ECE 695R:
SYSTEM-ON-CHIP DESIGN

Module 1: Introduction and Background
Lecture 1.2: Course Overview

Anand Raghunathan
raghunathan@purdue.edu

Fall 2014, ME 1052, T Th 12:00PM-1:15PM
Objective of this Course

• To give students an understanding and appreciation of the key design issues in SoC architecture, and hands-on experience with addressing some of them
  – Design above the Register-transfer Level
  – Designing systems that contain a mix of Digital Hardware and Software
Key SoC Design Issues

- MP211: Multi-core mobile phone SoC
  - 3 ARM CPUs, VLIW DSP, Programmable Security Processor, 2D/2D graphics accelerator, Image processor, DDR/LCD/SD/MS interfaces, peripherals

Key design issues:
Which / how many processors?
Partitioning functionality between CPU, DSP, accelerators?
Hardwired vs. programmable accelerators?
How to interconnect the components?
Re-use legacy / pre-designed components?
Topics / Schedule (Tentative)

• **Introduction** (1 lecture)

• **Background** (IC design flow, levels of abstraction), Quick tour through SoC design process (2 lectures)

• **Hardware/Software Partitioning** (8 lectures)
  - The tradeoff between efficiency and flexibility
  - Designing hardware accelerators, Interfacing accelerators to software, Automatic HW/SW Partitioning
  - Application-specific Instruction Processors, Basic approaches to ASIP design, Extensible processors, Automatic synthesis of custom instruction sets
  - Efficient software architectures for hardware acceleration

• **Behavioral Synthesis: Compiling software into hardware** (6 lectures)
  - Steps involved in behavioral synthesis, Scheduling, Resource sharing/binding, Advanced performance and power optimization techniques
Topics / Schedule (contd.)

- **On-chip Communication Architecture Design** (6 lectures)
  - Interconnect scaling challenges, on-chip buses – concepts and real examples, Optimizing bus-based communication architectures, Networks-on-chip

- **Modeling and Co-Simulating Hardware and Software** (2 lectures)
  - Modeling abstractions for hardware (FSM+Datapath, cycle-accurate functional, behavioral), Modeling abstractions for programmable processors (Instruction-set simulation, compiled simulation), HW/SW co-simulation

- **Advanced / current research topics** (2 lectures)
How does this fit-in with other Purdue ECE courses?

- **ECE 559** (MOS VLSI Design)
  - Design at the transistor-level
- **ECE 670K** (Modeling and Optimization of High Performance Interconnects)
  - Physical design (placement, routing, clock tree synthesis)
- **ECE 595Z** (Digital Logic Synthesis)
  - From register-transfer level to gates
  - Emphasis on automation (how the tools work)
- **ECE 565** (Computer Architecture)
  - Focuses on how to design a micro-processor
- **ECE 568** (Embedded Systems)
  - Software and system-level view (mostly outside the chip boundary)

This class addresses design at a different level of abstraction (what happens before RTL?)

In this class, the processor will be used as a component to design larger systems
Grading

- 45% - Homework/Lab Assignments
- 30% - Course Project
- 20% - Midterm
- 5% - Class participation

Late submissions: -20% per day
Homework/Lab Assignments

• **HW1:** 1-page introduction to yourself
• **HW2:** Introduction to the Altera Embedded Development Suite and DE2 FPGA platform
  – Build a simple system-on-chip using Altera SOPC Builder, generate RTL and synthesize to the FPGA, develop and run a simple test program on the DE2 board
• **HW3:** Designing a simple HW accelerator and integrating it into an SoC
  – Tools: Altera DE3 Board, SOPC Builder, NiosII Software IDE
Homework/Lab Assignments (contd.)

- **HW4**: Designing custom instructions to accelerate a program
  - Tools: Altera DE3 Board, SOPC Builder, NiosII Software IDE

- **HW5**: HW/SW co-simulation
  - Explore Transaction-level Modeling (TLM)
  - Tools: Synopsys Platform Architect

- **HW6**: C-to-Hardware synthesis
  - Tools: Mentor Graphics Catapult C
Course Project

- Design HW/SW architecture for an application of moderate complexity
  - Working prototype demonstrated at the end of the semester
  - Emphasis on architecture exploration / optimized designs, not just any working system
- Option 1: Standard (e.g., JPEG decoder)
- Option 2: Pick your own application (requires a proposal and discussion with me)
Platform for Labs/Project

- Altera
  - DE2 FPGA board
  - SOPC builder
  - NiosII Configurable Soft Processor
  - NiosII Software Integrated Development Environment
  - Optional: Digital Camera, LCD touch panel
Expected Background

• Hardware
  – Digital logic design, HW design using HDL and synthesis/simulation tools*, some exposure to FPGA prototyping*

• Software - C programming, write and debug programs of moderate complexity

• Comfortable with using and programming on Linux/UNIX computing platforms

• Above all, willingness to
  – Challenge yourself to go beyond your comfort zone
  – Put in the extra time to fill any gaps in background
Expectations

• This is a hands-on class, be prepared to implement and debug HW/SW systems of moderate complexity

• You will be using industrial tools and hardware platforms, without much hand-holding
  – Extensive documentation available, but ability to zoom in on what is relevant is critical
  – Strongly encouraged to collaborate and exchange information on tools / FPGA platform / debugging
    • However, actual labs and project should be your own effort

• Great opportunity to go beyond the minimum (push the limits of tools and hardware platform)
Reference Material

- Course Notes and Handouts
  - Primary source of information
- Online resources (manuals / documentation) will be provided for the labs and project
- Unfortunately, there is no good textbook that covers all the topics
Reference Books

• Reference Books – in no particular order (click on titles for hyperlinks)
  – Introduction to SoCs and Embedded Systems
    • Computers as Components: Principles of Embedded Computing System Design, Wolf
    • Embedded System Design: A Unified Hardware/Software Introduction, Vahid, Givargis (early edition available online)
    • Multiprocessor Systems-on-chips, Jerraya/Wolf
    • Surviving the SOC Revolution - A Guide to Platform-Based Design, Chang/Todd/Nelly/Martin/Cooke
    • Winning the SoC Revolution: Experiences in Real Design, Martin/Chang (Editors)
  – HW/SW Co-Design
    • Hardware/Software Co-Design - Principles and Practice, Straunstrup / Wolf
    • Readings in Hardware/Software Co-Design, De Micheli / Ernst / Wolf
  – Custom Processors and ASIPs
    • Processor Design: System-On-Chip Computing for ASICs and FPGAs, Nurmi (Editor)
    • Engineering the Complex SOC: Fast, Flexible Design with Configurable Processors, Rowen
    • Customizable Embedded Processors: Design Technologies and Applications, Ienne / Leupers
  – On-chip Communication Architecture
    • On-Chip Communication Architectures: System on Chip Interconnect, Pasricha/Dutt
    • Networks on Chips: Technology and Tools, De Micheli/Benini
  – Behavioral Synthesis
    • High-level Synthesis: Introduction to Chip and System Design, Gajski/Dutt/Wu/Lin
    • Synthesis and Optimization of Digital Circuits, De Micheli
Did You Know?

- The integrated circuit was first conceived by Geoffrey W. A. Dummer (British Ministry of Defense) and published on May 7th, 1952. He unsuccessfully attempted to build such a circuit in 1956.
- The IC was first realized by Jack Kilby (Texas Instruments) and Robert Noyce (Fairchild Semiconductor) around the same time.
- Robert Noyce came up with his idea for an integrated circuit ~6 months after Kilby. His invention solved the problem of interconnecting the components by adding a metal layer.
- Knowing that TI had already filed a patent on something similar, Fairchild wrote out a highly detailed application, hoping that it wouldn't infringe on TI's similar device.
- All that detail paid off. On April 25, 1961, the patent office awarded the first patent for an integrated circuit to Robert Noyce while Kilby's application was still being analyzed. Today, both men are acknowledged as having independently conceived of the idea.
- Noyce went on to be a co-founder of Intel.
- Kilby was jointly awarded the Nobel Prize in Physics in 2000.

Source: nobelprize.org, pbs.org