Universidad de Navarra

Sistemas Embebidos (MIT)
Guía docente 2026-27

PRESENTACIÓN

Breve descripción: Un Sistema Embebido (**Embedded System**) es un sistema basado en un microcontrolador con determinados periféricos, como audio, vídeo, algunos sensores, interfaces de comunicaciones tipo USB, I2C, etc. Está diseñado para realizar determinadas funciones, por lo cual en muchos casos tienen sus limitaciones. La asignatura profundizará en el diseño e implementación de un Sistema Embebido, basado en una plataforma comercial. El alumno aprenderá a agregar Hardware al sistema, como sensores digitales y analógicos. También aprenderá a trabajar en un entorno de programación basado en Linux y con lenguaje de programación C/C++

- **Titulación**: Máster en Ingeniería de Telecomunicación
- **Módulo/Materia**: Tecnología de Telecomunicación / Redes, Sistemas y Servicios
- **ECTS**: 5 ECTS
- **Curso, semestre**: 1º, Primer semestre
- **Carácter**: Obligatorio
- **Profesorado**:
  - Dr. Paul Bustamante / pbustamante@tecnun.es
  - Dr. Adam Podhorski / apodhorski@unav.es
- **Idioma**: Castellano / Inglés
- **Aula, Horario**: Lab. Informática Miramón / Horario a definir

RESULTADOS DE APRENDIZAJE

- RA2 - Que los estudiantes sepan aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios (o multidisciplinares) relacionados con su área de estudio.
- RA28- Capacidad para diseñar componentes de comunicaciones como por ejemplo encaminadores, conmutadores, concentradores, emisores y receptores en diferentes bandas.
- RA30 - Capacidad para desarrollar instrumentación electrónica, así como transductores, actuadores y sensores.

PROGRAMA

1. **Introducción**Sistemas embebidos
2. Microcontroladores
3. Periféricos: digitales y analógicos
4. Metodologías de programación
5. Buenas prácticas en el desarrollo de software para sistemas embebidos

1. **Hardware**:
   1. Descripción de la plataforma hardware de desarrollo
   2. Interrupciones y *Watchdog*
   3. Entradas I/O, PWM y Convertidores A/D
   4. Interfaces de comunicaciones: SPI, I2C, Serial, USB.
   5. JTAG y Herramientas de *Debug*
   6. Diseño de la arquitectura de un sistema
   7. Reducción de consumo energético
   8. **Software**Herramientas de desarrollo de software
   9. Control de versiones
   10. Bootloaders
   11. Metodologías de diseño de software
   12. Análisis estático del software
   13. Testeo del código
   14. Desarrollo de código para sistemas críticos de seguridad y certificación del software.

Universidad de Navarra

15. Linux para sistemas embebidos
16. **Procesamiento en Tiempo Real**Requisitos de un sistema en tiempo real
17. Sistemas operativos en tiempo real
18. *Preemptive scheduling and priority process*
19. Ejecución concurrente, *reentrancy* e instrucciones atómicas

2. Reducción de la carga de CPU
    1. PRU-ICSS
    2. Cache
    1. Acceso directo a memoria (DMA)
    2. Optimización
3. **Proyecto**

Desarrollo de un **proyecto con sensores**, aplicando las metodologías vistas en la asignatura.

PROGRAM:

- **Introduction**
    1. Embedded systems
    2. Microcontrollers
    3. Peripherals: digital and analog
    4. Programming methodologies
    5. Good practices in software development for embedded systems
- **Hardware:**
    1. Description of the development hardware platform
    2. Interrupts and Watchdog
    3. Inputs I / O, PWM and A / D Converters
    4. Communications interfaces: SPI, I2C, Serial, USB.
    5. JTAG and Debug Tools
    6. Design of the architecture of a system
    7. Reduction of energy consumption
- **Software**
    1. Software development tools
    2. Version control
    3. Bootloaders
    4. Software design methodologies
    5. Static software analysis
    6. Code testing
    7. Code development for critical security systems and software certification.
    8. Linux for embedded systems
- **Real Time Processing**
    1. Requirements of a real-time system
    2. Real-time operating systems
    3. Preemptive scheduling and priority process
    4. Concurrent execution, reentrancy, and atomic instructions
    5. CPU load reduction
    6. PRU-ICSS
    7. Cache
    8. Direct memory access (DMA)
    9. Optimization
- **Final Project**
    1. Development of a **project with sensors**, applying the methodologies seen in the subject.

ACTIVIDADES FORMATIVAS

La asignatura está orientada desde un punto de **vista práctico**. A lo largo del curso el alumno realizará una serie de prácticas formativas sobre cada elemento constituyente de un sistema embebido. Estas prácticas estarán acompañadas por **clases teóricas** sobre los conocimientos que los alumnos deberán adquirir en esta asignatura.

Universidad de Navarra

El conjunto de las prácticas servirá como base para la realización de un **proyecto final**.

EVALUACIÓN

CONVOCATORIA ORDINARIA

1. Actitud y Trabajo en Clase: 20%
2. Evaluaciones: 30%
3. Proyecto Final: 50%

CONVOCATORIA EXTRAORDINARIA

1. Proyecto: 50%
2. Examen teórico: 50%

HORARIOS DE ATENCIÓN

Dr. Paul Bustamante (pbustamante@unav.es)

- Despacho 212 Edificio Miramon. Planta 02
- Horario de tutoría: a definir

BIBLIOGRAFÍA

1. **Sistemas embebidos:**
    1. Elecia White, *Making Embedded Systems*, O'Reilly Media, 2011. Localízalo en la biblioteca / Localízalo en la biblioteca (2a. ed)
    2. Jack Ganssle, *The Firmware Handbook*, Newnes, 2004. Localízalo en la biblioteca (formato papel y electrónico)
    3. Jack Ganssle, *Embedded Systems: World Class Designs*, Newnes, 2007. Localízalo en la biblioteca (formato electrónico)
    4. David Simon, *An Embedded Software Primer*, Addison-Wesley Professional, 1999. Localízalo en la biblioteca
    5. James W. Grenning, *Test Driven Development for Embedded C*, Localízalo en la biblioteca
2. **Lenguajes de Programación:**
    1. Richard Reese, *Understanding and Using C Pointers*, O'Reilly Media, 2013.
    2. Stephen Kochan, *Programming in C*, Adison-Wesley, 4th Edition, 2014.
    3. Kyle Loudon, *Mastering Algorithms with C*, O'Reilly Media, 1999.
    4. Bjarne Stroustrup, *Programming: Principles and Practice Using C++*, Addison-Wesley Professional, 2nd Edition, 2014.
    5. Mark Lutz, *Learning Python*, O'Reilly Media, 5th Edition, 2013.
3. **Específica de la Plataforma de desarrollo:**

Derek Molloy, *Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux*, Wiley, 2015. Localízalo en la biblioteca (formato papel) Localízalo en la biblioteca (formato electrónico)