IAF0530 (MSc)
IAF9530 (PhD)

Süsteemide usaldusväärsus ja veakindlus
Dependability and fault tolerance

Gert Jervan
Department of Computer Engineering (ATI)
Tallinn University of Technology (TTÜ)

General Information

- Contents:
  Dependability and fault tolerance
  www.pld.ttu.ee/IAF0530
- Lecturer & Examiner:
  Gert Jervan
  IT-229 620 2261
  gert.jervan@ati.ttu.ee
  www.ati.ttu.ee/~gerje

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
- First PostDoc, then senior research fellow at TTÜ since 2005, full professor since 2012
- Vice-Dean for Research at the Faculty of IT (since 2012)
- Published more than 50 papers at international conferences and journals
- Organized many international conferences and coordinated several research projects, incl. 7-year project CEBE (Centre for Integrated Electronic Systems and Biomedical Engineering)

Course Plan

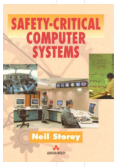
- 16 occasions, á 1,5 hours
  Thursdays 14:00-15:30
- 7-10 Lectures. No meetings on Feb 14, March 7, March 21 (Tentatively)
- Case Studies
  - Introductory presentation (5 min)
  - 20 min/30 min presentation of the final report
  - Written report (min. 6 pages, using predefined template; min. 10 pages for PhD students)
- Oral exam (discussion)

Reading

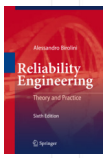
- **Various papers (on the course homepage)**
  www.pld.ttu.ee/IAF0530
- Textbooks
- Incident/accident reports
- Web pages

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

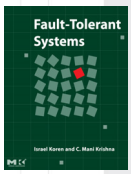


Textbooks



- Reliability Engineering: Theory and Practice.
  - Alessandro Birolini
  - Springer
  - 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 the 4th edition (2004).

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.

Case Studies

- 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

Case Studies

- Some examples:
  - Clock synchronization
  - Atomic and reliable broadcast
  - Algorithmic based fault-tolerance
  - System level diagnosis - distributed algorithms
  - Fault-tolerant transaction processing systems
  - Measures of software reliability
  - Validation and verification techniques
  - CAN (Controller Area Network) protocol
  - Fault-Tolerance in E-Commerce Web Servers
  - Fault tolerance in wired and wireless systems
  - Nano tubes
  - ...

Case Studies

- Topic selection:
  - February 14 (via e-mail, no lecture at that day)
- Draft of the report (1 page) + introductory presentation of the topic (5 min, 3 slides):
  - March 28
- Presentations (20/30 min):
  - Starting from the end of April
- Final Report (min. 6/10 pages):
  - May 30
- Final discussions:
  - June 3-7

Course Outline (Preliminary)

- Jan 31: Introduction
- Feb 7: Lecture II
- Feb 14: No lecture – topic selection (via e-mail)
- Feb 21, Feb 28, March 14: Lectures III – V
  - March 7, March 21: No lecture
- **March 28: Introductory presentation of the topic (5 min each)**
- April 4, ...: Lectures VI - ...
- **Starting from mid-April: Case study presentations (till May 16)**

Course overview

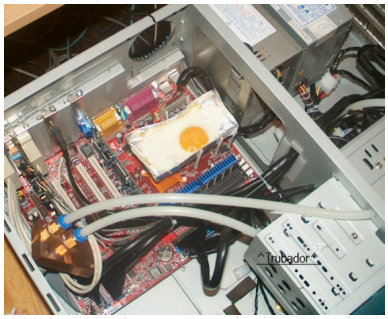
13

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

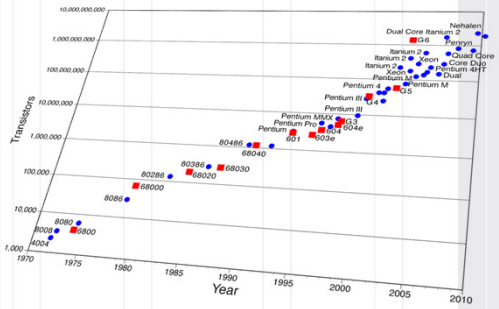
14

Today



15

Moore's Law



16

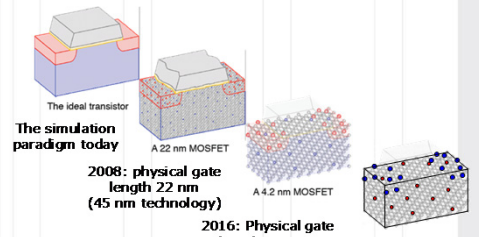
Moore's Law

- Growth rate:
  - 2x transistors/chip every 1.5 years over 50 years
  - 10x every 6-7 years

This won't last for long...

- 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 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 million pixels) HDTV resolution

Scaling



The simulation paradigm today

2008: physical gate length 22 nm (45 nm technology)

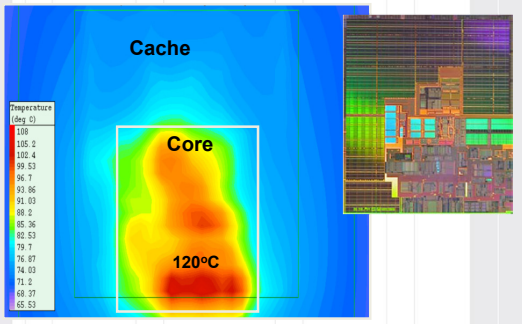
2016: Physical gate length 9 nm = 30x30x30 atoms (22 nm technology) 50 Si atoms in a channel

2025 (?): 4 nm technology 10 Si atoms in a channel

Courtesy: A. Asenov

18

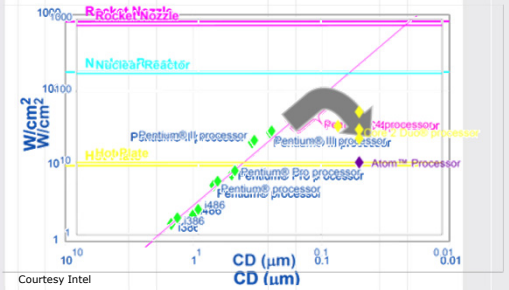
Thermal map: 1.5 GHz Itanium-2



[Source: Intel Corporation and Prof. V. Oklobdzija]

19

Power Density

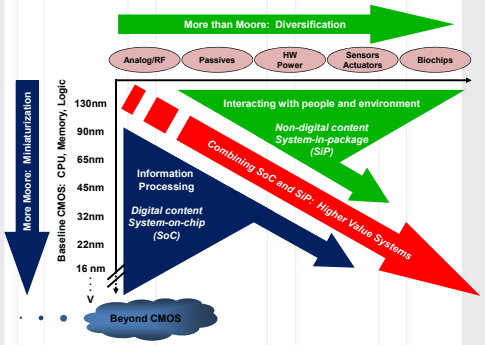


Courtesy Intel

Power density too high to keep junctions at low temp

20

Moore's Law & More



July 14, 2010 ITRS public conference – San Francisco

21

© Gert Jervan

Hardware - Background

- Chip designers, device engineers and the high-reliability community recognize that reliability concerns ultimately limit the scalability of any generation of microelectronics technology
- Statistical methods and reliability physics provide the foundation for better understanding the next generation of scaled microelectronics
  - Microelectronics device physics
  - Reliability analysis and modeling
  - Experimentation
  - Accelerated testing
  - Failure analysis
- The design, fabrication and implementation of highly aggressive advanced microelectronics requires expert controls, modern reliability approaches and novel qualification strategies

22

© Gert Jervan

Scaling Trends & Reliability Considerations

- A lot of technology concerns:
  - Reduced gate oxide thicknesses
  - Increased thermal/power densities
  - Reduced interconnect dimensions
  - Higher device operating temperatures
  - Increased sensitivity to defects and statistical process variations
  - Introduction of new materials with each new generation, replacing proven materials
    - e.g. Cu and low K inter-level dielectrics for Al and SiO2

23

© Gert Jervan

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

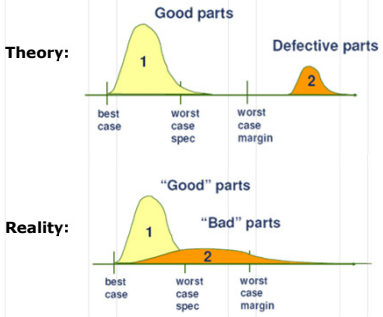
24

© Gert Jervan

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

Correct or Defective?



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.
Service life	*>20 yrs.*	*5-10 yrs.*	*<5yrs.*

*MRQW-2002, Bernstein

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
- In some areas code size is doubling every 9 months [ST Microelectronics, Medea Workshop, Fall 2003]
- … > 70% of the development cost for complex systems such as automotive electronics and communication systems are due to software development [A. Sangiovanni-Vincentelli, 1999]

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 barrier

Rob van Ommering, COPA Tutorial, as cited by: Gerrit Müller: Opportunities and challenges in embedded systems, Eindhoven Embedded Systems Institute, 2004

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.

Contents

- Fault tolerance
- System reliability
- Hardware redundancy
- Error detection techniques
- Coding techniques
- Processor-level detection and recovery
- Disk arrays
- Checkpointing and recovery
- Software fault tolerance
- Testing distributed real-time systems
- …

Lecture Outline

- ✓ **Historical perspective and famous incidents/accidents**
- **Basic terminology**

31

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"

32

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.

33

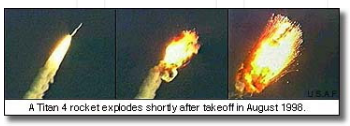
Mars Orbiter

- 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.

34

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.



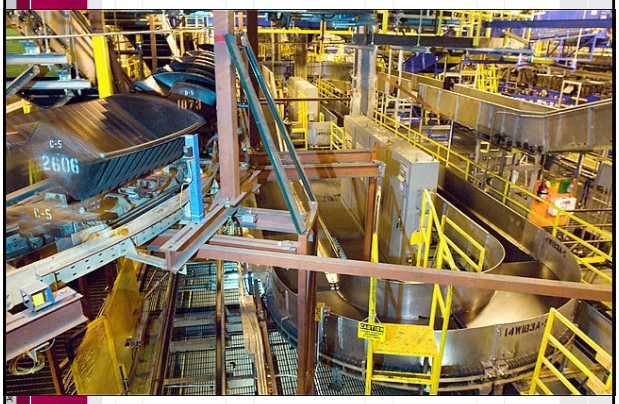
A Titan 4 rocket explodes shortly after takeoff in August 1998.

35

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

36



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).
- Abondoned 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 is 2014

Boeing 787 Dreamliner

- Program launched in 2003, roll-out in 2007, first delivery in 2011. 49 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
  - Comprehensive review of the 787's critical systems, including the design, manufacture and assembly.
  - Japanese ANA alone loses 1.1 M USD per day (17 aircrafts)

Lecture Outline

- ✓ **Historical perspective and famous incidents/accidents**
- **Basic terminology**

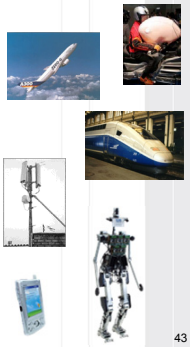
Embedded Systems

- Computing systems are everywhere
- Most of us think of "desktop" computers
  - PC's
  - Laptops
  - Mainframes
  - Servers
- But there's another type of computing system
  - Far more common...

General-Purpose vs. Embedded

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
  - Perhaps 50 per household and per automobile
  - „Internet of things"
  - SmartX (buildings, homes, communities, ...)



A "Short List" of Embedded Systems

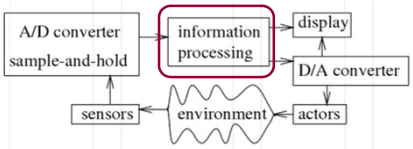
Anti-lock brakes
Auto-focus cameras
Automatic teller machines
Automatic transmission
Avionic systems
Battery chargers
Cameras
Cell phones
Cell-phone base stations
Cordless phones
Cruise control
Curbside check-in systems
Digital cameras
Disk drives
Electronic games
Electronic instruments
Electronic toys/games
Factory control
Fax machines
Fingerprint identifiers
Home security systems
Life-support systems
Medical testing systems
Modems
MP3 players
Network switches/routers
On-board navigation
Pagers
Photocopiers
Point-of-sale systems
Portable video games
Printers
Satellite phones
Scanners
Smart ovens/dishwashers
Speech recognizers
Stereo systems
Teleconferencing systems
Telephone switches
Theft-tracking controllers
Theft tracking systems
TV set-top boxes
Video cameras
Video game consoles
Video phones
Washers and dryers

Ambient Intelligence
Medical Systems
Telecommunications
Transport systems
Security & Military
Entertainment

Our daily lives depend on embedded systems

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.



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

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.

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**.

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

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.

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.

Basic Concepts

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

So called RAMS-studies

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

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.
- Example: Bit flips in hardware due to cosmic radiation
  - A person on an airplane over the Atlantic at 35,000 ft working on a laptop with 256 Mbytes (2 Gbits) of memory. At this altitude, the soft error rate (SER) of 600 FITs per megabit becomes 100,000 FITs per megabit, resulting in a potential error every five hours.

Fault Examples

- Year 2000 bug
- Loose wire
- Aircraft retracting its landing gear while on ground

- Effects in time:
  - Permanent
  - Transient
  - Intermittent

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).

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.

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

Electromagnetic interference (EMI)

Radiation

Lightning storms

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.

Intermittent faults

Crosstalk

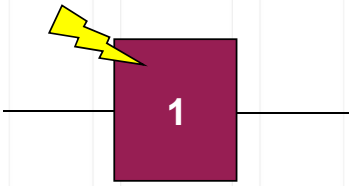
Internal EMI

- Manifest similar as transient faults
- Happen repeatedly
- Causes are inside system boundaries

Software errors (Heisenbugs)

Power supply fluctuations

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)

61

Sources

- First traced to alpha particlce 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

62

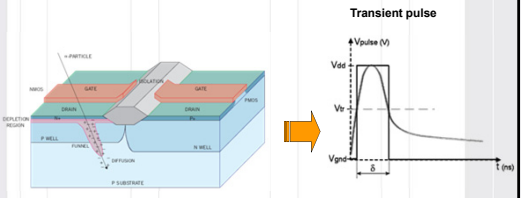
Sources (cont.)

- 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-meV neutrons	Collision with B10 in BPSG	Highest, always dominates if present

63

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

64

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 not solved)

65

Current Situation

- Soft errors induced the highest failure rate of all other reliability mechanisms combined

Rober Baumann, TI

66