

- Hardware always works.  
*Almost correct, but only almost.....*

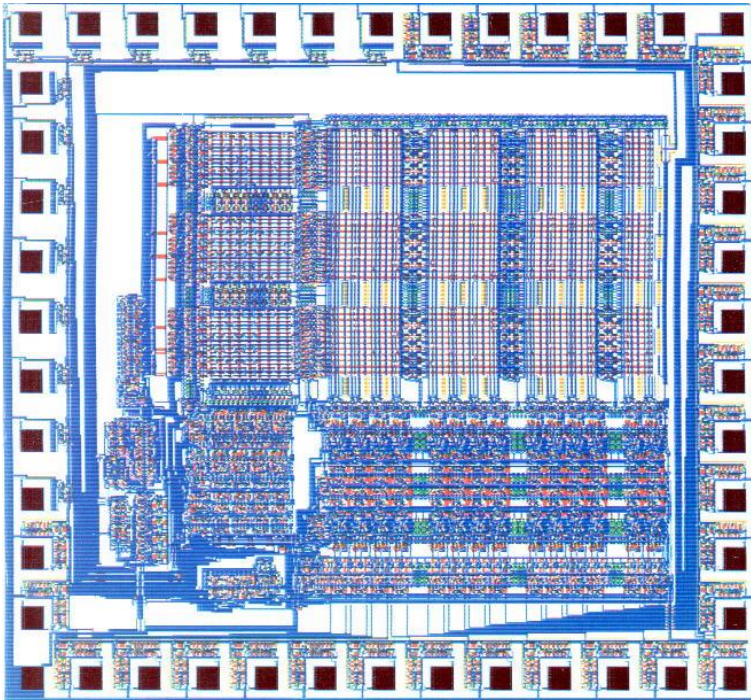
- Hardware is hard and cannot be modified.  
*No longer correct. Hardware may become soft.*

## *What Softies do not see:*

- Hardware can become faulty without a total breakdown. See transient faults !
- Hardware ages and does not become better when getting older (no banana effect).
- There are **software-based solutions** for hardware error management, but they are costly in terms of either extra hardware, power, or timing !

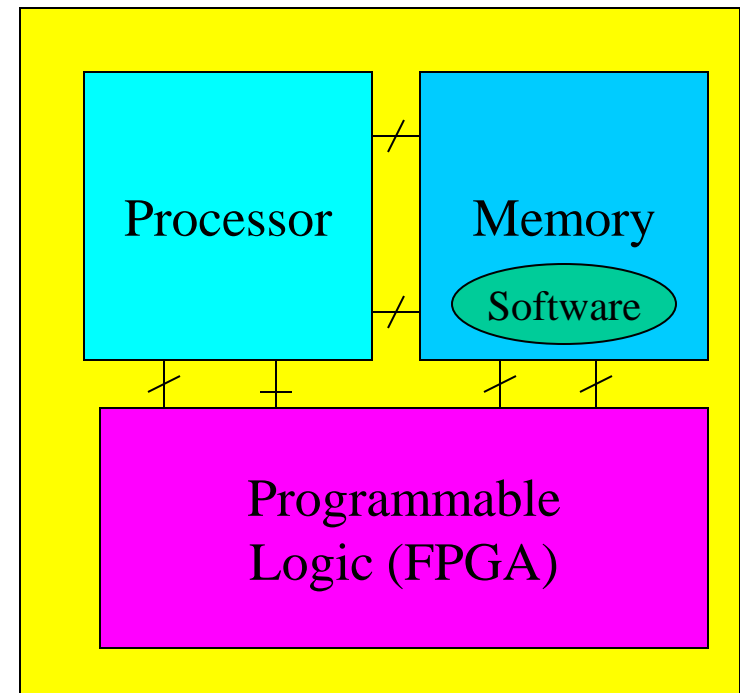
# From „hard“ Chip to „soft“ FPGAs

Application specific IC (ASIC)



Function fixed!

System on a Chip (SoC)

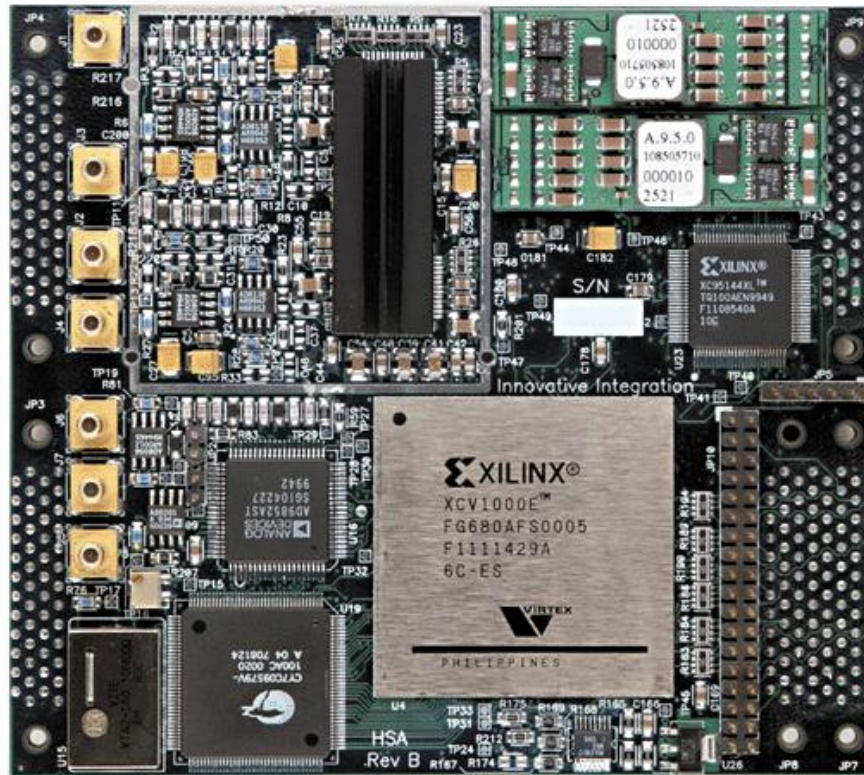


Variable (re-) configurable function!

*Hardware becomes soft!*



# FPGA-Board

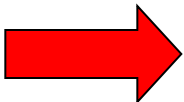


# Reconfigurable Systems

- Instead of „solid“ logic circuits programmable (via memory cells) configurable logic units are used. The cost is larger hardware (up to a factor of 10) and lower speed (again a factor of 10) plus extra power.
- Many FPGAs of today contain „embedded“ processors fabricated in „solid“ logic. But FPGAs can also implement „soft“ processors.
- Software may be modified *at run time* depending on changing parameters & requirements. *That is often forbidden in **real life** !*
- Hardware can be changed „in system“ and „at run time“ upon demand!

*Software and hardware that is self-modifying upon demand is not science fiction, but reality !*

# Curse or Blessing??

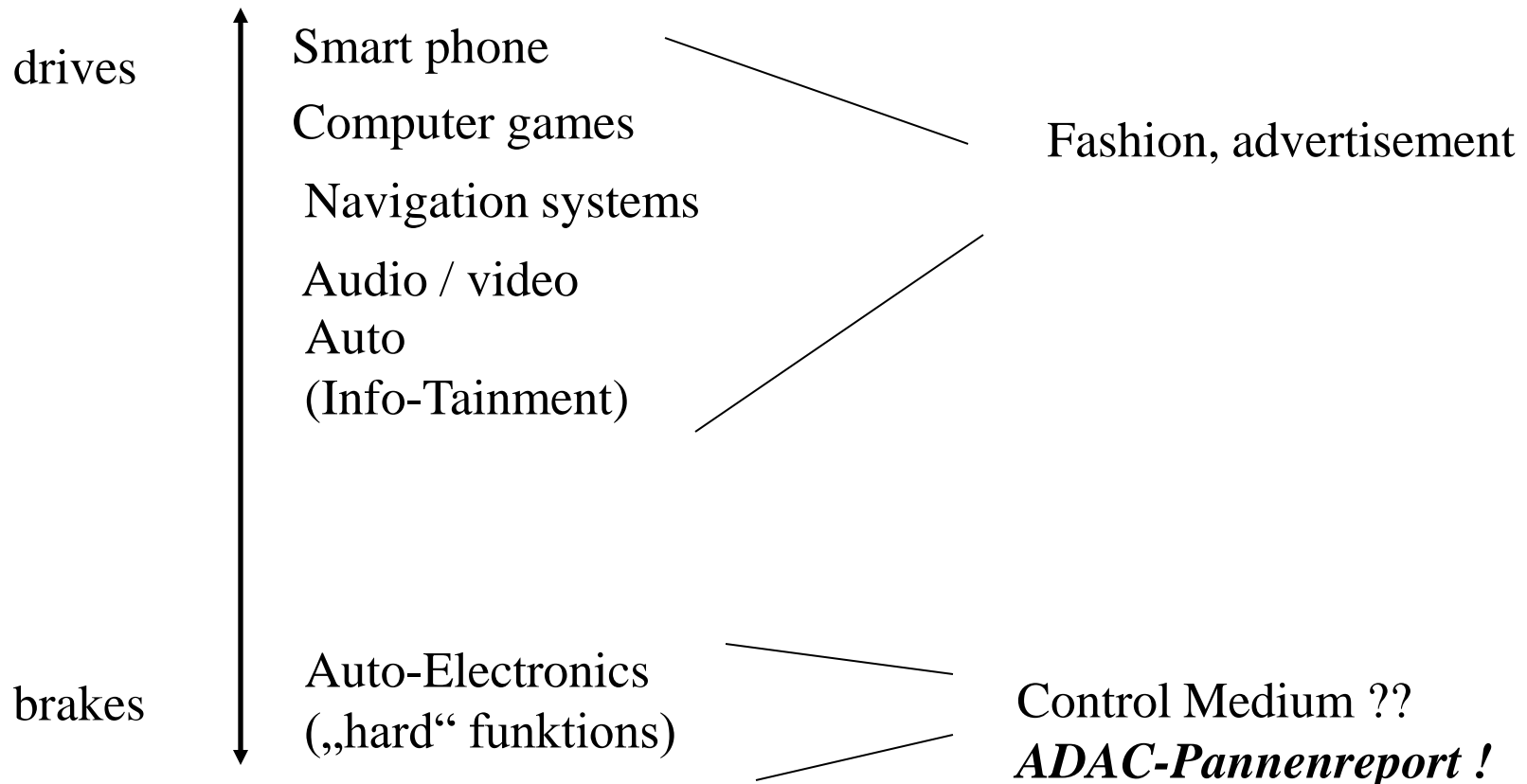
- Systems that can perform tasks of self-reconfiguration come close to the properties of living biological systems. This is self repair or self healing, but it can also be cancer !!
- Hardware is configured by Software. But software cannot be verified formally.  
 Adventure Playground! No FPGAs e. g. in automotive control !
- Not only Software, but also Hardware becomes vulnerable against a „Computer-Virus“!
- With strict restrictions on how to modify systems, self-repair and self-healing may become an attractive property!

# 6. Summary

There are always tendencies and opportunities in technology,  
which are prone to end in crashes ...

(likely or liable to suffer from, do, or experience something unpleasant or regrettable.)

# Complexity Drivers





# The Most Powerful Medium !





Unsere Tabelle zeigt, wie oft die Autos mit Erstzulassung 2001 bis 2006 im vergangenen Jahr (wegen technischer Pannen) liegen blieben. Die Zahlen geben an, wie viele Pannen auf jeweils 1000 Autos des gleichen Typs und Jahrgangs entfielen. Ausgewertet wurden Modellreihen, die mindestens drei aufeinanderfolgende Jahre prinzipiell unver-

ändert gebaut wurden und zumindest in einem Jahr 10 000 Neuzulassungen erreichten. Fällt die Zulassungszahl in einem Jahr unter 7500, wird dieses Jahr nicht bewertet. Je niedriger der Durchschnitt der Platzziffern in den jeweiligen Jahren ist, um so weiter vorne liegt das Fahrzeugmodell im Gesamtergebnis. Die farbige Bewertung von **Grün** (= zuverlässig) bis **Rot** (= unzuverlässig) ergibt sich aus der Spanne der Pannen-

zahlen des jeweiligen Jahres in der Fahrzeugklasse, für jedes Zulassungsjahr wird also eine individuelle Rangfolge erstellt. Die unterschiedlichen durchschnittlichen Laufleistungen der einzelnen Modellreihen innerhalb der Fahrzeugklassen werden durch einen entsprechenden Bonus-/Malusfaktor ausgeglichen, wobei einkalkuliert ist, dass nur ein Teil der Pannen durch die Laufleistung beeinflusst wird.

Platz/Modell	Erstzulassung:	2001	2002	2003	2004	2005	2006
<b>Kleine Klasse</b>							
1 Audi A2		12,2	12,3	8,3	5,1	2,2	
2 BMW Mini			12,3	7,7	4,9	2,6	2,0
3 Renault Modus					5,8	3,1	3,2
4 VW Lupo		23,4	17,4	13,3	8,6		
5 VW Polo		22,0	23,8	13,6	8,3	5,1	3,0
6 Toyota Yaris, Yaris Verso		15,1	11,6	11,2	6,9	6,1	7,5
7 Honda Jazz				9,3	7,8	5,5	3,4
8 Opel Corsa		27,8	24,6	13,8	8,8	5,7	2,2
8 Renault Twingo		37,7	26,7	16,8	6,7	4,3	2,6
10 Smart Forfour					9,6	6,3	0,9
11 Fiat Panda (neu)					8,4	4,1	5,6
12 Seat Ibiza/Cordoba		27,6	27,7	15,9	7,3	5,4	12,6
13 Ford Fiesta		16,4	16,3	15,9	10,7	9,0	6,0
13 Renault Clio		55,9	39,1	22,9	10,0	4,3	2,1
15 Skoda Fabia		26,6	24,9	16,0	9,8	7,4	3,7
16 Peugeot 206		37,2	25,7	19,9	10,0	6,6	2,9
17 Nissan Micra		25,8	20,2	26,1	18,0	9,9	1,5
18 Citroën C3				21,3	13,4	5,7	2,4
19 Citroën C2					13,8	5,7	2,6
20 Smart Fortwo		41,3	35,9	30,6	13,5	6,0	1,7
21 Fiat Punto		52,5	51,3	25,1	12,0	7,1	1,6
22 Ford Ka/Streetka		11,5	10,9	20,0	22,0	14,6	8,8
23 Mazda 2				15,4	12,6	8,0	3,4
24 Hyundai Getz				18,7	14,4	10,7	8,0
25 Kia Picanto					11,3	11,5	9,1
<b>Untere Mittelklasse</b>							
1 BMW 1er					4,5	3,0	1,6
2 Mazda 3					5,6	3,8	1,9
3 Audi A3		23,6	19,0	12,7	5,2	3,0	1,6
4 VW Golf/Bora		23,2	17,6	11,8	7,2	4,9	4,0
5 VW New Beetle				13,1	7,2	5,3	
6 Toyota Corolla, Corolla Verso		28,6	11,2	9,8	7,8	6,2	8,0
7 Mercedes A-Klasse		37,0	21,5	14,7	8,6	3,9	2,5
8 Seat Leon/Toledo		23,2	22,1	11,7	9,4	5,2	8,0
9 Honda Civic		18,3	17,8	15,9	8,9	8,5	8,1
10 Ford Focus		18,7	16,9	15,1	11,9	12,2	4,8
11 Citroën Xsara/Xsara Picasso		36,6	24,3	16,2	10,3	5,2	8,1
11 Opel Astra		32,5	29,9	17,7	13,3	5,4	3,7
13 Nissan Almera/Almera Tino		22,4	21,2	20,1	16,2	11,4	
14 Peugeot 307		52,2	34,5	23,9	9,9	7,7	3,0
15 Renault Mégane		53,6	60,1	41,6	11,1	5,1	4,4
16 Fiat Stilo			47,3	38,5	22,2		
<b>Mittelklasse</b>							
1 Audi A4		20,4	12,0	7,7	5,2	4,0	2,8
2 Mercedes C-Klasse		23,2	13,6	8,1	4,0	1,9	2,8
3 BMW 3er		18,9	17,1	12,1	6,5	3,1	2,6
4 VW Passat		19,9	14,7	10,6	7,9	5,4	4,3

Platz/Modell	Erstzulassung:	2001	2002	2003	2004	2005	2006
<b>Mittelklasse</b>							
5 Mazda 6			12,7	10,7	8,2	5,5	4,0
6 Skoda Octavia		20,2	17,4	12,5	6,8	8,0	6,1
7 Toyota Avensis, Aven. Verso		23,3	16,8	12,3	9,2	8,4	7,1
8 Peugeot 407					10,3	6,6	6,3
9 Ford Mondeo		21,9	23,5	21,6	13,6	9,3	5,1
10 Nissan Primera		21,9	26,7	32,3			
11 Opel Vectra		34,1	30,5	26,0	18,4	7,9	3,2
12 Volvo S40/V40/V50		24,2	21,7	18,3	15,4	12,3	5,2
13 Renault Laguna		58,8	61,3	41,0	19,0	7,6	4,7
<b>Obere Mittelklasse/Oberklasse</b>							
1 Audi A6		28,4	21,3	14,9	6,1	3,4	2,3
2 Mercedes E-Klasse		41,2	24,8	11,5	5,5	3,4	3,2
3 BMW 5er		39,7	31,9	18,0	7,6	4,4	2,7
4 Opel Signum				34,8	21,6	8,6	1,0
5 Volvo S60/S70/S80/V70		36,4	27,0	20,4	11,6	12,9	8,0
<b>Sportwagen/Cabrios</b>							
1 Mercedes CLK		31,6	15,3	7,3	2,8	2,3	1,8
2 BMW 3er Cabrio		13,1	8,7	8,4	5,0		
3 Mercedes SLK		22,9	18,6	12,1	3,9	2,5	1,2
4 Peugeot 206 CC		34,0	27,3	16,1	8,6	5,5	2,9
<b>Geländewagen</b>							
1 BMW X3					3,1	1,8	1,2
2 Mercedes ML		27,8	20,7	13,6	6,8	3,5	2,0
3 Toyota RAV4		25,4	16,9	13,5	9,4	5,9	3,1
4 BMW X5				18,7	6,8	5,6	
5 VW Touareg					7,1	4,5	3,2
6 Nissan X-TRAIL				13,2	9,7	7,8	3,6
<b>Kleine Vans</b>							
1 Mazda Premacy		23,0	16,3	9,02			
1 Mitsubishi Space Star		14,9	15,0	12,4	9,2		
3 VW Touran				13,7	10,2	7,2	3,3
4 VW Caddy					14,1	8,5	2,5
5 Ford Fusion				20,5	10,3	8,0	5,7
5 Seat Altea					10,7	8,5	5,0
7 Opel Agila		33,5	28,9	24,2	17,1		
7 Opel Meriva				13,9	9,7	11,0	5,7
9 Renault Scénic		53,3	50,4	43,5	17,9	6,0	2,5
10 Opel Zafira		35,1	32,8	22,7	14,9	12,0	5,6
11 Citroën Berlingo		51,1	39,0	26,0	18,1	9,3	5,8
12 Renault Kangoo		77,9	56,8	37,1	21,5	11,1	3,5
13 Ford C-Max					18,3	13,3	6,3
<b>Große Vans</b>							
1 VW Sharan		35,7	30,1	24,3	13,8	8,3	6,2
2 VW T4/T5		34,6	29,8	26,7	22,3	15,4	4,9
3 Fiat Ducato		35,4	35,2	32,0	21,1	13,4	3,3
3 Mercedes Vito/Viano		59,7	33,5	27,2	19,1	11,6	6,0
5 Ford Galaxy		47,8	44,5	32,0	23,3	11,4	9,6

# Und what about the Future ??



Unreliable hardware / software may give a lot of fun sometimes..

*.. But you depend on the rescue helicopter, and that one needs to be dependable at least !*