

IAF0530 (MSc, 5 ECTS)
IAS0530 (MSc, 6 ECTS)
IAF9530 (PhD, 6 ECTS)


Dependability and fault tolerance

Gert Jervan
Department of Computer Systems
Tallinn University of Technology (TTÜ)




General Information

- Homepage:
www.pld.ttu.ee/IAF0530
- Lecturer & Examiner:
Gert Jervan
ICT-527 620 2261
gert.jervan@ttu.ee
www.pld.ttu.ee/~gerje
- Guest lectures:
Maksim Jenihhin




Gert Jervan

- MSc from TTÜ in 1998
 - Exchange student at
TIMA Labs (Grenoble, France), Fraunhofer Institute
(Dresden, Germany), Linköping University (Sweden)
- PhD from Linköping University (Sweden) in 2005
- Senior research fellow at TTÜ since 2005, professor
since 2012
- Vice-Dean for Research at the Faculty of IT (2012),
Dean (2013)
- Published more than 80 papers at international
conferences and journals
- Organized many international conferences and
coordinated several research projects




Course Plan

- Lectures on weeks 2,4,6,7,9,10 (2x per week)
 - Tuesdays and Thursdays 12:00 – 13:30
 - Lectures combined with groupwork
- Case study presentations on weeks 12-16
 - Thursdays 10:45 – 13:30
- Always check the course homepage!!
- Individual project work
- Oral exam (individual discussions)



Individual work

- Reading
- Writing
- Presenting
- The course requires weekly reading and
participation in discussions
 - All missing assignments have to be compensated
during the exam



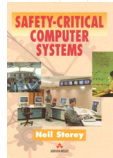
Reading

- Various papers
(on the course homepage)**
www.pld.ttu.ee/IAF0530
- Textbooks
- Incident/accident reports
- Web pages



Textbooks

- Safety-Critical Computer Systems
 - Neil Storey
 - Addison Wesley, 1996.
 - An introductory text which provides overview of safety related aspects and methods in computer systems development.
 - Available in the TTÜ library

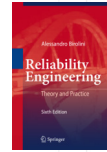


7



Textbooks

- Reliability Engineering: Theory and Practice.
 - Alessandro Birolini
 - Springer
 - 2014 (7th ed.) 2010 (6th ed.), 2007 (5th ed.)
 - This book shows how to build in, evaluate, and demonstrate reliability & availability of components, equipment, systems. It presents the state-of-the-art of reliability engineering, both in theory and practice
 - TTÜ library has several copies of the latest edition.

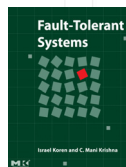


8



Textbooks

- Fault-Tolerant Systems
 - Israel Koren and C. Mani Krishna
 - Morgan-Kaufman Publishers, 2007



This book covers comprehensively the design of fault-tolerant hardware and software, use of fault-tolerance techniques to improve manufacturing yields and design and analysis of networks. Additionally it includes material on methods to protect against threats to encryption subsystems used for security purposes.

9



Textbooks

- Fault-Tolerant Design
 - Elena Dubrova
 - Springer, 2013
 - This textbook serves as an introduction to fault-tolerance, intended for upper-division undergraduate students, graduate-level students and practicing engineers in need of an overview of the field. Readers will develop skills in modeling and evaluating fault-tolerant architectures in terms of reliability, availability and safety. They will gain a thorough understanding of fault tolerant computers, including both the theory of how to design and evaluate them and the practical knowledge of achieving fault-tolerance in electronic, communication and software systems. Coverage includes fault-tolerance techniques through hardware, software, information and time redundancy. The content is designed to be highly accessible, including numerous examples and exercises.



10



Case Studies

- The exact format will be announced during the second week of lectures (and it depends of the number of students we will have)
- Topic categories:
 - Accident analysis
 - System safety analysis
 - Literature survey
 - Something else (implementation, tool study, etc.)
 - Requires prior ack.
- Literature and sample (!) topics on the webpage

11



Case Studies

- Some examples (from 2016):
 - Estimating availability of the KSI service.
 - Dependability and Fault Tolerance of PaaS
 - Real-time Transport Protocol security considerations in Source-Specific Multicast topology
 - Fault tolerance on Cryptography
 - Automatic train protection systems
 - Software Fault Injection Methods
 - Safety and reliability of autonomous vehicle technologies
 - Evolution of Fault Tolerance in PostgreSQL
 - Self-checking network-on-chip layout design
 - Verified compilation
 - Fault tolerance in wireless systems
 - Critical Information Infrastructure vulnerability analysis methods

12

Course overview

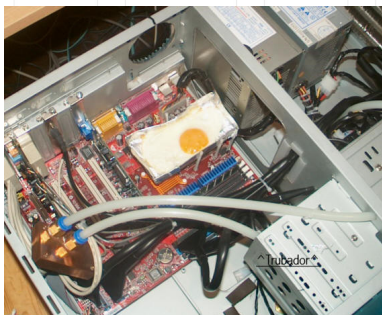
13

Course Overview

- Reliability: increasing concern
 - Historical
 - High reliability in computers was needed in critical applications: space missions, telephone switching, process control, medical applications etc.
 - Contemporary
 - Extraordinary dependence on computers: on-line banking, commerce, cars, planes, communications etc. Emergence of internet-of-things.
 - Hardware is increasingly more fault-prone (complexity, technology, environment)
 - Software is increasingly more complex
 - Things simply do not work without special reliability measures

14

Already yesterday



15



The Silicon Engine

A Timeline of Semiconductors in Computers

1950s	1960s	1970s	1980s	1990s	2000s	2010s
1 Transistor	16 Transistors	4500 Transistors	275,000 Transistors	3,100,000 Transistors	59,000,000 Transistors	8,000,000,000 Transistors

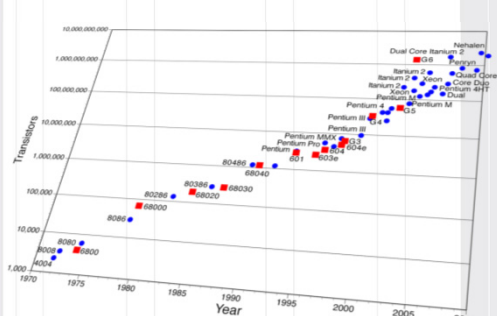
Moore's Law "Transistor density on integrated circuits doubles about every two years." (Source: "Moore's Law: Raising the Bar" Intel Corporation 2002)

Microelectronic silicon computer "chips" have grown in capability from a single transistor in the 1950s to hundreds of millions of transistors per chip in today's microprocessor and memory devices. From the first documented semiconductor effect in 1833 to the transition from transistors to integrated circuits in the 1960s and 70s, this website explores key milestones in the development of these extraordinary engines that power the computing and communications revolution of the information age.

© Computer History Museum | Credits

16

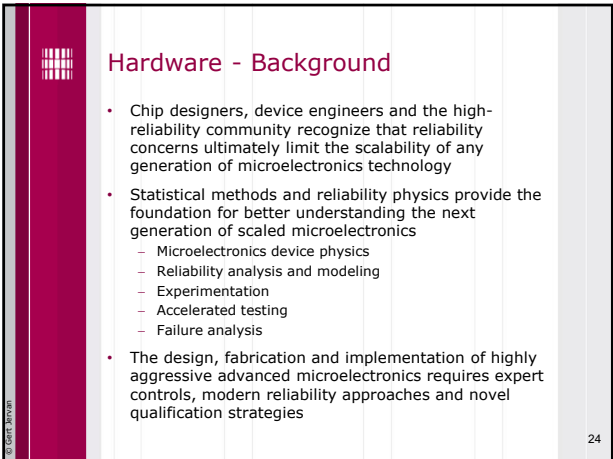
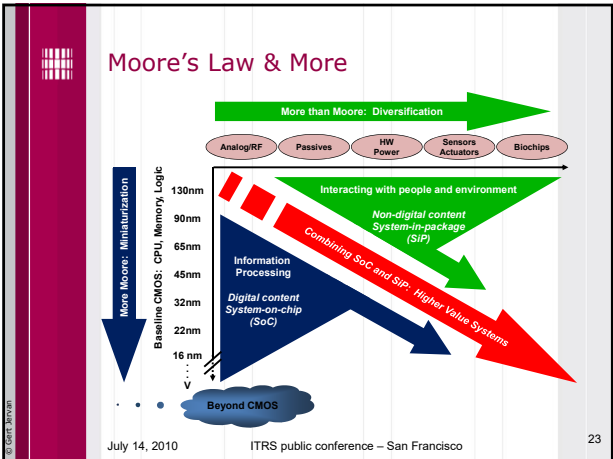
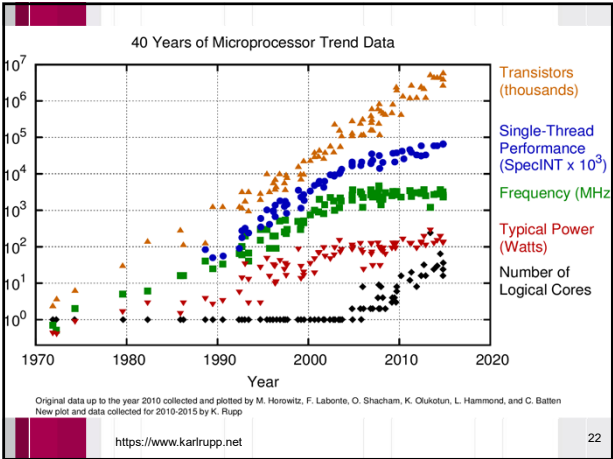
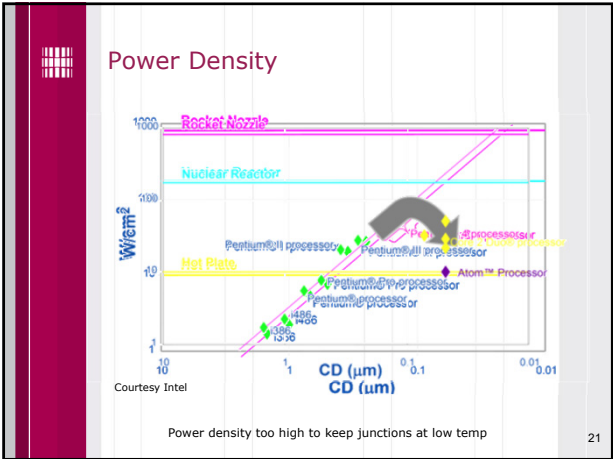
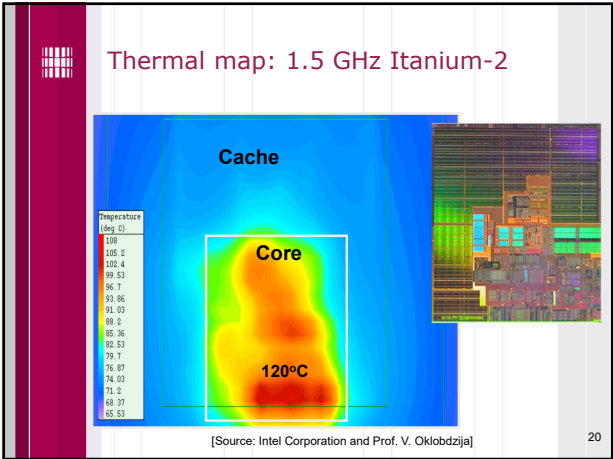
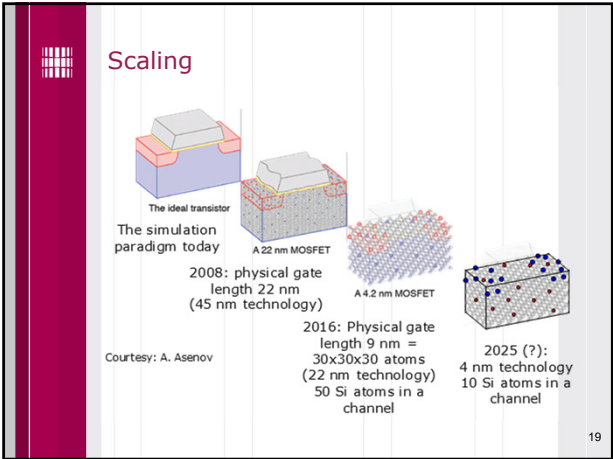
Moore's Law



17

Moore's Law

- Growth rate
 - 2x transistor count every 2 years over 50 years
 - 10x every 10 years
- Dramatically more complex algorithms previously not feasible
 - Dramatically more realistic video games and graphics animation (e.g. Playstation 4, Xbox 360 Kinect, Nintendo Wii)
 - 1 Mb/s DSL to 10 Mb/s Cable to 2.4 Gb/s Fiber to Homes.
 - 2G to 3G to 4G to 5G wireless communications
 - MPEG-1 to MPEG-2 to MPEG-4 to H.264 video compression
 - 480 x 270 (0.13 million pixels) NTSC to 1920x1080 (2 megapixels) HDTV resolution to 4K UHD 3840 x 2160 (8.3 megapixels)



Scaling Trends & Reliability Considerations

- Dramatic increase in processing steps with each new generation
 - approx. 50 more steps per generation and a new metal level every 2 generations
- Rush to market - Less time to characterize new materials than in the past
 - e.g. reliability issues with new materials not fully understood and potential new failure modes
- Manufacturers' trends to provide 'just enough' lifetime, reliability, and environmental specs for commercial & industrial applications
 - e.g. 3-5 yr product lifetimes, trading off 'excess' reliability margins for performance

25

Scaling Trends & Reliability Considerations

- Significant rise in the amount of proprietary technology and data developed by manufacturers, reluctance to share information with hi-relevance customers
 - e.g. process recipes, process controls, process flows, design margins, MTTF
- Next generation microelectronics focus on the performance needs of the commercial customer, with little or no emphasis on the extreme needs
 - e.g. extended life, extreme environments, high reliability
- Increasingly difficult testability challenges due to device complexity

26

Correct or Defective?

Theory:

Reality:

27

Product Technical Trends

	1990	2000	2010
Operating temperature, °C	-55 to 125	-40 to +85	0 to 70
Supply voltage	5v	1.5v	0.6v
Max. power (high perf.)	5	100	170
No. of package types	<10	<60	??
Design support life	>10 yrs.	1-5 yrs.	<1yr.
Production life	>10 yrs.	3-5 yrs.	<3yrs.
<u>Service life</u>	<u>>20 yrs.</u>	<u>5-10 yrs.</u>	<u><5yrs.</u>

*MRQW-2002, Bernstein

28

Growing Internet Traffic

Year	Global Internet Traffic
1992	100 GB/Day
1997	100 GB/Hour
2002	100 GB/Sec
2007	2 000 GB/Sec
2012	12 000 GB/Sec
2017	35 000 GB/Sec

Cisco VNI, 2013

29

Ubiquitous Computing

30



Software complexity is a challenge

Aviation:

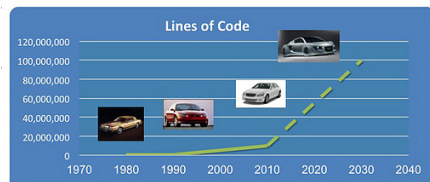
- Boeing 747 → 0.4 M LOC
- Boeing 777 → 4 M LOC
- Technology Review 2002

Software:

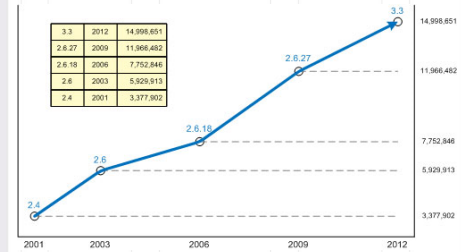
- Exponential increase in software complexity

Automotive:

- ✓ 2010 Premium → 100 M LOC
- ✓ 1995 – 2000 → 52%/Year
- ✓ 2001 – 2010 → 35%/Year
- Tony Scott, GM CIO
- ✓ 2011 – BMW is the first manufacturer to break the 1Gb



Linux Growth



32



Big Data

- An increasingly sensor-enabled and instrumented business environment generates HUGE volumes of data with MACHINE SPEED characteristics



- 1 Billion lines of code
- EACH engine (A380 has 4 of them) generating 10 TB every 30 minutes!

33



Course Overview

- To get an insight into the broad area of system safety
- We cover techniques for high availability, fault tolerance, monitoring, detection, diagnosis, and confinement of failure, ways to improve availability through fast recovery and graceful service degradation, and techniques for using redundancy and replication.
- We also discuss the utopia of flawless software, the impact of scale on availability, ways to cope with human operator error, and metrics for evaluating dependability.

34



Contents

- Fault tolerance, dependability
- Safety, hazards, risks
- Safety-critical systems
- System reliability
- Hardware reliability
- Hardware redundancy
- Software fault tolerance
- ...

35



Lecture Outline



- ✓ Historical perspective and famous incidents/accidents

- Basic terminology

36



Murphy's Law

- "If something can go wrong, it will go wrong"
Major Edward A. Murphy, Jr.
US Air Force, 1949
- "Every component than can be installed backward, eventually will be"

37



Genesis Space Capsule

- \$260 million Genesis capsule was collecting samples of the solar wind over 3 years period
- Crashed in Sept 2004 due to the failure of the parachutes
- Reason:
 - the deceleration sensors — the accelerometers — were all installed backwards. The craft's autopilot never got a clue that it had hit an atmosphere and that hard ground was just ahead.



38



Mars Orbiter

- One of the Mars Orbiter probes crashed into the planet in 1999.
- It did turn out that engineers who built the Mars Climate Orbiter had provided a data table in "pound-force" rather than newtons, the metric measure of force.
- NASA flight controllers at the Jet Propulsion Laboratory in Pasadena, Calif., had used the faulty table for their navigation calculations during the long coast from Earth to Mars.

39



Lockheed Martin Titan 4

- In 1998, a LockMart Titan 4 booster carrying a \$1 billion LockMart Vortex-class spy satellite pitched sideways and exploded 40 seconds after liftoff from Cape Canaveral, Fla.
- Reason: frayed wiring that apparently had not been inspected. The guidance systems were without power for a fraction of a second.



A Titan 4 rocket explodes shortly after takeoff in August 1998.

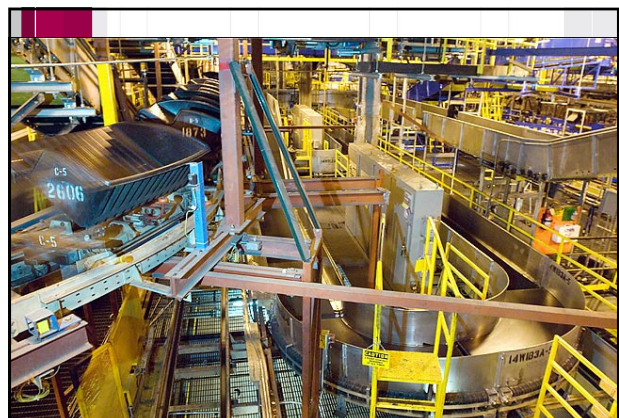
40



Therac-25

- Therac-25:
 - the most serious computer-related accidents to date (at least nonmilitary and admitted)
 - machine for radiation therapy (treating cancer)
 - between June 1985 and January 1987 (at least) six patients received severe overdoses (two died shortly afterward, two might have died but died because of cancer, the other two had permanent disabilities)
 - scanning magnets are used to spread the beam and vary the beam energy
 - dual-mode: electron beams for surface tumors, X-ray for deep tumors

41



42



Denver Airport

- Denver International Airport, Colorado: intelligent luggage transportation system with 4000 "Telecars", 35km rails, controlled by a network of 100 computers with 5000 sensors, 400 radio antennas, and 56 barcode readers. Price: \$186 million (BAE Automated Systems).
- Due to SW problems about one year delay which costs \$1.1 million per day (1993).
- Abandoned in 2005 to save \$1 million per month on maintenance
- Today we have the on-going story with the new Berlin Brandenburg Airport
 - Scheduled to open in 2011, the new estimate was 2014, 2017, 2020, 2021, ...

43



Boeing 787 Dreamliner

- Program launched in 2003, roll-out in 2007, first delivery in 2011. 114 delivered so far.
- Grounded on January 16, 2013 due to the problems with electrical circuitry
 - Leading to thermal runaway of Li-ion batteries and causing several fires in the battery compartment (several emergency landings, one aircraft (ET) was heavily damaged on ground)
 - Comprehensive review of the 787's critical systems, including the design, manufacture and assembly.
 - Japanese ANA alone lost 1.1 M USD per day (17 aircrafts)
- Grounding lifted on April 26, 2013



44



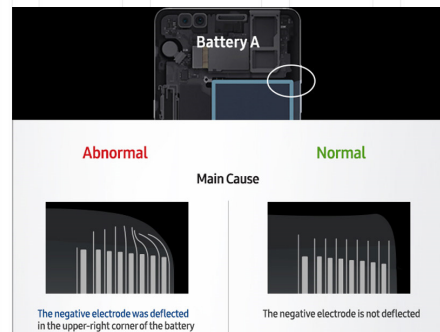
LAX airport ATC software failure

- 2,4 billion USD system (developed by Lockheed Martin) crashed on April 30, 2014.
 - Reason: U-2 spy plane that was Flying „too high“
 - Result: The system attempted to calculate all possible flight paths and run out of memory
- The "new \$40 billion air traffic control system, known as NextGen, which encompasses ERAM, including its reliance on Global Positioning System data that could be faked" is "very over-budget and behind schedule," Moss (founder of Def Con) told Reuters. It "doesn't surprise me that it's got some bugs - it's the way it presented itself that's alarming." You can expect at least two upcoming Def Con talks to delve into exploiting weaknesses in the system.

45



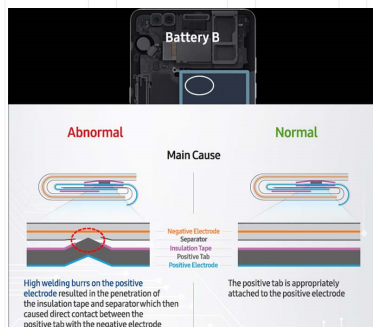
Samsung Galaxy Note 7



46



Samsung Galaxy Note 7



47



Lecture Outline



- ✓ Historical perspective and famous incidents/accidents

- Basic terminology

48



History

- Early computer systems
 - Basic components had very low reliability (de-bug)
 - Fault tolerant techniques were needed to overcome it by
 - Adding redundant structures with voting
 - Error-detection and error correction Codes
 - EDVAC (1949)
 - Duplicate ALU and compare results of both
 - Continue processing if agreed, else report error
 - Bell Relay Computer (1950)
 - 2 CPUs
 - One unit begin executing the next instruction if the other encounters an error
 - IBM 650, UNIVAC (1955)
 - Parity check on data transfers

49



History

- Advent of transistors
 - more reliable components
 - led to temporary decrease in the emphasis on fault-tolerant computing
 - designers thought it is enough to depend on the improved reliability of the transistor to guarantee correct computations
- last decades
 - more critical applications
 - space programs, military applications
 - control of nuclear power stations
 - banking transactions
- VLSI made the implementation of many redundancy techniques practical and cost effective

50



Applications

- Safety-critical applications
 - critical to human safety
 - aircraft flight control
 - environmental disaster must be avoided
 - chemical plants, nuclear plants
 - requirements
 - 99.99999% probability to be operational at the end of a 3-hour period

51



Applications

- Mission-critical applications
 - it is important to complete the mission
 - repair is impossible or prohibitively expensive
 - Pioneer 10 was launched 2 March 1970, passed Pluto 13 June 1983
 - requirements
 - 95% probability to be operational at the end of mission (e.g. 10 years)
 - may be degraded or reconfigured before (operator interaction possible)

52



Applications

- Business-critical applications
 - users want to have a high probability of receiving service when it is requested
 - transaction processing (banking, stock exchange or other time-shared systems)
 - ATM: < 10 hours/year unavailable
 - airline reservation: < 1 min/day unavailable

53



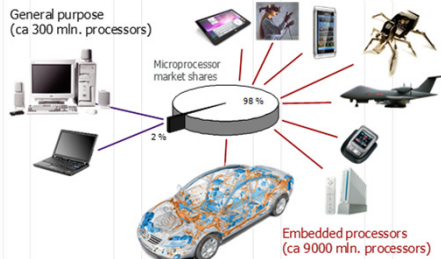
Applications

- Maintenance postponement applications
 - avoid unscheduled maintenance
 - should continue to function until next planned repair (economical benefits)
 - examples:
 - remotely controlled systems
 - telephone switching systems (in remote areas)

54



General-Purpose vs. Embedded

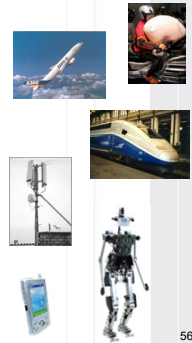


55



Embedded Systems, cont.

- Embedded computing systems
 - Computing systems embedded within electronic devices
 - Hard to define. Nearly any computing system other than a desktop computer
 - Billions of units produced yearly, versus millions of desktop units
 - „Internet of things“
 - SmartX (buildings, homes, communities, ...)

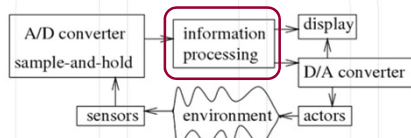


56



What is an Embedded System?

- Definition
 - an **embedded system** special-purpose computer system, part of a larger system which it controls.
- Notes
 - A computer is used in such devices primarily as a means to simplify the system design and to provide flexibility.
 - Often the user of the device is not even aware that a computer is present.



57



Characteristics of Embedded Systems

- Single-functioned
 - Dedicated to perform a single function
- Complex functionality

Many new challenges that all have effect on dependability

At the same time all these devices are around us, maybe even inside us

 - environment
 - Must compute certain results in real-time without delay
- Safety-critical
 - Must not endanger human life and the environment

58



Real-Time Systems

- **Time**
 - The correctness of the system behavior depends not only on the logical results of the computations, but also on the *time* at which these results are produced.
- **Real**
 - The reaction to the outside events must occur *during* their evolution. The system time must be measured using the same time scale used for measuring the time in the controlled environment.

59



Hard vs. Soft Real-Time

- Definitions
 - A real-time task is said to be **hard** if missing its deadline may cause catastrophic consequences on the environment under control.
 - A real-time task is said to be **soft** if meeting its deadline is desirable for performance reasons, but missing its deadline does not cause serious damage to the environment and does not jeopardize correct system behaviour.
- Definition
 - A real-time system that is able to handle hard real-time tasks is called a **hard real-time system**.

60



Hard vs. soft, cont.

- Examples of hard activities
 - Sensory data acquisition
 - Detection of critical conditions
 - Actuator serving
 - Low-level control of critical system components
 - Planning sensory-motor actions that tightly interact with the environment
- Examples of soft activities
 - The command interpreter of the user interface
 - Handling input data from the keyboard
 - Displaying messages on the screen
 - Representation of system state variables
 - Graphical activities
 - Saving report data

61



Functional vs. Non-Functional Requirements

- Functional requirements
 - output as a function of input
- Non-functional requirements:
 - Time required to compute output
 - Reliability, availability, integrity, maintainability, dependability
 - Size, weight, power consumption, etc.

62



Fault Tolerance

- A fault-tolerant system is one that can continue to correctly perform its specified tasks in the presence of failures:
 - hardware
 - software
 - user errors
 - environmental, input, ...
- Fault tolerance is the attribute that enables a system to achieve fault tolerant operation.

63



Basic Concepts

- *Fault Tolerance* is closely related to the notion of "Dependability". This is characterized under a number of headings:
 - **R**eliability – the system can run continuously without failure.
 - **A**vailability – the system is ready to be used immediately.
 - **M**aintainability – when a system fails, it can be repaired easily and quickly (and, sometimes, without its users noticing the failure).
 - **S**afety – if a system fails, nothing catastrophic will happen.

So called RAMS-studies

64



Faults, Errors & Failures

- Fault: a defect within the system or a situation that can lead to the failure
- Error: manifestation of the fault – an unexpected behavior
- Failure: system not performing its intended function

Fault → Error → Failure

65



Measuring

- Failures are measured in FITs
 - 1 FIT (failures in time), is the number of failures in 1 billion device-operation hours. A measurement of 1000 FITs corresponds to a MTTF (mean time to failure) of approximately 114 years.

66



Fault Examples

- Particle strike
 - Broken wire
 - Missing component
 - Aircraft retracting its landing gear (while on ground)
- Effects in time:
 - Permanent
 - Transient
 - Intermittent



67



Permanent

- A permanent fault or failure is one which is stable and continuous.
- Permanent hardware failures require some component to be replaced or repaired.
- An example of a permanent fault would be a VLSI chip with a manufacturing defect, causing one input pin to be stuck high (stuck-at-1).

68



Transient

- A transient fault is one which results from a temporary environmental condition.
- For example, a voltage spike might cause a sensor to report an incorrect value for a few milliseconds before reporting correctly.

69



Transient faults

- Happen for a short time
- **Corruptions of data, miscalculation in logic**
- Do not cause a permanent damage of circuits
- Causes are outside system boundaries



Radiation



Lightning storms

70



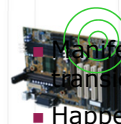
Intermittent

- An intermittent fault is one which only manifests occasionally, due to unstable hardware or certain system states.
- A loose contact on a connector will often cause an intermittent fault.
- Intermittent electrical faults, as a rule, are notoriously difficult to detect. Typically, whenever the fault doctor shows up, the system works fine.

71



Intermittent faults



Internal EMI



Power supply fluctuations



Crosstalk



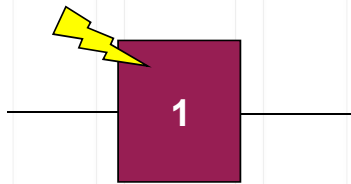
Init (Data)

Software errors (Heisenbugs)

72



Soft Errors



- Transient bit-flip (soft memory error)
 - Random event
 - Corrupts the value but not the cell
 - Can be corrected (in contrast to hard errors caused by faults in the hardware itself)
 - Happen continuously during system lifetime (i.e., can not be screened by burn-in tests)

73



Sources

- First traced to alpha particle emissions from chip packaging materials
 - Most sources removed (pure materials, different designs, shielding)
- Today's main problem: cosmic radiation
 - Cosmic particles from deep space (actually 5th- or 6th-hand collision particles)
 - At ground level ca 95% neutrons, 5% protons
 - Radioactive material in manufacturing process

74



Sources (cont.)

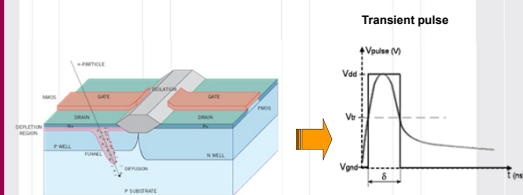
- Four main sources:
 - Low-energy alpha particles
 - High-energy cosmic particles
 - Thermal neutrons
 - Poor system design

SER type	Source	Mechanism	Trend
Alpha	Thorium and uranium contamination in-mold compound, silicon, or lead bumps	2- to 9-MeV alpha particle creating electron-hole tunnel traveling 25 microns in silicon	Exponential increase with scaling
Cosmic	Intergalactic sources modulated by solar flares	High-energy neutrons/protons (10 MeV to 1 GeV) colliding with silicon nuclei	Decrease in failures in time per megabit
Thermal neutron	Boron present in BPSG25-mV neutrons	Collision with B10 in BPSG	Highest, always dominates if present

75



Soft Errors



The electric field in the depletion region directly generates electron-hole pairs in its wake, causing the charges to drift so that the transistor sees a current disturbance

76



Evidence of Cosmic Ray Strikes

- Documented strikes in large servers found in error logs
 - Normand, "Single Event Upset at Ground Level," IEEE Transactions on Nuclear Science, Vol. 43, No. 6, December 1996.
- Sun Microsystems, 2000 (R. Baumann, Workshop talk)
 - Cosmic ray strikes on L2 cache with defective error protection
 - caused Sun's flagship servers to suddenly and mysteriously crash!
 - Companies affected
 - Baby Bell (Atlanta), America Online, Ebay, & dozens of other corporations
 - Verisign moved to IBM Unix servers (for the most part)
- 2005 – Los Alamos 2048-CPU HP server system crashed frequently due to defective cache
- 2010 Toyota brake problem (still no agreement)
- More recently: problems with GPGPU based HPC

77



Example

- It is practically impossible to build a perfect system
 - Suppose a component has the reliability of 99.99%
 - A system consisting of 100 non-redundant components will have the reliability 99.01%
 - A system consisting of 10 000 non-redundant components will have the reliability 36.79%
- It is hard to foresee all the factors

78