CMP3201 Embedded Systems

Period per Week			Contact Hour per Semester	Weighted Total Mark	Weighted Exam Mark	Weighted Continuous Assessment Mark	Credit Units
LH	PH	TH	CH	WTM	WEM	WCM	CU
45	30	00	60	100	60	40	4

Rationale

Almost every electronic appliance and device today uses embedded systems. Cell phones, automobiles, toasters, televisions, airplanes, medical equipment, and a host of other devices, products, and applications use embedded systems. Such systems include microcontrollers, embedded programs, and real-time operating systems. These systems require a conscious effort to produce the most reliable product possible requiring the utmost diligence in system design and in design methodologies. Indeed, these designs often reflect the design of low power systems and tool support.

Objectives

By covering the course in Embedded Systems, the student will be able to:

- Identify some contributors to embedded systems and relate their achievements to the knowledge area, describe the meaning of an embedded system, explain the reasons for the importance of embedded systems, describe the relationship between programming languages and embedded systems and describe how computer engineering uses or benefits from embedded systems.
- Understand the CPU in the context of a complete system with I/O and memory, understand how the CPU talks to the outside world through devices, and understand how memory system design (caches, memory management) affect program design and performance.
- Understand how high-level language programs convert into executable code, know the capabilities and limits of compilers, and comprehend basic representations of programs used to manipulate programs either in a compiler or by hand.
- Distinguish RTOSs from workstation/server OS, distinguish real-time scheduling from traditional OS scheduling, understand major real-time scheduling policies and understand interprocess communication mechanisms.
- Understand why low-power computing is important, identify sources of energy consumption and identify possible remedies for energy consumption at various levels of design abstraction.
- Understand the variety of sources of faults in embedded computing systems, identify strategies to find problems and identify strategies to minimize the effects of problems.

- Understand why real-world projects are not the same as class projects, identify important goals of the methodology and understand the importance of design tracking and documentation.
- Understand role of hardware and software tools in system development and understand how to use tools to support the methodology.
- Understand the use of multiple processors in embedded systems, identify trade-offs between CPUs and hardwired logic in multiprocessors, and understand basic design techniques.

Subject Content

1. History and Overview of Embedded

- Indicate some reasons for studying embedded systems
- Highlight some people that influenced or contributed to the area of embedded systems
- Indicate some important topic areas such as mapping between language and hardware, classifications, influence of software engineering, applications and techniques, and tool support
- Contrast between an embedded system and other computer systems
- Mention the role of programming and its associated languages as applied to embedded systems
- Explore some additional resources associated with embedded systems
- Explain the purpose and role of embedded systems in computer engineering

2. Embedded Microcontrollers

- Structure of a basic computer system: CPU, memory, I/O devices on a bus
- CPU families used in microcontrollers: 4-bit, 8-bit, 16-32-bit
- Basic I/O devices: timers/counters, GPIO, A/D, D/A
- Polled I/O vs. interrupt-driven I/O
- Interrupt structures: vectored and prioritized interrupts
- DMA transfers
- Memory management units
- Memory hierarchies and caches

3. Embedded Programs

- The program translation process: compilation, assembly, linking
- Representations of programs: data flow and control flow
- Fundamental concepts of assembly language and linking: labels, address management
- Compilation tasks: mapping variables to memory, managing data structures, translating control structures, and translating expressions
- What can and cannot be controlled through the compiler; when writing assembly language makes sense

4. Real-Time Operating Systems

- Scheduling policies
- Rate-monotonic scheduling: theory and practice
- Priority inversion
- Other scheduling policies such as EDF
- Message-passing vs. shared memory communication
- Interprocess communication styles such as mailbox and RPC

5. ***Low-Power Computing***
   - Sources of energy consumption: toggling, leakage
   - Instruction-level strategies for power management: function unit management
   - Memory system power consumption: caches, off-chip memory
   - Power consumption with multiple processes
   - System-level power management: deterministic, probabilistic methods
6. ***Reliable System Design***
   - Transient vs. permanent failures in hardware
   - Sources of errors from software
   - The role of design verification in reliable system design
   - Fault-tolerance techniques
   - Famous failures of embedded computers
7. ***Design Methodologies***
   - Multi-person design projects
   - Designing on-time and on-budget
   - Design reviews
   - Tracking error rates and sources
   - Change management
8. ***Tool Support***
   - Compilers and programming environments
   - Logic analyzers
   - RTOS tools
   - Power analysis
   - Software management tools
   - Project management tools
9. ***Embedded Multiprocessors***
   - Importance of multiprocessors as in performance, power, and cost
   - Hardware/software partitioning for single-bus systems
   - More general architectures
   - Platform FPGAs as multiprocessors
10. ***Networked Embedded Systems***
    - Why networked embedded systems
    - Example networked embedded systems: automobiles, factory automation systems
    - The OSI reference model
    - Types of network fabrics
    - Network performance analysis
    - Basic principles of the Internet protocol
    - Internet-enabled embedded systems
11. ***Interfacing and Mixed-Signal Systems***
    - Digital-to-analog conversion
    - Analog-to-digital conversion
    - How to partition analog/digital processing in interfaces
    - Digital processing and real-time considerations

Recommended and Reference Books

[1] Frank Vahid and Tony Givargis, 2002. *Embedded System Design: A Unified Hardware/Software Introduction.* John Wiley & Sons; ISBN: 0471386782.

[2] Peter Marwedel, 2006. *Embedded System Design*. Birkhäuser, ISBN 0387292373, 9780387292373

[3] Arnold S. Berger, and Arnold S. Berger, 2001. *Embedded Systems Design: An Introduction to Processes, Tools and Techniques*. CMP Books; 1st edition. ISBN-10: 1578200733, ISBN-13: 978-1578200733

[4] John Catsoulis, 2005. *Designing Embedded Hardware*. 2nd Edition. O'Reilly Media, Inc. ISBN-10: 0596007558. ISBN-13: 978-0596007553.

[5] Tammy Noergaard, 2005. *Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers.* Newnes. ISBN-10: 0750677929. ISBN-13: 978-0750677929.