Technische Informatik

Pflichtfächer

1. Semester
   - Mathematische Grundlagen
   - Grundlagen der Technischen Informatik
   - Programmiermethodik I
   - Programmiertechnik
   - Grundlagen der Elektrotechnik I
2. Semester
   - Automaten und Formale Sprachen
   - Programmiermethodik II
   - Datenbanken
   - Grundlagen der systemnahen Programmierung
   - Grundlagen der Elektrotechnik II
3. Semester
   - Analysis und lineare Algebra
   - Algorithmen und Datenstrukturen
   - Software Engineering I
   - Betriebssysteme
   - Digitaltechnik
4. Semester
   - Signaltheorie und Regelungstechnik
   - Embedded System Engineering
   - Rechnernetze
   - Computer Engineering
5. Semester
   - Verteilte Systeme
   - Betriebswirtschaft

Modulbezeichnung	**Mathematische Grundlagen**	Kürzel	MG / MGÜ
Lehrveranstaltung(en)	Vorlesung: Mathematische Grundlagen Übung: Mathematische Grundlagen	Semester	1
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Stephan Pareigis	SWS	3+1
Dozenten	Prof. Dr. Stephan Pareigis , Prof. Dr. Reinhard Baran, Prof. Dr. Julia Padberg	Sprache	deutsch
Voraussetzungen	Inhalte des Mathematik-Vorkurs	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können wichtige mathematische Strukturen sicher verwenden • beherrschen die logischen und algebraischen Grundlagen der theoretischen Informatik • können Definitionsprinzipien und Beweistechniken in unterschiedlichen Bereichen und an typischen Beispielen anwenden • können die Methoden der linearen Algebra anwenden		
Inhalte	• Mathematische Grundlagen: Mengen, Relationen, Abbildungen, Funktionen und deren Operatoren, Aussagenlogik, Boolsche Algebra • Mathematische Techniken: Grundlegende Beweisstrategien, Vollständige Induktion • Mathematisch Strukturen: Lösung von linearen Gleichungssystemen. Vektoren, Matrizen, Determinanten • Vertiefung in folgende Richtungen: o Konvergenz und Grenzwerte von Folgen und Reihen o Schaltalgebra o Graphentheorie • Kombinatorik, Diskrete Stochastik		
Lehr- und Lernformen	Vorlesung: Tafel, Präsentation, Vorrechnung von Beispielaufgaben, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium Übung: selbstständiges Lösung von Übungsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreich absolvierte Übung (PVL)		
Literatur	• Peter Hartmann. Mathematik für Informatiker. Vieweg 4. Auflage Februar 2006 Kapitel 1-3, 12-14		

Modulbezeichnung	**Grundlagen der Technischen Informatik**	Kürzel	GT/GTP
Lehrveranstaltung(en)	Vorlesung: Grundlagen der Technischen Informatik Praktikum: Grundlagen der Technischen Informatik	Semester	1
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Franz Korf	SWS	3+1
Dozenten	Prof. Dr. Reinhard Baran, Prof. Dr. Andreas Meisel, Prof. Dr. Franz Korf	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können grundlegenden Begriffe der Informatik benennen • können grundlegende Rechnerarchitekturkonzepte wie zum Beispiel die von-Neumann-Architektur beschreiben • verstehen die Grundkonzepte der Instruction-Set-Architektur eines Prozessors • können einfache Assembler-Programme für einen ausgewählten Prozessors erstellen		
Inhalte	Elementare Begriffe und Definitionen der Informatik wie zum Beispiel: • Informatik, Information, Daten, Algorithmus, Programm, Theoretische Informatik, Praktische Informatik, Technische Informatik, Hochsprachen, • Assembler, Syntax und Semantik von Programmiersprachen • Darstellung von Daten im Computer • Rechnerarchitekturgrundlagen auf Instruction Set Architecture Ebene: • Elemente eines Rechners, von Neumann, Harvard • Grundlegende Konzepte der Instruction Set Architecture einer ausgewählten Mikroprozessorfamilie • Abbildung von Daten- und Kontrollstrukturen prozeduraler Hochsprachen in maschinennahe Implementierungen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• A. S. Tanenbaum, James Goodman: Computerarchitektur • J. L. Hennessy, D. A. Patterson: Computer Architecture • W. Hohl: ARM Assembly Language: Fundamentals and Techniques • eigene Skripte der Dozenten		

Modulbezeichnung	**Programmiermethodik I**	Kürzel	PM1
Lehrveranstaltung(en)	Vorlesung: Programmiermethodik I	Semester	1
Arbeitsaufwand	48 Std. Vorlesung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Schäfers	SWS	4+0
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Philipp Jenke, Prof. Dr. Michael Schäfers, Prof. Dr. Bernd Kahlbrandt, Prof. Dr. Axel Schmolitzky, N.N.	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • sind vertraut mit unterschiedlichen Konzepten und Modellen von Programmiersprachen (Programmierparadigmen) • können Objektorientierte Modellierung von Systemen im Kleinen durchführen • können eine abstrakte Problembeschreibung in einen programmierbaren Algorithmus übertragen		
Inhalte	• Abstraktionsmechanismen - Funktionale Abstraktion - Datenabstraktion (ADT) - Kontrollabstraktion (z.B. Iteratoren, Streams) • Objektorientierung (prozedural und funktional) - Polymorphie: Overloading, Overriding, Dynamische Bindung • Ausgewählte Elemente objektorientierter Bibliotheken, z.B. - Collections - Streams, Channels • Typisierungskonzepte - Dynamische vs. statische Typisierung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentat		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat		
Literatur	• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	**Programmiertechnik**	Kürzel	PT/PTP
Lehrveranstaltung(en)	Vorlesung: Programmiertechnik Praktikum: Programmiertechnik	Semester	1
Arbeitsaufwand	24 Std. Vorlesung, 24 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Schäfers	SWS	2+2
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Philipp Jenke, Prof. Dr. Michael Schäfers, Prof. Dr. Bernd Kahlbrandt, Prof. Dr. Axel Schmolitzky, N.N.	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden beherrschen handwerklichen Programmierfertigkeiten, elementaren Programmiertechniken sowie eine modernen Entwicklungsumgebung mit den zugehörigen Werkzeugen (z.B. Editor, Debugger, Testautomation, Version Control System,...) • können eine technische Realisierung von Systemen im Kleinen durchführen		
Inhalte	• Elementare Programmiertechniken und Syntax einer modernen Programmiersprache: - primitive Datentypen, - Unicode, - Arrays, - Referenztypen, - Sequenz, Selektion, Iteration		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, Beispielaufgaben, freiwillige Übungsaufgaben, Tutorium Praktikum: Bearbeitung der Aufgaben in 2-er Gruppen, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	**Grundlagen der Elektrotechnik I**	Kürzel	GE1
Lehrveranstaltung(en)	Vorlesung: Grundlagen der Elektrotechnik I	Semester	1
Arbeitsaufwand	48 Std. Vorlesung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Wolfgang Fohl	SWS	4+0
Dozenten	Prof. Dr. Wolfgang Fohl, Prof. Dr. Andreas Meisel, Prof. Dr. Reinhard Baran	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können Grundschaltungen aus linearen Bauelementen im Gleichstromkreis berechnen • können transiente Vorgänge mit einem kapazitiven oder induktiven Speicher verstehen und berechnen • sind in der Lage, Netzwerkanalysen für Schaltkreise mit kapazitiven und induktiven Speichern bei Sinus-Wechselstromanregung durchzuführen		
Inhalte	• Elementare Begriffe, wie Einheiten, Konstanten, Ladung, Spannung, Strom, Ohmsches Gesetz, Energie, Leistung, Wirkungsgrad • Einführung in die Analyse von Gleichstromschaltungen: Kirchhoff-Gleichungen, Grund- und Teilerschaltungen mit Widerständen, Ersatzquellen, Superposition, Nichtlinearitäten • Einführung in die Berechnung von Schalt- und Ausgleichsvorgängen in kapazitiven und induktiven Schaltungen • Einführung in die Wechselstromrechnung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, Vorrechnen von Übungsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat		
Literatur	• Wolf-Ewald Büttner: Grundlagen der Elektrotechnik 1, Oldenbourg, 2011 • Manfred Albach: Elektrotechnik 1, Pearson, 2011 • Horst Clausert, Gunther Wiesemann, Volker Hinrichsen: Grundgebiete der Elektrotechnik 1, Oldenbourg, 2008 • eigene Skripte der Dozenten		

Modulbezeichnung	**Automatentheorie und Formale Sprachen**	Kürzel	AF/AFÜ
Lehrveranstaltung(en)	Vorlesung: Automatentheorie und Formale Sprachen Übung: Automatentheorie und Formale Sprachen	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Bettina Buth	SWS	3+1
Dozenten	Prof. Dr. Bettina Buth, Prof Dr. Köhler-Bussmeier, Prof. Dr. Franz Korf, Prof. Dr. Michael Neitzke, Prof. Dr. Julia Padberg	Sprache	deutsch
Voraussetzungen	Modul Mathematische Grundlagen oder äquivalente Kenntnisse Anmerkung: benötigt werden Kenntnisse von formalen Beweisen, Prädikatenlogik, speziell Induktion	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können formale Beweise erläutern und selber durchführen • können formale Modelle in Form von Automaten, Regulären Ausdrücken und Grammatiken verstehen und erstellen • können Zusammenhänge zwischen Automatenmodellen, regulären Sprachen und Grammatiken herstellen und Modelle ineinander überführen • können formale Spezifikationen auf Problemstellungen der realen Welt anwenden		
Inhalte	• Grundlagen der Automatentheorie • Endliche Automaten • Reguläre Ausdrücke und Sprachen • Kontextfreie Grammatiken und Sprachen • Eigenschaften Kontextfreier Sprachen • Vertiefung in einer der folgenden Richtungen: Kellerautomaten, zelluläre Automaten, Zeitautomaten, Modellierung mit formalen Methoden		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben, Gruppenarbeit Übung: selbstständiges Lösen von Aufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreiche Bearbeitung der Übungsaufgaben (PVL)		
Literatur	• J.E. Hopcroft, P. Motwani, J.D. Ullmann: Einführung in die Automatentheorie, Formalen Sprachen und Komplexitätstheorie, Pearson Studium • M.Kreuzer, S. Kühling: Logik für Informatiker, Pearson Studium		

Modulbezeichnung	**Programmiermethodik II**	Kürzel	PM2/PMP2
Lehrveranstaltung(en)	Vorlesung: Programmiermethodik II Praktikum: Programmiermethodik II	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Michael Schäfers	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Philipp Jenke, Prof. Dr. Michael Schäfers, Prof. Dr. Bernd Kahlbrandt, Prof. Dr. Axel Schmolitzky	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • beherrschen fortgeschrittene Fertigkeiten und Programmiertechniken • beherrschen die Objektorientierte Modellierung und technische Realisierung von Systemen, die nur Team-orientiert erarbeitet werden können		
Inhalte	• Ausgewählte Elemente objektorientuerter Bibliotheken, z.B.: - GUI-Frameworks - Generics • Metasprachliche Konzepte - Annotationen, XML, - Reflection • Vertiefungen - Typ- vs. Implementierungshierarchie - elementare Design Pattern - Modellierungen (anhand UML) - nebenläufige bzw. asyncrone Programmierung - Deployment • Correctness - Design by Contract (Assertions, Invarianten, Teststrategien)		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, Beispielaufgaben, freiwillige Übungsaufgaben, evtl. Tutorium Praktikum: Bearbeitung der Aufgaben in 2-er Gruppen, Begutachtung der Lösungen, Gesprächsführung		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Literaturhinweise werden je nach Programmiersprache und aktuellem Stand in der Vorlesung gegeben		

Modulbezeichnung	**Datenbanken**	Kürzel	DB/DBP
Lehrveranstaltung(en)	Vorlesung: Datenbanken Praktikum: Datenbanken	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Thomas Thiel-Clemen	SWS	3+1
Dozenten	Prof. Dr. Wolfgang Gerken, Prof. Dr. Stefan Sarstedt , Prof. Dr. Thomas Thiel-Clemen, Prof. Dr. Olaf Zukunft	Sprache	deutsch
Voraussetzungen	Programmieren I (PR1, PRP1), Mathematische Grundlagen (MG)	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden: • kennen die Einsatzgebiete und Grenzen von Datenbanken • beherrschen des Prozess des Datenbankentwurfs • kennen die theoretischen Grundlagen von Datenbanksystemen • können einfache Datenbankanwendungen entwickeln • beherrschen die relationale Anfragesprache SQL im Rahmen des Standards • verstehen Datenschutzmechanismen und gesellschaftliche Auswirkungen großer Datensammlungen		
Inhalte	• Grundkonzepte relationaler Datenbanksysteme • der logische Entwurf und die Überführung in das technische Design • Implementierung und Befüllung von Datenbanksystemen • Anfragen und Transaktionen • programmiersprachliche Schnittstellen • Alternativen zum relationalen Modell		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Kleingruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• A. Kemper / A. Eickler, Datenbanksysteme – Eine Einführung, Oldenbourg 2011. • R.A. Elmasri / S.B. Navathe, Grundlagen von Datenbanksystemen, Pearson 2009 • Eigene Skripte der Dozenten		

Modulbezeichnung	**Grundlagen der systemnahen Programmierung**	Kürzel	GS/GSP
Lehrveranstaltung(en)	Vorlesung: Grundlagen der systemnahen Programmierung Praktikum: Grundlagen der systemnahen Programmierung	Semester	2
Arbeitsaufwand	24 Std. Vorlesung, 24 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Franz Korf	SWS	2+2
Dozenten	Prof. Dr. Reinhard Baran, Prof. Dr. Andreas Meisel, Prof. Dr. Franz Korf	Sprache	deutsch
Voraussetzungen	Grundlagen der Technischen Informatik	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können prozedurale Programme in C erstellen • verstehen die Zugriffe auf Hardwareschnittstellen • können Anwendungen zum Ansteuern einfacher Sensoren und Aktoren erstellen • verstehen die Schnittstelle zwischen einer Hochsprache und einem Assembler		
Inhalte	• Methoden und Techniken zur prozeduralen - und maschinennahen Programmierung • Weiterführende, für die technische Informatik relevante Konzepte einer hardwarenahen Programmiersprache wie zum Beispiel C • Speicherverwaltung auf Hochsprachen- und Maschinenebene • C Projekte: Verwaltung, Modultechnik, Bibliotheken, Standardbibliotheken • Interfaces zur Verzahnung von Hochsprachen und Assembler • elementare Zeitmessungen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, freiwillige Übungsaufgaben Praktikum: Programmieren in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• A. S. Tanenbaum, James Goodman: Computerarchitektur • Groll, U. Bröckl, M. Dausmann: C als erste Programmiersprache • B.W. Kernighan, D.M. Ritchie: Programmieren in C • W. Hohl: ARM Assembly Language: Fundamentals and Techniques • eigene Skripte der Dozenten		

Modulbezeichnung	**Grundlagen Elektrotechnik II**	Kürzel	GE2/GEP2
Lehrveranstaltung(en)	Vorlesung: Grundlagen Elektrotechnik II Praktikum: Grundlagen der Elektrotechnik II	Semester	2
Arbeitsaufwand	24 Std. Vorlesung, 24 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Wolfgang Fohl	SWS	2+2
Dozenten	Prof. Dr. Reinhard Baran, Prof. Dr. Andreas Meisel, Prof. Dr. Wolfgang Fohl	Sprache	deutsch
Voraussetzungen	Grundlagen der Elektrotechnik I	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • besitzen Grundkenntnisse der Netzwerkanalyse für Schaltkreise mit kapazitiven und induktiven Speichern bei Sinus-Wechselstromanregung, • besitzen Kenntnisse über elektrische Messverfahren und Messgeräte, • können Strom-, Spannungs- und Widerstandsmessungen sowie Messungen nichtelektrischer Größen durchführen, • sind in der Lage, messtechnische Grundschaltungen zu berechnen und aufzubauen, • können Messergebnisse protokollieren und präsentieren, • können Messabweichungen abschätzen und bewerten.		
Inhalte	• Grundlagen der Messtechnik, d.h.: Messabweichungen, Fehlerrechnung, Erstellen von Versuchs-protokollen, grafische Auswertung von Messergebnissen, Berechnen von Messabweichungen. • Einführung in den Einsatz und die Wirkungsweise elektrischer Messgeräte: Amperemeter, Voltmeter, Ohmmeter, Oszilloskop, Analog-Digital-Umsetzung, • Wechselstromschaltungen mit sinusförmiger Anregung • Zeigerdarstellung (komplexe Darstellung) sinusförmiger Größen, Impedanz von Induktivitäten, Admittanz von Kapazitäten, Leistung • Grundlegende Messschaltungen, Gleichstrom-Brückenschaltung, AC- und DC-Betrieb, strom- und spannungsrichtige Messung, systematische Messabweichungen, • Messverfahren für nichtelektrische Größen, Messung transienter Vorgänge mit dem Speicheroszilloskop, • Grundschaltungen mit Operationsverstärkern		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, freiwillige Übungsaufgaben Praktikum: Versuche zur elektrischen Messtechnik in Gruppen zu zwei bis drei Personen.		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Wolf-Ewald Büttner. Grundlagen der Elektrotechnik 2, Oldenbourg, 2011 • Manfred Albach, Elektrotechnik 2, Pearson, 2011 • Horst Clausert, Gunther Wiesemann, Volker Hinrichsen: Grundgebiete der Elektrotechnik 2, Oldenbourg, 2008 • eigene Skripte der Dozenten		

Modulbezeichnung	**Analysis und lineare Algebra**	Kürzel	AA / AAÜ
Lehrveranstaltung(en)	Vorlesung: Analysis und lineare Algebra Übung: Analysis und lineare Algebra	Semester	2
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Stephan Pareigis	SWS	3+1
Dozenten	Prof. Dr. Stephan Pareigis , Prof. Dr. Reinhard Baran, Lehrbeauftragte	Sprache	deutsch
Voraussetzungen	Mathematische Grundlagen	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • beherrschen die analytischen Rechentechnik mit elementaren Funktionen einer Variablen • können mit Parametern, Transformationen und graphischen Darstellungen umgehen		
Inhalte	• Kombinatorik, Teilbarkeit, Restklassen • Gruppen, Vektorräume, Homomorphismen • Skalarprodukt, Orthogonalität, Norm und Metrik • Differenzialrechnung, Integralrechnung, Fourierreihen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, Vorrechnen von Beispielaufgaben, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium Übung: selbstständiges Lösung von Übungsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreiche Bearbeitung aller Aufgaben (PVL)		
Literatur	• Peter Hartmann. Mathematik für Informatiker. Vieweg 4. Auflage Februar 2006, Kapitel 4-8, 10, 15-17		

Modulbezeichnung	**Algorithmen und Datenstrukturen**	Kürzel	AD / ADP
Lehrveranstaltung(en)	Vorlesung: Algorithmen und Datenstrukturen Praktikum: Algorithmen und Datenstrukturen	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Stephan Pareigis	SWS	3+1
Dozenten	Prof. Dr. Stephan Pareigis , Prof. Dr. Bernd Kahlbrandt	Sprache	deutsch
Voraussetzungen	Mathematische Grundlagen, Programmieren I + II	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • haben Kenntnisse zur Bewertung und selbstständigen Entwicklung von Algorithmen und beherrschen die dazu erforderlichen Datenstrukturen.		
Inhalte	• Abstrakte Datentypen Abstrakte Datentypen, Signatur, Vor- und Nachbedingungen, Lineare Listen, Stack, Queue • Algorithmen und Komplexität Komplexität, Aufwandsfunktion, Asymptotischer Aufwand, Landau-Notation, Darstellung in logarithmischen Skalen, rekursive Verfahren • Sortierverfahren Nicht-rekursive Sortierverfahren und deren Komplexität, Rekursive Sortierverfahren (Quicksort und Mergesort) und deren Komplexität • Bäume und Graphen Implementationsmöglichkeiten, Binäre Suchbäume und Komplexität, Graphen und kürzeste Wege (Dijkstra) • Hashfunktionen offene Adressierung und separate chaining, Kollisionsvermeidungsstrategien		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafel, Präsentation, Vorrechnen von Beispielaufgaben, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium Praktikum: selbstständiges Lösung von Praktikumsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• T.Ottman und P.Widmayer Algorithmen und Datenstrukturen, Spektrum Verlag, 4.Auflage, 2002		

Modulbezeichnung	**Software Engineering I**	Kürzel	SE1/SEP1
Lehrveranstaltung(en)	Vorlesung: Software Engineering I Praktikum: Praktikum Software Engineering I	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Bettina Buth	SWS	3+1
Dozenten	Prof. Dr. Bettina Buth , Prof. Dr. Zhen Ru Dai, Prof. Dr. Thomas Lehmann, Prof. Dr. Stephan Pareigis, Prof. Dr. Philipp Jenke	Sprache	deutsch
Voraussetzungen	Programmieren I und II	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können die Zusammenhänge zwischen Programmierung und Software Engineering erläutern • können Fachbegriffe des Bereichs Software Engineering erläutern • können vorgegebene Modelle (insbesondere Prozessmodelle und Modell der UML Notationsfamilie) im Rahmen einer Softwareentwicklung interpretieren und analysieren, anwenden und umsetzen. • können die Besonderheiten des Software Engineering bei technischen Anwendungen benennen und erläutern • können programmiernahe Techniken und Werkzeuge im Rahmen des Software Engineering einsetzen, speziell für die Anforderungsanalyse, die Modellierung und den Test. • können statische Analysen ebenso wie grundlegende dynamische Testtechniken anwenden • können den Wert von Softwareentwicklungswerkzeugen einschätzen • können sich eigenständig neue Techniken und Methoden und deren Anwendung an konkreten Beispielen aneignen • können fachliche Zusammenhänge schriftlich ausdrücken, speziell Dokumentation von Anforderungen, Architekturen, Tests erstellen und bewerten • können englische Dokumentation interpretieren		
Inhalte	• Grundkonzepte und Ziele des Software Engineering, insbesondere Lebenszyklusmodelle, Anforderungserhebung- und Analyse, Architekturen, Qualitätssicherung, • Modellierungstechniken, speziell auch im Hinblick auf technische Systeme, • Spezielle Entwicklungstechniken, wie z. B. Refactoring, Code Analyse, Continuous Integration, DSLs, Code Generierung, • Requirements Engineering, speziell Anforderungserhebung und -analyse • Prozessmodelle, speziell traditionelle Lebenzyklusmodelle im Vergleich zu agilen Entwicklungsansätzen • Im Praktikum: Arbeiten mit aktuellen Software-Entwicklungsumgebungen speziell zu Modellierung, Test, Refactoring, Requirements Engineering – auch: Beherrschen von Werkzeugketten (Konfiguration, Einsatz)		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in 2er-4er Teams, Diskussionsforen mit Betreuer		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Sommerville: Software Engineering, Pearson Education • Spillner/Linz: Basiswissen Softwaretest, dpunkt Verlag • Hammerschall, Beneken: Software Requirements		

Modulbezeichnung	**Betriebssysteme**	Kürzel	BS/BSP
Lehrveranstaltung(en)	Vorlesung: Betriebssysteme Praktikum: Betriebssysteme	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Franz Korf	SWS	3+1
Dozenten	Prof. Dr. Bettina Buth, Prof. Dr. Wolfgang Fohl, Prof. Dr. Martin Hübner, Prof. Dr. Franz Korf	Sprache	deutsch
Voraussetzungen	Programmieren, Grundkurs Technische Informatik, Grundlagen der systemnahen Programmierung	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen die Architektur, die Konzepte und die Funktionsweise moderner Betriebssysteme sowie des Zusammenspiels von Hard- und Software, • verstehen die Konzepte zur Implementierung systemnaher Software • können das Verhalten von Computersystemen analysieren und beschreiben • können Grundkonzepte der nebenläufigen Programmierung anwenden		
Inhalte	• Architekturen und Betriebsarten • Prozess- und Thread-Konzept, Scheduling • Synchronisation, Interprozesskommunikation, Deadlocks • Hauptspeicherverwaltung, Virtueller Speicher • Verwaltung externer Geräte • Dateisysteme • Schutzmechanismen, Sicherheitsaspekte • Exemplarische Betrachtung aktueller Betriebssysteme		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Praktikum: Programmieren in 2-er Gruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Andrew S. Tanenbaum, Modern Operating Systems, Pearson Studium Verlag • Abraham Silberschatz, Peter Galvin, Greg Gagne: Operating System Concepts with Java, John Wiley & Sons • Eduard Glatz: Betriebssysteme: Grundlagen, Konzepte, Systemprogrammierung, dpunkt Verlag • Williams Stallings. Operating Systems • Eigene Skripte der Dozenten		

Modulbezeichnung	**Digitaltechnik**	Kürzel	DT/DTP
Lehrveranstaltung(en)	Vorlesung: Digitaltechnik Praktikum: Digitaltechnik	Semester	3
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Bernd Schwarz	SWS	3+1
Dozenten	Prof. Dr. Bernd Schwarz, Prof. Dr. Michael Schäfers	Sprache	deutsch
Voraussetzungen	Grundlagen der Technischen Informatik	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können kombinatorische Schaltungen analysieren und mit Minimierungsverfahren entwerfen • verstehen die Funktion von Latches und Flipflops • können einfache Schaltwerke entwerfen • besitzen die Fähigkeit zur Modellierung von Schaltnetzen und Schaltwerken mit einer Hardware-Beschreibungssprache (HDL) • können digitale Schaltungen mit CPLD-Hardware implementieren		
Inhalte	• Entwurf kombinatorischer Schaltungen mit Wahrheitstabellen • Logikbeschreibungen mit der disjunktiven Normalform • Schaltnetzentwurf auf Basis von Minimierungstechniken • Übertragungseigenschaften und Signalverzögerungen von Logikgattern • Ausgangstreiber: Open Drain, Tri-State • HDL-Modellierung von Schaltnetzen • Beschreibung sequentiell arbeitender Digitalschaltungen mit • Zustandsdiagrammen und Zustandsfolgetabellen • Mealy- und Moore-Zustandsautomaten • Gesteuerte Zähler und Schieberegister • HDL-Modellierung von Schaltwerken im RTL-Stil • Schaltnetz- u. Schaltwerkimplementierung mit CPLDs • HDL-basierte Testsignalgeneratoren (Testbenches) für synthesefähige RTL-Modelle		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium Praktikum: Aufgabenbearbeitung in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Reichardt: Lehrbuch Digitaltechnik • Woitowitz: Digitaltechnik • Wakerly: Digital Design, Principles and Practices		

Modulbezeichnung	**Signaltheorie und Regelungstechnik**	Kürzel	SR / SRÜ
Lehrveranstaltung(en)	Vorlesung: Signaltheorie und Regelungstechnik Übung: Signaltheorie und Regelungstechnik	Semester	4
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Andreas Meisel, Prof. Dr. Stephan Pareigis	SWS	3+1
Dozenten	Prof. Dr. Stephan Pareigis, Prof. Dr. Wolfgang Fohl Prof. Dr. Andreas Meisel, Prof. Dr. Bernd Schwarz	Sprache	deutsch
Voraussetzungen	Analysis und lineare Algebra	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können auf der Basis von Elementarsignalen und einfachen Grundoperationen (Gewichtung, Verschiebung, Dehnung) periodische und abgetastete Funktionen beschreiben. • können mit Hilfe der Fouriertransformation (von Elementarsignalen) und den Theoremen der FT die Frequenzeigenschaften von Impulsen, periodischen und abgetasteten Signalen angeben. • können den Einfluss von Frequenzbegrenzungen (Filter: TP, HP, BP) und Zeitbegrenzungen (Fensterfunktionen) auf die Zeit- und Frequenzeigenschaften der Signale beschreiben. • können Abtastsignale mit Hilfe der z-Transformation beschreiben und sowohl die Differenzengleichungen sowie die Übertragungsfunktion digitaler Filter angeben. • können mit Hilfe von Werkzeugen digitale Filter (FIR, IIR) entwerfen und diese auf Zielsystemen (z.B. eingebetteten Systemen) unter Benutzung von Hardwaretimern und Interrupts realisieren. • können Verfahren zur Bestimmung von statistischen Signalkenngrößen (Erwartungswert, Varianz, Momente) und Ähnlichkeitsmaßen (Kreuzkorrelation, Autokorrelation) realisieren und die Ergebnisse interpretieren. • können ausgewählte Merkmale (zerocrossings, Momente, usw.) aus verschiedenen Anwendungsbereichen (Audioprocessing, medizinische Signalverarbeitung, usw.) aus Signalen extrahieren.		
Inhalte	• Elementarsignale, Grundoperationen (Verschieben, Dehnen, Abtastung, ..), Faltungsalgebra • Fouriertransformation, Faltungssatz, Diracstoß und Diracstoßfolgen • Abtastung, Periodisierung, ideale Filter, diskrete FT • Abtastsysteme, z-Transformation, Differenzengleichungen, Übertragungsfunktion, FIR- und IIR-Filter • Diskrete Fouriertransformation, Bandbegrenzung (Filterung), Zeitbegrenzung (Fensterung) • Entwurf und Realisierung digitaler Filter, schmalbandige Filter, Görtzel-Algorithmus • Realisierung von Abtastsystemen (Filter, Audioeffekte, …) mit Hardwaretimern und Interrupts unter Nutzung einer hardwarenahen Programmiersprache (z.B. C). • Grundbegriffe der Statisitik: Wahrscheinlichkeitsbegriff, Zufallsvariablen, Verteilungs- und Verteilungsdichtefunktion, stat. Unabhängigkeit, bedingte Wahrscheinlichkeit, Satz von Bayes • Signale als Zufallsprozesse: Erwartungswert, Varianz, Schätzung stat. Signalkenngrößen • Ausgewählte Signalmerkmerkmale aus verschiedenen Anwendungsbereichen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, Matlab Beispiele, freiwillige Übungsaufgaben Übung: Bearbeitung der Übungsaufgaben in Zweiergruppen mit Abnahmegespräch		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführte Übung Übung: erfolgreiche Bearbeitung der Übungsaufgaben (PVL)		
Literatur	• Hoffmann und Quint : Signalverarbeitung mit MATLAB und Simulink, Oldenbourg Verlag • D.Ch. von Grüningen: Digitale Signalverarbeitung, Fachbuchverlag Leipzig • eigene Skripte der Dozenten		

Modulbezeichnung	**Embedded System Engineering**	Kürzel	ESE/ESEP
Lehrveranstaltung(en)	Vorlesungen: Software Engineering II, Embedded Programming, System- und Echtzeitprogrammierung Praktikum: Praktikum Embedded System Engineering	Semester	4
Arbeitsaufwand	72 Std. Vorlesung, 24 Std. Praktikum, 204 Std. Eigenarbeit/Selbststudium	CP	10
Modulverantwortliche(r)	Prof. Dr. Franz Korf, Prof. Dr. Thomas Lehmann, Prof. Dr. Stephan Pareigis	SWS	6+2
Dozenten	Prof. Dr. Bettina Buth , Prof. Dr. Zhen Ru Dai, Prof. Dr. Wolfgang Fohl, Prof. Dr. Franz Korf, Prof. Dr. Thomas Lehmann, Prof. Dr. Stephan Pareigis, Prof. Dr. Bernd Schwarz, Prof. Dr. Philipp Jenke	Sprache	deutsch
Voraussetzungen	Programmieren, Betriebssysteme, Software Engineering I, Grundlagen der Systemprogrammierung	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • können konkrete Projekte unter Anwendung von Projektmanagementmethoden planen und durchführen, insbesondere unter Berücksichtigung unterschiedlicher Entwicklungsprozesse • können spezielle Aspekte eingebetteter Systeme, wie z. B. Safety, Robustness oder Fault Toleranz, beim Entwurf und Entwicklung berücksichtigen • können technische Systeme auf der Systemebene modellieren und eingebettete Software als Teilsystem modellbasiert entwerfen, realisieren und testen • können verteilte nebenläufige Echtzeitsysteme auf Basis objektorientierter Architekturen unter Verwendung spezialisierter implementierungsnaher Pattern entwerfen und implementieren • können Echtzeitsysteme entsprechen ihres Einsatzes und Anforderungen klassifizieren • können ausgewählter Aspekte eingebetteter Echtzeitsysteme im Zusammenhang mit einem Echtzeitbetriebssystems (z. B. Zeitmanagement, Interrupt Management, Kommunikation, I/O) konzipieren und umsetzen		
Inhalte	• Vorlesung Embedded Programming (EP) - C++ Sprachprinzipien und Eignung für Echtzeitprogrammierung - Kapselung von Systemaufrufen, Plattformunabhängige Programmierung - Reaktor Pattern, Speicherverwaltung, Fabrikpattern, Functor Pattern - Kommunikationsprinzipien und Message Passing - Generische Programmierung mit Templates - Code Instrumentalisierung, Logging, Debuggen von Echtzeitsystemen • Vorlesung Software Engineering 2 (SE2) - Einführung in das Projektmanagement und Projektplanung - Grundlagen des Konfigurations- und Versionsmanagement - Einführung Prozessverbesserung - Einführung in das Qualitätsmanagement - Spezielle Aspekte des Software Engineering bei eingebetteten Systemen (wie Safety, Fault Tolerance, usw.) - Spezialisierungen von Techniken der Modellierens und des Testens auf eingebettete Echtzeitsysteme • Vorlesung System- und Echtzeitprogrammierung (SY) - Grundlagen, Klassifizierung, Einsatz, Anforderungen von Eingebetteten Systemen. - Vertiefung ausgewählter Aspekte eingebetteter Echtzeitsysteme unter Einsatz eines Echtzeitbetriebssystems (z. B.: Zeitmanagement, Interrupt Management, Kommunikation, I/O). - Scheduling Techniken für Echtzeitanwendungen - Umsetzung der Aspekte anhand eines ausgewählten Echtzeitbetriebssystems (z. B. QNX) ---- Fortsetzung auf nächster Seite ----		

Modulbezeichnung	**Embedded System Engineering**	Kürzel	ESE/ESEP
	• Praktikum - Projektorientierte Software-Entwicklung für eine ES auf Basis eines ausgewählten Echtzeitbetriebssystems (z. B. QNX). Das Praktikum erfordert im besonderen Maße auch Techniken und Fähigkeiten der vorausgesetzten Module. - Teamorientierte Entwicklung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht mit verschiedenen Medien Praktikum: Projektorientiertes Entwicklung eines System in kleinen Teams (4-6 Personen)		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Sommerville: Software Engineering. 9. Auflage, 2012, Pearson Education • Leveson N.G.: Safeware – System Safety and Computers. Addison-Wesley Professional, 1995 • (GOF) Gamma, Erich et al.: Design Patterns. Addison Wesley, 1995 • Douglas Schmidt, Stephen Huston: C++ Network Programming. Volume 1, Addison Wesley, 2002 • (POSA 2) Schmidt, Stal, Rohnert, Buschmann: Pattern-Oriented Software Architecture. Wiley, 1999 • Bruce Powell Douglas: Real-Time Design Patterns. Addison-Wesley, 2003 • Andrei Alexandrescu: Modernes C++ Design. MITP, 2003 • Kalev, Danny: The ANSI/ISO C++ Profesional Programmer's Handbook.Que Corperation. 1999 • Meyers, Scott: Effective C++. 2nd Edition. Addison Wesley, 1998 • Meyers, Scott: More Effective C++. Addison Wesley, 1998 • A. Burns und A. Wellings: Real-time systems and programming languages : Ada 95, real-time Java and real-time POSIX. 3. ed, Pearson Addison-Wesley, 2003. • G. C. Buttazzo: Hard real-time computing systems: predictable scheduling algorithms and applications. Nr. 23 in Real-time systems series. 2nd, Springer, 2005. • H. Kopetz: Real-time systems : design principles for distributed embedded applications. 8. ed, Kluwer Acad., 2004. • R. Krten: Getting started with QNX Neutrino : a guide for realtime programmers. QNX Software Systems, 2009. • W. Stallings: Operating systems : internals and design principles. 7. ed., Pearson, 2012.		

Modulbezeichnung	**Rechnernetze**	Kürzel	RN/RNP
Lehrveranstaltung(en)	Vorlesung: Rechnernetze Praktikum: Rechnernetze	Semester	4
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof . Dr. Thomas Schmidt	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Thomas Schmidt, Prof. Dr. Martin Becke	Sprache	deutsch
Voraussetzungen	Programmieren, Betriebssysteme	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen die Konzepte und die Funktionsweisen von Rechnernetzen • können auf der Socket-Schnittstelle basierende Client- / Server-Anwendungen erstellen • können Methoden und Werkzeuge für die Konfiguration und Administration von Rechnernetzen anwenden • können die Leistungsdaten von Rechnernetzen bewerten		
Inhalte	• Grundlagen der Datenübertragung • Protokolle der Sicherungsschicht • Protokolle und Dienste der Netzwerk- und Transportschicht, insbesondere die TCP/IP-Protokollsuite • Einführung in wichtige Anwendungsschichtprotokolle • Sicherheit in Netzwerken • Einführung in Netzwerkmanagement • Socket-Programmierung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Andrew S. Tanenbaum: Computer Networks • Larry L. Peterson, Bruce S. Davie: Computer Networks – A Systems Approach • James F. Kurose, Keith W. Ross: Computer Networking: A Top-Down Approach Featuring the Internet • eigene Skripte der Dozenten		

Modulbezeichnung	**Computer Engineering**	Kürzel	CE/CEP
Lehrveranstaltung(en)	Vorlesung: Computer Engineering Praktikum: Computer Engineering	Semester	4
Arbeitsaufwand	48 Std. Vorlesung, 24 Std. Praktikum, 168 Std. Eigenarbeit/Selbststudium	CP	8
Modulverantwortliche(r)	Prof. Dr. Bernd Schwarz	SWS	4+2
Dozenten	Prof. Dr. Bernd Schwarz, Prof. Dr. Michael Schäfers	Sprache	deutsch
Voraussetzungen	Grundkurs der Technischen Informatik, Grundlagen systemnahes Programmieren, Grundlagen der Elektotechnik, Digitaltechnik	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • haben vertiefte Kenntnisse der Architektur und Funktionsweise von Mikroprozessoren und eingebetteten Systemen • können Komponenten von Mikrocontrollersystemen entwerfen und implementieren • können eingebettete Systeme in Betrieb nehmen • können eine RTL-Modellierung von Prozessorelementen mit einer HDL vom Entwurf bis zur Implementierung mit FPGAs durchführen		
Inhalte	• Entwurf von digitalen Systemen: Modellierung von ausgewählten Prozessorelementen und Peripheriemodulen • Architektur von Mikroprozessoren • Zeit- und ereignisgesteuerte digitale und analoge Ein- und Ausgabe • Programmierung von eingebetteten Systemen: • Hardwarekomponenten, Bootkonzepte, einfache Anwendungen • Aspekte von ausfallsicheren und zuverlässigen Systemen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, Applets zur Veranschaulichung, freiwillige Übungsaufgaben, evtl. Tutorium Praktikum: Aufgabenbearbeitung in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• Patterson/Hennessy: Computer Organization and Design. • Bähring: Anwendungsorientierte Mikroprozessoren: Mikrocontroller und Digitale Signalprozessoren. • Wolf: FPGA-based System Design. • Sloss: ARM System Developers's Guide.		

Modulbezeichnung	**Verteilte Systeme**	Kürzel	VS/VSP
Lehrveranstaltung(en)	Vorlesung: Verteilte Systeme Praktikum: Verteilte Systeme	Semester	5
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Praktikum, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Thomas Schmidt	SWS	3+1
Dozenten	Prof. Dr. Thomas Schmidt, Prof. Dr. Martin Becke, NN.	Sprache	deutsch
Voraussetzungen	Programmieren, Betriebssysteme, Rechnernetze	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • kennen und verstehen die Verteilten Systemen zugrunde liegenden Probleme sowie die einschlägigen Leistungskategorien zu ihrer Lösung • beherrschen und verstehen einschlägige verteilte Programmiermodelle, können eine exemplarische Auswahl praktisch anwenden und können den zugehörigen Lösungsraum technisch beurteilen • beherrschen und verstehen einschlägige Algorithmen zur Realisierung verteilter Anwendungsszenarien und können diese auf reale Probleme übertragen • können eine System-Infrastruktur eines VS entwerfen und realisieren • können eine Middleware eines VS entwerfen und realisieren • können ein Konzept für replizierte Daten entwerfen und realisieren		
Inhalte	• Eine Einführung im Sinne einer Beschreibung der charakteristischen Eigenschaften verteilter Systeme • Interprozesskommunikation und einschlägige Programmiermodelle • Namensdienste und exemplarische Anwendungen • Zeit , Koordination und Übereinstimmung • Wahlen, Wechselseitiger Ausschluss und Verteilte Transaktion • Verteilte Dateisysteme und Replikation • Ausgewählte Anwendungen verteilter replizierender Systeme • Sicherheit in verteilten Systemen		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Overhead-/Rechnerpräsentationen, freiwillige Übungsaufgaben Praktikum: Aufgabenbearbeitung in Zweiergruppen		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich durchgeführtes Praktikum Praktikum: erfolgreiche Bearbeitung der Praktikumsaufgaben (PVL)		
Literatur	• G. Coulouris, J. Dollimore, T. Kindberg: Verteilte Systeme, Pearson Studium • Andrew S. Tanenbaum, Maarten van Stehen: Verteilte Systeme - Grundlagen und Paradigmen, Pearson Studium • eigene Skripte der Dozenten		

Modulbezeichnung	**Betriebswirtschaft**	Kürzel	BW/BWÜ
Lehrveranstaltung(en)	Vorlesung: Betriebswirtschaft Übung: Betriebswirtschaft	Semester	5
Arbeitsaufwand	36 Std. Vorlesung, 12 Std. Übung, 132 Std. Eigenarbeit/Selbststudium	CP	6
Modulverantwortliche(r)	Prof. Dr. Martin Hübner	SWS	3+1
Dozenten	Prof. Dr. Martin Hübner, Prof. Dr. Gerken	Sprache	deutsch
Voraussetzungen	keine	Häufigkeit	semesterweise
Lernziele und Kompetenzen	Die Studierenden • verstehen rechtliche, finanzielle und organisatorische Strukturen von Unternehmen, • verstehen die Bedeutung von wirtschaftlichen Vorgehensweisen und können entsprechende Controlling-Instrumente anwenden • können Kostenberechnungen selbstständig durchführen, • können Investitionsentscheidungen nach betriebswirtschaftlichen Kriterien treffen.		
Inhalte	• Das Unternehmen als System • Rechtsformen und Aufbauorganisation • Ablauforganisation und Methoden zu ihrer Beschreibung • Grundlagen der Finanzbuchhaltung (Buchführung und Jahresabschluss) • Kosten- und Leistungsrechnung • Finanzierung und Investitionsrechnung		
Lehr- und Lernformen	Vorlesung: Seminaristischer Unterricht, Tafelarbeit, Multimedia-Präsentationen, freiwillige Übungsaufgaben Übung: selbstständiges Lösung von Übungsaufgaben		
Studien- und Prüfungsleistungen	Vorlesung: nach Festlegung als benotete Klausur, benotete mündliche Prüfung oder benotetes Referat Prüfungsvorleistung (PVL): erfolgreich absolvierte Übung Übung: erfolgreiche Bearbeitung aller Aufgaben (PVL)		
Literatur	• G. Wöhe: Einführung in die allgemeine BWL, Verlag Franz Vahlen • A. J. Schwab: Managementwissen für Ingenieure, Springer-Verlag • Dietmar Vahs, Jan Schäfer-Kunz: Einführung in die BWL, Schäffer-Poeschel Verlag • Siegfried Schmolke, Manfred Deitermann: Industrielles Rechnungswesen IKR, Winklers Verlag • Eigene Skripte der Dozenten		