Master of Science

Computer Engineering (Technische Informatik)

(MScTI)

Description of the course modules (Modulhandbuch)

Heidelberg University

Faculty of Engineering Sciences

Version 2.3

Key Information

Name of university	Heidelberg University
Name of department	Faculty of Engineering Sciences
Name of degree course	Master of Science Computer Engineering
Type of degree course	Consecutive
Acronym	MScTI
Formats of studies	Full time or part time
Prescribed duration of study	2 years, i.e. 4 semesters
Total number of credit points	120
Location	Heidelberg
University places	Unlimited
Target group	Holders of Bachelor of Science, Magister, Staatsexamen, Diploma or equivalent final degree of at least 6 semester study. Major in computer science, Mathematics, Natural Sciences or Engineering with proven knowledge in computer science (modules with $\approx$ 24 CP).
Date of version	17.01.2025

1 Qualification Goals and Profile

1.1 Präambel – Qualification Goals at Heidelberg University

In keeping with Heidelberg University's mission statement and constitution, degree programs are designed to provide a comprehensive academic education, incorporating subject-specific, cross-disciplinary, and career-related objectives that prepare students for their future professional careers. The resulting skills profile is included in the course description for all university disciplines and is implemented in each degree program's specific qualification objectives, curricula, and modules.

The main points of the competence profile are:

- Development of subject-specific skills, with a particular emphasis on research
- Development of the skills required for trans-disciplinary dialogue
- Development of practical problem-solving skills
- Development of personal and social skills
- Promotion of students' willingness to assume social responsibility on the basis of the skills acquired.

1.2 Profile of the MScTI

The research oriented master program in Computer Engineering (MScTI) at Heidelberg University is organized by the Institute of Computer Engineering and the Faculty of Engineering Sciences. Its educational objective is to deepen and broaden the student's expertise and prepare them for a research or development oriented professional career in the field of Computer Engineering as well as for participation in PhD programs. The students develop a thorough understanding of various approaches and methods and are able to assess their advantages and drawbacks in order to develop the best solution for a given problem. They are able to realize which solutions are inappropriate or suboptimal and have the necessary skills to devise novel approaches. The MScTI has a focus on practical skills. Students learn to work with state-of-the-art tools from computer engineering research and are able to apply these skills to develop working solutions for application-oriented problems efficiently.

Students can choose between two specializations *Emerging Computing, Microelectronics*, which are sub-fields of 'computer engineering'. Each specialization consists of a set of modules on an advanced level, which cover the field to a large extent. By following a sufficient number of modules in such a specialization, students reach the state-of-the art in the area to become fully competitive.

The master includes a research phase, consisting of a Seminar, a Study Project and the Master Thesis, where students acquire the ability to do independent research and to document and publish their own research work. They deepen their knowledge on scientific methods, information engineering, hardware and software, interdisciplinary system thinking, experience in practical applications as well as the communication competence and the ability to work in teams.

1.3 Subject-specific Qualification Objectives

Graduates have a strong foundation in microelectronics and computing systems, enabling them to design and implement both analogue and digital circuits as well as develop efficient hardware and software solutions. They are proficient in working with microprocessors, peripheral circuits, and reconfigurable architectures such as FPGAs. Additionally, they understand modern computing architectures and algorithms, including parallel and embedded systems, and can optimize performance, power efficiency, and scalability. Their expertise allows them to address hardware-specific challenges and leverage emerging technologies to develop innovative and high-performance computing solutions.

1.4 Transdisciplinary Qualification Objectives

Graduates of the MScTI possess the skill to work independently with a variety of tools for various special applications and to choose the appropriate ones to solve specific problems. They are able to work in a structured way and can organize complex professional projects. Additionally, they can acquire a basic understanding about legal and financial aspects of creating and running a company and are able to apply marketing strategies and tools.

1.5 Employment Opportunities

The increasing demand for advanced information technology requires a deep understanding of the underlying compute hardware, whether in data centers, automotive systems, mobile and edge devices, or specialized high-performance computing. Optimizing performance, power efficiency, and cost is essential for selecting the right hardware for each application. Graduates with expertise in microelectronics and computing systems are well-equipped to meet these challenges, making them valuable to a wide range of industries. They can pursue careers in semiconductor design, embedded systems, and hardware acceleration, working for both established tech companies and innovative startups. Additionally, Europe's growing focus on microelectronics independence is driving a rising demand for experts in hardware design and development. As a result, graduates will find opportunities in both emerging companies and major industry players developing next-generation computing solutions.

2 Structure

The modules in the MScTI belong to one of the categories[1]

- *Fundamentals* (Grundlagen),
- *Specializations* (Vertiefung),
- *Free Courses* (frei wählbare Veranstaltungen),
- *Seminar*,
- *Soft Skills*,
- *Study Project* (Studienarbeit) and
- *Master Thesis.*

These categories are described in the following subsections:

Fundamentals

The modules in category Fundamentals shall contribute to establish a well defined background knowledge and competences for all students of the MScTI, who may have completed Bachelor studies in very different fields.

- Advanced Computer Architecture
- System Design (can be substituted, see note)

are mandatory. Because all new students must attend, these modules also help to know each other.

Note, that due to the closure of the Specialization "Biorobotics", the lecture "System Design", that is mandatory according to the most recent examination regulations will not be offered anymore. A revised version of the examination regulations is currently compiled. Until it is finalized, the Study commission and the Examination board agreed that the lecture "System Design" can be substituted by any additional lecture from Specializations.

Specializations

These modules treat, on an advanced level, topics from the main research directions of ZITI:

- *Microelectronics* and
- *Emerging Computing.*

The available modules and their assignment to one or more specializations are listed in the table on page 8. Students can chose freely among all modules in this category. A total of 7 modules must be completed. We strongly recommend, however, to concentrate on one specialization topic. Completing a significant number of modules from one specialization provides a broad knowledge base and eases the completion of the Master Thesis. The successful completion of a specialization will be documented explicitly in the Master Grade Report as a 'major' if

- 5 modules from the specialization are completed
- including the 2 modules of that specialization labeled as 'compulsory'.

[1]This version of the module handbook was created as part of a general revision of the degree program in the context of the HeiQuality quality assurance procedures. Since the examination regulations take longer to change than the module handbook, there might be a time window, where module handbook and examination regulations are out of sync. In case of questions, please contact the ZITI study program advisors.

Free Courses

The aim of the Free Courses is to broaden expertise and knowledge. The prime intention is to look beyond computer science and engineering, but such modules can be chosen as well. Students can benefit from the huge lecture program of Heidelberg University and gain for instance background knowledge in fundamentals or applications of their research track. The lectures can be chosen freely from the course catalog of the Heidelberg University. Modules on Bachelor level are allowed, but should not repeat existing knowledge. The modules chosen as Free Courses must meet the following conditions:

- they are graded,
- the sum of credit points is 12 CP or more,
- they broaden expertise beyond the status quo, i.e. they do not repeat already completed modules, for instance from a previous Bachelor program.

Obviously, Bachelor modules completed in a previous study do not broaden knowledge and cannot be counted. In case of doubt, the Prüfungsausschuss decides on the approval. The application must be filed before the module is taken.

It is also possible to select modules from the MScTI for the Free Courses. This can be useful if the choice of a preferred track is not yet clear in the first semesters. In this situation, Specializations modules from several tracks can be completed in the early phase of the studies. If one track is then chosen, the modules which are not from that track can be assigned to the Free Courses.

Soft Skills

10 CP must be completed in the field of soft skills, the following courses can be chosen:

- Modules from the MScTI classified as 'Soft Skill' in the table below,
- Two introductory modules from the Entrepreneurship certificate of UHD (6 CP),
- Courses from the University course program classified as soft skill courses,
- Language courses up to a maximum of 6 CP.

Note: The well established module 'Tools' (page 12) gives student a quick overview and introduction to useful software and methods. The module relies on practical, supervised, on-site participation with no significant homework, so that only 4 CP are justified. This smaller number of 'soft-skill' points also makes it more compatible to other BSc or MSc programs.

Study Project

This module introduces the student to the work in a chosen research group. The topic is typically on an introductory level so that background knowledge for the specific task can be gained and tools required to complete the task can be learned. The topic can be (but does not have to) an introduction to the subject which will later be treated in the Master Thesis. The Study Project is completed by a report. As 14 CP are assigned, the work load is roughly 50% of the 3rd semester load, i.e. significant.

Master Thesis

The master thesis spans a duration of 6 months and is usually carried through in the specialization (if one is chosen) in the 4th semester. After handing in the written thesis document, the results are presented in a final, public colloquium which enters into the grading by the reviewers.

Summary

In summary, the following number of modules have to be completed successfully in the various categories in order to reach the required 120 CP:

2 modules from Fundamentals *(see note above)*	12 CP
6 modules from Specializations	36 CP
2 modules from Free Courses	12 CP
2 or more modules from Soft Skills	10 CP
Seminar	6 CP
Study Project	14 CP
Master Thesis with final colloquium	30 CP
Sum	120 CP

Mobility Window

The best choice for a semester abroad are the 2nd or the 3rd semester. The 12 CP of the Free Courses as well as, for instance, 6 CP in the area of Soft Skills can easily be accomplished abroad because they can be accepted with few restrictions. If the students finds typically 2 modules of 2×6 CP abroad which match the topics in the Specializations (to be agreed upon with the ERASMUS-Coorinator beforehand), the full 30 CP of the respective semester can be obtained abroad.

Modules Overview and Relationships

The following table shows all regular modules of the MScTI and their assignment to the Fundamentals, Soft Skills or one of the three Specializations. The circle ($\circ$) indicates that the module is compulsory in the respective category or track. The same module cannot be counted twice.

Note that additional modules which are held exceptionally for instance by guest lecturers can be accepted and assigned to one of the categories by the Studienkommission.

Module	Fundamentals	Soft Skills	Spec.: Microelec.	Spec.: Em. Comp.	Module Responsible	page
Advanced Computer Architecture	∘				NT	10
Seminar					AS	11
Tools		×			PF/AS	12
Components, Circuits & Simulation			∘		PF	13
Digital Hardware Description and Verification			∘		DK	14
Full Custom VLSI Design			×		PF	15
Advanced Analogue Building Blocks			×		PF	16
Digital Semicustom Design Flow			×		AG	17
Reconfigurable Embedded Systems			×	×	DK	18
Performance Essentials for CPUs and GPUs				∘	RS	19
GPU Computing (Architecture + Progr.)				∘	HF	20
Embedded Machine Learning				×	HF	21
Scalable and Robust Embedded Machine Learning				×	HF	22
CPU Algorithm Design				×	RS	23
GPU Algorithm Design				×	RS	24
Consistency and Coherency				×	HF	25
High Perf. and Distr. Comp.				×	HF	26
Emerging Computing Paradigms			×	×	NT	27
Architecture and CAD for FPGA			×	×	DK	28
Energy Efficient Computing			×	×	DK	29
Memory-Centric Computing			×	×	NT	30
Biosignal Processing				×	NT	31
Research Project						32
Master Thesis						33

Table 1: Modules in the MScTI and assignment to the various categories. Compulsory modules in a category are labeled with ∘.
AA = Dr. Amin Aminifar AG = Dr. Andreas Grübl, AS = Dr. Alexander Schubert, DK = Prof. Dr. Dirk Koch, HF = Prof. Dr. Holger Fröning, MH = Dr. Mostafa Haghi, NT = Prof. Dr. Nima TaheriNejad, PF = Prof. Dr. Peter Fischer, RS = Prof. Dr. Robert Strzodka.

3 Module Descriptions

The following pages contain the descriptions of all modules offered primarily by the MScTI.

Modules are open to non-MScTI students, as long as sufficient room space and infrastructure (computer seats, lab places, compute resources...) area available. In the *Seminar*, the number of participants is limited by the time available for presentations. MScTI students are accepted with priority. Free slots are available for non-MScTI students. The *Study Project* and the *Master Thesis* are intended only for students of the MScTI.

The modules *'Current topics in'* serve as flexible modules to allow the inclusion of non-regular or spontaneous lectures (e.g. given once only by a guest professor) in the Specializations. The list of possible lectures to be taken in this module is published on the ZITI website. Additional lectures can be accepted by the examination committee (Prüfungsausschuss) on request.

None of the modules is compulsory in other bachelor or master programs.

An optional inclusion of modules in other study programs is left to these programs.

Temporary deviations from the default semester assignment or of the lecturer may sometimes be required for organizational reasons. Longer term changes will be amendment in updated versions of the document.

In the following module descriptions
- 'ST' is summer term ('Sommersemester'), lectures starting mid April,
- 'WT' is winter term ('Wintersemester'), lectures starting mid October.

Code: MScTI_ACA		**Course Title:** Advanced Computer Architecture	
Module Coordinator: Prof. Dr. Nima TaheriNejad		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:

- Lecture (3 h / week)
- Practical exercises with homework (1 h / week)

Objectives: Students...

- By the end of this lecture, the students will be able to:
- list and discuss advantages of and challenges in parallel computing
- describe at least one method of managing performance of multiprocessor systems
- list at least two interconnect technology and a method to select the suitable solution for a system
- discuss various memory systems and their performance figures
- name at least two emerging memory technologies and elaborate on their advantages and disadvantages

Content:

- Processor Architectures
- Parallel Computing
- Programming models & architectures
- Multiprocessor architectures
- Interconnects (incl. Network-on-Chip)
- Caching in Multiprocessors
- Multi-threading
- Dataflow
- Memory & storage
- Emerging memory technologies
- Introduction to emerging computing paradigms (approximate and in-memory computing)
- Heterogeneous Multiprocessor Systems

Prerequisites: none	**Recommended Knowledge:** Basic Computer Architecture

Literature:

- Lecture Notes and Handouts
- A list of sources that will be provided in the course

Testing: Defined by the lecturer before the beginning of the course

<table>
<tr><td colspan="2">Code:
MScTI_SEMINAR</td><td colspan="2">Course Title:
Seminar</td></tr>
<tr><td colspan="2">Module Coordinator:
All Groups</td><td colspan="2">Type:
Seminar</td></tr>
<tr><td>Credit Points:
4+2</td><td>Workload:
180 h</td><td>Teaching Hours:
Block</td><td>Term:
ST/ WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Presentation
• Paper
• Review of paper of other participants</td></tr>
<tr><td colspan="4">Objectives: Students...
• can search literature for a specific subject,
• can select and structure material for a presentation,
• can make slides for a presentation,
• can give a scientific presentation,
• learn to summarize a topic in a publication-type writeup,
• are familiar with the typical reviewing process of publications.</td></tr>
<tr><td colspan="4">Content:
• Literature research
• Preparation of presentation
• Oral Presentation (≈ 45 Minutes)
• Preparation of a short summary ’paper’ (≈ 10 pages)
• reviewing process of several other papers
• Active participation in other student’s presentations and discussion</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Initial knowledge in chosen field</td></tr>
<tr><td colspan="4">Literature:
• Provided by supervisor and by own literature search</td></tr>
<tr><td colspan="4">Testing: Presentation, paper, review, regular active participation</td></tr>
</table>

Code: MScTI_TOOLS		**Course Title:** Tools	
Module Coordinator: Prof. Dr. Peter Fischer		**Type:** Lecture with exercise	
Credit Points: 4	**Workload:** 120 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:
- Lecture with on-site exercises (4 h / week)

Objectives: Students...
- have an overview of the functionalities of various software tools suited to accomplish frequent tasks, like the creation of drawings and illustrations, programming, solving of mathematical problems, analysis and visualization of data, search for literature or working in a team.
- are able to improve their work flows by choosing an appropriate tool
- are aware that application of a suited tool improves their working quality and efficiency
- are able to deepen their knowledge and skill in the presented tools on the basis of the give introduction
- are familiar with the concepts of good scientific practice and can apply them to their own scientific work.
- know how to find scientific literature in professional library systems

Content:
- Topics are chosen from a pool of possibilities as a function of interest of students and availability of knowledgeable lecturers. The list of topics is regularly adapted to new developments. Example Topics are listed below:
- Introduction to Linux
- Version control tools (git, svn,..)
- Introduction to python
- Mathematical software (Mathematica)
- Data evaluation and plotting (gnuplot, root)
- 2D & 3D drawing, construction and visualization (PovRay, OpenSCAD, PostScript, pdf)
- styles and templates (powerpoint, word)
- introduction to Latex
- team work
- project planning and management
- Good scientific practice
- literature search, bibliometric measures, open access concepts,...

Prerequisites: none	**Recommended Knowledge:**

Literature:
- announced in lecture per topic

Testing: No testing. Regular attendance required.

<table>
<tr><td colspan="2">Code:
MScTI_CCS</td><td colspan="2">Course Title:
Components, Circuits and Simulation</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Peter Fischer</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
ST</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week)
• Practical exercise with homework (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• can design simple analog circuits by combining elementary building blocks
• can predict the properties (gain, frequency behavior) of simple circuits and give analytical approximate expressions for gain, bandwidth, output resistance, etc.
• can use analogue simulators to analyze circuits in the time and frequency domain
• know what an operation point is, how it affects circuit behavior and how it can be set
• can relate the geometry and operation point of transistors to their small- and large signal properties</td></tr>
<tr><td colspan="4">Content:
• Diode and transistor operation principle
• Modeling of Diode and MOS, large / small signal models
• Voltage and current sources, Thévenin equivalent
• Component and circuit description with complex variables
• Bode plot, transfer function
• Analogue simulation (dc, ac, transient)
• Basic circuits: current mirror, gain stage, cascode, source follower, differential pair
• Practical exercises with professional simulation tools</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Introduction to Physics</td></tr>
<tr><td colspan="4">Literature:
• P. R. Gray, P. J. Hurst, S. H. Lewis, R. G. Meyer: Analysis and Design of Analog Integrated Circuits (Wiley & Sons, New York, 1993)
• D. A. Johns, K. Martin: Analog Integrated Circuit Design (Wiley & Sons, 1997)</td></tr>
<tr><td colspan="4">Testing: Defined by lecturer before beginning of course</td></tr>
</table>

Code: MScTI_HDL		**Course Title:** Digital Hardware Description and Verification	
Module Coordinator: Prof. Dr. Dirk Koch		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** WT

Module Parts and Teaching Methods:
- Lecture (2 h / week)
- Practical exercise with homework (2 h / week)

Objectives: Students...
- can describe digital circuits using hardware description languages
- know typical languages like Verilog, System-Verilog, VHDL
- can meaningfully structure a large design into smaller design blocks
- have experience in designing state machines
- can set up simulation environments (test benches) for verification
- know about advanced verification methods
- realise the difference betwwen synthesiable and non-synthesizable constructs

Content:
- Syntax and semantics of Verilog, VHDL, Systemverilog
- Modules and their instantiation, simple and advanced ports
- Structural description, behavioural description
- Description of common blocks (RAM, FIFO, Register file, counter,...)
- Robust description of state machines
- Instantiation of hard macros (in FPGAs or VLSI)
- Simulation, test patterns, test benches,...
- Introduction to UVM

Prerequisites: none	**Recommended Knowledge:**

Literature:
- Defined in lecture

Testing: Defined by lecturer before beginning of course

<table>
<tr><td colspan="2">Code:
MScTI_ANADESIGN</td><td colspan="2">Course Title:
Full Custom VLSI Design</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Peter Fischer</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week)
• Practical exercise with homework (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• can carry out the complete design process from a circuit idea to a final, checked layout,
• understand how design rules are related to semiconductor properties or manufacturing issues,
• are able to practically carry out a mixed mode simulation,
• are able to extract parasitic values and perform a simulation with these parasitics,
• can program simple automatized scripts using SKILL.</td></tr>
<tr><td colspan="4">Content:
• Semiconductor manufacturing
• Technology and design rules, technology files
• Layout of components, rules, matching
• Design Rule Check
• Extraction, Layout versus Schematic Check
• ESD and Antenna rules, latchup
• Test equipment and test procedures
• Script programming using SKILL
• Parasitic extraction and simulation
• Mixed Mode simulation</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
MScTI_CCS</td></tr>
<tr><td colspan="4">Literature:
• Lecture script available online</td></tr>
<tr><td colspan="4">Testing: Design (schematic entry, simulation and layout) of a simple circuit with a short presentation.</td></tr>
</table>

Code: MScTI_AABB	**Course Title:** Advanced Analogue Building Blocks		
Module Coordinator: Prof. Dr. Peter Fischer	**Type:** Lecture with exercise		
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** WT

Module Parts and Teaching Methods:

- Lecture (2 h / week)
- Practical exercise with homework (2 h / week)

Objectives: Students...

- know a variety of advanced circuit topologies
- get a deeper qualitative and quantitative understanding of the behavior of analogue circuits,
- can define performance measures, extract them from simulations and optimize circuit parameters
- are able to chose an appropriate circuit for a given problem

Content:

- Circuit families and topics that will be treated in some detail will be chosen from:
- Noise of components and circuits
- Charge amplifier with feedback and filter
- Logic Families (NMOS, Dynamic, Pass Gate, Differential)
- PLL
- Cascaded amplifiers
- Advanced current mirrors
- Differential circuits, common mode feedback
- DACs and ADCs
- Switches
- Switched Capacitor Circuits

Prerequisites: none	**Recommended Knowledge:** MScTI_ANADESIGN

Literature:

- Razavi : *Design of analog CMOS integrated circuits*
- J. Millman: *Microelectronics*

Testing: Oral Examination

Code:
MScTI_DIGDF

Course Title:
Digital Semicustom Design Flow

Module Coordinator:
Prof. Dr. Dirk Koch, A. Grübel

Type:
Lecture with exercise, lab, project

Credit Points:	**Workload:**	**Teaching Hours:**	**Term:**
6	180 h	4 / week	ST

Module Parts and Teaching Methods:
- Lecture (2 h / week)
- Practical and practical chip project (2 h / week)

Objectives: Students...
- deepen their knowledge of the methodology for semi-custom ASIC design,
- are able to use their acquired knowledge to design very complex chips,
- can run the complete backend design process for modern chip technology.

Content:
- Advanced methods for design of application specific ICs
- Synthesis of complex hardware systems
- Static Timing Analysis (STA)
- Place & Route of modules and standard cells
- Signal integrity analysis
- Design rule checks
- Generation of mask data
- The SEED-2002 agreement between Cadence Design Systems and the University of Heidelberg allows our students to work and learn with the most modern EDA tools that are usually only used in chip industry.

Prerequisites:
none

Recommended Knowledge:
deeper knowledge of Digital Hardware Design

Literature:
- A reading list will be provided in the lecture

Testing: 30' oral exam at the end of the semester. At least 50% of the exercises and the chip project must be passed.

<table>
<tr><td colspan="2">Code:
MScTI_RES</td><td colspan="2">Course Title:
Reconfigurable Embedded Systems</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Dirk Koch</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
ST</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week)
• Practical exercise with homework (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• understand elements, properties and requirements of embedded systems
• get basic understanding of reconfigurable architectures
• learn application design methodologies for microprocessors and FPGAs
• create custom IP cores using structural data-flow and FSM based control-flow design techniques,
• use IP cores to create hybrid applications for processor and reconfigurable coprocessor with appropriate interface mechanisms,
• implement and program a sample embedded FPGA platform</td></tr>
<tr><td colspan="4">Content:
• Requirements and specific properties of embedded systems
• Overview on hardware components: microcontrollers, peripherals, FPGAs
• Real-time issues and scheduling
• FPGA design tools: HDL (incl. VHDL tutorial), simulator, debugger
• High-level FPGA design methodologies (incl. HLS)
• CAD Tool basics: how is code translated into an FPGA configuration?
• System-on-Chip architecture – controller, buses and peripherals
• HW/SW co-design
• Embedded system software (stand-alone and real-time kernels)
• Reconfigurable custom Instruction Set Architecture (ISA) extensions</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
none</td></tr>
<tr><td colspan="4">Literature:
• Provided by the lecturer or on lecture web site</td></tr>
<tr><td colspan="4">Testing: Defined by lecturer before beginning of course</td></tr>
</table>

Code: MScTI_PERF		**Course Title:** Performance Essentials for CPUs and GPUs	
Module Coordinator: Prof. Dr. Robert Strzodka		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST
Module Parts and Teaching Methods: • Lecture 2 h / week • Exercise 2 h / week plus homework			
Objectives: Students... • understand performance implications of code constructs, • design better programs following guidelines of an effective programming style, • know how to program CPUs and GPUs with the same source code, • can select efficient parallel algorithms from existing libraries.			
Content: • Most important dos and donts of efficient code • Unified high level programming of CPUs and GPUs • Efficient allocation, data access, computation • Clear and effective programming style • Parallel data structures and algorithms • Libraries for dense and sparse linear algebra • Specialized libraries			
Prerequisites: none		**Recommended Knowledge:** Programming experience	
Literature: • Bjarne Stroustrup: *A Tour of C++* (2rd ed, Addison-Wesley, 2022) • Additional material will be provided through the learning platform			
Testing: 50% of points from the exercises are required for participation in the oral or written exam.			

<table>
<tr><td colspan="2">Code:
MScTI_CEGPU</td><td colspan="2">Course Title:
GPU Computing - Architecture + Programming</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Holger Fröning</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week)
• Practical exercise with homework (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• know GPU architectures and associated design decisions,
• know the factors that determine the performance of GPU programs, and are able to program GPUs to solve computing problems,
• can design and optimize CUDA programs for compute- or memory-intensive problems,
• know how to use CUDA tools to aid in programming, debugging and performance tuning,
• are capable to solve compute- or memory-intensive problems using GPUs with objectives including performance in terms of time and energy,
• are capable to decide when accelerators like GPUs are suitable for a given computing problem.</td></tr>
<tr><td colspan="4">Content:
• Basics of GPU architecture and its programming model
• Introduction to CUDA
• Performance optimization techniques
• Consistency and coherence of GPUs
• Alternatives to CUDA and advanced GPU concepts</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Computer architecture basics, parallel programming principles, C, C++, OS basics</td></tr>
<tr><td colspan="4">Literature:
• N.W. Wilt: The CUDA Handbook: A Comprehensive Guide to GPU Programming (Addison-Wesley, 2013)
• D.B. Kirk, W.W. Hwu: Programming Massively Parallel Processors (Morgan- Kaufmann, 2010)
• T.G. Mattson, B.A. Sanders, B.L. Massingill: Parallel Patterns for Parallel Programming (Addison Wesley, 2004)
• J.L. Hennessy, D.A. Patterson: Computer Architecture: A Quantitative Approach (Morgan Kaufmann, 2017)</td></tr>
<tr><td colspan="4">Testing: The module is completed with a graded written or oral exam. The final grade of the module is determined by the grade of this exam. The requirements for the assignment of credits follows the regulations in section modalities for exams. Details will be given by the lecturer at the start of the semester.</td></tr>
</table>

Code: MScTI_CEEML		**Course Title:** Embedded Machine Learning	
Module Coordinator: Prof. Dr. Holger Fröning		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST
Module Parts and Teaching Methods: • Lecture (2 h / week) • Practical exercise with homework (2 h / week)			
Objectives: Students... • Know machine learning (ML) basics • Are familiar with neural network architectures for image, signal, and speech processing • Can design such model architectures for simple problems • Understand the computational requirements of such architectures • Know different processor and system architectures to execute ML models • Are capable of finding solutions to deploy ML models on resource-constrained processors			
Content: • Basics of ML • Neural architecture design, including basics of neural networks, automatic differentiation and optimization, regularization and generalization • Applications including time series and computer vision • Safe optimizations with regard to SW and HW • Unsafe optimizations with regard to SW and HW • Future directions of embedded ML			
Prerequisites: none		**Recommended Knowledge:** Computer architecture (e.g. courses “GPU Computing”, “Introduction to HPC”, or ”Advanced Computer Architecture”), Python	
Literature: • I. Goodfellow, Y. Bengio and A. Courville: *Deep Learning* (MIT Press, 2006) • B. Reagen et.: *Deep Learning for Computer Architects (Synthesis Lectures on Computer Architecture)* (Morgan & Claypool. 2017) • C. M. Bishop: *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer. 2006)			
Testing: The module is completed with a graded written or oral exam. The final grade of the module is determined by the grade of this exam. The requirements for the assignment of credits follows the regulations in section modalities for exams. Details will be given by the lecturer at the start of the semester.			

Code: MScTI_SREML		**Course Title:** Scalable and Robust Embedded Machine Learning	
Module Coordinator: Prof. Dr. Holger Fröning		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:
- Lecture (2 h / week)
- Practical exercise with homework (2 h / week)

Objectives: Students...
- Know advanced machine learning (ML) techniques and emerging hardware technologies
- Are familiar with advanced neural architectures for various tasks
- Can design such neural architectures for advanced problems
- Understand the computational requirements of such architectures
- Know conventional and non-conventional processor and system architectures to execute advanced neural architectures
- Are capable of finding solutions to deploy advanced neural architectures on resource-constrained processors

Content:
- Introduction to the intersection of emerging hardware technologies and advanced neural architectures
- Scalable machine learning methods
- Learning uncertainties
- Safe and unsafe optimizations for advanced neural architectures and conventional and nonconventional processor and system architectures
- Cost-quality trades
- Future trends including probabilistic processors and scaling rules

Prerequisites:	**Recommended Knowledge:**
MScTI CEEML or equivalent	Computer architecture (e.g. courses “GPU Computing”, “Introduction to HPC”, or ”Advanced Computer Architecture”), Python

Literature:
- Richard McElreath: *Statistical Rethinking* (2020, Chapman & Hall)
- Kevin Patrick Murphy: *Probabilistic Machine Learning – An Introduction* (MIT Press, 2022)
- Kevin Patrick Murphy: *Probabilistic Machine Learning: Advanced Topics* (MIT Press, 2023)

Testing: The module is completed with a graded written or oral exam. The final grade of the module is determined by the grade of this exam. The requirements for the assignment of credits follows the regulations in section modalities for exams. Details will be given by the lecturer at the start of the semester.

Code: MScTI_CPUALG		**Course Title:** CPU Algorithm Design	
Module Coordinator: Prof. Dr. Robert Strzodka		**Type:** Lecture with exercises and project	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:
- Lecture 2 h / week
- Exercise 2 h / week plus homework in first half of semester
- Project 2 h / week plus homework in second half of semester

Objectives: Students...
- are able to exploit all the parallelism and bandwidth in large CPUs,
- can make design decisions depending on tradeoffs in parallel algorithms,
- know how to apply advanced transformations for higher parallelism and data locality,
- understand parallel performance models and scalability.

Content:
- Parallel performance models and scalability
- Multiple levels of parallelism
- Parallel design patterns
- Parallel data access
- Communication vs. computation
- Latency vs. throughput
- Work efficiency vs. step efficiency
- Locality vs. parallelism
- Tools for parallel programming

Prerequisites: Familiarity with C++	**Recommended Knowledge:** C++17 and STL (e.g. MScTI_PERF)

Literature:
- Michael McCool, Arch Robison, James Reinders: Structured Parallel Programming, Morgan Kaufmann, 2012
- Michael Voss, Rafael Asenjo, James Reinders: ProTBB, Springer Nature, 2019
- Additional material will be provided through the learning platform

Testing: 50% of points from the exercises are required for participation in the project exam, which consists of a software design with documentation and a presentation of results. Alternatively to the project exam, an oral (20-30 min) exam may be announced by the Module Coordinator.

<table>
<tr><td colspan="2">Code:
MScTI_GPUALG</td><td colspan="2">Course Title:
GPU Algorithm Design</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Robert Strzodka</td><td colspan="2">Type:
Lecture with exercises and project</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture 2 h / week
• Exercise 2 h / week plus homework in first half of semester
• Project 2 h / week plus homework in second half of semester</td></tr>
<tr><td colspan="4">Objectives: Students...
• are able to exploit all the parallelism and bandwidth in large CPUs,
• can make design decisions depending on tradeoffs in parallel algorithms,
• know how to apply advanced transformations for higher parallelism and data locality,
• understand how to balance numerical efficiency and parallel efficiency.</td></tr>
<tr><td colspan="4">Content:
• Most recent developments in GPUs
• On-the-fly data transformations
• Data locality optimizations
• Hierarchical algorithms
• SIMD utilization
• Precision, accuracy and numerical schemes
• Numerical efficiency vs. parallel efficiency
• Data representation</td></tr>
<tr><td colspan="2">Prerequisites:
Familiarity with C++ and CUDA</td><td colspan="2">Recommended Knowledge:
C++17 and STL, GPU and CUDA</td></tr>
<tr><td colspan="4">Literature:
• David B. Kirk, Wen-mei W. Hwu: Programming Massively Parallel Processors (3rd ed, Morgan Kaufmann, 2017)
• Additional material will be provided through the learning platform</td></tr>
<tr><td colspan="4">Testing: 50% of points from the exercises are required for participation in the project exam, which consists of a software design with documentation and a presentation of results. Alternatively to the project exam, an oral (20-30 min) exam may be announced by the Module Coordinator.</td></tr>
</table>

Code: MScTI_CECOCO		**Course Title:** Consistency and Coherence	
Module Coordinator: Prof. Dr. Holger Fröning		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:
- Lecture (2 h / week)
- Practical exercise with homework (2 h / week)

Objectives: Students...
- know coherence and consistency principles of parallel architectures,
- know design techniques for scalable synchronization and communication primitives,
- are familiar with advanced concepts like transactional memory, relaxed consistency, and scalable coherence
- know how to design and optimize complex parallel code for particular communication and synchronization problems,
- are capable of solving complex computing problems using massively parallel processors, understanding the implications of architectural design decisions on performance in terms of time and energy, and reasoning about the suitability of certain processor architectures for a given computing problem.

Content:
- Shared memory architectures
- Programming paradigms, communication and synchronization concepts and algorithms
- Consistency models and scalable cache coherence
- Multi-/many-core and multi-threading architectures
- Emerging architectures

Prerequisites: none	**Recommended Knowledge:** Computer architecture (e.g. “GPU Computing”, “Introduction to HPC”, or ”Advanced Computer Architecture), parallel programming principles, C, C++, OS basics

Literature:
- M. Herlihy, N. Shavit: *The Art of Multiprocessor Programming* (Morgan Kaufmann, 2012)
- J.L. Hennessy, D.A. Patterson: *Computer Architecture: A Quantitative Approach* (Morgan Kaufmann, 2017)

Testing: The module is completed with a graded written or oral exam. project. Details will be given by the lecturer at the start of the semester.

<table>
<tr><td colspan="2">Code:
MScTI_CEHPDC</td><td colspan="2">Course Title:
High-Performance and Distributed Computing</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Holger Fröning</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
ST</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week)
• Practical exercise with homework (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• know message passing, cluster computing architectures, and scalable programming
• are familiar with the most important past and present concepts for large-scale computing problems
• can design and optimize solutions for large-scale computing problems
• know how to use MPI and related software tools to implement large-scale computing problems
• are capable to solve large-scale computing problems with objectives including performance in terms of time or energy, and scalability in terms of time and capacity</td></tr>
<tr><td colspan="4">Content:
• HPC architectures and message passing
• Parallel algorithm design and Message Passing Interface (MPI)
• MPI internals
• Characterization and benchmarking
• Short introduction to parallel training of machine learning models
• Practical problems and their solutions</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Computer architecture basics, parallel programming principles, C, C++, OS basics</td></tr>
<tr><td colspan="4">Literature:
• G. Hager and G. Wellein: Introduction to High Performance Computing for Scientists and Engineers (Taylor & Francis Inc, 2011)
• I. Goodfellow, Y. Bengio and A. Courville: Deep Learning (MIT Press, 2006)</td></tr>
<tr><td colspan="4">Testing: The module is completed with a graded written or oral exam. The final grade of the module is determined by the grade of this exam. The requirements for the assignment of credits follows the regulations in section modalities for exams. Details will be given by the lecturer at the start of the semester.</td></tr>
</table>

Code: MScTI_ECP	**Course Title:** Emerging Computing Paradigms		
Module Coordinator: Prof. Dr. Nima TaheriNejad	**Type:** Lecture with exercise		
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** WT

Module Parts and Teaching Methods:
- Lecture (2 h / week)
- Practical exercises with homework (2 h / week)

Objectives: Students...
- By the end of this lecture, the students will be able to:
- explain the fundamental principle of neuromorphic computing
- name and describe at least one approximate computing solution at software, architecture, and circuit level.
- discern whether approximate computing may be used for a specific application
- describe the fundamental advantage of in-memory computing
- list five emerging memory technologies
- solve (evaluate the state of) memristive circuits
- name at least one advantage and disadvantage of stochastic computing
- describe the principle of quantum computing
- name at least one advantage and disadvantage of optical computing

Content:
- Challenges in computing
- ML accelerators and neuromorphic computing
- Approximate computing
- In-memory computing
- Emerging memory technologies
- Memristive Computing
- Stochastic computing
- Quantum computing
- Optical computing

Prerequisites: None	**Recommended Knowledge:** Basic Computer Architecture. Ideally: CCS, HDL, VLSI, ACA, and Embedded Machine Learning

Literature:
- A list of sources that will be provided in the course

Testing: Defined by the lecturer before the beginning of the course

<table>
<tr><td colspan="2">Code:
MScTI_ACF</td><td colspan="2">Course Title:
Architecture and CAD for FPGAs</td></tr>
<tr><td colspan="2">Module Coordinator:
Prof. Dr. Dirk Koch</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week), Practical exercise with homework (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• The overall objective of Architecture and CAD for FPGAs – ACF is to provide a deep understanding of the ecosystem that is required to make user logic working on an FPGA. This includes the FPGA hardware itself but also the tools that are required to map user logic onto an FPGA. In detail the course unit has the objectives:
• list and explain advanced components of FPGA devices
• list and explain architectural details of FPGA devices
• understand the design factors that drive FPGA development
• understand the ecosystem that is required to build FPGA chips as well as the CAD tools to program them
• designing own customized FPGAs, including the logic and special primitives, the routing fabric and the configuration logic
• understand and program tools for logic synthesis, technology mapping and place and route,
• understand test strategies of FPGAs, including factory tests
• understand Hardware verification, including the internals of a simulator</td></tr>
<tr><td colspan="4">Content:
• Hardware programmability: reconfigurable logic and routing structures
• Dedicated blocks for memory and arithmetic (why and how)
• Understanding FPGA application requirements
• Performace tuning: how to improve area, speed and power consumption
• Physical implementations of FPGAs and embedded FPGAs versus other ASICs
• Logic synthesis and technology mapping (translate user logic into FPGA primitives)
• Place & Route (of user circuits)
• Bitstream Assembly, Partial reconfiguration
• Factory test and chip characterization (how do we ensure that our fabricated chip works as expected)
• Simulation: simulation techniques and accuracy versus simulation speed
• Robustness and security in FPGAs
• Outlook emerging technologies: resistive RAM, phase change memory, spintronics for reconfigurable computing</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Hardware design (Verilog or VHDL), Algorithms and data structures, FPGA Basics</td></tr>
<tr><td colspan="4">Literature:
• An updated list of papers will be provided in the classes</td></tr>
<tr><td colspan="4">Testing: Defined by lecturer before beginning of course</td></tr>
</table>

Code: MScTI_EEC	Course Title: Energy Efficient Computing		
Module Coordinator: Prof. Dr. Dirk Koch		**Type:** Lecture with exercise	
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:
- Lecture (2 h / week)
- Practical exercise with homework (2 h / week)

Objectives: Students...
- understand the importance and impact of energy efficiency in computing systems
- learn the sources of energy usage in a computing system
- understand the similarities and differences of computing in datacenters and in embedded systems
- understand and evaluate hardware and software technologies and methods for building energy efficient systems
- develop and implement high performance and energy efficient code

Content:
- Energy-efficiency is the probably most important objective in virtually all computing systems. It not only enables the mobile revolution, it is also key to cramp billions of transistors into a single processor chip, and is consequently a performance driver. Moreover, embedded systems usually have limited power envelopes and global warming requires us to rethink our computing systems.
- The need for energy efficient computing
- Impact of technology and system-level design choices on energy-efficiency
- Design factors impacting cost, performance and energy-efficiency
- The impact of memory on performance and energy-efficiency
- Performance optimization techniques, including frequency voltage scaling and SIMD and multi-core processing
- Specialized hardware and acceleration for performance and more energy-efficient computing
- Existing (e.g., chiplets and 3D integration) and emerging technologies for energy efficient computing
- Performing design-space exploration
- Programming and evaluation of different performance and energy-efficiency tuning techniques

Prerequisites: none	**Recommended Knowledge:** none

Literature:
- to be defined in lecture or on web page

Testing: Defined by lecturer before beginning of course

Code: MScTI_MCC	**Course Title:** Memory-Centric Computing		
Module Coordinator: Prof. Dr. Nima TaheriNejad	**Type:** Lecture with exercise		
Credit Points: 6	**Workload:** 180 h	**Teaching Hours:** 4 / week	**Term:** ST

Module Parts and Teaching Methods:

- Lecture (2 h / week)
- Practical exercises with homework (2 h / week)

Objectives: Students...

By the end of this lecture, the students will be able to:

- Describe the basic ideas and the evolution of memory technologies.
- List at least three architectural designs used for memory centric computing.
- Name the principles and advantages of memory-centric computing paradigms and their implications for energy efficiency and performance.
- Use one or more established memory-centric computing solutions.
- Name at least three emerging memory-centric computing solutions as well as their advantages and disadvantages.
- Match at least two specific memory technologies to their most suitable applications, such as AI, machine learning, or edge computing.
- State three major challenges in memory-centric computing and suggest one potential solution for each challenge.
- Summarize two key advancements shared through case studies.

Content:

- Overview of different memory technologies, including SRAM and DRAM, Flash Memory, PCM, RRAM, MRAM, etc.
- Fundamentals of the Memory-Centric Computing design principles and architectures.
- Energy-efficient and high-performance solutions through memory-centric computing.
- System-level integration of memory technologies and their applications in AI, machine learning, and edge computing.
- Challenges and open research problems in memory-centric computing.
- One or more case studies on advanced memory-centric applications.

Prerequisites: None	**Recommended Knowledge:** Advanced computer architecture and/or Emerging computing paradigms.

Literature:

- A list of sources that will be provided in the course

Testing: Defined by the lecturer before the beginning of the course

<table>
<tr><td colspan="2">Code:
MScTI_BIOSIG</td><td colspan="2">Course Title:
Biosignal Processing and Machine Learning</td></tr>
<tr><td colspan="2">Module Coordinator:
Dr. Mostafa Haghi and Dr. Amin Aminifar</td><td colspan="2">Type:
Lecture with exercise</td></tr>
<tr><td>Credit Points:
6</td><td>Workload:
180 h</td><td>Teaching Hours:
4 / week</td><td>Term:
ST</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Lecture (2 h / week) with seminar/exercises (2 h / week)</td></tr>
<tr><td colspan="4">Objectives: Students...
• understand and describe the most contributing biosignals in biomedical applications
• perform the preprocessing and processing of biosignals such as electrocardiogram
• describe and implement the biosignal processing techniques such as discrete wavelet transform to adjust with the frequency boundary of cardiorespiratory parameters
• apply the biosignal processing techniques in cardiac abnormality detection
• identify and distinguish the well-known biosignal data formats
• explain the fundamental concepts in machine learning
• implement and use deep learning for medical applications
• describe at least one machine learning solution for addressing noise concern in biosignal processing
• apply machine learning in the context of biosignal processing</td></tr>
<tr><td colspan="4">Content:
• Introduction to biosignals and signal processing (general briefing)
• Signal generation and improvement
• Signal visualization
• Basic of R wave and R wave detection in electrocardiogram
• Computer Aided Detection (CAD) – biosignal, Atrial Fibrilation detection
• Data formats for biosignals
• Introduction to the application of machine learning in the biomedical/health domain
• Classical machine learning in the biomedical domain
• Deep learning in the biomedical domain
• Reinforcement learning in the biomedical domain
• How to treat noisy data using machine learning?
• Current and future challenges for machine learning in biomedical applications</td></tr>
<tr><td colspan="2">Prerequisites:</td><td colspan="2">Recommended Knowledge:
Basic Calculus, Algebra, Python Programming</td></tr>
<tr><td colspan="4">Literature:
• Will be suggested in lecture</td></tr>
<tr><td colspan="4">Testing: Defined by lecturer before beginning of course</td></tr>
</table>

<table>
<tr><td colspan="2">Code:
MScTI_SA</td><td colspan="2">Course Title:
Study Project</td></tr>
<tr><td colspan="2">Module Coordinator:
All Groups</td><td colspan="2">Type:
Practice Course</td></tr>
<tr><td>Credit Points:
14</td><td>Workload:
420 h</td><td>Teaching Hours:
n.a.</td><td>Term:
ST/ WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Practical Course</td></tr>
<tr><td colspan="4">Objectives: Students...
• can dig into scientific and technical aspects of a selected topic,
• manage and carry through a small research project,
• write a medium length report.</td></tr>
<tr><td colspan="4">Content:
• Research work on a specific topic.
• Management of work.
• Preparation of a medium length report.</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Knowledge in research field</td></tr>
<tr><td colspan="4">Literature:
• Depending on subject, provided by supervisor</td></tr>
<tr><td colspan="4">Testing: Written Report</td></tr>
</table>

<table>
<tr><td colspan="2">Code:
MScTI_THESIS</td><td colspan="2">Course Title:
Master Thesis</td></tr>
<tr><td colspan="2">Module Coordinator:
All Groups</td><td colspan="2">Type:
Practice Course</td></tr>
<tr><td>Credit Points:
30</td><td>Workload:
900 h</td><td>Teaching Hours:
n.a.</td><td>Term:
ST/ WT</td></tr>
<tr><td colspan="4">Module Parts and Teaching Methods:
• Master Thesis</td></tr>
<tr><td colspan="4">Objectives: Students...
• manage and carry through a large research project,
• write an extended thesis,
• report on own scientific work in an oral presentation.</td></tr>
<tr><td colspan="4">Content:
• Research work on a specific topic.
• Management of work.
• Preparation of a longer written thesis.
• Oral presentation in the colloquium.</td></tr>
<tr><td colspan="2">Prerequisites:
none</td><td colspan="2">Recommended Knowledge:
Knowledge in research field</td></tr>
<tr><td colspan="4">Literature:
• Depending on subject, provided by supervisor</td></tr>
<tr><td colspan="4">Testing: Written Thesis, Colloquium</td></tr>
</table>