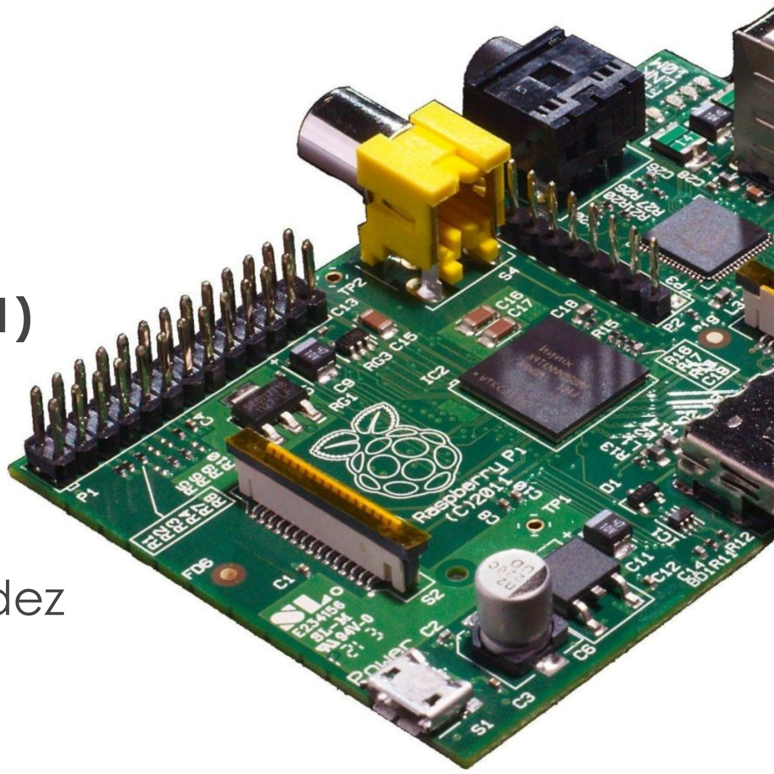
Embedded Systems (SEMP 1)



José Manuel Moya Fernández

Alvaro Araujo Pinto

Departamento de Ingeniería Electrónica

Universidad Politécnica de Madrid

Máster Oficial en Ingeniería de Sistemas Electrónicos

www.die.upm.es

Index

Embedded Systems

Semester: 1

Number of credits: 4 ECTS (3 ECTS theory and 1 ECTS practical)

Type: mandatory I1

Objectives

This course covers two aspects simultaneously: computing and restrictions. It is clear that computer systems have a major impact on our lives, and it is clear that any engineer or scientist should have a basic knowledge of its inner workings. But why should we worry about the restrictions?

Embedded systems, like any computer system, they have to perform a function. But we also have to meet very strict restrictions often:

• Time constraints: The ABS of a car has to activate the brakes in a very short time to avoid accidents.

• A reduction in memory requirements and size means lighter devices, more portable and cheaper.

Why study Embedded Systems?

Embedded systems are by far away the most numerous computer systems. Embedded systems is often found in many common products, including automobiles, toys and household appliances. They are also used to monitor health parameters sick and elderly, to improve the safety of our homes to save energy, to prevent accidents, to increase comfort by automatically adjusting the heat or light, or by the purchase automatically when need. Embedded systems are changing our lives and the ability to design, implement and analyze embedded systems is often demanding from the industry and the University.

• Mobile phones, portable media devices and wireless sensor networks often have very strong restrictions on power consumption.

• Finally, with so few resources, security becomes a very difficult challenge.

In addition, an embedded system has to work in the worst case scenario, should be designed to meet the restrictions even in the worst case.

In this course students will learn to program microprocessor-based embedded systems and hardware design extensions to run in the worst case, considering all the constraints for the design and implementation. We begin with the most basic concepts to soon move to more advanced techniques.

This course provides the theoretical content required for the course "Electronic Systems Laboratory," which is taught in the second semester. The development environment and tools presented in this course will also be used in the laboratory. And this laboratory practices are designed to complement the approach taken in this subject.

We believe in learning by doing. There is no better way to learn how to build an embedded system to building it. Therefore, the course is organized around several projects using the Raspberry-Pi, a computer system the size of a credit card and very cheap that plugs into your TV and a keyboard.

At the end of the course, the student:

1. Efficiently use the tools most widely used development (development tools from the GNU project): GCC compiler, GNU make, binutils, profilers and debuggers.

2. Efficiently use the Linux operating system, including real-time extensions based Xenomai, and be able to describe the inner workings.

3. Be able to write well-structured programs in C, formally correct and efficient, considering hard real-time constraints, memory constraints, and consumption constraints of physical security restrictions.

4. Be able to design and implement complete embedded systems based on the Raspberry-Pi, connecting other hardware components.

Program

Program description with approximate distribution of class hours per subject:

1. Introduction to embedded systems and basic concepts. 4h (11%)
Definition of embedded system. Cyber-physical systems. Basics architecture, compilers, operating systems for embedded systems. Introduction to the Raspberry-Pi and Linux for embedded systems.

2. Microprocessors and platforms for embedded systems. Programming embedded systems. 10h (26%)
Microprocessors, microcontrollers and peripherals. Data Path and segmentation. Development Environment. Elements of the toolchain, error analysis. Initialization kernel and user space.

3. Design and analysis of programs. Concurrent and real time systems. 8h (21%)
Planning multi-tasking software. Real-time systems. Cyclic Executives. Planning priorities. Methods of Analysis of the execution time in worst case. Shares. Calculation of maximum blockage. Priority ceiling protocols.

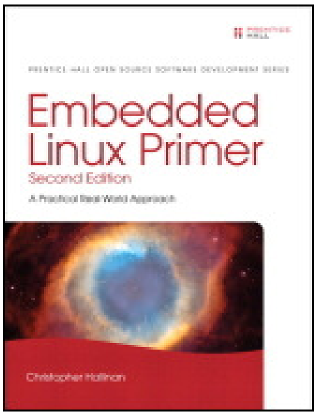
4. Systems design techniques. Modeling (models of computation). 4h (11%)
Models of computation. Invariant. Equivalence and refined. Reliability. Accessibility Analysis. Model Checking. Quantitative analysis programs. Runtime analysis in worst case.

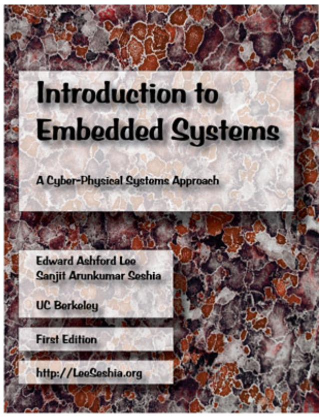
5. Low Power Design. Consumption optimization. 4h (11%)
Basics consumption in integrated circuits. Models of high-level consumption. Consumption reduction techniques in hardware. Consumption reduction techniques in software.

6. Design techniques to reduce memory usage. Memory Optimization. 4h (10%)
Design patterns to reduce memory consumption. Memory hierarchies. Technical architectural memory optimization. Scratchpad memories. Loop buffers.

7. Security in embedded systems. 4h (10%)
Introduction to security in embedded systems. Logical security and physical security. Auxiliary channel attacks. Countermeasures and design recommendations.

Bibliography





Main

1. Christopher Hallinan, Embedded Linux Primer: A Practical, Real-World Approach, Second Edition, Prentice Hall, ISBN-13: 978-0-13-701783-6, 2010.
2. Edward A. Lee and Sanjit A. Seshia, Introduction to Embedded Systems, A Cyber-Physical Systems Approach, http://LeeSeshia.org, ISBN 978-0-557-70857-4, 2011.

Complementary

3. Marilyn Wolf, Computers as Components: Principles of Embedded Computing System Design, 3rd edition, Morgan Kaufmann, ISBN 978-0-12-388436-7, 2012.
4. Jane W. S. Liu, Real-Time Systems, Prentice Hall, ISBN 0-13-099651-3, 2000.
5. Karim Yaghmour, Jon Masters, Gilad Ben-Yossef, and Philippe Gerum, Building Embedded Linux Systems, 2nd edition, O'Reilly, ISBN 978-0-596-52968-0, 2008.
6. Artículos y documentación del portal moodle de la asignatura.

Teachers

Coordinator: José Manuel Moya Fernández

Teachers: José Manuel Moya Fernández
Alvaro Araujo Pinto

Teaching Methodology

Proposal simple exercises embedded systems-based Raspberry Pi to approach different issues, making explicit the difficulties and challenges.

Classes theoretical exposition of the topics by the teachers.

Personal work to solve the exercises, delivered by the portal of the course moodle.

Pooling the results of the exercises and practical aspects of design and optimization.

Continued use of the forums moodle portal of the subject as basic communication mechanism.

Evaluation

• Proposed exercises throughout the course 50%

• Final exam without books or notes 50%

Contact

José Manuel Moya Fernández
e-Mail: jm.moya@upm.es
Desk: B104.1b
Telephone: 915495700, ext 4224

Alvaro Araujo Pinto
e-Mail: alvaro.araujo@upm.es
Desk: B104.1b
Telephone: 915495700, ext 4224