Embedded Systems

Introduction to Embedded Systems

Lesson 01

Francesco Menichelli

francesco.menichelli@uniroma1.it

Professor

Francesco Menichelli

Email	francesco.menichelli@uniroma1.it
Office	DIET, 4° floor, room 412

Website for course material download:
sites.google.com/site/embeddedsystemssapienza

Available after google group subscription:
https://groups.google.com/forum/#!forum/embsyssapienza

Course UNprogram

- System design process
- Design methodology and flow
- Functional and non functional requirements
  - Performance, cost, size, power consumption
- Requirements analysis and validation
- Specification derivation
- Formalism for system design
- Quality assurance techniques
  - ISO9000
- Verification
- Design review

Course program

- Embedded system definition and application domains
- Embedded systems components (hardware)
  - Microprocessors (architecture, Instruction set, interface)
  - Peripherals (communication, acquisition, power)
- Hardware/software partitioning
- Embedded systems software
  - Software abstraction layers (driver, HAL, application)
  - Interrupts and concurrency
  - Compilation, C/assembly language interaction
  - Executable code analisys
  - Embedded software debug

Course textbook:

Wayne Wolf - *Computers as Components: Principles of Embedded Computing System Design*, Morgan Kaufman Publishers

Goals of the Course

High-Level Goals

- Understand the principles and concepts behind embedded systems
- Obtain hands-on experience in programming embedded systems.

By the end of the course, you will be able to

- Understand the “Big Ideas" in embedded systems
- Obtain direct experience on both hardware and software elements commonly used in embedded system design.
- Understand basic resource management theory
- Understand the basics of embedded system application concepts such as signal processing and feedback control
  - Understand, and be able to discuss and communicate intelligently about embedded processor architecture and programming
  - I/O and device driver interfaces to embedded processors with peripherals
  - OS primitives for concurrency, scheduling, communication and synchronization

Course pre-requisite

- Basic of microelectronics
  - CPU, programmable architectures, memory systems, buses
  - Instruction set and its definition
- Basic of programming language
  - C language,
  - C++ can be useful, anyway it is not taken for granted
  - ~~Java~~
- Negative pre-requisite
  - Arduino programming

What is an embedded system?

What is an Embedded System?

- Based on their structure:
  - “Electronic systems which use a computer to perform a specific function, but are neither used nor perceived as a computer, are generally known as embedded systems" (Edwards et al.)
- Based on their function:
  - "It performs a dedicated function [...], real-time behavior must conform to very strict requirements, correctness of the design is essential due to the potential impact on the surrounding environment or the person using the equipment" (Paulin et al.)

- Embedded System or “*Sistema dedicato*” or “*Sistema immerso*”

What is an Embedded System?

- An electronic system being part of a larger (physical) system
  - neither the electronic nor the physical system are capable of providing the same functionality, reliability or other aspects of the composed system
  - Computation is required to interact with the environment
- Any system where the user doesn't want to know that it includes a processor

Embedded Systems parts

- Computational
  - but not first-and-foremost a computer
- Integral with physical processes
  - sensors, actuators
- Reactive
  - at the speed of the environment
- Heterogeneous
  - hardware/software, mixed architectures
- Networked
  - shared, adaptive

Embedded Systems

- An embedded system
  - uses a (network of) computer(s) to perform some functions, but
  - is not used (nor perceived) as a computer
- Typical characteristics:
  - it performs a single function (non user programmable)
  - it is part of a larger (controlled) system

  cost and reliability are often the most significant aspects

Four General Categories of Embedded Systems

- General Computing
  - Applications similar to desktop computing, but in an embedded package
  - Video games, set top boxes, wearable computers, automatic tellers
- Control Systems
  - Closed loop feedback control of real time system
  - Vehicle engines, chemical processes, nuclear power, flight control
- Signal Processing
  - Computations involving large data stream
  - Radar, Sonar, video compression
- Communication & Networking
  - Switching and information transmission
  - Telephone system, Internet

Distinctive Embedded System Attributes

- Reactive: computations occur in response to external events
  - Periodic events (e.g., rotating machinery and control loops)
  - Aperiodic events (e.g., button closures)
- Real-Time: timing correctness is part of system correctness
  - Hard real-time
    - Absolute deadline, beyond which answer is useless
    - May include minimum time as well as maximum time
  - Soft real-time
    - Missing a deadline is not catastrophic
    - Utility of answer degrades with time difference from deadline
  - Example:
    - a train is entering an urban area...
    - the railway gate in the city allows automotive traffic to go over the tracks
    - when should the railway gate close?

In general, Real Time does not equal “Real Fast”

Embedded System Requirements

- Functional Requirements:
  - Functionality that the real-time system must perform
- Temporal Requirements:
  - Timing constraints to be satisfied: hard and soft
- Dependability Requirements
  - Quality of the implemented behavior (service)

Typical Embedded System Constraints

- Small Size, Low Weight
  - Handheld electronics
  - Transportation applications weight costs money
- Low Power
  - Battery power for 8+ hours (laptops often last only 2 hours)
  - Limited cooling may limit power even if AC power available
- Harsh environment
  - Heat, vibration, shock
  - Power fluctuations, RF interference, lightning
  - Water, corrosion, physical abuse
- Safety critical operation
  - Must function correctly
  - Must not function incorrectly
- Extreme cost sensitivity
  - $.05 adds up over 1,000,000 units