EE382M.20: System-on-Chip (SoC) Design

Lecture 0 – Class Overview

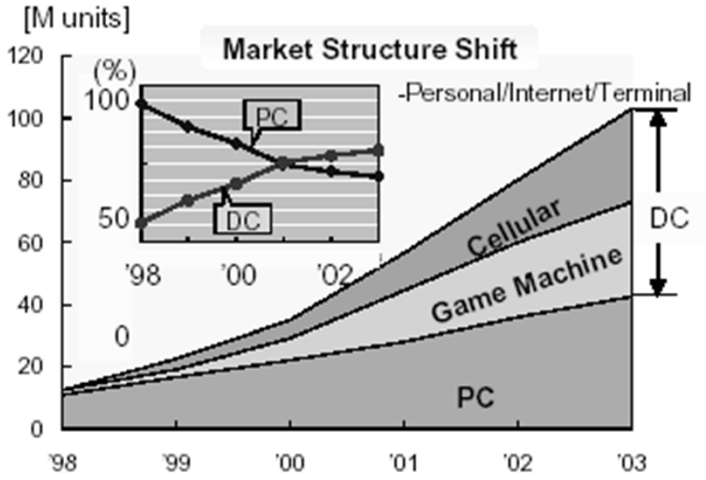
Andreas Gerstlauer
Electrical and Computer Engineering
University of Texas at Austin
gerstl@ece.utexas.edu

Lecture 0: Outline

- **Introduction**
  - Systems-on-Chip (SoCs)
  - Design flow
- **Course information**
  - Overview, goals, topics
  - Administration
  - Labs and project
- **Class project**
  - Deep learning based visual object recognition
  - SoC implementation

Industrial Structure Shift

- **Ubiquitous, embedded computing**
  - From personal to dedicated computers

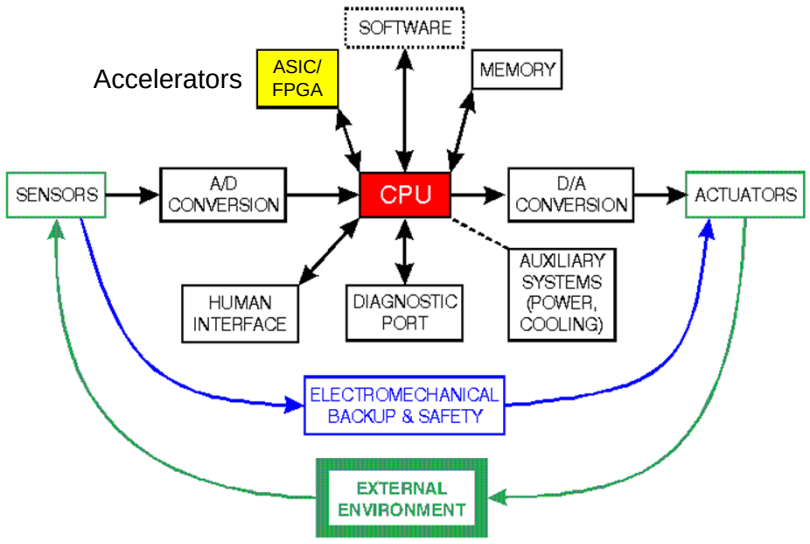


Source: SONY Corp & Market Estimates

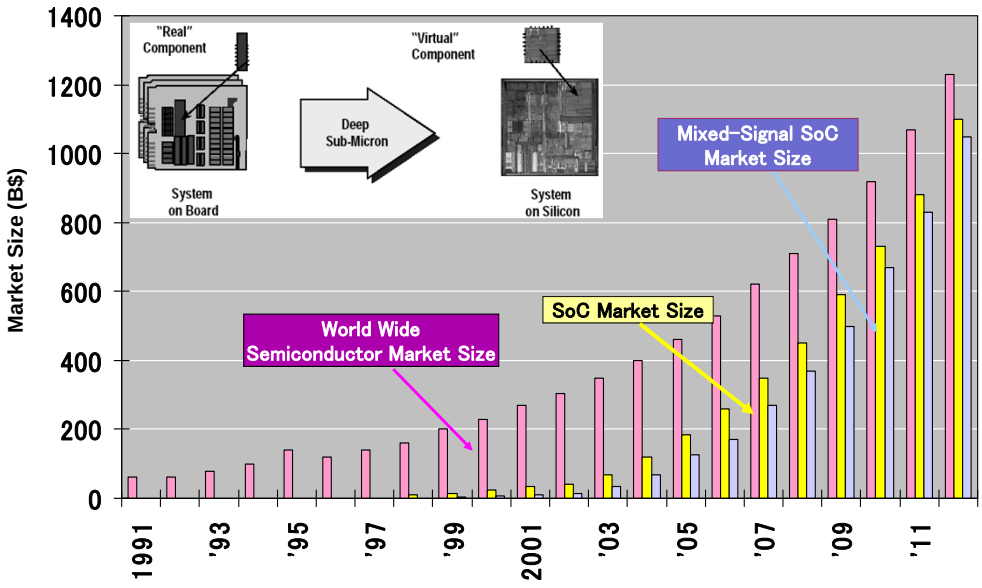
Embedded Systems

- **Embedded computing**
  - Cyber-physical systems (CPS)
    - Signal processing & control, robotics, autonomous intelligence
  - Internet-of-Things (IoT)
    - Sensing & actuating
  - "Small" systems
    - Cellphones, appliances, toys, MP3, PDAs, digital cameras, smart badges
- **Part of a larger system - masquerading as non-computers**
  - Not a "computer with keyboard, display, etc."
  - Continuously interact with external world (never terminate)
- **Application-specific – not general-purpose**
  - Application is known a priori, but some re-programmability
- **Tightly constrained**
  - TReal-time, power, size, weight, heat, reliability, etc.

(Embedded) Computer System Architecture



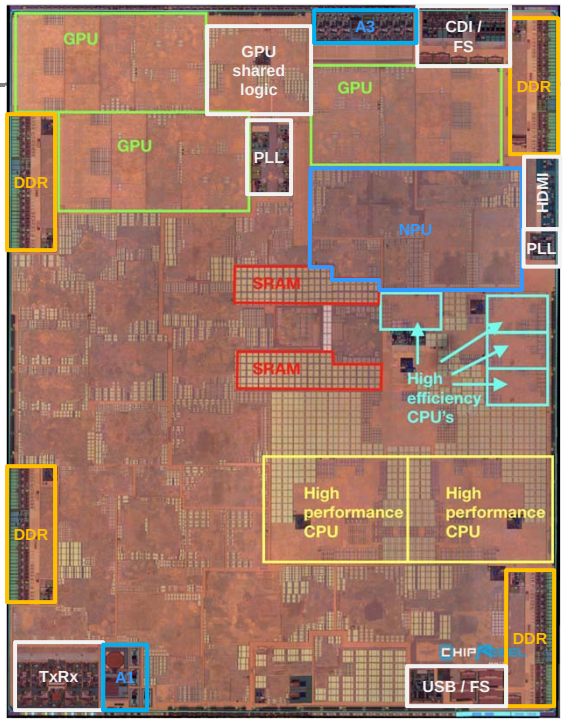
System-on-Chip (SoC) Era



Source: SONY Corp & Market Estimates

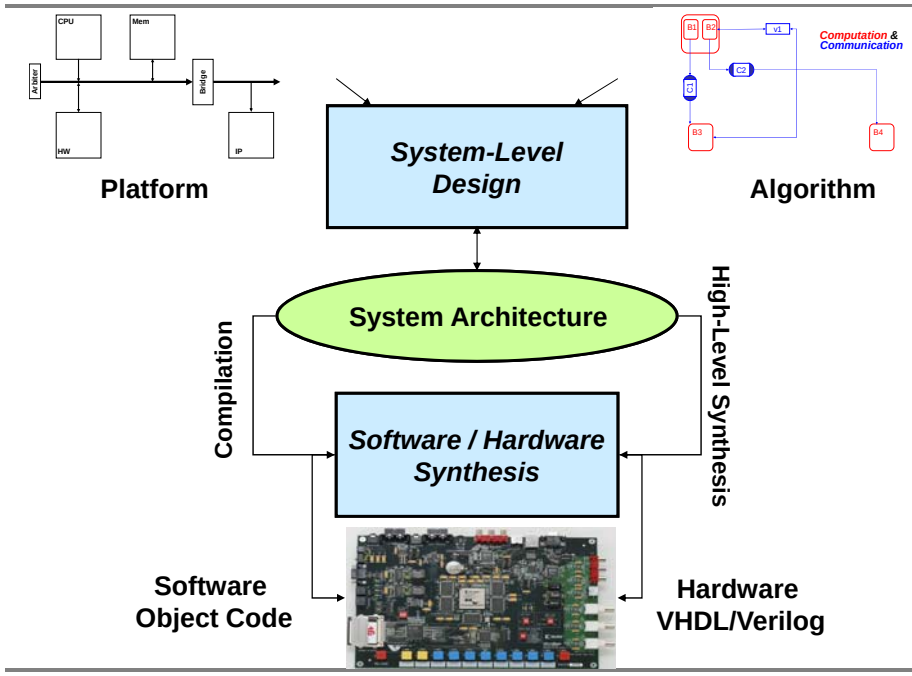
Apple A11 SoC

- 6-core ARMv8
  - 2 big
  - 4 little
- 3-core GPU
  - Proprietary
- Co-processor
  - M-class ARM
- Neural processing unit (NPU)
  - "Neural engine"
- Accelerators
  - Lots…

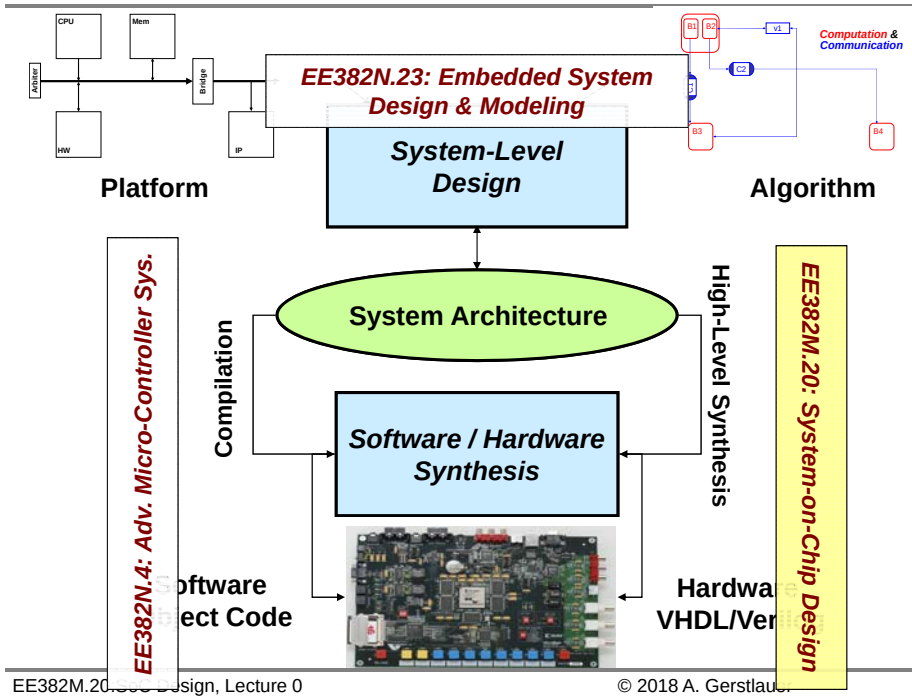


Source: ChipRebel & TechInsights

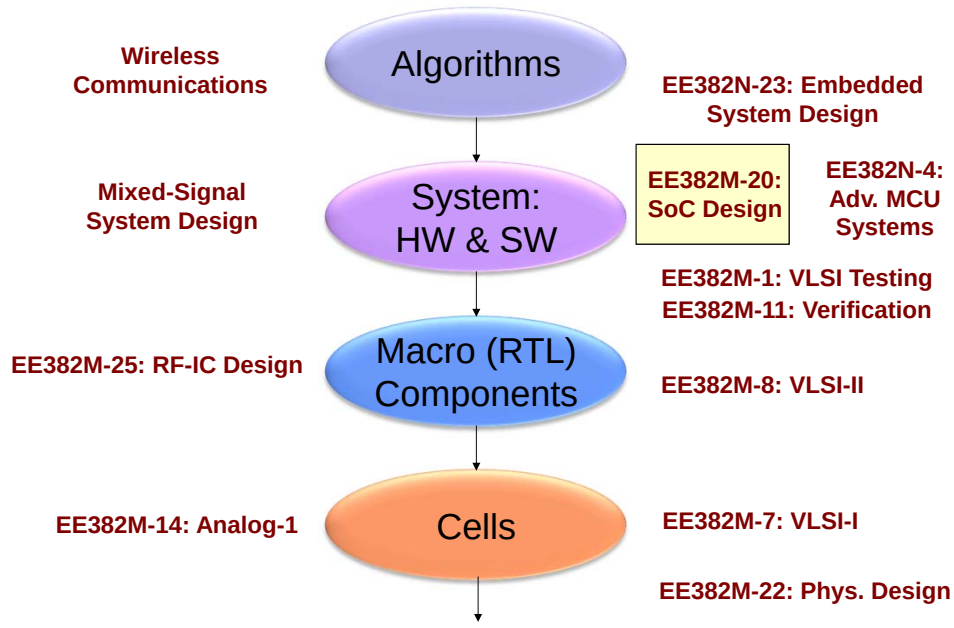
System Design Flow



UT ECE Courses

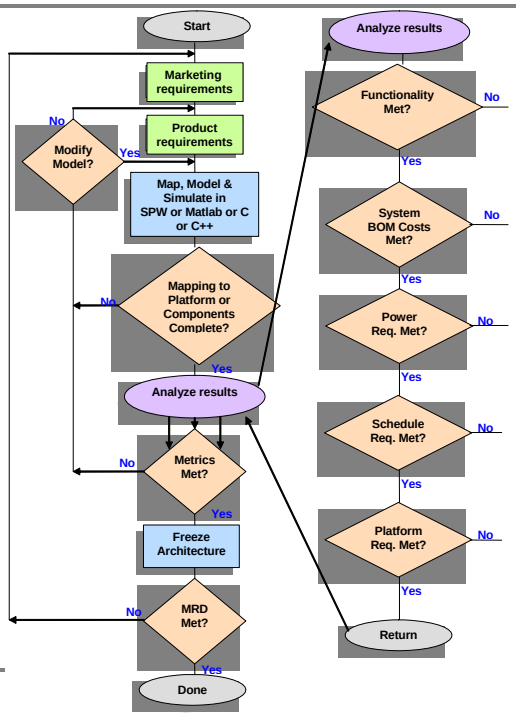


Integrated Circuits and Systems at UT



SoC Design Flow

- **In this class:**
  - Industry practice
  - Manual SoC arch. exploration
  - HW/SW comp. synthesis
- **In EE382N-23:**
  - Academic Research
  - System-level design automation
  - Formal methods & tools



Course Overview

- **Provide an understanding of the concepts, issues, and process of designing highly integrated SoCs**
  - Systematic hardware/software co-design & co-verification
  - Industry-standard/advanced SoC design flow
- **Class labs and project: visual object recognition SoC**
  - Deep/Convolutional Neural Network (DNN/CNN) inference
    - Hardware/software co-design
  - State-of-the-art synthesis and verification tools
    - Virtual platform prototyping and simulation in SystemC/TLM
    - High-level hardware synthesis from C++ to RTL
  - ARM-based FPGA prototyping platform
    - ARM processor, I/O devices, memory components, hardware accelerators

Course Goals

- **Course is designed to learn about:**
  - Early functional and nonfunctional performance analysis to support design decisions.
  - Analysis and optimization of hardware/software tradeoffs, algorithms, and architectures based on requirements and implementation constraints.
  - Analyze tradeoffs and explore architecture and mico-architecture design spaces to develop and synthesize custom hardware accelerators.
  - Hardware, software, and interface synthesis.
  - Interface design.
  - Co-simulation to validate system functionality.
  - Examples of applications and systems developed using a co-design approach.
  - Intellectual property, reuse, and verification issues.

Course Description and Topics

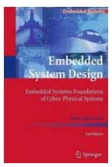
- **Covered in class:**
  - System-level and SoC design methodologies and tools;
  - HW/SW co-design: analysis, partitioning, real-time scheduling, hardware acceleration;
  - Virtual prototyping: virtual platform models, electronic system-level languages, and HW/SW co-simulation;
  - High-Level Synthesis (HLS): allocation, scheduling, binding algorithms for C-to-RTL synthesis
  - SoC integration: SoC communication architectures, IP interfacing, verification and test.
  - FPGA prototyping of HW/SW systems;

Course Administration

- **Instructors**
  - Instructor: Andreas Gerstlauer <gerstl@ece.utexas.edu>
    - Office hours: EER 5.882, TTh 2-3:30pm, or after class/by appointment
  - Guest lecturers from industry and academia
  - TA: Mochamad Asri <asri@utexas.edu>
    - Office hours: TBD
- **Information**
  - Web: http://www.ece.utexas.edu/~gerstl/ee382m_f18
    - Lecture notes, homework and lab exercises.
  - Canvas
    - Announcements, assignments, grades.
  - Piazza: https://piazza.com/utexas/fall2018/ee382m20
    - All non-personal class discussions.
- **Dates (tentative)**
  - Exam: November 15, in class
  - Project design reviews: December 4 & 6, in class
  - Final project presentations: December 19, 2-5pm

Textbooks

- **No required textbook**
- **Suggested references**
  - P. Marwedel, *Embedded System Design: Embedded Systems Foundations of Cyber-Physical Systems*, 3rd edition, Springer, 2018
    - http://ls12-www.cs.tu-dortmund.de/~marwedel/es-book/
  - D. Black, J. Donovan, B. Bunton, A. Keist, *SystemC: From the Ground Up*, 2nd edition, Springer, 2010
    - Electronic version available from the library
  - G. De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, 1994
    - Out of print, but can be ordered
    - On reserve in library



Course Policies

- **Grading:**
  - Homework: 15%
  - Lab assignments: 30%
  - Exam: 20%
  - Project: 35%
- **Late penalties:**
  - 20% per day (24 hours)
- **Academic dishonesty**
  - Homeworks
    - Ok to discuss questions and problems with others
    - But turn in own, independent solution
  - Labs and project
    - In teams, one report and presentation
    - Collaboration encouraged and desired
  - Plagiarism
    - Use of any outside source of information without quoting or referencing is cheating

Labs (tentative)

- **Lab 1: Software/algorithm (3 weeks, due Sep 23)**
  - Profiling of code on ARM board
    - Identify time consuming bottlenecks to optimize
  - Software optimization
    - Floating-point to fixed-point conversion
    - Improve performance, performance vs. detection accuracy tradeoffs
- **Lab 2: System architecture (3 weeks, due Oct 14)**
  - Hardware acceleration and system architecture
    - Identify and partition code into hardware and software
    - Hardware/software interface, communication architecture and drivers
  - Virtual platform modeling and simulation
    - Isolate and interface hardware as SystemC module
    - Co-simulation w/ software and firmware on ARM-Linux CPU
- **Lab 3: Accelerator design (3 weeks, due Nov 4)**
  - High-level synthesis of accelerator from C++ to RTL
    - Xilinx Vivado HLS tool
  - Verification of synthesis results
    - Testbench around standalone C++ vs. RTL hardware module

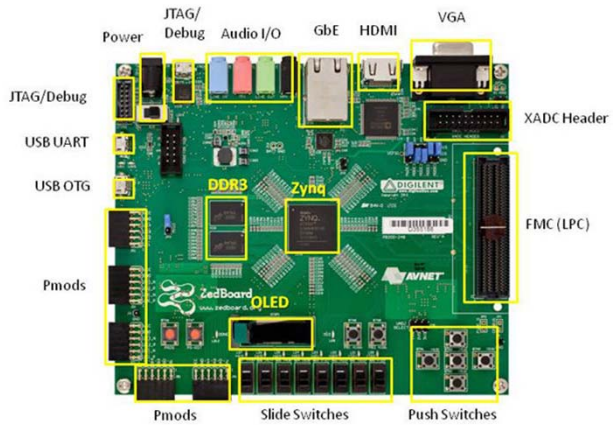
Class Project

- **Implementation on prototyping board**
  - Synthesizing hardware accelerator(s) into the FPGA
    - Synthesis and download using Xilinx software
  - C/C++ program on the ARM host processor
    - Cross-compilation or development under Ubuntu Linux on the ARM
  - Hardware/software interfacing and communication
    - Software drivers and hardware bus interfaces
  - Analysis and validation of product metrics
    - Estimate timing, area and power consumption

➢ **Low-power SoC implementation of real-time obj. detection**
  - Reference design of visual objection recognition ASIC
    - FPGA-based implementation as prototype of ASIC design
  - To be incorporated into cars, drones, …
    - Satisfy market and product requirements

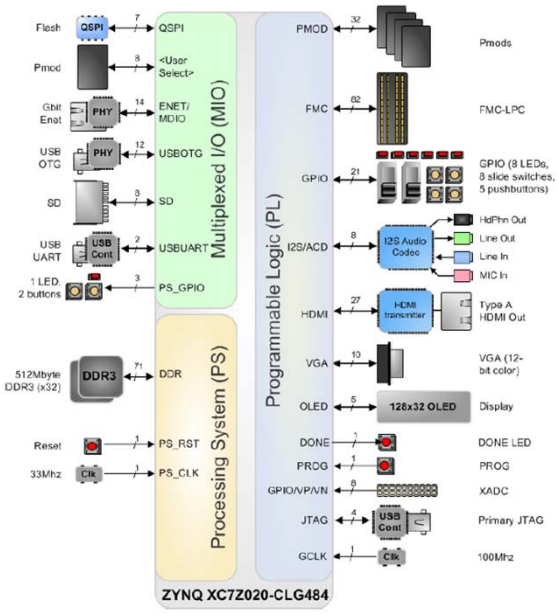
Prototyping Board

- **Zedboard by Digilent**
  - 10 boards, one per team



➢ **Based on Xilinx Zynq-7000 All-Programmable SoC**
  - Dual-core ARM Cortex A9 in Virtex 7 FPGA fabric

Zedboard Block Diagram



Zynq-7000 All-Programmable SoC