

# EECS 222: Embedded System Modeling Lecture 1

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering  
Electrical Engineering and Computer Science  
University of California, Irvine

## Lecture 1: Overview

- Course Overview
  - Context, content
- Course Administration
  - Schedule, assignments, communication
- Introduction to Embedded System Design
  - Embedded computer systems
  - Levels of abstraction
  - System design flow
- Project Assignment 1
  - Setup and introduction to application example

## Course Context

- Graduate courses on Embedded System Design
  1. **EECS 222: Embedded System Modeling**
  2. **EECS 225: Embedded System Design**
  3. **EECS 226: Embedded System Software**
  4. **EECS 227: Cyber-Physical System Design**

## Course Context

- Graduate courses on Embedded System Design
  1. **EECS 222: Embedded System Modeling**
    - Computation models for embedded systems.
    - System-level specification and description languages.
    - Concepts, requirements, examples.
    - Embedded system models at different levels of abstraction.
    - Modeling of test benches, design under test, IP components.
    - Discrete event simulation, semantics, and algorithms.
  2. **EECS 225: Embedded System Design**
  3. **EECS 226: Embedded System Software**
  4. **EECS 227: Cyber-Physical System Design**

## Course Context

- Graduate courses on Embedded System Design
  1. **EECS 222: Embedded System Modeling**
  2. **EECS 225: Embedded System Design**

Embedded system design flow and methodology.  
Design space exploration. Co-design of hardware and software, embedded architecture and network exploration and synthesis.  
System software/hardware interface generation.  
Real-time constraints, specification-to-architecture mapping, design tools and methodologies.
  3. **EECS 226: Embedded System Software**
  4. **EECS 227: Cyber-Physical System Design**

EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

5

## Course Context

- Graduate courses on Embedded System Design
  1. **EECS 222: Embedded System Modeling**
  2. **EECS 225: Embedded System Design**
  3. **EECS 226: Embedded System Software**

Embedded system software concepts, requirements, examples, for engineering application such as multi-media and automotive.  
Software generation methodology.  
Algorithmic specification, design constraints.  
Embedded operating systems.  
Static, dynamic, real-time scheduling.  
Input/output, interrupt handling.  
Code generation, compilation, instruction set simulation.
  4. **EECS 227: Cyber-Physical System Design**

EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

6

## Course Context

- Graduate courses on Embedded System Design
  1. **EECS 222: Embedded System Modeling**
  2. **EECS 225: Embedded System Design**
  3. **EECS 226: Embedded System Software**
  4. **EECS 227: Cyber-Physical System Design**

Model-based design of cyber-physical systems including, e.g., plant, sensing, control, actuation, embedded hardware/software, communication, real-time analysis, various levels of simulation (MILS, SILS, HILS), tools and methodologies for automatic synthesis, and application from various interdisciplinary domains.

## EECS 222 Course Content

1. Embedded system concepts, abstraction levels, computational models
2. The SpecC system-level description language
3. The SystemC system-level description language
4. Embedded system specification, modeling guidelines
5. Validation, execution and simulation semantics
6. Top-down design methodology
7. System-level architecture modeling
8. Embedded system communication modeling
9. Cycle-accurate modeling, implementation
10. UML and other system-level description languages

## Course Administration

- Course web pages at <http://eee.uci.edu/17s/18425/>
  - Instructor information
  - Course description and policies
  - Objectives and outcomes
  - Contents and schedule
  - Resources and communication
  - Assignments

## Embedded System Design

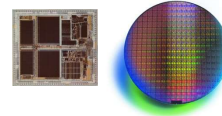
- Embedded Computer Systems
- System-on-Chip (SoC) Design
- Abstraction Levels
- Embedded System Design Flow

## Embedded Computer Systems

- Computers are ubiquitous, omnipresent...



- *System-on-Chip (SoC) Design*: Design of complex embedded systems on a single chip



EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

11

## Embedded Systems

- System embedded into cyber-physical system
  - Constraints from external input (often real-time)
  - Application specific (not general purpose)
- Omnipresent in our environment
  - Pervasive in many application domains
  - In 2005 [Source Netrino]
    - Only 2% of all processors in workstations
    - Remaining 8.8 billion in embedded systems
  - Most computers are embedded systems!



Source: P. Chou, UCI



Source: Edumaticator



Source: Miele



Source: Philips



Source: www.trouper.com



Source: www.medicacorp.com/

EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

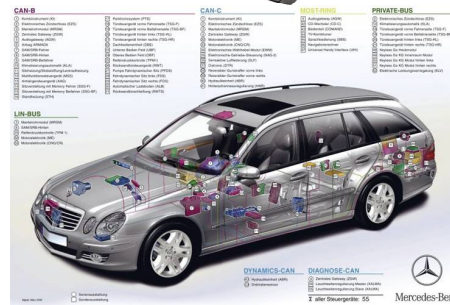
12

## Embedded System Design

- Design challenges
  - Often mobile
    - Battery powered (low power)
  - Often highly reliable
    - Extreme environment (e.g. temperature)
  - High performance constraints
    - Often real-time requirements
  - High complexity
    - E.g. Mercedes Benz E-class
      - 55 electronic control units
      - 5 communication busses
  - Tightly coupled
    - Software
    - Hardware
  - Rapid development for low price...



Source: Motorola Inc



Source: Daimler

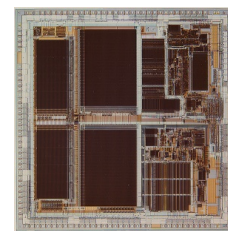
(c) 2017 R. Doemer

13

EECS222: Embedded System Modeling, Lecture 1

## Embedded System Design

- Design Advantages
  - *Application known at design time*
  - *Environment known at design time*
  - Allows for customized / optimized solution
    - Improved performance
    - More functionality
    - At lower power
- Custom Platform, SW and HW components
  - Multi-Processor System-on-Chip (MPSoC),
    - Complete embedded system integrated on a chip
  - General-purpose and application-specific processors
  - Application Specific Integrated Circuit (ASIC)
  - Field Programmable Gate Array (FPGA)
  - Circuit board with off-the-shelf-components



Source: simh.trailing-edge.com

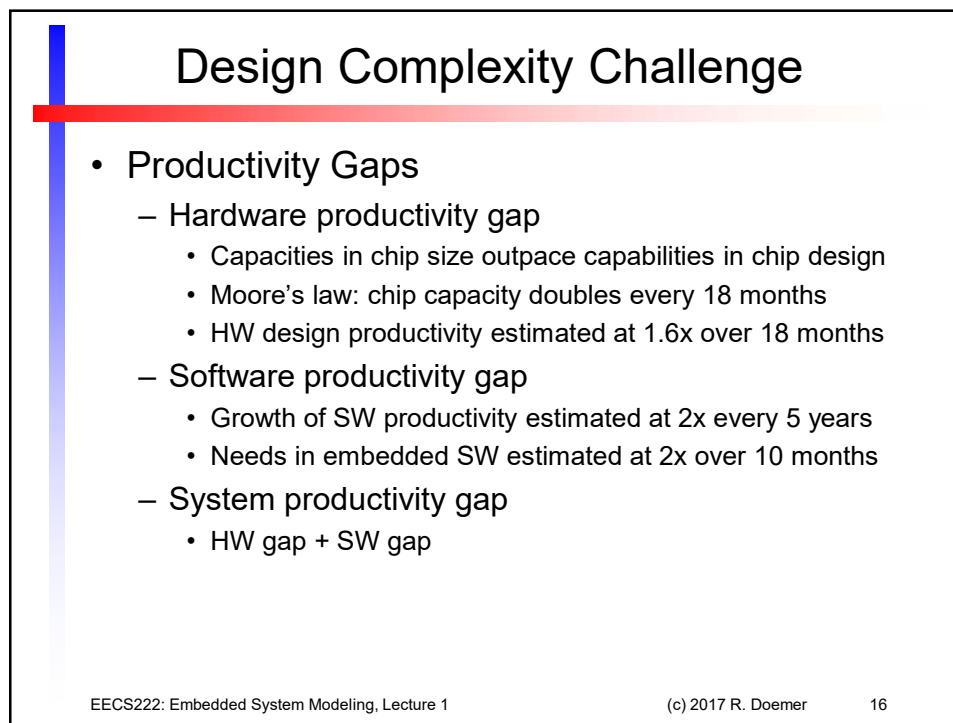
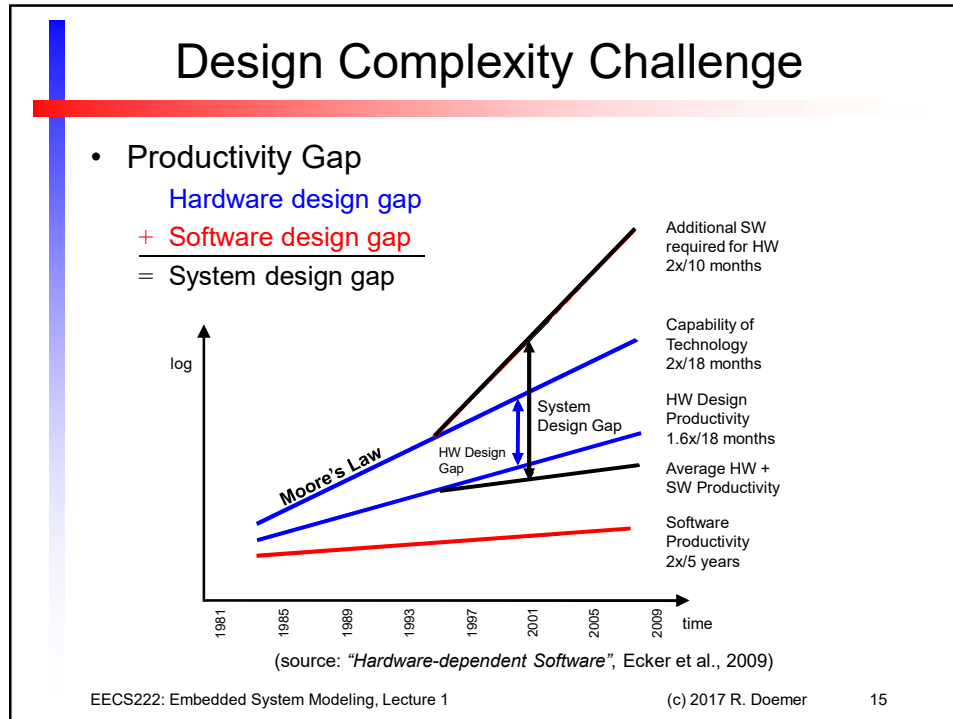


Source: Xilinx

EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

14





## Abstraction Levels

- System-on-Chip (SoC) design faces tremendous increase of design complexity

Level	Number of components
System	1E0
Algorithm	1E1
RTL	1E2
Gate	1E3
Transistor	1E4
	1E5
	1E6
	1E7

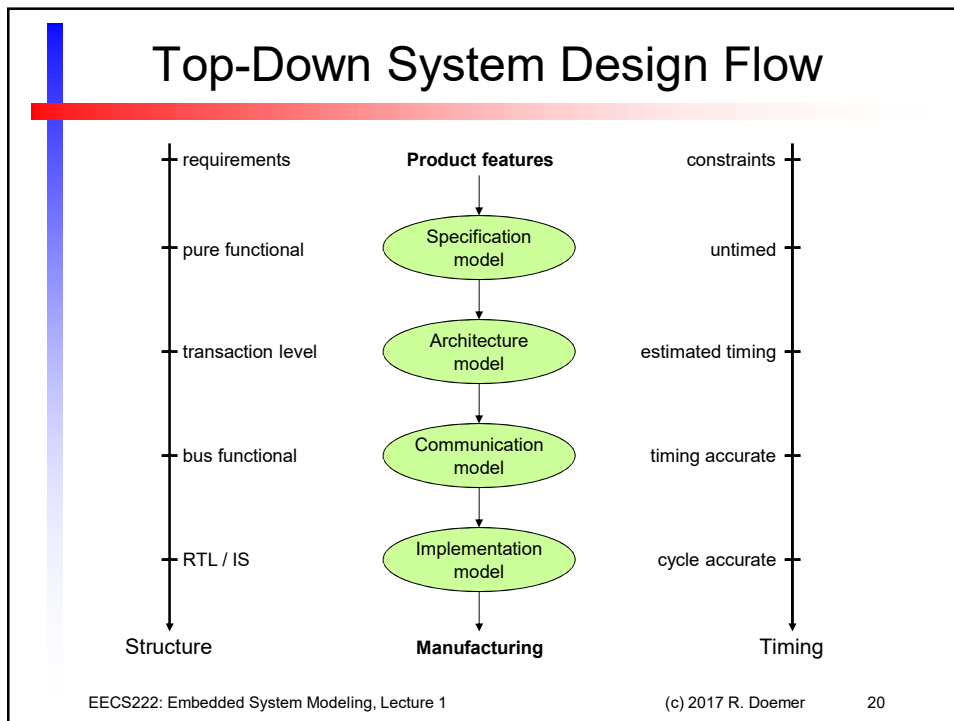
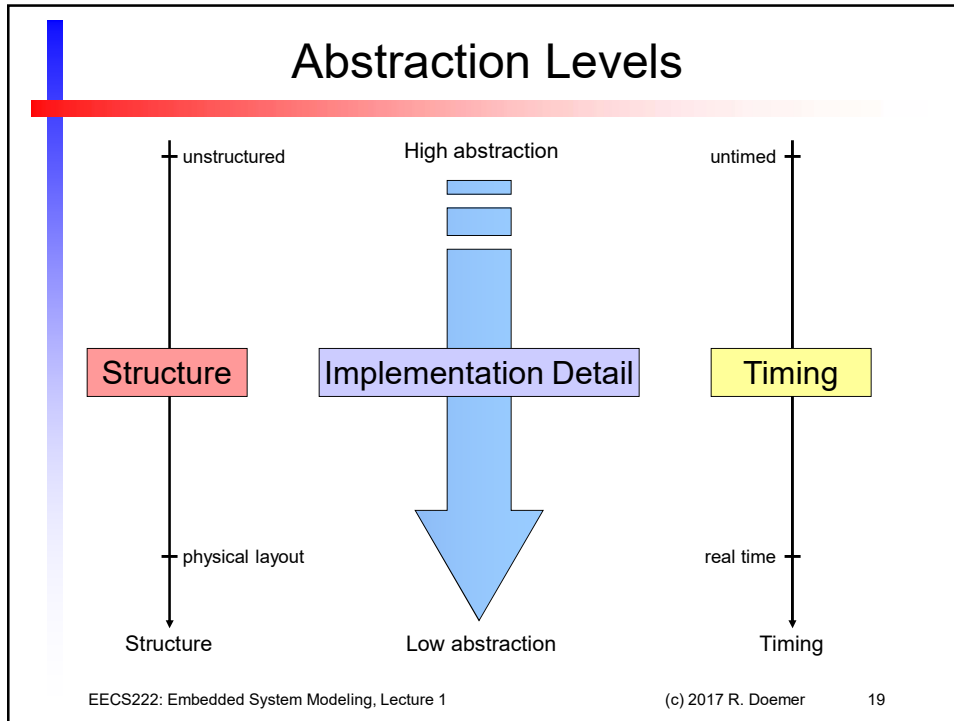
EECS222: Embedded System Modeling, Lecture 1
(c) 2017 R. Doemer
17

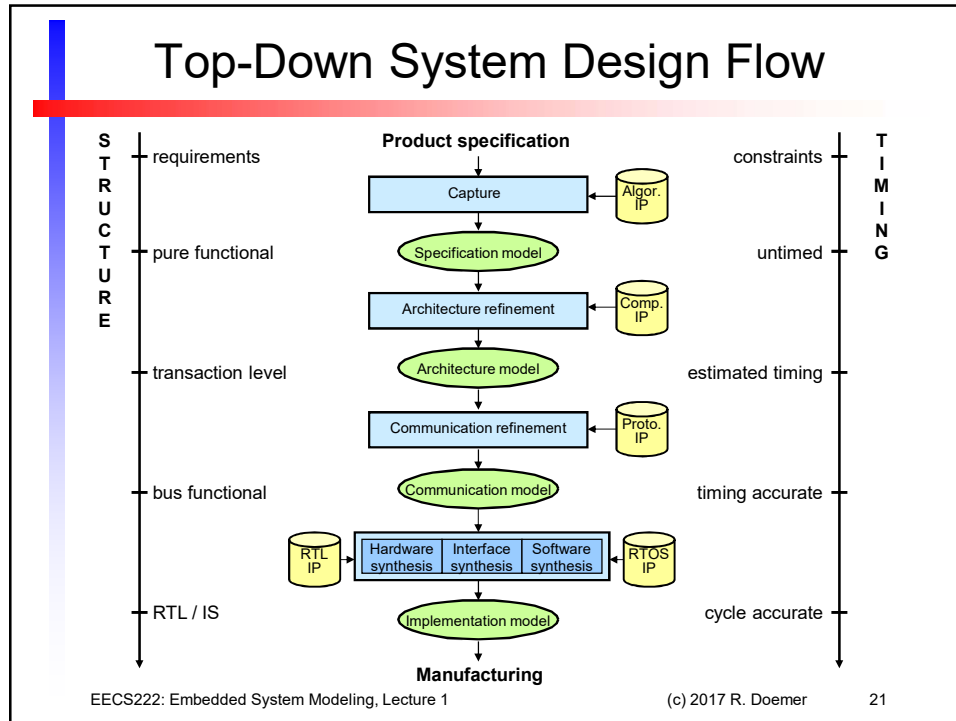
## Abstraction Levels

- System-on-Chip (SoC) design faces tremendous increase of design complexity
- Move to higher levels of abstraction!

Level	Number of components
System level	1E0
Algorithm	1E1
RTL	1E2
Gate	1E3
Transistor	1E4
	1E5
	1E6
	1E7


EECS222: Embedded System Modeling, Lecture 1
(c) 2017 R. Doemer
18



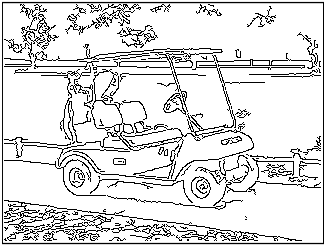


## EECS 222 Project

- Application Example: Canny Edge Detector
  - Embedded system model for image processing:  
Automatic Edge Detection in a Digital Camera



golfcart.pgm



golfcart.pgm\_s\_0.60\_l\_0.30\_h\_0.80.pgm

- Application Source and Documentation:
  - [http://marathon.csee.usf.edu/edge/edge\\_detection.html](http://marathon.csee.usf.edu/edge/edge_detection.html)
  - [http://en.wikipedia.org/wiki/Canny\\_edge\\_detector](http://en.wikipedia.org/wiki/Canny_edge_detector)

EECS222: Embedded System Modeling, Lecture 1 (c) 2017 R. Doemer 22

## Homework Assignment 1

- Administration
  - EECS Department Linux Servers
    - `crystalcove.eecs.uci.edu`, and others
    - Linux environment (CentOS 6.8)
    - Access via secure shell protocol (SSH)
  - Accounts
    - User ID same as your UCInetID
    - Password same as your EEE password
  - Login and make yourself familiar with
    - Command-line tools and GUI tools (which need X client)
    - Text editing and C/C++ programming
    - Image processing tools

EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

23

## Homework Assignment 1

- Task: Introduction to Application Example
  - Canny Edge Detector
  - Algorithm for edge detection in digital images
- Steps
  1. Setup your Linux programming environment
  2. Download, adjust, and compile the application C code with the GNU C compiler (`gcc`)
  3. Study the application
- Deliverables
  - None at this time (preparation for following assignments)
- Due
  - By next week: April 10, 2017, 12pm (noon!)

EECS222: Embedded System Modeling, Lecture 1

(c) 2017 R. Doemer

24