Embedded Systems and Software

Course website:
http://www.liacs.leidenuniv.nl/~stefanovtp/courses/ES

Main Lecturer: Todor Stefanov (t.p.stefanov@liacs.leidenuniv.nl)
Assistants:
Roozbeh Siyadatzadeh (s.r.siyadatzadeh@liacs.leidenuniv.nl)
Fatemeh Mehrafrooz

Leiden Embedded Research Center (LERC)
Leiden Institute of Advanced Computer Science (LIACS)
Leiden University, The Netherlands

Course Organization: Structure

- ***On-campus* Class Lectures**
- Main Goal
  - To introduce students to state-of-the-art in Embedded MPSoC
    - methods, techniques, and tools
    - to design, program, and implement
- Format
  - Oral presentations supported by PPT slides
  - Slides available on website after each lecture
  - Interactive discussions during lectures

Course Organization: Structure

- ***On-campus* Hands-on lab sessions with practical assignments**
  - DAEDALUS framework (http://daedalus.liacs.nl)
  - Used to design and prototype a relatively simple Embedded MPSoC
- ***On-campus* Student Symposium**
  - Read and study scientific papers on topics related to the course material
  - Presentation on the topic under study
  - Read carefully the schedule and instructions at **http://liacs.leidenuniv.nl/~stefanovtp/courses/ES/symposium.html**

Course Organization: Grading

- There is NO traditional written exam
- Final grade is a combination of grades for
  - Presentation on a given topic related to the course material (40%)
  - Performance during the hands-on sessions (40%)
  - Pro-active attitude during lectures and paper discussions (20%)

Course Literature and Material

- Some reference books
  - ***"Embedded System Design"* by Peter Marwedel,**
    2nd, 3rd , or 4th edition, Springer (available as eBook or on-line)
  - ***"Embedded System Design: Modeling, Synthesis and Verification"* by Daniel Gajski, Samar Abdi, Andreas Gerstlauer, and Gunar Schirner,**
    Springer, 2009, XXVI, 358 p. 185 illus., Hardcover ISBN: 978-1-4419-0503-1
  - ***"Embedded System Design: A Unified Hardware/Software Introduction"* by Frank Vahid and Tony Givargis,**
    John Wiley & Sons; ISBN: 0471386782. Copyright (c) 2002.
  - ***"Embedded Multiprocessors: Scheduling and Synchronization"* by Sundararajan Sriram, Shuvra S. Bhattacharyya,** (Marcel Dekker)
  - ***"Computers as Component, Principles of Embedded Computer Systems Design"* by Wayne Wolf,** (Morgan Kaufman Publishers)
    http://www.ee.edu/~wolf/embedded-book/about.html
- The slides contain (copyright) material by
  - Peter Marwedel, Lothar Thiele, Wayne Wolf, Daniel Gajski, Shuvra Bhattacharyya, Andy Pimentel, Edward Lee, Stephan Edwards
  - and from the above books …

Contents of the Course (1)

- Introduction to Embedded Systems
- Embedded Systems Components
- Embedded Systems Specification and Modeling
  - Models of Computation
  - Specification Languages
- Basic concepts, methods, techniques, and tools for design of Embedded MPSoCs
  - Y-chart approach by Gajski
  - Y-chart approach by Kienhuis

Contents of the Course (2)

- DAEDALUS framework for MPSoC design
  - Automatic parallelization of streaming applications
  - System-level modelling and simulations for DSE
  - System-level synthesis in a “plug-and-play” fashion
- DAEDALUS^{RT} framework
  - Extensions for Real-Time MPSoC design
- Other System-level design frameworks for MPSoCs
  - System on Chip Environment from the UC Irvine
  - HOPES from the Seoul National University
  - …

Introduction to Embedded Systems and Software

Outline

- What is an Embedded System?
- Examples
- Characteristics of Embedded Systems
- Comparison
  - Embedded Systems vs. General Purpose Systems
- Trends in Embedded Systems
- What is Embedded Systems Design?
- Future of Embedded Systems

What is an Embedded System?

Many Definitions exist:

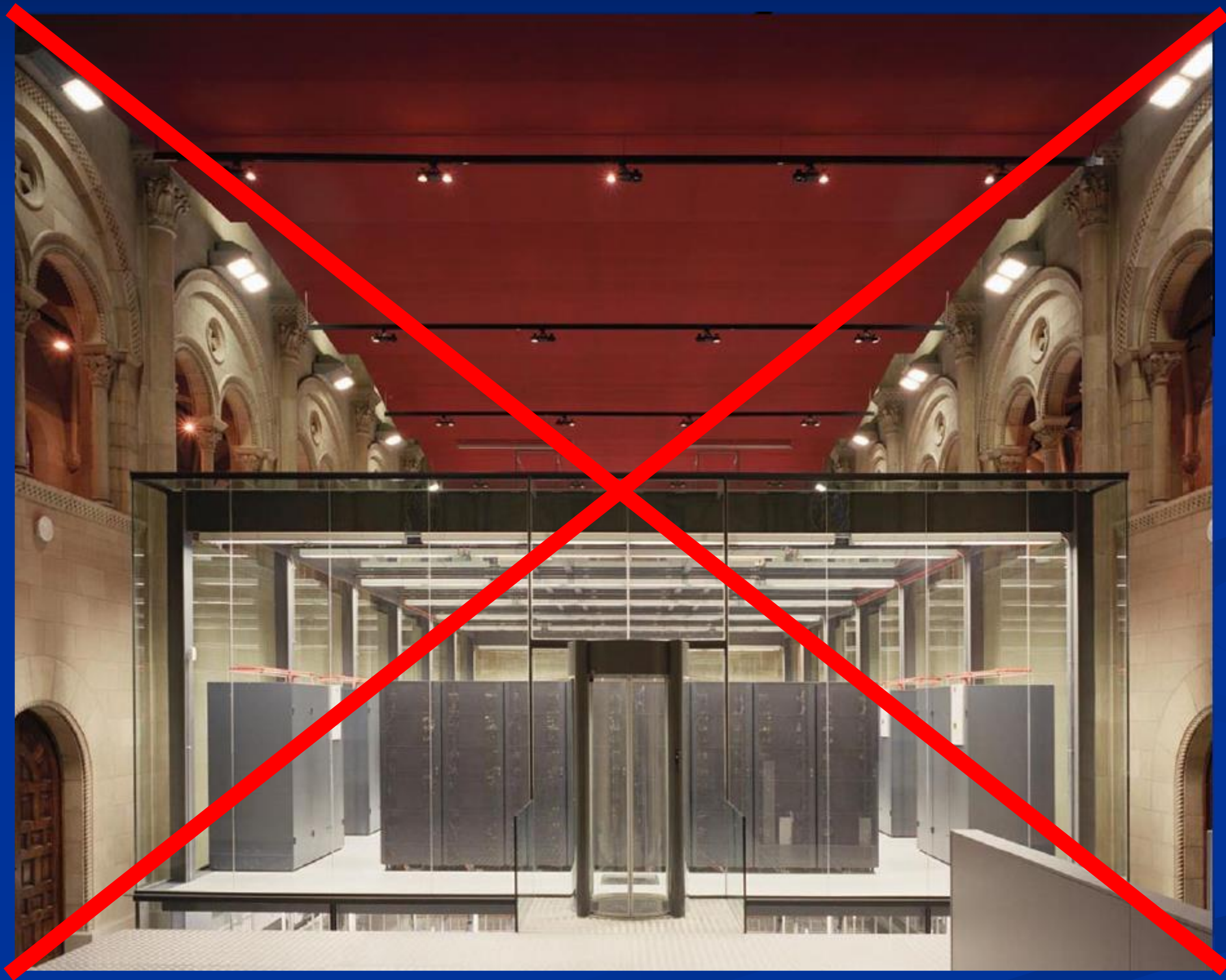
[Peter Marwedel, TU Dortmund]

Embedded Systems = **Information processing systems embedded into a larger product**

[Edward A. Lee, UC Berkeley]

Embedded Software = **Software integrated with physical* processes. The technical problem is managing time and concurrency in computational systems.**

Is this an Embedded System?



Barcelona SuperComputer Center

Is this an Embedded System?

Yet Another Definition …

Embedded Systems = **Information processing systems that are:**

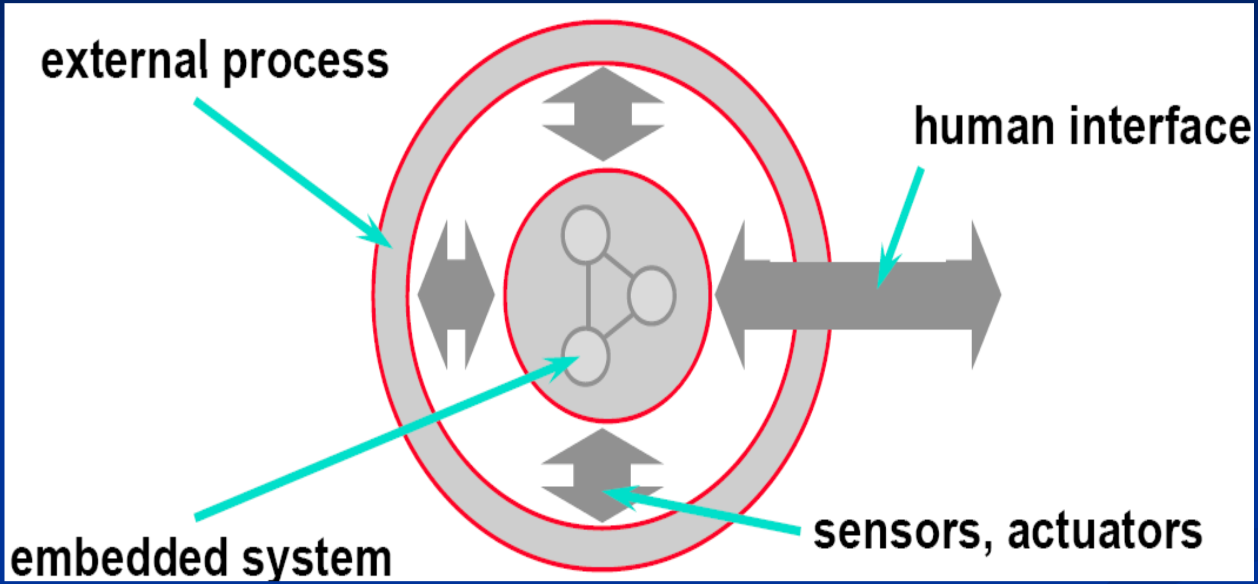
- ***application domain* specific (*not* general purpose)**
- ***tightly coupled*** to their ***environment***

Examples of application domains: automotive electronics, avionics, multimedia, consumer electronics, etc.

Environment: type and properties of input/output information.

Tightly coupled: the environment *dictates* what the system *response* behavior must be. (*"ES cannot synchronize with environment"*)

Embedded Systems



What they do:

- **Sense** environment (input signals)
- **Process** input information
- **Respond** in real-time (output signals)

In Embedded Systems ***time matters***:

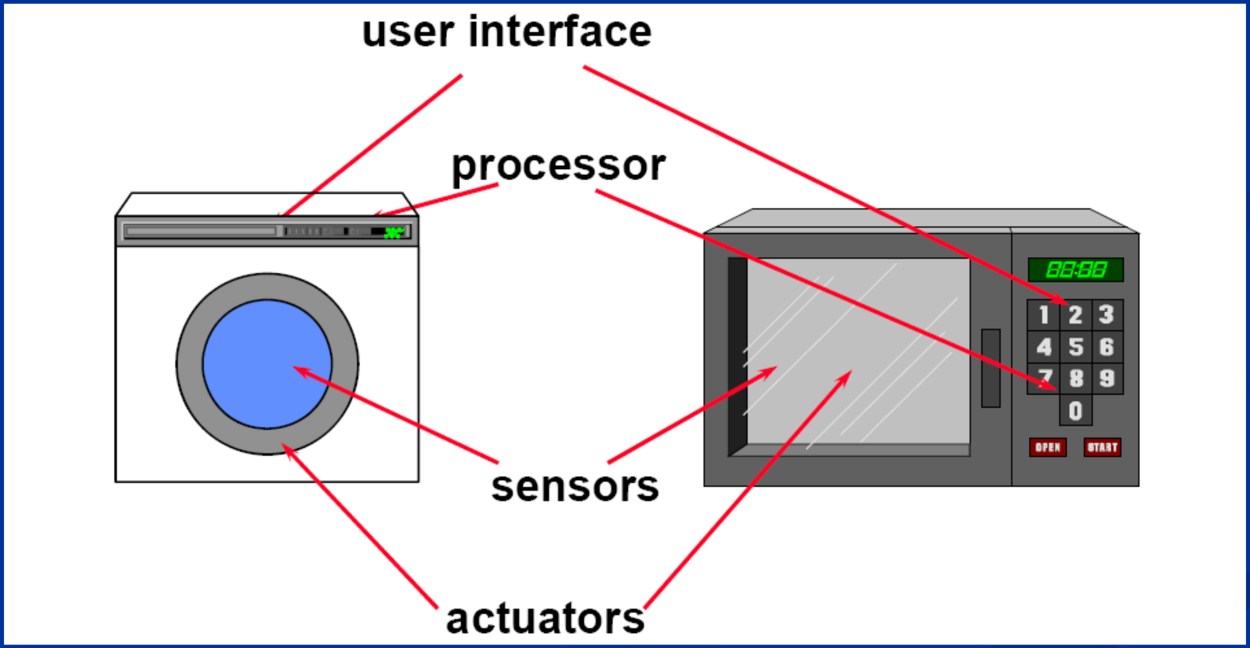
NOT in the sense that information processing should be always very fast

BUT in the sense that information processing should be:

determinate (bounded by definite limits) and *time predictable*

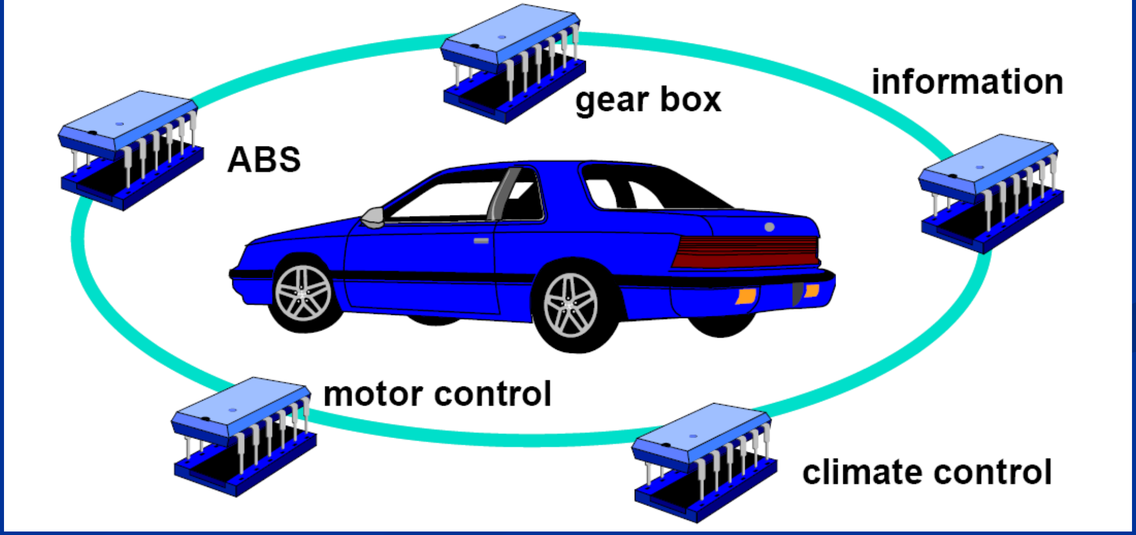
Examples of Embedded Systems: Consumer Electronics

- Examples:
  - Home electronics (washing machine, microwave cooker/oven, …)
  - Video electronics (digital camera, …)



Examples of Embedded Systems: Automotive Electronics

- Functions by embedded processing:
  - ABS: Anti-lock braking systems
  - ESP: Electronic stability control
  - Efficient automatic gearboxes
  - Theft prevention with smart keys
  - Blind-angle alert systems
  - Airbags
  - ... etc ...
- Multiple Processors
  - Up to 100
  - Networked together
- Multiple Networks
  - Body, engine, media, safety

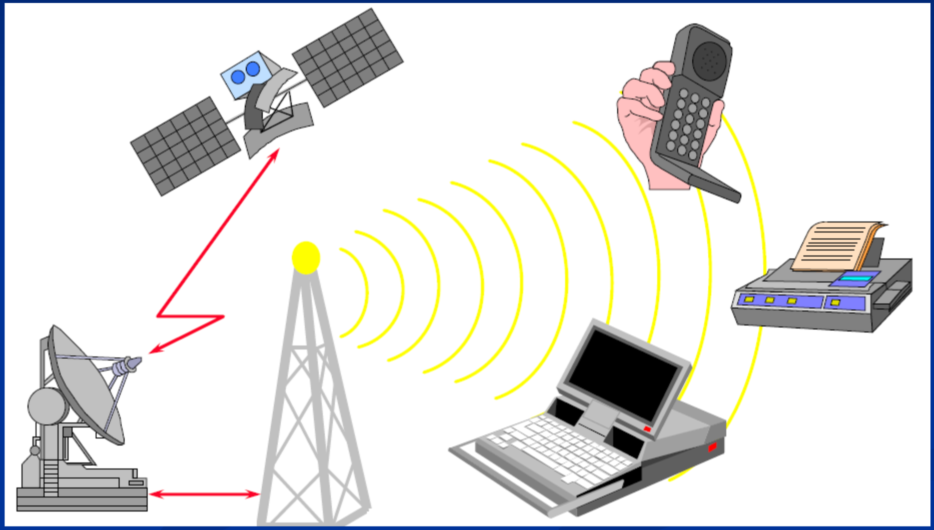


Examples of Embedded Systems: Avionics

- Anti-collision systems,
- Flight control systems,
- Pilot information systems,
- Power supply system,
- Flap control system,
- …

Examples of Embedded Systems: Telecommunication

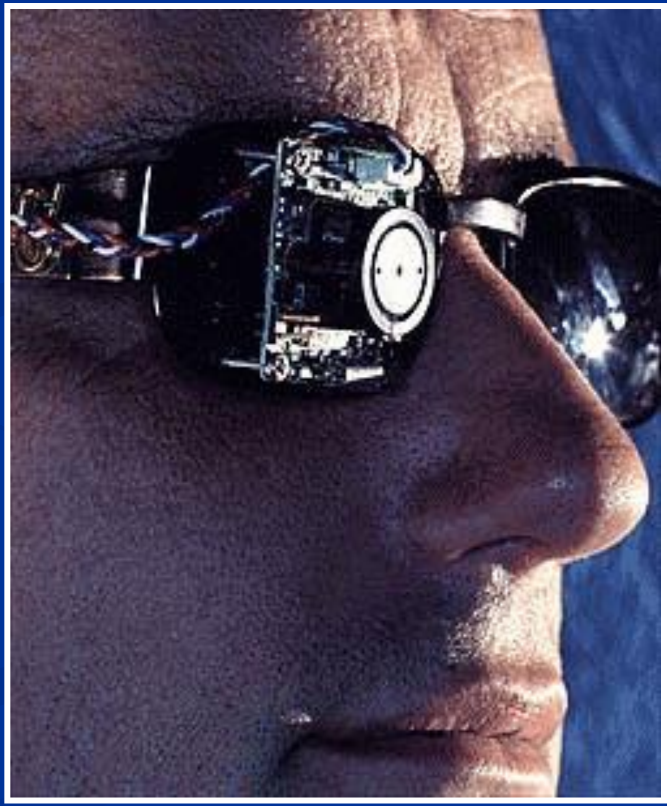
- Information systems
  - Wireless communication
    - Mobile phone
    - Wireless LAN
    - Closed systems for police, ambulance, rescue staff



- Geo-positioning systems
  - Navigation
  - etc

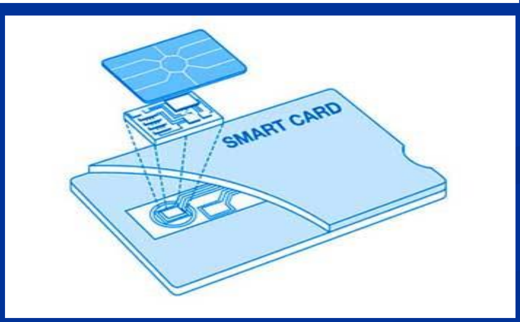
Examples of Embedded Systems: Medical Systems

- For example
  - Artificial Eye
    - Camera is attached to glasses
    - Computer is worn on belt
    - Output directly connected to the brain



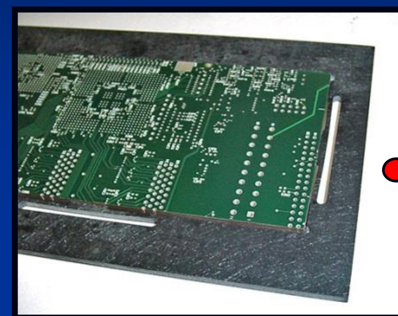
Examples of Embedded Systems: Authentication Systems

- Finger print sensors
- Airport security systems
- Smartpen®
- Access control
- Smart cards

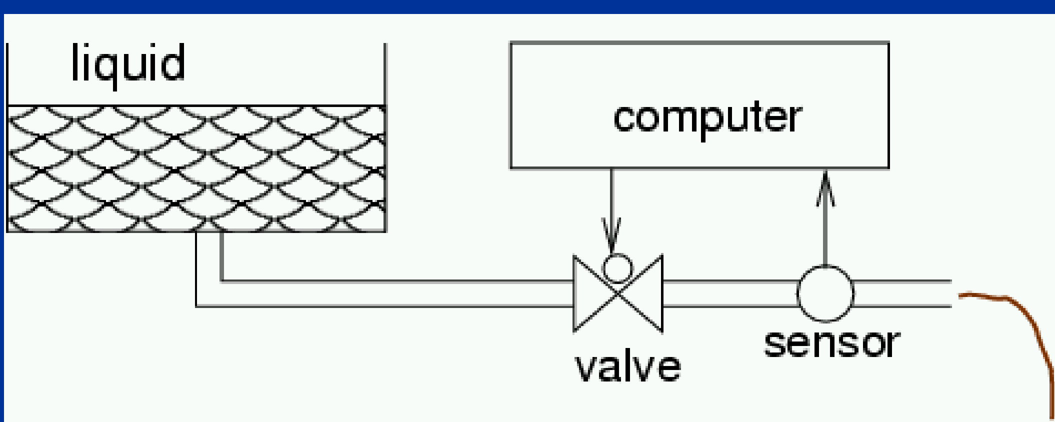


Examples of Embedded Systems: Industrial Production Systems

Chemical Installation



High-speed bonder (SMD/PCB)



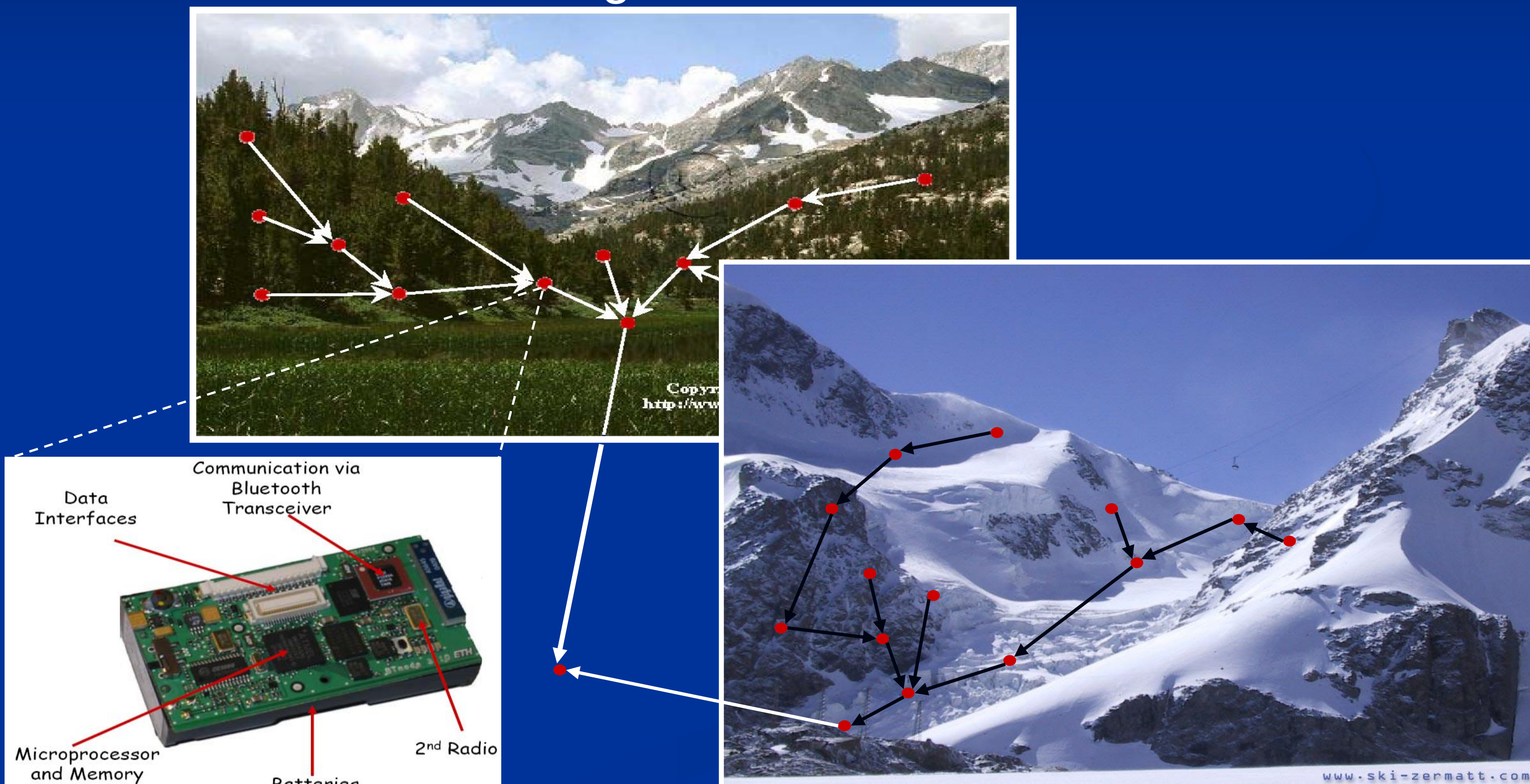
Examples of Embedded Systems: Robotics

NASA's Mars Sojourner Rover

Sony Aibo ERS-110 Robotic Dog

Examples of Embedded Systems: Sensor Networks

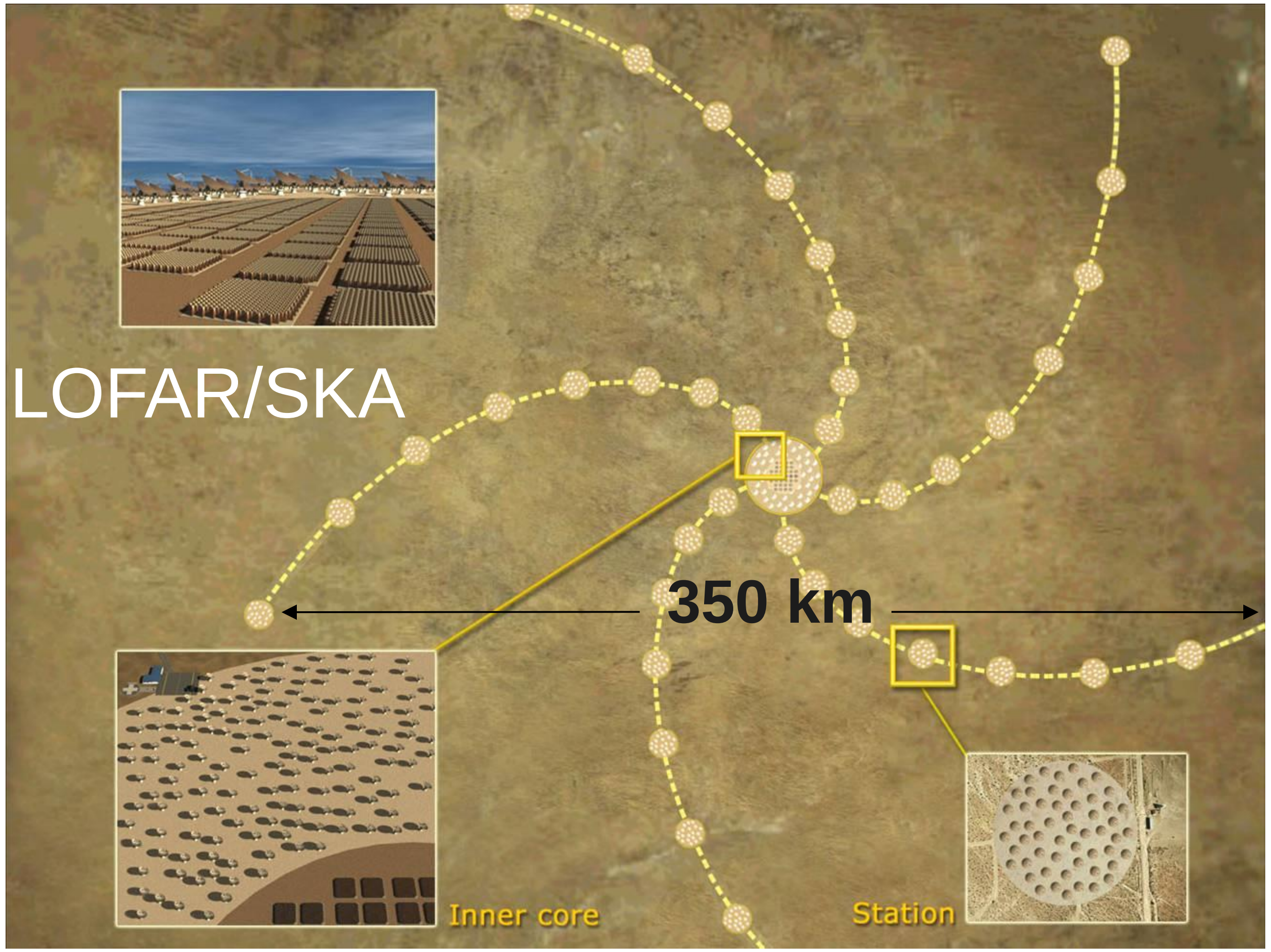
Communicating ES used in civil engineering, buildings, environmental monitoring, traffic, etc.



Examples of Embedded Systems: Very Large Distributed ES

Embedded does not mean SMALL!

Distributed Hierarchical Radio Telescope



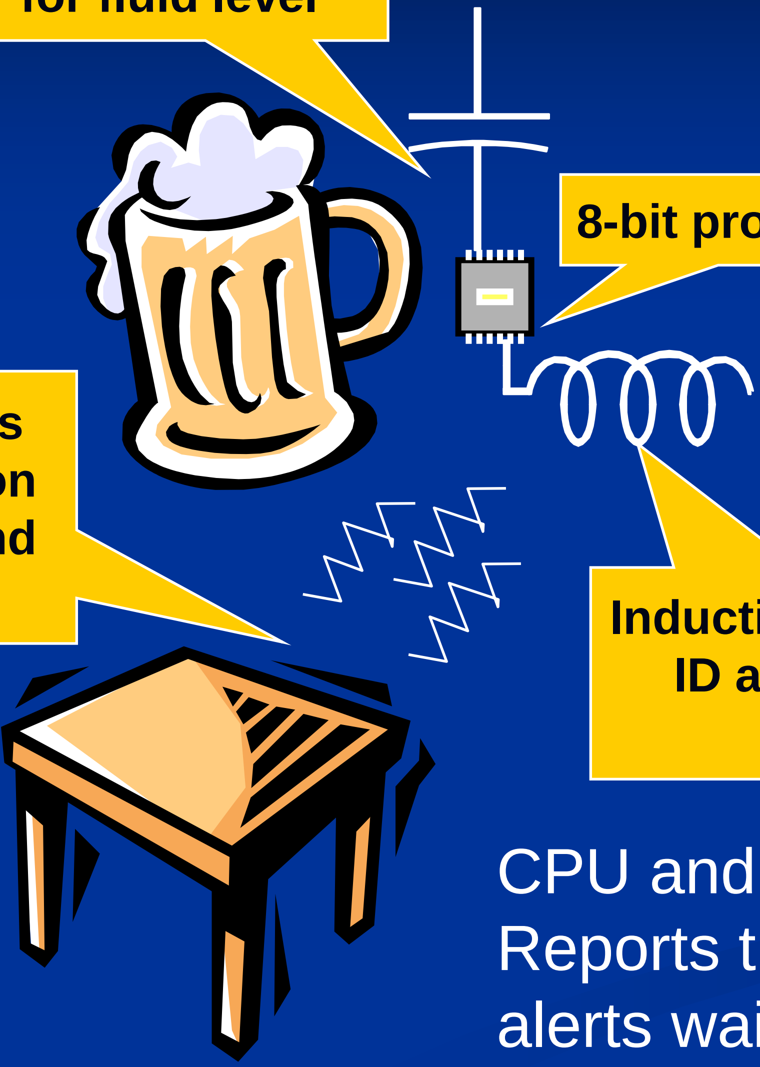
Station contains:
- **100 LF antennas**
- **100 HF antennas**

Inner core contains:
- **supercomputer**

Embedded Systems are Everywhere

- Every object or device that is called “Smart” has an Embedded System in it:
  - Smart Grids
  - Smart Meters
  - Smart Phones
  - Smart TVs
  - etc …
- Attaching an Embedded System to an object makes it “Smart”!
  - Can a Beer Glass become “Smart”?

“Smart” Beer Glass



- Integrates several technologies:
  - Radio transmissions
  - Sensor technology
  - Magnetic inductance for power
  - Computer used for calibration
- Impossible without the computer
- Meaningless without the electronics

CPU and reading coil in the table. Reports the level of fluid in the glass, alerts waiters when close to empty

Characteristics of Embedded Systems (1)

- Must be ***dependable,*** *i.e.*,
  - ***Highly Reliable***
    - ***Reliability:* R(T) =** probability that a system will not fail for a given period of time **T**
  - ***Highly Maintainable***
    - ***Maintainability:* M(d) =** probability that a system can be repaired within ***d*** time units after a failure
  - ***Highly Available***
    - ***Availability:* A(*t*)** = probability that a system is operational at a given point in time ***t***
  - ***Safe:*** no harm to be caused by a failing system
  - ***Secure:*** a system is resilient/protected to/from attacks

- Making the system dependable must not be an after-thought, it must be considered from the very beginning.

- Even perfectly designed systems can fail if the assumptions about the workload and possible errors turn out to be wrong.

Characteristics of Embedded Systems (2)

- Must be ***efficient,*** *i.e.*,
  - ***Energy*** efficient (also when "doing nothing" in standby mode!)
    - Many ES are mobile systems powered by batteries
    - Customers expect long run-times from batteries but
    - Battery technology improves at a very slow rate
  - ***Code-size*** efficient (especially for Systems on a Chip)
    - Typically there are no hard discs or huge memories to store code
  - ***Run-time*** efficient
    - Meet time constraints with the least amount of HW and energy
    - Only necessary HW should be present working at as low as possible Voltage and Clock Frequency
  - ***Weight*** efficient (especially for portable ES)
  - ***Cost*** efficient (especially for high-volume ES)

Characteristics of Embedded Systems (3)

- Many ES must meet ***real-time constraints***
  - The system must ***respond to stimuli*** coming from the environment within the time interval ***dictated*** by the environment.
  - For real-time systems, correct responses arriving too late are wrong.

"A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe" [Kopetz, 1997].

  - All other time-constraints are called ***soft***.
  - A ***guaranteed system response*** has to be explained without statistical arguments.

Characteristics of Embedded Systems (4)

- ES are ***connected to physical environment*** through sensors and actuators
- ES are ***hybrid systems***, i.e., composed of analog and digital parts
- Typically, ES are ***reactive systems***

"A reactive system is one which is in continual interaction with its environment and executes at a pace determined by that environment" [Bergé, 1995].

Characteristics of Embedded Systems (5)

- All ES are ***dedicated systems***
  - ***Dedicated*** towards a certain ***application***:
    Knowledge about the application at design time can be used to minimize resources and to maximize robustness
  - ***Dedicated*** user ***interface:***
    No mice, no large keyboards and fancy monitors

Not every ES has all of the above characteristics, thus

We can define the term "Embedded System" as follows: Information processing systems having most of the above characteristics are called embedded systems.

Comparison

- Embedded Systems
  - Execute few applications that are known at design-time
  - Non programmable by the end user
  - Fixed run-time requirements (additional computing power not useful)
  - Important criteria
    - Cost
    - Power consumption
    - Predictability
    - …

- General Purpose Systems
  - Execute broad class of applications
  - Programmable by the end user
  - Faster is better
  - Important criteria
    - Cost
    - Average speed

Trends in Embedded Systems

- In the past Embedded Systems were called Embedded (micro-)Controllers
- Appeared typically in control dominated applications
  - Traffic lights control
  - Elevators control
  - Washing machines and dishwashers
  - …
- Implemented using a single µProcessor or dedicated HW (sequential circuits)

All this has been rapidly changing!
Let us see How and Why?

Trends in Embedded Systems: Towards Multi-Processor Systems

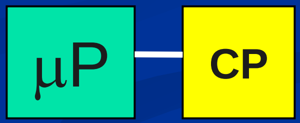
***Complexity* of ES is increasing, thus**

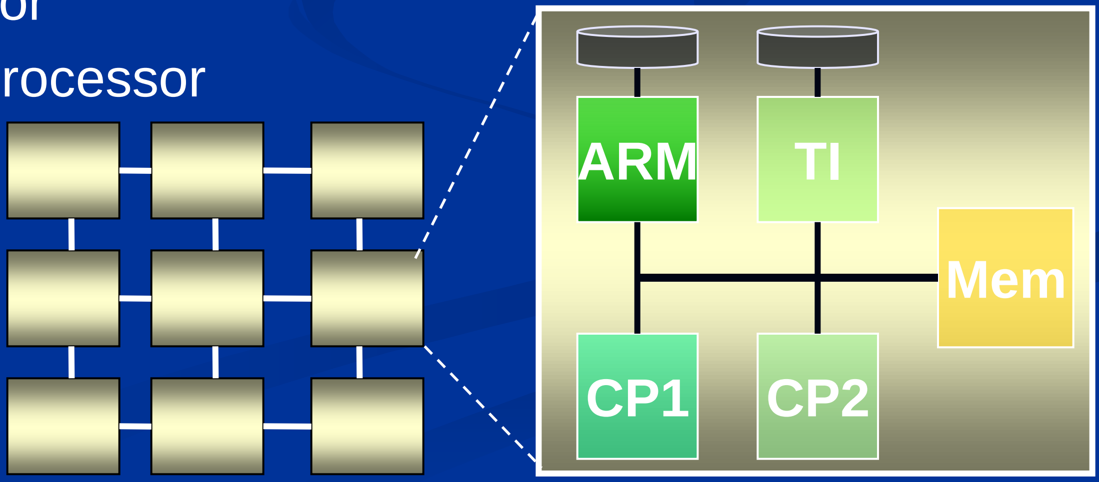
- A single uProcessor is only sufficient for some consumer products/applications
  - Performance requirements are relatively low
- For other systems – such as cars and aircrafts – network of processors is needed
  - Due to performance requirements
  - Due to safety requirements - single failed component should not cause total system failure
- For some systems – such as mobile devices – network of heterogeneous processors is needed
  - Due to run-time efficiency requirements
  - Due to power efficiency

Trends in Embedded Systems:
Higher Degree of Integration on a Chip

Moore's Law: number of transistors that can be placed on a chip doubles approximately every two years

- In 70s, only a Microprocessor, microcontroller on a Chip
- In 90s, System-on-Chip (SoC)
  - Processor + memory + I/O-units + communication structure
- Multi-processor System-on-Chip (MPSoC)
  - Processor – Co-processor
  - (Heterogeneous) Multi-processor
  - Network-on-Chip
    - Identical tiles
    - Scalable system

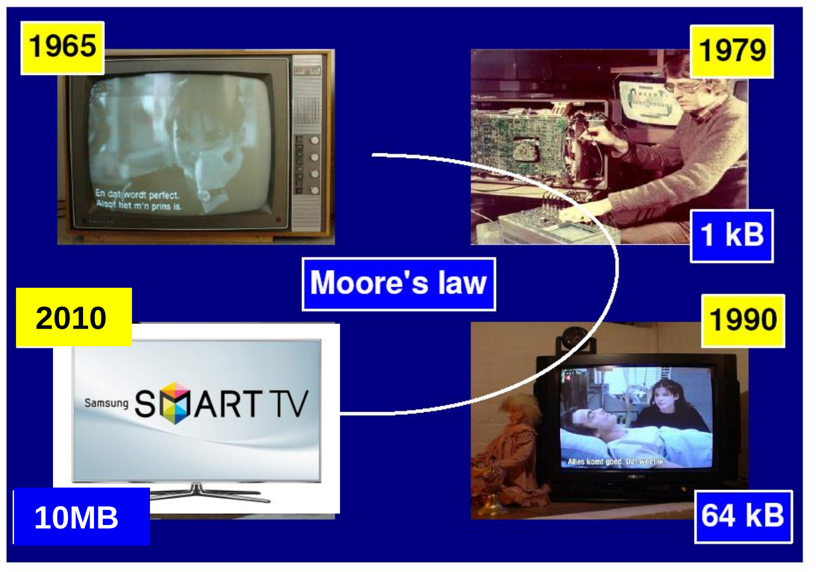




Trends in Embedded Systems:
Software Increasing (amount and complexity)

Implementing ES in specialized HW brings
-- *lack of flexibility* (changing standards)
-- *very expensive masks*, thus

- Most of the functionality has been implemented in software
  - On the average, a human "touches" about 50 to 100 micro-processors per day
  - Average car has 15 micro-processors, luxurious one ~ 100!
- Exponential increase in software complexity



What is Embedded Systems Design?

Embedded Systems Design is NOT just a special case of either hardware or software design! <u>It is Multidisciplinary!!!</u>

- **<u>Computer Science</u>** deals with (software) functionality
  - Independent of any HW implementation and physical realization
- **<u>Computer/Electrical Engineering</u>** deals with hardware
  - Logical implementation and Physical realization
- **<u>Embedded Systems design discipline</u>** needs to combine these two approaches because
  - Functional behavior (deadlock-free execution, functional correctness, etc.) is provided by Computer Science
  - Non-functional behavior (performance, cost, power, robustness, etc.) is crucial and provided by Computer Engineering

Future of Embedded Systems

- Embedded Systems are everywhere
- ES market will be much larger than the market of PC-like, general purpose systems
  - Information processing is more and more moving away from just PCs/servers to embedded systems
- ES provide basic technology for Ubiquitous/Pervasive computing:
  - Information processing thoroughly integrated into everyday objects and activities
  - Key goal is to make information available *anytime*, *anywhere*
  - Building Ambient Intelligence into our environment
- ES are the Edge of Internet of Things (IoT)!

☞ **The future is embedded, embedded is the future**