Course guides
230707 - AES - Automotive Embedded Systems

Last modified: 29/04/2020

Unit in charge: Barcelona School of Telecommunications Engineering
Teaching unit: 744 - ENTEL - Department of Network Engineering.

Degree: MASTER'S DEGREE IN TELECOMMUNICATIONS ENGINEERING (Syllabus 2013). (Optional subject).
MASTER'S DEGREE IN ELECTRONIC ENGINEERING (Syllabus 2013). (Optional subject).
MASTER'S DEGREE IN ADVANCED TELECOMMUNICATION TECHNOLOGIES (Syllabus 2019). (Optional subject).

Academic year: 2020 **ECTS Credits:** 5.0 **Languages:** English

LECTURER

Coordinating lecturer: de la Cruz Llopis, Luis J.

Others: Moreno Arostegui, J. Manuel
Madrenas Boadas, Jordi
Franch Gutiérrez, Xavier
Abella Ferrer, Jaume

TEACHING METHODOLOGY

Lectures
Application classes
Laboratory classes
Laboratory sessions
Individual work (not presential)
Group work (not presential)
Short-answer tests (Control)
Short-answer tests (Test)
Extended-response tests (Final Exam)

LEARNING OBJECTIVES OF THE SUBJECT

Nowadays there is an evident evolution in the automotive sector towards vehicles that make an exhaustive use of electronic and communications technologies. Vehicles with all kinds of sensors and actuators (temperature, proximity, cameras, driving assistance, parking, ...), long and short range communications technologies (4G, 5G, RFID/NFC, WiFi, ...) and its associated services (Internet access, infotainment, teleoperation, ...) make driving more comfortable, reliable and safe. To provide vehicles with all these new functionalities, there are many companies, both national and international, which focus their efforts on the production of systems that allow vehicle factories to be at the vanguard in the market. These companies are currently an attractive destination for telecommunications and computer engineers with special interest in the development and programming of embedded systems.
This subject arises from the need to offer a first specialization for engineers who wish to guide their professional career towards the contribution in the development of these electronic communication systems. It has been designed in collaboration between research groups from different departments of the UPC and working groups of leading companies in the electronics sector for the automotive industry. Its contents include aspects related to embedded software development, real-time operating systems, communication buses and reference architectures. In addition, the processes related to the evaluation, verification, validation and functional safety of the developed software are studied. As a result, an interesting preliminary training is offered that allows graduates to enter with guaranteed success in this exciting industry.

STUDY LOAD

Type	Hours	Percentage
Hours small group	13,0	10.40
Self study	86,0	68.80
Hours large group	26,0	20.80

Total learning time: 125 h

CONTENTS

1. Introduction

Description:
1. Introduction.
1.1. Opening.
1.2. The V Model.
1.3. Emerging concepts.
1.3.1. Embedded software and telematics.
1.3.2. Functional safety, software architecture and testing.
1.3.3. Process assessment.
1.4. Structure of the course. Practicalities.

Full-or-part-time: 0h 45m
Theory classes: 0h 45m

2. Embedded software.

Description:
2.1. Embedded software design principles.
2.1.1. Algorithm design and coding practices.
2.1.2. Advanced I/O techniques.
2.1.2.1. DMA-handled I/O.
2.1.2.2. Interrupt-handled I/O.
2.1.3. MISRA-C design rules and good practices.
2.2. RTOS.
2.2.1. Introduction.
2.2.2. Kernel.
2.2.3. Tasks, multitasking and multithreading.
2.2.4. Scheduler.
2.2.5. Inter-process communication.
2.3. The CAN communication protocol.
2.3.1. Introduction.
2.3.2. Bus topology.
2.3.3. CAN messages.
2.3.4. Physical layer.
2.3.5. Bit Timing.
2.3.6. Error handling.
2.3.7. Protocol versions (2.0A, 2.0B, Open).
2.4. Laboratory sessions.
2.4.1. Introduction to the laboratory and the design tools.
2.4.2. Design of a standalone software application.
2.4.3. Design of a software application based on an RTOS.

Full-or-part-time: 8h 30m
Theory classes: 8h 30m

3. Autosar.

Description:
3.1. Reference architectures and their role in software Systems.
3.2. AUTOSAR: a software reference architecture for the automotive industry.
3.2.1. Goals.
3.2.2. Chronology. Releases.
3.2.3. Partnership.
3.3. Background.
3.3.1. Automotive communication protocols: CAN, LIN, Flexray.
3.3.2. Diagnostics. UDS ISO 14229. Adaptation of UDS to CAN.
3.4. Constituent elements of AUTOSAR.
3.4.1. The layers.
3.4.1.1. Basic software. Dependencies.
3.4.1.2. Runtime Environment and its configuration.
3.4.1.3. Application layer.
3.4.2. The Virtual Functional Bus.
3.4.3. Interfaces.
3.5. AUTOSAR methodology.
3.5.1. Defining the architecture.
3.5.2. Development processes.
3.5.3. Software production. Code generation (model-based).
3.5.4. Data interchange.
3.5.5. Tool support.
3.6. Conclusions.

Full-or-part-time: 6h 15m
Theory classes: 6h 15m

4. Telematics.

Description:
4.1. V2X communications.
4.1.1. Intelligent Transportation Systems (ITS).
4.1.2. ETSI Architecture.
4.1.2.1. Application Layer.
4.1.2.2. Facilities Layer.
4.1.2.2.1. Basic Services and Messages.
4.1.2.3. Networking and Transport Layer.
4.1.2.3.1. Basic Transport Protocol (BTP).
4.1.2.3.2. GeoNetworking.
4.1.2.4. Acces Layer.
4.1.2.4.1. IEEE 802.11p / ITS-G5.
4.1.2.4.2. Cellular V2X.
4.2. Embedded Linux on automotive telematics.
4.2.1. Linux kernel architecture: essential points for adapting the kernel to a custom embedded platform.
4.2.2. Techniques for right-sizing the system to meet project constraints.
4.2.3. Yocto Distribution: Cross development environment for embedded projects.
4.2.4. Bootloaders. Focus on U-Boot and Android Fastboot.
4.2.5. Flash storage and file systems.
4.2.6. Developing and debugging applications for the embedded system.
4.3. Laboratory sessions.
4.3.1. Develop a Linux application for launching and interact with a Qualcomm Linux modem.

Full-or-part-time: 7h 45m
Theory classes: 7h 45m

5. Verification and validation.

Description:
5.1. Introduction.
5.1.1. Definition and importance of Software Quality Assurance & Testing.
5.1.2. Managing risks.
5.1.3. Testing in Agile & DevOps.
5.1.4. Software testing economics.
5.2. Test levels (unit testing, system testing, integration testing, ...).
5.2.1. Ways of testing software.
5.2.2. The seven principles of testing.
5.2.3. Software testing levels and responsibilities.
5.2.4. Software testing types.
5.3. Test methods (black box, white box, grey box, ...).
5.3.1. V-model and test methods.
5.3.2. The testing lifecycle.
5.3.3. Testing Management.
5.3.4. Defect Management.
5.3.5. Test cases design.
5.3.6. Practicing the testing lifecycle (test case design, test case execution and defect reporting).
5.4. Test automation.
5.4.1. Introduction to test automation.
5.4.2. Towards an acceptance test automation framework.
5.4.3. Recording vs. Layered automation.
5.4.4. Basic concepts for JUnit+Selenium automation.
5.4.5. Automated testing for APIs.
5.4.6. Practicing test cases automation (the 10-levels challenge).
5.4.7. Automated testing in mobile devices (demo).
5.5. Test-driven development.
5.5.1. The concept of Test-Driven Development.
5.5.2. Test-First & Acceptance Test-Driven Development.
5.5.3. Practicing TDD through a small example.
5.6. Conclusions.
5.6.1. Conclusions.

Full-or-part-time: 6h 15m
Theory classes: 6h 15m

6. Functional safety.

Description:
6.1. Introduction.
6.1.1. What is Functional Safety.
6.1.2. Functional Safety and Product Safety / Cybersecurity.
6.1.3. Functional safety standards and definitions.
6.1.4. ISO26262 overview.
6.2. Safety Concepts.
6.2.1. Hazard & Risk Analysis and determination of ASILs.
6.2.2. System-level architectures & examples.
6.3. Software safety.
6.3.1. Safety Requirements.
6.3.2. Sw Architectural descriptions for functional safety.
6.3.3. Patterns in Sw Architecture, E-Gas koncept.
6.3.4. Freedom from Interference concepts.
6.3.5. Safety Analysis at the sw level.
6.3.6. Autosar & Functional Safety.
6.3.7. Software safety process overview.

Full-or-part-time: 5h 30m
Theory classes: 5h 30m

7. SPICE methodology.

Description:
7.1. Introduction.
7.2. Process Maturity Models. CMM. SPICE.
7.3. Automotive SPICE.
7.3.1. Process Groups.
7.3.2. Work Products.
7.3.3. Maturity Levels.
7.4. Conclusions.

Full-or-part-time: 4h
Theory classes: 4h

GRADING SYSTEM

The evaluation is based on individual works carried out by the students, corresponding to each of the different parts of the subject.

BIBLIOGRAPHY

Basic:
- ETSI EN. EN 302 636-3-V.1.1.2 Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 3: Network Architecture [on line]. 2014 [Consultation: 16/05/2018]. Available on: http://www.etsi.org/standards.
- AUTOSAR [on line]. Available on: www.autosar.org.
- SPICE. Automotive SPICE® Process Reference and Assessment Model [on line]. RELEASE 3.1. 2017 [Consultation: 16/05/2018]. Available on: http://www.automotivespice.com/download/.
- Skiena, Steven S. The Algorithm design manual. 2nd ed. London: Springer, 2012. ISBN 9781848000698.
- Motor Industry Software Reliability Association. Guidelines for the use of the C Language in critical systems [on line]. Misra, 2012 [Consultation: 16/05/2018]. Available on: https://cds.cern.ch/record/2017046. ISBN 978-1-906400-11-8.
- International Organization for Standardization. ISO/DIS 26262. Road Vehicles - Functional Safety [on line]. 2009 [Consultation: 16/05/2018]. Available on: https://www.iso.org/obp/ui/#iso:std:iso:26262:-2:dis:ed-2:v1:en.
- Koomen, Tim; Van der Aalst, Leo; Broekman, Bart; Vroon, Mitchiel. TMap Next, for result-driven testing. Tutein Nolthenius, Uitgeverij, 2007. ISBN 9789072194800.