# EECS 221: System-on-Chip Software Synthesis 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 administration
  - Communication
- Course overview
  - Context
  - Contents
  - Objectives and Outcomes
  - Literature
- Introduction to Embedded Systems
  - Overview
  - Characteristics and Applications
  - Problems

## Course Administration

- Course web pages at **`http://eee.uci.edu/06f/18415/`**
  - Instructor information
  - Course description
  - Course syllabus
  - Course objectives and outcomes
  - Course resources
  - Assignments
- Course communication
  - Noteboard
  - Email



## Course Context

- Set of 4 courses on SoC Design

**A. System-on-Chip Description and Modeling**
**B. System-on-Chip Design and Exploration**
**C. System-on-Chip Software Synthesis**
**D. System-on-Chip Hardware Synthesis**

- **Starting as EECS 222 in Fall 2007**
- **Course A is prerequisite for B, C, and D, or consent of instructor**

## Course Context

- Set of 4 courses on SoC Design
  - **A. System-on-Chip Description and Modeling**
    Computational models for System-on-Chip (SoC). System-level specification and description languages and execution semantics. Concepts, requirements, examples. SoC modeling at different levels of abstraction (untimed, approximate time, cycle-accurate). Modeling of IP (IP wrappers), design constraints, test benches. Simulation semantics and algorithms. Co-simulation methodology.
  - **B. System-on-Chip Design and Exploration**
  - **C. System-on-Chip Software Synthesis**
  - **D. System-on-Chip Hardware Synthesis**

EECS221: SoC Software Synthesis, Lecture 1 (c) 2006 R. Doemer 5

## Course Context

- Set of 4 courses on SoC Design
  - **A. System-on-Chip Description and Modeling**
  - **B. System-on-Chip Design and Exploration**
    System-on-Chip design flow and methodology. Design space exploration. Co-design of hardware and software, hardware/software partitioning. System-on-Chip architecture exploration and synthesis. On-chip network and communication design and synthesis. On-chip software/hardware interface generation.
  - **C. System-on-Chip Software Synthesis**
  - **D. System-on-Chip Hardware Synthesis**

EECS221: SoC Software Synthesis, Lecture 1 (c) 2006 R. Doemer 6

## Course Context

- Set of 4 courses on SoC Design

**A. System-on-Chip Description and Modeling**
**B. System-on-Chip Design and Exploration**

**C. System-on-Chip Software Synthesis**
System-on-Chip software concepts, requirements, examples, for engineering applications such as automotive and communication. Software synthesis methodology. Algorithmic specification, design constraints. Applications using embedded operating systems. Static, dynamic, real-time scheduling. Input/output, interrupt handling. Code generation, retargetable compilation. Instruction set simulation. Debugging and prototyping.

**D. System-on-Chip Hardware Synthesis**



## Course Context

- Set of 4 courses on SoC Design

**A. System-on-Chip Description and Modeling**
**B. System-on-Chip Design and Exploration**
**C. System-on-Chip Software Synthesis**

**D. System-on-Chip Hardware Synthesis**
Hardware IP specification. Real-time constraints. Cycle-accurate languages and modeling. Target architectures, data path and control unit. Design tasks and design methodology. Behavioral synthesis. Resource allocation, operation scheduling, binding of operations and variables to functional units, storage units and busses. Communication protocol and interface synthesis. Arbiter, bridge, Transducer, Controller design and synthesis. Net list generation.

## Course Contents

- EECS 221: SoC Software Synthesis
  - System-on-Chip software
    - concepts, requirements, and examples,
    - for engineering applications such as automotive and communication.
  - Software synthesis methodology.
  - Algorithmic specification and design constraints.
  - Applications using embedded operating systems.
  - Static, dynamic, real-time scheduling.
  - Input/output, interrupt handling.
  - Code generation, retargetable compilation.
  - Instruction set simulation.
  - Debugging and prototyping.



## Course Goals

- Objectives
  - To learn embedded software concepts in System-on-Chip designs
  - To be able to design, develop and debug software in SoC designs
  - To understand software code generation for SoC
- Outcomes
  - Students understand
    - the special requirements of software for SoC.
    - the process of code generation and integration for SoC.
  - Students are able to
    - develop application SW, middleware, and/or drivers for SoC.
    - implement, test and debug a software application for a SoC.

## Course Outline

Week 1 – Embedded software concepts, requirements
Week 2 – Real-time requirements
Week 3 – Real-time scheduling
Week 4 – SoC software specification
Week 5 – Embedded software design flow
Week 6 – Software synthesis
Week 7 – RTOS targeting and mapping
Week 8 – Target processors
Week 9 – Code generation and compilation
Week 10 – Instruction-set simulation



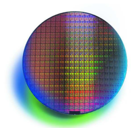
## Course Literature

- Primary Textbooks
  - P. Marwedel:
    *"Embedded System Design"*,
    Kluwer Academic Publishers, Boston, 2003.
  - A. Jerraya, S. Yoo, D. Verkest, N. Wehn (editors):
    *"Embedded Software for SoC"*,
    Kluwer Academic Publishers, Boston, 2003.
  - P. Marwedel, G. Goosens (editors):
    *"Code Generation for Embedded Processors"*,
    Kluwer Academic Publishers, 1995.
  - A. Gerstlauer, R. Doemer, J. Peng, D. Gajski:
    *"System Design: A Practical Guide with SpecC"*,
    Kluwer Academic Publishers, Boston, June 2001.
- Additional Reading
  - F. Vahid, T. Givargis:
    *"Embedded System Design: A Unified Hardware/Software Introduction"*,
    John Wiley and Sons, New York, 2002.
  - J. Staunstrup, W. Wolf (editors):
    *"Hardware/Software Co-Design: Principles and Practice"*,
    Kluwer Academic Publishers, Boston, 1997.
  - H. Kopetz:
    *"Real-time Systems"*,
    Kluwer Academic Publishers, Boston, 1997.
  - C. Krishna, K. Shin:
    *"Real-Time Systems"*,
    McGraw-Hill, 1997.

## Introduction to Embedded Systems

- Embedded systems are everywhere…

- Deep sub-micron design enables System-on-Chip (SoC)





## Introduction to Embedded Systems

- Chapter 1 of
  *"Embedded System Design"*
  by P. Marwedel (Univ. of Dortmund, Germany),
  Kluwer Academic Publishers, 2003.