System-on-a-Chip (SoC) Design

EE382V, Unique: 16835, Spring 2010

Lectures: TTh 5-6:30pm, ENS 116

Instructor: Andreas Gerstlauer <gerstl@ece.utexas.edu>**,** Office hours: W 2-4pm, ACES 6.118

Teaching Assistant: Hyungman Park <hpark@cerc.utexas.edu>, Office hours: TBD, ENS 113A

Guest Lecturers (tentative): Jacob Abraham, Steven Smith, Mark McDermott, Xtreme EDA

Class website: Blackboard and http://www.ece.utexas.edu/~gerstl/ee382v_s10/

Description

With technological advances that allow us to integrate complete multi-processor systems on a single die, Systems-on-Chip (SoCs) are at the core of most embedded computing and consumer devices, such as cell phones, media players and automotive, aerospace or medical electronics. This course will provide an understanding of the concepts, issues, and process of designing highly integrated SoCs following systematic hardware/software co-design & co-verification principles. Specifically, the class project involves taking public domain C++ code for a DRM (Digital Radio Mondiale) PC based system and mapping it to an ARM-based virtual and FPGA prototyping platform using state-of-the-art synthesis and verification tools and design flows.

Goals

This course is designed for students to learn and be able to:

- Model and specify embedded systems at high levels of abstraction.
- Analyze the functional and nonfunctional performance of the system early in the design process to support design decisions.
- Analyze hardware/software tradeoffs, algorithms, and architectures to optimize the system based on requirements and implementation constraints.
- Describe architectures for control-dominated and data-dominated systems and real-time systems.
- Understand hardware, software, and interface synthesis.
- Understand issues in interface design.
- Use co-simulation to validate system functionality.
- Describe examples of applications and systems developed using a co-design approach.
- Appreciate issues in system-on-a-chip design associated with co-design, such as intellectual property, reuse, and verification.

Topics

Likely to be covered in class:

- System-level and SoC design methodologies and tools;
- HW/SW co-design: analysis, partitioning, real-time scheduling, hardware acceleration;
- Virtual platform models, co-simulation and FPGAs for prototyping of HW/SW systems;
- Transaction-Level Modeling (TLM), Electronic System-Level (ESL) languages: SystemC;
- High-Level Synthesis (HLS): allocation, scheduling, binding, resource sharing, pipelining;
- SoC and IP integration, verification and test.

Prerequisites

- Working knowledge of C and C++, including software development and debugging (e.g. EE322C Data Structures, or equivalent);
- Embedded real-time system design and hardware/software interfacing (e.g. EE345M Embedded & Real-time Systems, or equivalent);
- Digital hardware design and hardware description languages (e.g. EE360M Digital System Design using VHDL, or equivalent);
- It is helpful to have some basic knowledge of communication systems.

Textbooks

No required textbook. Suggested reference books:

- D. Black, J. Donovan, *SystemC: From the Ground Up*, Springer, 2004.
- R. Zurawski (Editor), *Embedded Systems Handbook*, CRC Press.
- D. Gajski, S. Abdi, A. Gerstlauer, G. Schirner, *Embedded System Design: Modeling, Synthesis, Verification*, Springer, 2009.
- P. Marwedel, *Embedded System Design*, Springer, 2003.
- G. De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994.
- T. Noergaard, *Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers*, Newnes.
- B. Eckel, *Thinking in C++*, 2nd Edition, Prentice Hall.

Grading

Homework:	15%
Exam:	20%
Labs:	30%
Project:	35%

Oral discussion of homework problems is encouraged but solutions have to be submitted individually and independently. Labs and the final project will be done in teams.

Electronic Mail Notification Policy

In this course e-mail will be used as a means of communication with students. You will be responsible for checking your e-mail regularly for class work and announcements. The complete text of the University electronic mail notification policy and instructions for updating your e-mail address are available at http://www.utexas.edu/its/policies/emailnotify.html.

Use of Blackboard and Class Web Site

This course uses the class web page and Blackboard to distribute course materials, to communicate and collaborate online, to submit assignments and to post solutions and grades. You will be responsible for checking the class web page and the Blackboard course site regularly for class work and announcements. As with all computer systems, there are occasional scheduled downtimes as well as unanticipated disruptions. Notification of disruptions will be posted on the Blackboard login page. Scheduled downtimes are not an excuse for late work. However, if there is an unscheduled downtime for a significant period of time, I will make an adjustment if it occurs close to the due date.

Students with Disabilities

The University of Texas at Austin provides upon request appropriate academic accommodations for qualified students with disabilities. For more information, contact the Office of the Dean of Students at 471-6259, 471-4641 TTY or the College of Engineering Director of Students with Disabilities at 471-4382.

Religious Holidays

Religious holy days sometimes conflict with class and examination schedules. If you miss an examination, work assignment, or other project due to the observance of a religious holy day you will be given an opportunity to complete the work missed within a reasonable time after the absence. It is the policy of The University of Texas at Austin that you must notify each of your instructors at least fourteen days prior to the classes scheduled on dates you will be absent to observe a religious holy day.

Tentative Course Outline and Schedule

Week	Lecture Topic
1 (Jan 19/21)	Class Overview, C++ Review
2 (Jan 26/28)	SystemC Tutorial
3 (Feb 2/4)	Project Overview and DRM Tutorial
4 (Feb 9/11)	Requirements Analysis and Architecture Mapping
5 (Feb 16/18)	System-Level Design Methodology
6 (Feb 23/25)	HW/SW Partitioning and Real-Time Scheduling
7 (Mar 2/4)	Platform Simulation and Prototyping
8 (Mar 9/11)	SoC Models, Design Languages and Hardware Architectures
9 (Mar 16/18)	*Spring Break*
10 (Mar 23/25)	C-to-RTL High-Level Synthesis
11 (Mar 30/Apr 1)	Synthesis Models and Architectures
12 (Apr 6/8)	High-Level Synthesis Algorithms
13 (Apr 13/15)	Review, **Exam**
14 (Apr 20/22)	SoC Verification and Testing
15 (Apr 27/29)	System Integration, Wrapup
16 (May 4/6)	**Project Design Reviews**
Finals (TBD)	**Project Presentations**