

# ECPS 203

## Embedded Systems Modeling and Design

### Lecture 1

Rainer Dömer

doemer@uci.edu

Center for Embedded and Cyber-physical Systems  
University of California, Irvine



## Lecture 1: Overview

- **Course Overview**
  - Context, description, content
- **Course Administration**
  - Schedule, assignments, communication
  - Academic honesty
- **Introduction to Embedded System Design**
  - Embedded computer systems
  - Design challenges
- **Project Assignment 1**
  - Setup and introduction to application example

## Course Context

---

- Professional Master Degree Program
  - Master of Embedded and Cyber-Physical Systems (MECPS)
- Courses
  - Fall quarter
    - **ECPS 203: Embedded systems modeling and design**
    - ECPS 205: Sensors, actuators and sensor networks
    - ECPS 206: Real-time and distributed systems
  - Winter quarter
    - ECPS 202: Cyber-physical systems design
    - ECPS 204: Embedded system software
    - ECPS 209: CPS case studies
  - Spring quarter
    - ECPS 207: Security, privacy and regulatory considerations
    - ECPS 208: Control systems for CPS
    - ECPS 210: Project

ECPS203: Embedded Systems Modeling and Design, Lecture 1
(c) 2018 R. Doemer
3

## Course Context

---

- Embedded Systems in the Big Picture

Cyber-Physical System

The diagram illustrates a Cyber-Physical System. It consists of three main components: Sensors on the left, an Embedded Computer System in the center, and Actuators on the right. The Embedded Computer System is further divided into Software (green) and Hardware (blue) layers. A large gray arrow loops from the Actuators back to the Sensors, indicating a control loop. The word 'Control' is written below the Embedded Computer System.

- Positioned in a networked world:  
From System-on-Chip (SoC) to Internet of Things (IoT)

ECPS203: Embedded Systems Modeling and Design, Lecture 1
(c) 2018 R. Doemer
4

## Course Description

- ECPS 203:  
Embedded Systems Modeling and Design (4)
  - Embedded systems definition, system-level specification, models and languages.
  - Concepts, requirements, examples.
  - Embedded system models at different levels of abstraction.
  - Test benches, design under test, IP components.
  - Discrete event simulation, semantics, and algorithms.

## Course Content

1. Embedded system concepts, models of computation
2. Application introduction, case study example
3. IEEE SystemC system-level description language
4. Application specification, modeling guidelines
5. Validation, discrete event simulation, estimation
6. Levels of abstraction, co-design methodology
7. Top-down system design methodology
8. Embedded system architecture, HW/SW partitioning
9. Cycle-accurate, register transfer level models
10. Project discussion, completion, report

## Course Administration

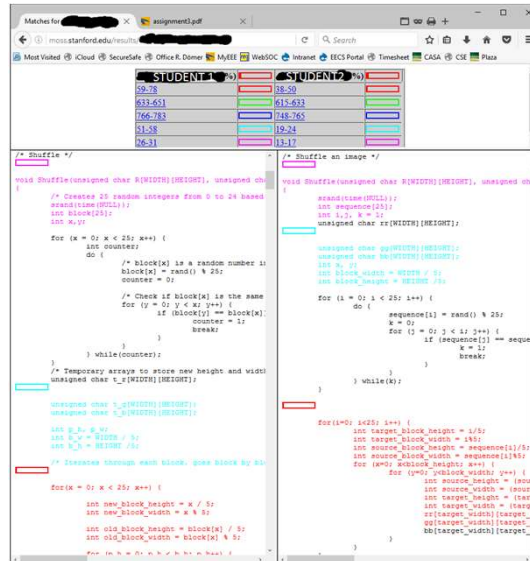
- Course web pages at <http://eee.uci.edu/18f/16905/>
  - Instructor information
  - Course syllabus and policies
  - Objectives and outcomes
  - Contents and schedule
  - Resources and communication
  - Assignments

## Academic Honesty

- Honesty and Integrity are Required
  - See UCI Office of Academic Integrity & Student Conduct
  - See course policy on course web site
- Plagiarism
  - Theft of intellectual property
  - Taking someone else's work or ideas and passing them off as one's own
  - *Do not copy code!*
- Violations will be reported
  - Academic misconduct report to UCI Office of AISC
    - Interview, written report, AISC staff meeting, decision, ...
  - Possible sanctions
    - Warning, probation, suspension, dismissal

# Academic Honesty

- Example (F'16):
  - **Moss:**  
Automatic system for determining similarity of program code



ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

# Academic Honesty

- Example (S'16): Automatic text transformation
  - Technical Report 1:

**1 INTRODUCTION**  
 With complexities of Systems-on-Chip rising almost daily, the design community has been searching for new methodology that can handle given complexities with increased productivity and less time. The modeling and design of embedded systems can be performed at several abstraction levels. The highest level of abstraction is the System level, where the functionality is described using "system-level specification" (in the case of VLSI design, description languages like VHDL or Verilog) and the architecture is seen as building blocks consisting of processors, memories, etc. ...

- Technical Report 2:

**INTRODUCTION:**  
 SOC challenges are changing day by day, the plan group has been hunting down new philosophy that can deal with given complexities with expanded efficiency and less time. The displaying and plan of implanted frameworks can be performed at a few reflection levels. The most abnormal amount of reflection is the System level, where the usefulness is portrayed utilizing "framework level determination" (on account of VLSI plan, depiction dialects like VHDL or Verilog) and the design is viewed as building pieces comprising of processors, recollections, and so on...

ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

## Embedded System Design

- Embedded Computer Systems
  - Examples
  - Design Challenges
  - Design Advantages
- Design Complexity
  - Hardware design gap
  - Software design gap
  - System design gap

Cyber-Physical System

The diagram illustrates a Cyber-Physical System. It consists of three main components: Sensors on the left, an Embedded Computer System in the center, and Actuators on the right. The Embedded Computer System is further divided into Software and Hardware layers. A circular arrow labeled 'Control' indicates a feedback loop from the Actuators back to the Sensors, passing through the Embedded Computer System.

ECPS203: Embedded Systems Modeling and Design, Lecture 1
(c) 2018 R. Doemer
11

## Embedded Computer Systems

- Computers are ubiquitous, omnipresent...

A collage of images illustrating the ubiquity of embedded systems. It includes a PDA, a camcorder, a mobile phone, a laptop, a satellite dish, a space shuttle, a car, and a satellite ground station.


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

Two images representing SoC design: a physical chip and a colorful circular visualization of a chip layout.


ECPS203: Embedded Systems Modeling and Design, Lecture 1
(c) 2018 R. Doemer
12

## 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: Miele



Source: Philips



Source: www.trouper.com



Source: www.medicacorp.com/

ECPS203: Embedded Systems Modeling and Design, Lecture 1 (c) 2018 R. Doemer 13



Source: P. Chou, UCI



Source: Edumaticator



Source: Miele



Source: Philips



Source: www.trouper.com




Source: www.medicacorp.com/

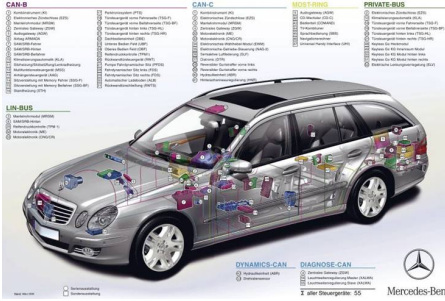
ECPS203: Embedded Systems Modeling and Design, Lecture 1 (c) 2018 R. Doemer 13

## 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

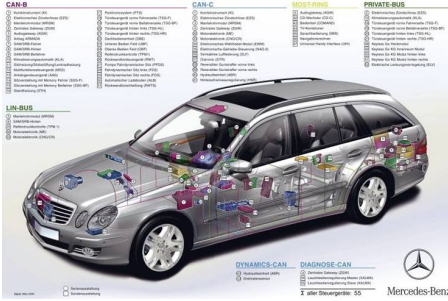


Source: Daimler

ECPS203: Embedded Systems Modeling and Design, Lecture 1 (c) 2018 R. Doemer 14



Source: Motorola Inc

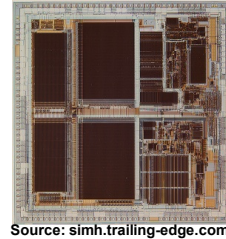


Source: Daimler

ECPS203: Embedded Systems Modeling and Design, Lecture 1 (c) 2018 R. Doemer 14

## 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

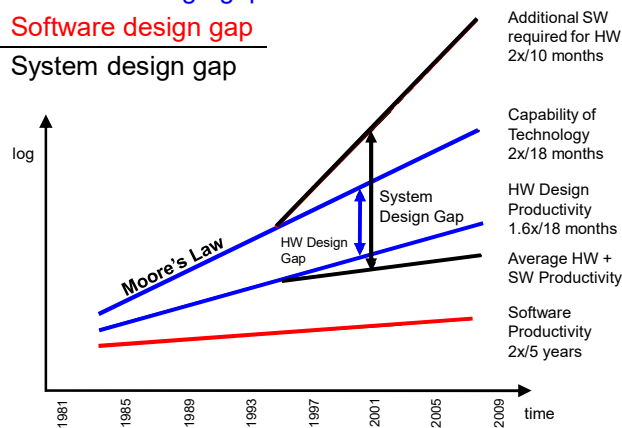
ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

15

## Design Complexity Challenge

- Productivity Gap
  - Hardware design gap
  - + Software design gap
  - = System design gap



(source: "Hardware-dependent Software", Ecker et al., 2009)

ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

16



## Design Complexity Challenge

- Productivity Gaps
  - Hardware productivity gap
    - Capacities in chip size outpace capabilities in chip design
    - Moore's law: chip capacity doubles every 18 months
    - HW design productivity estimated at 1.6x over 18 months
  - Software productivity gap
    - Growth of SW productivity estimated at 2x every 5 years
    - Needs in embedded SW estimated at 2x over 10 months
  - System productivity gap
    - HW gap + SW gap

ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

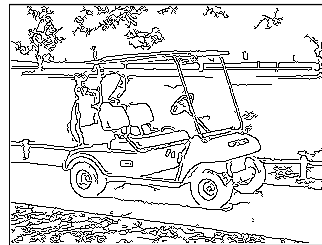
17

## ECPS 203 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)

ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

18

## Homework Assignment 1

- Setup
  - EECS Department Linux Servers
    - `crystalcove.eecs.uci.edu`, and other beaches
    - Linux environment (CentOS 6.10)
    - Remote access via secure shell protocol (SSH)
  - Accounts
    - Login ID is your UCInetID
    - Password is the 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

ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

19

## 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, determine function-call tree
- Deliverables
  - Source code and text file: `canny.c`, `canny.txt`
- Due
  - Wednesday, next week: October 10, 2018, 6pm

ECPS203: Embedded Systems Modeling and Design, Lecture 1

(c) 2018 R. Doemer

20