Embedded Systems Engineering

3. Semester

Subject to Approval by the Relevant Bodies

Course	Type	THW	ECTS
Project Management	SE	2	3
Electronic Engineering Project	PT	0,5	6
System-on-Chip Design	IC	4	6
Design Verification	IC	2	3
Continuous Delivery in Embedded Systems	IC	4	6
Realtime Computing	IC	4	6
		16,5	**30**

SE Seminar

PT Project

IC Integrated course

THW Hours per week

ECTS European Credit Transfer and Accumulation System

Project Management 2 SWS/3 ECTS

Teaching Content

- Project types and methodologies
- Project phases, milestones and gates
- Project plans
- Effort estimation methodologies
- Communication plans and meeting structures
- Resource-based and skill-based planning
- Risk management
- Failure mode and effect Analysis
- Agile project management

Competence Acquisition

After finishing this course, students can

- explain the nature of projects,
- compare different methods of project management,
- describe the different roles of persons involved into a project,
- create a project plan and add phases, milestones and gates of projects,
- estimate the efforts for tasks, work packages, and projects, and
- explain the risk management of a project.

Electronic Engineering Project 0,5 SWS/6 ECTS

Teaching Content

- Solving a specific technical task on the under supervision that corresponds to the level of education.

Competence Acquisition

After finishing this course, students can

- work on and solve a technical problem independently and,
- document and present the solution.

System-on-Chip Design 4 SWS/6 ECTS

Teaching Content

- Fundamentals of a modern instruction set architecture (e.g. RISC-V, ARM, etc.).
- Design and simulation of custom bus peripherals
- Development of a top-level chip design with a processor core, custom bus peripherals as well as existing IP from libraries
- Instantiation of the architecture on a programmable logic device (FPGA) and verification via execution of software applications
- Introduction to a chip design workflow and toolchain for application specific integrated circuits (ASICs)
- Implementation of the top-level chip design for production

Competence Acquisition

After finishing this course, students can

- are familiar with a modern instruction set architecture and its various extensions,
- will be able to design and verify custom bus addressable peripherals,
- will be able to assemble a microarchitecture in an FPGA device for prototyping,
- can use existing hardware IPs via embedding them within their design,
- can program and execute software for the designed architecture to verify its general functionality, and
- are familiar with a chip design workflow and can prepare a digital design for production.

Design Verification 2 SWS/3 ECTS

Teaching Content

- Introduction to functional verification
- Functional verification in electronic circuit development: system level, schematic and PCB
- Functional verification in the System-on-Chip (SoC) design flow
- Ensure the correctness and reliability of a design
- Requirements driven verification: requirement specification and verification plan, simulation, and creating testbenches
- Verification methods like pseudo-random stimuli generation, coverage-driven verification, assertion-based verification, and self-checking mechanisms
- RTL und gatelevel-simulation
- Hardware description language usage for verification
- Reporting the findings and addressing any identified issues

Competence Acquisition

After finishing this course, students can

- analyse the requirements of an electronic device design,
- create a verification plan,
- create a testbench architecture for a SoC,
- setup a unit level or SoC level testbench with all the required components by a hardware description language,
- setup a regression to simulate the unit with several seeds, and
- analyse the failures and report them, apply required changes.

Continuous Delivery in Embedded Systems 4 SWS/6 ECTS

Teaching Content

- IoT architectures: selection of architectural patterns with a focus on IoT systems, message-oriented architectures and microservices
- Architectural risk analysis: attack surface analysis, and performing threat modeling to assess potential security risks
- Software development lifecycle: phases of software development and process models used to manage the software development process
- Continuous integration pipeline: pre-commit stage (code inspection, test-driven development), commit stage (unit testing, security testing, static code analysis, artifact repository)
- Continuous delivery pipeline: acceptance stage (deployment into the test environment, acceptance testing, penetration testing), production stage (release into production)

Competence Acquisition

After finishing this course, students can

- master selected architecture patterns and communication protocols for designing embedded and IoT systems,
- analyse the risk of embedded and IoT architectures to identify potential security issues in the design,
- set up a CI/CD pipeline and carry out or automate the activities of the individual stages (learned in the previous courses),
- master the concept of infrastructure as code and are able to use concrete container technologies to facilitate the deployment of release candidates (for test and productive environments), and
- design and automate complex acceptance and penetration test cases for embedded and IoT systems.

Realtime Computing 4 SWS/6 ECTS

Teaching Content

- Computer Architecture Review
- HW / SW partitioning and synchronization of HW and SW tasks to avoid race conditions
- Multitasking/Multithreading and Scheduling, Interprocess Communication (Pipes, Queues, Shared Memories)
- Operating Systems (Embedded Linux, Free RTOS)
- Unix/Linux Ecosystem
- Programming Toolchains
- Debugging and Profiling
- Networking
- HW Acceleration, custom circuits, and hardened communication interfaces in the FPGA
- SW-HW Communication using Registers, Dualport-RAMs, FIFO-Queues and Direct Memory Access (DMA) as well as Scheduling in the FPGA device

Competence Acquisition

After finishing this course, students can

- explain the implications of embedded systems with respect to software development,
- use operating systems for task scheduling, profiling, and debugging,
- analyse real-time requirements and the available strategies to meet them,
- apply high-level programming languages for implementing typical tasks of embedded systems and know how to deploy them,
- design and use hardware interfaces and custom circuits in an FPGA, and
- access FPGA resources with software applications.