Technology
Arts Sciences
TH Köln

# Lehrveranstaltung
# DR - Digitalrechner

Version: 7 | Letzte Änderung: 19.09.2019 11:40 | Entwurf: 0 | Status: vom verantwortlichen Dozent freigegeben

## Allgemeine Informationen

| | |
|---|---|
| **Langname** | Digitalrechner |
| **Anerkennende LModule** | DR_BaTIN |
| **Verantwortlich** | Prof. Dr. Lothar Thieling<br>Professor Fakultät IME |
| **Niveau** | Bachelor |
| **Semester im Jahr** | Wintersemester |
| **Dauer** | Semester |
| **Stunden im Selbststudium** | 60 |
| **ECTS** | 5 |
| **Dozenten** | NF Hartung |
| **Voraussetzungen** | keine |
| **Unterrichtssprache** | deutsch |
| **separate Abschlussprüfung** | Ja |

### Abschlussprüfung

#### Details

Die Studierenden sollen in einer schriftlichen Klausur folgende Kompetenzen nachweisen: 1.) Sicherer Umgang mit grundlegenden Begrifflichkeiten, Mechanismes und Konnzepten. 2.) Analyse gegebener digtaler Schaltungen. 3.) Entwurf digitaler Systeme (Schaltnetze, Zähler, Automaten) in VHDL auf Basis von textuellen Problembeschreibungen (Textaufgaben). 4.) Umsetzung von Hochsprachenkontrukten in Assembler oder vice versa.

#### Mindeststandard

Mindestens 50% der möglichen Gesamtpunktzahl.

#### Prüfungstyp

Die Studierenden sollen in einer schriftlichen Klausur folgende Kompetenzen nachweisen: 1.) Sicherer Umgang mit grundlegenden Begrifflichkeiten, Mechanismes und Konnzepten. 2.) Analyse gegebener digtaler Schaltungen. 3.) Entwurf digitaler Systeme (Schaltnetze, Zähler, Automaten) in VHDL auf Basis von textuellen Problembeschreibungen (Textaufgaben). 4.) Umsetzung von Hochsprachenkontrukten in Assembler oder vice versa.

## Vorlesung / Übungen

### Lernziele

#### Kenntnisse

Boolesche Algebra
Grundfunktion
Axiome und Gesetze
Disjunktive Normalform, Minterme
Konjunktive Normalform, Maxterme
Systematische Vereinfachung

---

Schaltnetze
Logische Gatter, Tri-State-Buffer
Beschreibungsformen
boolsche Gleichung
Tabelle
KV-Diagramm
Schaltplan
Umformungen zwischen den Beschreibungsformen
Analyse
Synthese (inkl. Transfer von "Textaufgaben")
Don't-Care-Bedingungen
Typische Schaltnetze
1-aus-n-Decoder
Multiplexer
Demultiplexer
Addierer

---

Zahlendarstellung in Rechnersystememem
Dual-Code, Hexadezimal-Code, Basiswechsel
Zweierkomplement
Fixkommadarstellung
Gleitkommadarstellung
ASCII-Code

---

Speicherlemente
Flip-Flops
RS
D
Taktzustabdssteuerung
Taktflakensteuerung
Register

parallele Schreibleseregister
Schieberegister
Parallel-Seriell-Wandlung
Seriell-Parallel-Wandlung
praxisrelevante Spezifikationen
setup time
hold time
minimum puls width

---

synchrone Zähler
Grundidee
Aufbau unter Verwendung von D-Flip-Flops
Analyse
Synthese
Spezifikation in VHDL
siehe VHDL

---

synchrone Schaltwerke (Automaten)
Beschreibung von Automaten mittels Zustandsüberführungsdiagrammen nach Moore
Entwurf von Automaten als Problemlösung
Implementierung mittels VHDL

---

Zustandsüberführungsdiagramme
Modellierung nach Moore
zu beachtende Eigenschaften (Determinismus, Vollständigkeit)

---

VHDL
VHDL für Schaltnetze
Aufbau eines VHDLK-Programms (entity, port, architecture, signale, in, out)
Signale (Typ stdlogic: 1, 0, Tri-State, Don't-Care)
Einfache Signalzuweisung f.d. direkte Umsetzung Boolescher Funktionen
Bedingte Signalzuweisung f.d. direkte Umsetzung von Tabellen
Signal-Vektor
Datentyp Integer sowie Umwandlung von/nach Signal-Vektoren
Nutzung von VHDL im Entwurfswerkzeug (Design Entry VHDL)
VHDL für Zähler und Automaten
Prozesse und sequentielle Anweisungen (process, variable, if, case, event, type)
Realisierung regulärer Zähler in VHDL
Realisierung von Zustandsüberführungsdiagrammen in VHDL
Hierarchisches VHDL-Desin
Packages, Components, Portmaps, Generics

---

Programmierbare Bausteine
Aufbau
Grundidee
Technologie
Zellbegriff
CPLD versus FPGA
Entwurfswerkzeug
Spezifiation mittels Schaltplan
Erstellen eines Schaltplans (Design Entry Schematic)
elementare Bibliothek (Gatter, IN, OUT, Buffer, MUX, Decoder, Flip-Flops)
Sammelleitungen (Busse)
Hierarchische Schaltpläne

Spezifikation mittels VHDL
Details hierzu siehe VHDL
Synthese
Simulation

---

Grundaufbau und Arbeitsweise eines einfachen Rechnersystems
Aufbau eines Von-Neumann-Rechners (Register, Rechenwerk, Steuerwerk, Speicher, Busstruktur)
Funktionsweise, d.h. Ablauf einer Programmabarbeitung auf Basis von Registertransfers
Konkretisierung der Arbeits- und Funktionsweise anhand eines minimalen simulierten Von-Neumann-Rechners
Programmierung des Minimalrechners in Assembler (einfache Schleifen, Adressierungsarten absolut, direkt, indirekt)

---

Aufbau und Funktionsweise einer dedizierten CPU (z.B. IA32E-Architectur)
Architekturübersicht
Funktionsweise, d.h. Ablauf einer Programmabarbeitung auf Basis von Registertransfers
elementare Grundlagen zur deren Programmierung in Assembler

---

### Fertigkeiten

Systemverhalten aus spezifizierenden Texten herleiten
technische Texte erfassen
implizite Angaben erkennen und verstehen
fehlende Angaben
erkennen
ableiten
erfragen

---

Erarbeitung von Problemlösungen, die sich mit Schaltnetzen realisieren lassen
Austellen von Wahrheitstabelle
Spezifikation boole'scher Gleichungen
Minimierung boole'scher Gleichungen
Implementierung mittels Schaltplan
Implementierung mittels VHDL

---

Interpretation und Umwandlung von Codes
Codieren
Decodieren
Fehlererkennung und Fehlerkorrektur

---

Erarbeitung von Problemlösungen, die sich mit synchronen Zählern realisieren lassen
Aufstellen des Zustandüberführungsdiagramms
Implementierung mittels Schaltnetz und Flip-Flops
Implementierung in VHDL

---

Erarbeitung von Problemlösungen, die sich mit synchronen Automaten realisieren lassen
Aufstellen des Zustandüberführungsdiagramms
Auswahl der geeigneten Spezifikationsform (Moore versus Mealy)
Bewertung der Spezifikation
Vollständigkeit
Determiniertheit
Lebendigkeit
Implementierung mittels Schaltnetz und Flip-Flops
Implementierung mittels VHDL

Erläutern der Funktionsweise eines Von-Neumann-Rechners
Teilimplementierungen der Rechnerkomponenten
Implementierung des Rechenwerks als Schaltnetz
Implementierung der Register auf Basis von Flip-Flops
Implementierung des Speichers auf Basis von 1-aus-n-Decodern und Registern
Implementierung des Steuerwerks als Automat
Implementierung einfacher Hochsprachenkonstrukten in Assembler
Variable und Konstante
Felder
Kontrollstrukturen (if, while, do while, switch case, for)

### Aufwand Präsenzlehre

| Typ | Präsenzzeit (h/Wo.) |
|---|---|
| Vorlesung | 2 |
| Übungen (ganzer Kurs) | 1 |
| Übungen (geteilter Kurs) | 1 |
| Tutorium (freiwillig) | 0 |

### Separate Prüfung

keine

## Praktikum

### Lernziele

#### Fertigkeiten

digitale Systeme entwerfen
kommerzielles Entwurfswerkzeug verstehen und einsetzen
wesentliche Eigenschaften von Standardkomponetnen kennen
Hardwarebeschreibungssprache VHDL auf Basis von Design-Pattern kennen und anwenden können

Funktionsweise eines Von-Neuman-Rechners beschreiben

Teilsysteme eines Von-Neumanrechners implemetieren

Programmierung einfacher Hochsprachen-Sequenzen in Assembler

komplexere Aufgaben in Kleinteam bewältigen

komplexere Problemlösungen erarbeiten
komplexeren Problemstellungen verstehen und analysieren
Systemverhalten aus spezifizierenden Texten herleiten
technische Texte erfassen
implizite Angaben erkennen und verstehen
fehlende Angaben
erkennen
ableiten
erfragen
System strukturiert analysieren
sinnvolle Teilsysteme (Schaltnetze, Zähler, Automaten) erkennen
Schnittstellen zwischen Teilsystemen erfassen
Teilsysteme modellieren
Zustandsüberführungsdiagramme erstellen
Wahrheitstabellen erstellen
Problemlösung mittels Entwurfswerkzeug implementieren, testen und am Zielsystem in Betrieb nehmen
Spezifiation von Teilsystemen
Schaltplan
VHDL
Synthese von Teilsystemen
Auswahl geeigneter Bibliotheksfunktionalitäten
Finden syntaktischer Fehler und deren Behebung
Simulation von Teilsystemen
Erstellen von Teststimuli
Finden semantischer Fehler und deren Behebung
Spezifikation des Gesamtsystems
Simulationdes Gesamtsystems
Erstellen von Teststimuli
Finden semantischer Fehler und deren Behebung
Gesamtsystem am Zielsystem in Betrieb nehmen

## Aufwand Präsenzlehre

| Typ | Präsenzzeit (h/Wo.) |
|---|---|
| Praktikum | 1 |
| Tutorium (freiwillig) | 0 |

## Separate Prüfung

keine