TECHNISCHE UNIVERSITÄT
CHEMNITZ

Fakultät für Informatik
Professur Technische Informatik

Informatik

Professur Technische Informatik
Prof. Dr. Wolfram Hardt

Hardware /Software Codesign I

Introduction

Prof. Dr. Wolfram Hardt
Dipl.-Inf. Michael Nagler

Contents

- Embedded Systems (ES)
  - characterisation
  - mechatronics
  - requirements
  - classification
- Development and CoDesign
- Content of Lecture
- Organisational

Characterisation of Processor (SW)

- executes sequential commands
- consists of
  - set of registers, instruction pointer register
  - computational unit(s)
  - memory access (data and program)
  - control unit
- advantages
  - easy to program
  - can implement almost every application
  - application can be changed easily
- disadvantages
  - sequential (slow)
  - high energy consumption

Characterisation of (Digital) Hardware

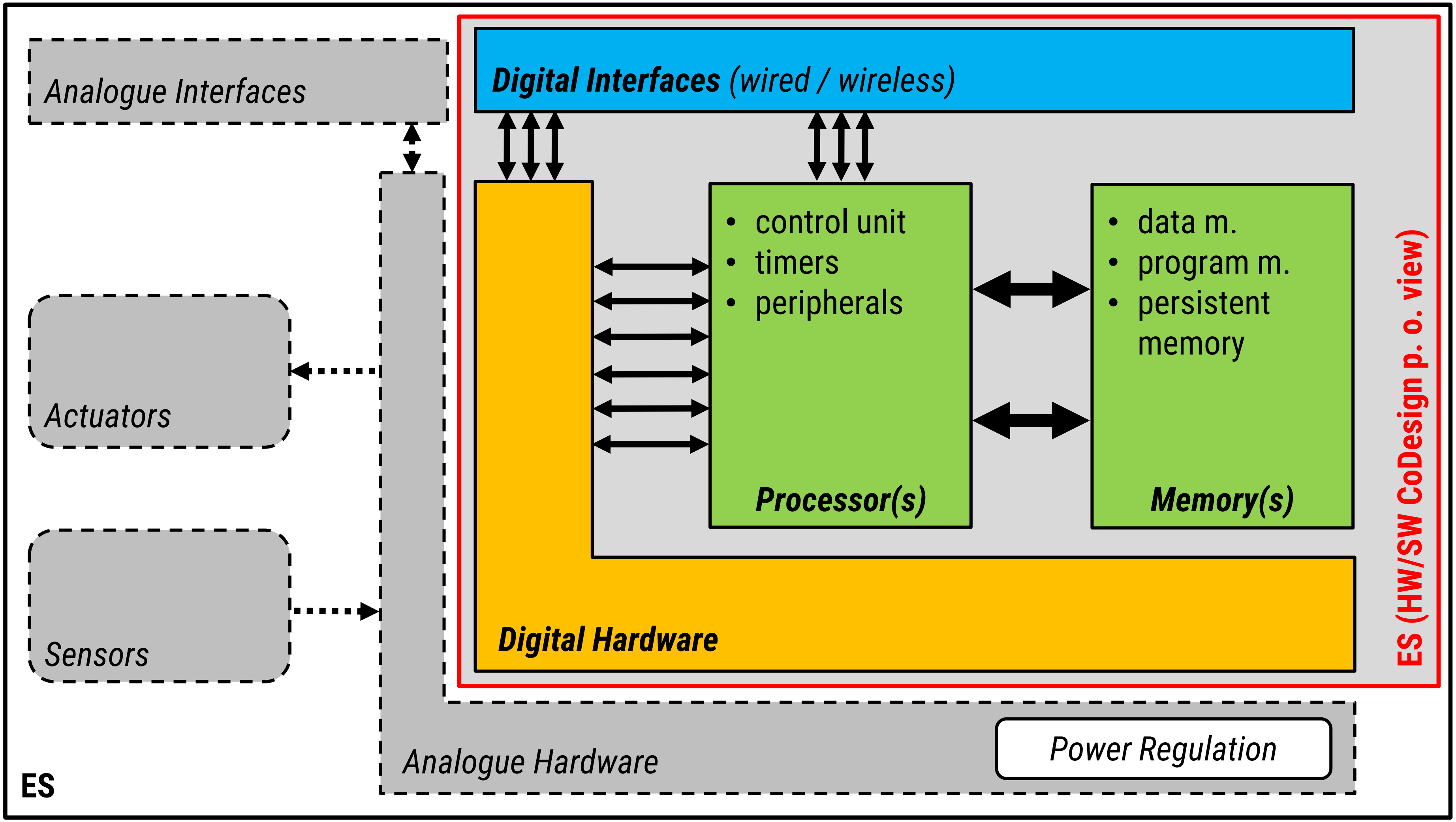
- implements finite state machines with elements of RTL
- can be implemented differently
  - dedicated board layouts
  - FPGA
  - ASIC
- advantages
  - all RTL elements work in parallel
  - optimisable energy consumption
- disadvantages
  - difficult and expensive development
  - changes of implementation / layout difficult

Characterisation of ES

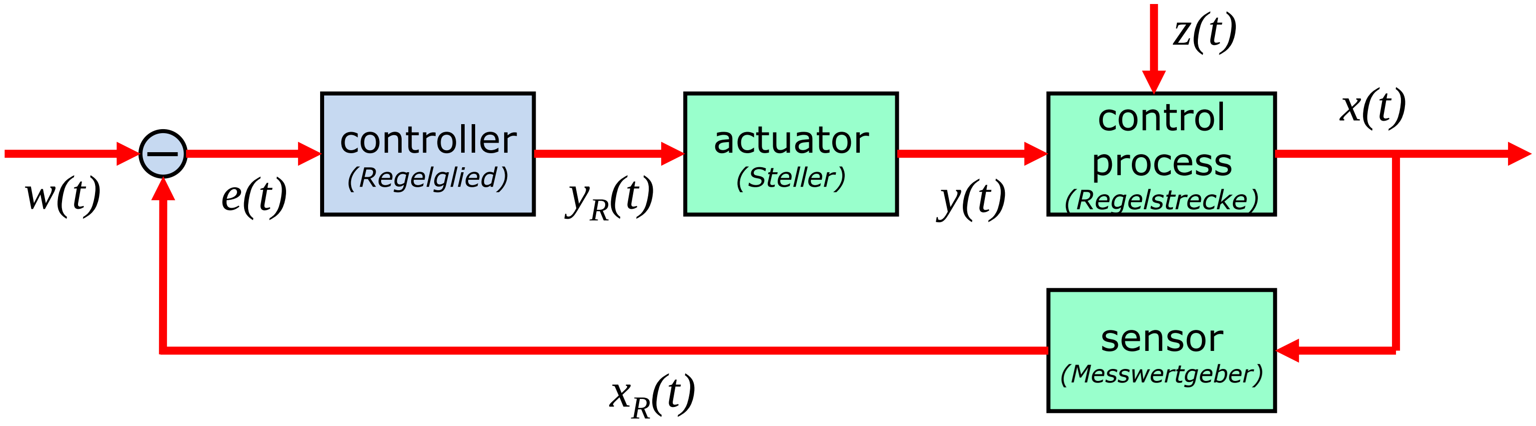
- Hansson[1]: *“Embedded systems are computers not looking like computers.”*
- Broy[2]: *“An embedded systems is a SW/HW unit connected via sensors and actors with a whole system and scans, controls and adjusts therein.”*
- more often used properties:
  - reactive, hybrid and distributed systems
  - real-time requirements
- human user often interacts unconscious with these systems, they are invisible for him
- Examples: mobile phone, TV, ECUs in cars, telephone switchboard, …

[1] Hans Hansson, Mälardalen University
[2] Manfred Broy, TU München

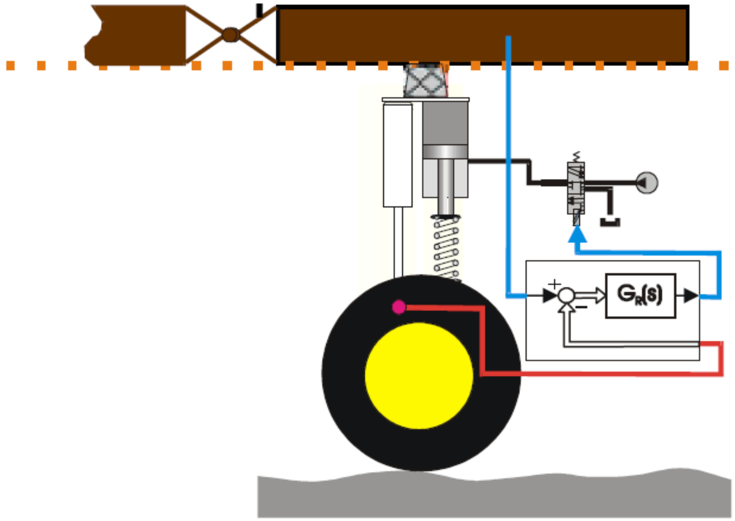
Structure of Embedded Systems



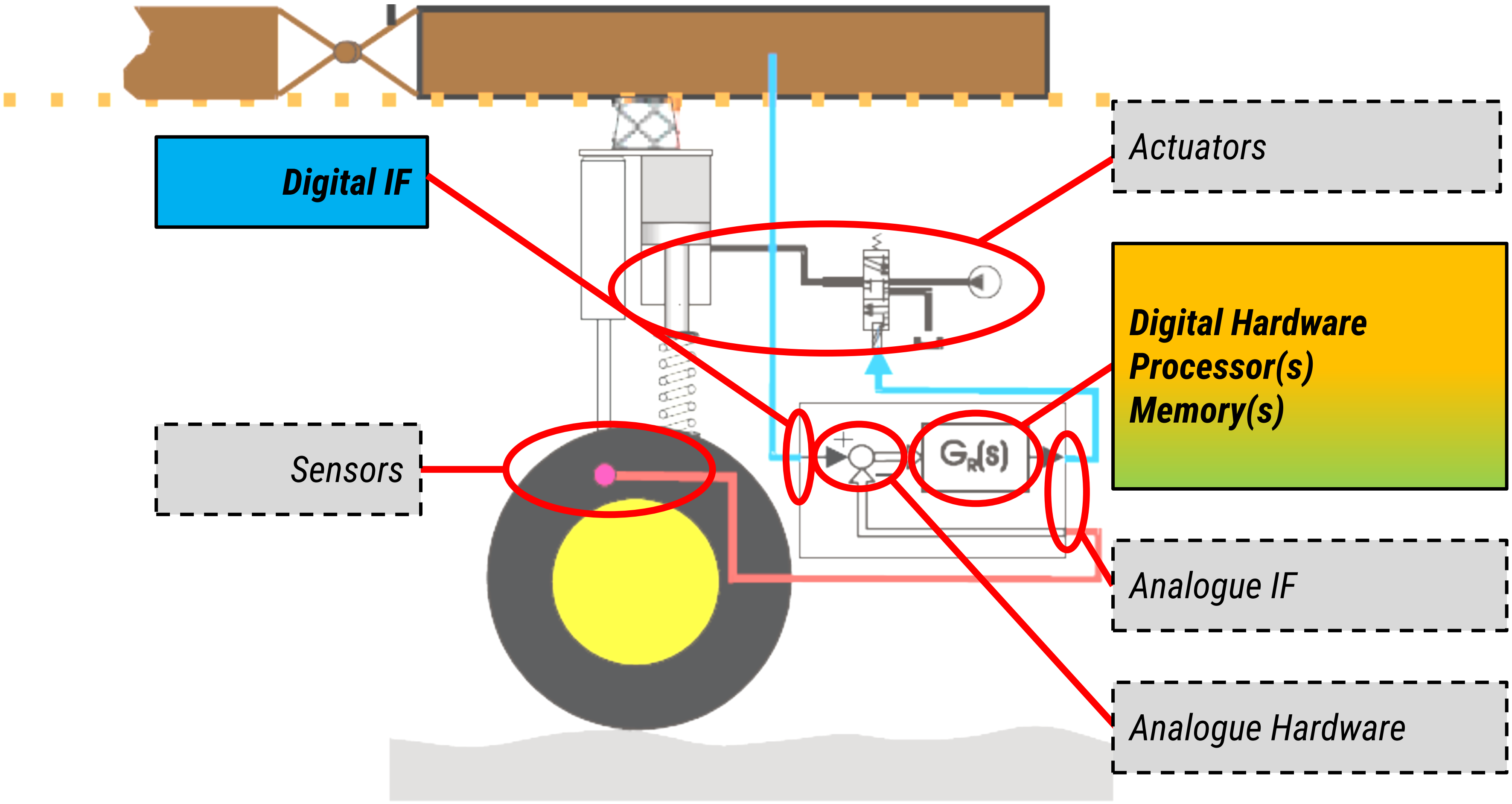
Control Loop



reference variable	*(Führungsgröße)*	w
error signal	*(Regeldifferenz)*	e
controller output variable	*(Reglerausgangsgröße)*	y_R
manipulated variable	*(Stellgröße)*	y
disturbance variable	*(Störgröße)*	z
control variable	*(Regelgröße)*	x
measured control variable	*(Erfasste Regelgröße)*	x_R



Realisation as ES



Digital IF

Actuators

Digital Hardware
Processor(s)
Memory(s)

Sensors

Analogue IF

Analogue Hardware

Implementation of ES

Domain of ES	Behaviour (developer point of view)	Typical Implementation (developer point of view)
Actuators / Sensors	converter between voltage / current and other physical values (kinetics, optics, …)	• purchase
Analogue HW / IF	physics	• differential equations • simulation by framework (SPICE, …)
Digital Hardware	finite state machine	• Boolean equations • HW description language synthesis framework
Processor / Memory	sequential execution of machine code	• assembly language • high-level language + compiler
Digital Interfaces	(partial) embedded system	• standardised → purchase • integrated in other parts • synthesise

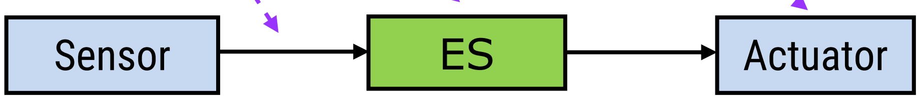
basic knowledge / THIS LECTURE

Requirements for ES

- functional requirements
  - data acquisition
  - digital control
    - calculation of control values for actuator
    - …
  - (graphical) user interface (GUI)
    - displaying of current system variables and states
    - logging
- temporal requirements
- reliability requirements

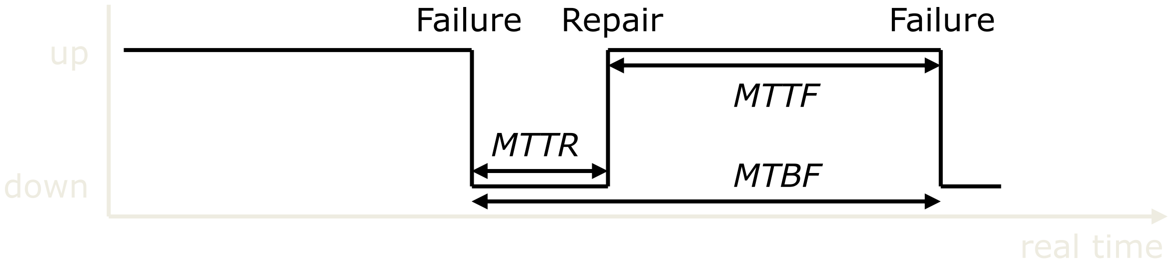
Temporal Requirements

- important values:
  - delay (d^{object}) time between setup of new actuating variables and beginning reaction of controlled object
  - response time time needed by controlled object to get to target value
  - sampling period (d^{sample}) frequency of actual value scan
  - processor delay ($d^{computer}$) necessary: $d^{computer} < d^{sample}$
  - dead time $= d^{object} + d^{computer}$



Reliability Requirements

- reliability
  - probability that ES is able to achieve the specified service to point in time t
- safety
  - reliability in respect of critical errors
- maintainability
  - value for repair time after an error
- availability



- security
  - protection against unauthorised usage

Classification of ES

Trade-Off

Fail-Safe ↔ **Fail-Operational**

error detection leads to change in safe state

minimal functionality is assured even in case of error

Guaranteed-Response ↔ **Best-Effort**

service is also in case of maximum load or error assured

system tries to execute services as good as possible – no warranty!

Resource-Adequate ↔ **Resource-Inadequate**

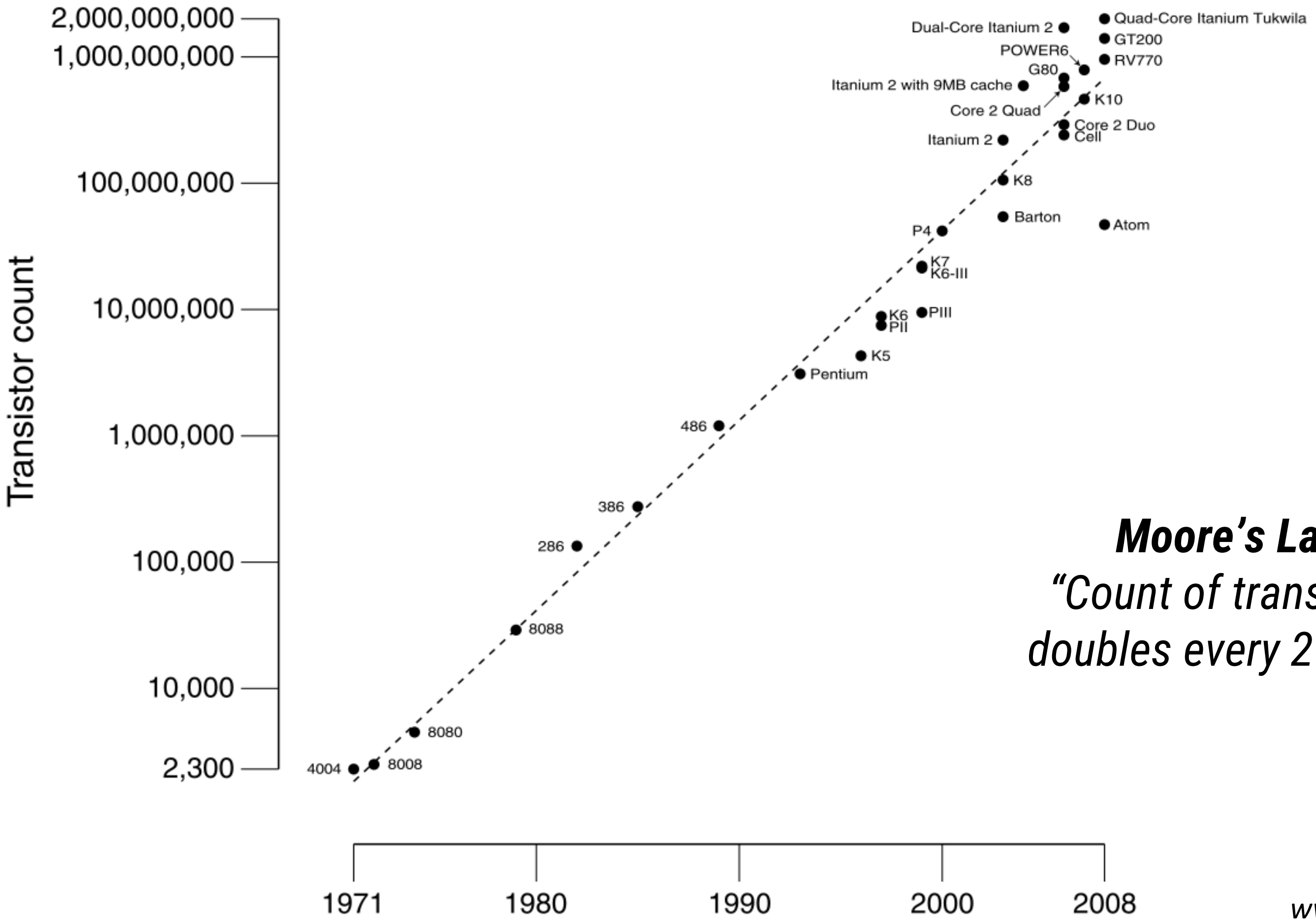
enough resources to execute service in every case

enough resources to execute service in usual cases

Contents

- Embedded Systems (ES)
  - characterisation
  - mechatronics
  - requirements
  - classification
- Development and CoDesign
- Content of Lecture
- Organisational

Increasing Technological Complexity

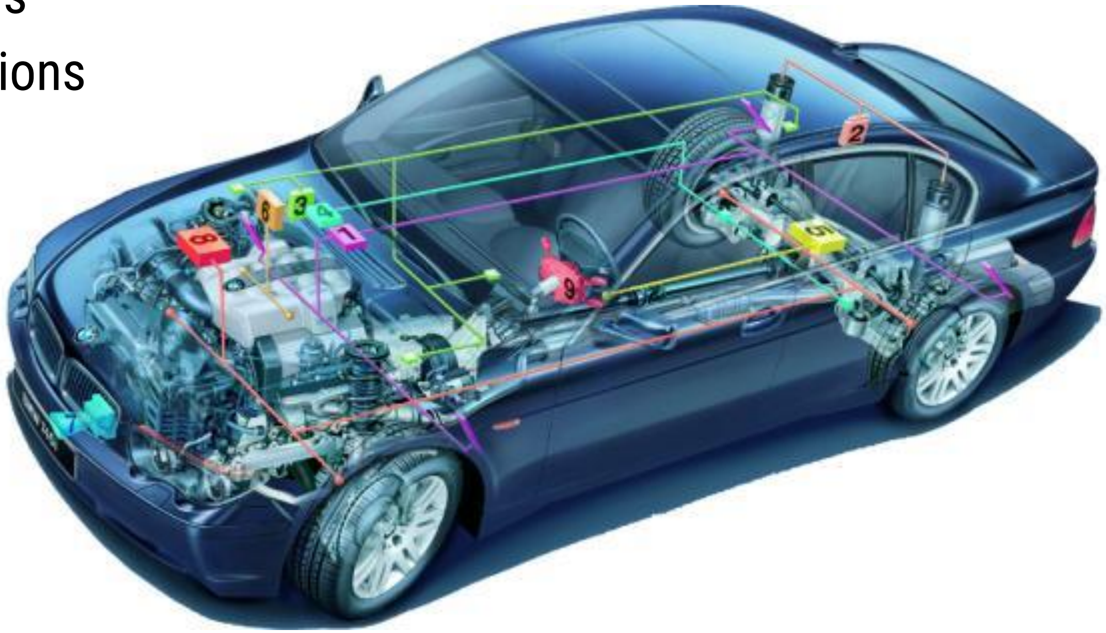


Moore's Law:
"Count of transistors doubles every 2 years."

www.wikipedia.org

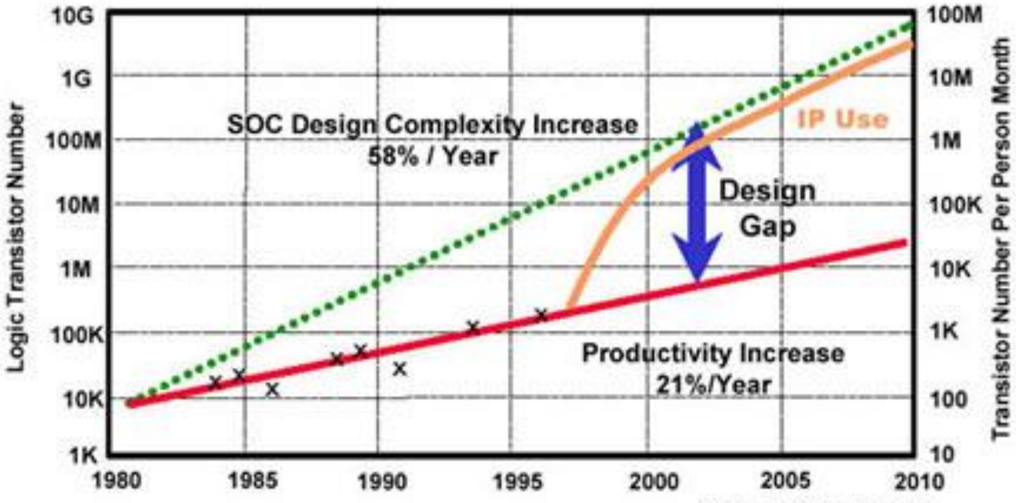
Increasing Functional Requirements

- in modern cars more than 100 digital systems (control boxes)
  - motor control
  - safety systems
  - comfort functions
  - Infotainment



Design Gap

- Can we develop systems which needs this complexity?



Source : SEMATECH

- problems to solve
  - develop in an efficient way
  - ensure security
  - ensure reliability

Common Question (of Students)

1. We have powerful high-level programming languages and compilers to write powerful programs for microprocessors!

2. We have powerful programs to support the development of digital systems and software!

Why do we have to talk about the development of Digital and Embedded Systems?

That‘s why!

1. Someone has to develop the microprocessors that execute programs.

   Someone has to develop compilers to generate executable machine code of the programs for the microprocessor.

   If you want to write good programs for micro-processors, you need to know how it work.

2. Someone has to develop the development tools.

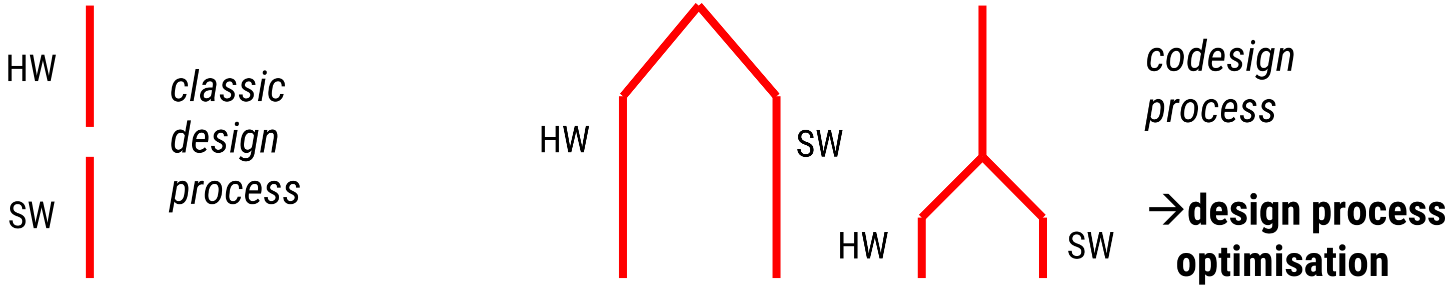
   If you want to implement good Digital and Embedded systems you need to know how the tools work.

What is CO-Design?

CO	mplex	- komplex
	ncurrent	- nebenläufig
	rrect	- korrekt
	ordinated	- koordiniert

What does it mean?

- integrated development of embedded system consisting of
  - hardware-components (HW) ***and***
  - software-components (SW)



classic design process

codesign process

→**design process optimisation**

- special requirements to designer
  - analyse the restrictions of HW and SW
  - evaluation of alternative development solutions
  - integration of HW and SW components

Restrictions of HW/SW

- general purpose systems
  - example: PC, Workstation
  - *trade-off:*

processor ←→ **compiler/ operating system**

- embedded systems
  - example: mobile phone, motor control unit
  - *trade-off for special processors:*

processor ←→ **compiler**

  - *trade-off for system development:*

dedicated HW ←→ **processors**

And what is MY advantage?

- understanding of modern system development
- insight into a present field of research
- alternatives in HW/SW implementations
- algorithms for many application areas
- jobs

Contents

- Embedded Systems (ES)
  - characterisation
  - mechatronics
  - requirements
  - classification
- Development and CoDesign
- Content of Lecture
- Organisational

Lecture Content

- system development – models and methods
- target architectures for HW/SW systems
- compiler and code generation
- partitioning, generally
- HW/SW partitioning

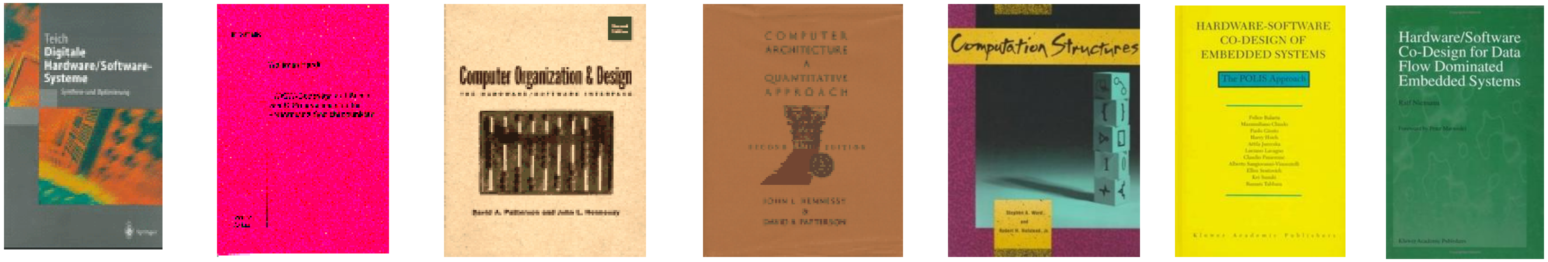
HW/SW Codesign I

- estimation of design parameters
- Co-Simulation
- Co-Specifiation (SystemC)
- interfaces
- interface synthesis

HW/SW Codesign II
(next semester)

Literature

- Teich, Jürgen: Digitale Hardware/Software-Systeme. Berlin: Springer, 1997
- Hardt, Wolfram: HW/SW-Codesign auf Basis von C-Programmen unter Performanz-Gesichtspunkten. Aachen: Shaker Verlag, 1996
- Patterson, David A.; Hennessy, John L.: Computer Organization and Design: The Hardware/Software Interface. 2. Auflage. Oxford: Elsevier Books, 1997
- Hennessy, John L.; Golderberg, David; Patterson, David A.: Computer Architecture: A Quantitive Approach. 2. Auflage. San Francisco: Morgan Kaufmann Publishers Inc, 1996
- Ward, Stephen A., Halstead, Robert H.: Computation Structures. Cambridge: The MIT Press, 1990
- Balarice, Felice: Hardware-Software Co-Design of Embedded Systems – the POLIS approach. Boston: Kluwer Academic, 1997
- Niemann, Ralf: Hardware/Software Co-Design for Data Flow Dominated Embedded Systems. Boston: Kluwer Academic, 1998


Teich
Digitale Hardware/Software-Systeme
Computer Organization & Design
COMPUTER ARCHITECTURE A QUANTITATIVE APPROACH
Computation Structures
HARDWARE-SOFTWARE CO-DESIGN OF EMBEDDED SYSTEMS
The POLIS Approach
Hardware/Software Co-Design for Data Flow Dominated Embedded Systems

Organisational (I)

- **Lecture (weekly):** Prof. Dr. Wolfram Hardt

Thursday	09:15 – 10:45	1/201	English

- **Exercises (weekly):** Michael Nagler, Kwame Nyarko

Tuesday	13:45 – 15:15	1/205	English
Wednesday	09:15 – 10:45	1/205	English
Thursday	07:30 – 09:00	1/367a	**German**
Monday	15:30 – 17:00	1/367a	**German**
Thursday	07:30 – 09:00	1/346	English
Monday	09:15 – 10:45	1/205	English

Exercises will start in week 44: 26th October, 2015

Organisational (II)

- **Exam:** written test (English or German), 90 minutes
- **Contact:** Michael Nagler, Room 1/023a, Tuesday: 12:30 – 13:30
- **ALL mails:** **ce-teaching@informatik.tu-chemnitz.de**
- **Content:** **www.tu-chemnitz.de/cs/ce/lectures/lectures.php**
  - **register for exercise group** (by OPAL)
  - download slides **before** lecture (They'll be incomplete, so you have to complete them!)
  - download and **prepare** exercise sheets