

Lehrveranstaltungshandbuch DR

Digitalrechner

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

– Allgemeine Informationen

Langname	Digitalrechner
Anerkennende LModule	<u>DR_BaTIN</u>
Verantwortlich	Prof. Dr. Lothar Thieling Professor Fakultät IME
Gültig ab	Wintersemester 2020/21
Niveau	Bachelor
Semester im Jahr	Wintersemester
Dauer	Semester
Stunden im Selbststudium	60
ECTS	5
Dozenten	NF Hartung
Voraussetzungen	keine
Unterrichtssprache	deutsch
separate Abschlussprüfung	Ja

Literatur

Urbanski K., Woitowicz R.: Digitaltechnik, 4. Auflage
Springer 2004

Beuth K.: Elektronik Bd. 4 Digitaltechnik, Vogel
Verlag 2001

Lipp H.M.: Grundlagen der Digitaltechnik, 4. Auflage
Oldenbourg 2002

Tanenbaum A. S.; Austin T.: Rechnerarchitektur: Von
der digitalen Logik zum Parallelrechner Pearson
Deutschland 2014

Abschlussprüfung

Details

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

Mindeststandard

Mindestens 50% der möglichen Gesamtpunktzahl.

Prüfungstyp

Klausur

– Vorlesung / Übungen

Lernziele

Zieltyp	Beschreibung
Kenntnisse	Boolesche Algebra Grundfunktion Axiome und Gesetze Disjunktive Normalform, Minterme Konjunktive Normalform, Maxterme Systematische Vereinfachung
Kenntnisse	Schaltnetze Logische Gatter, Tri-State-Buffer Beschreibungsformen boolesche 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
Kenntnisse	Zahlendarstellung in Rechnersystemem Dual-Code, Hexadezimal-Code, Basiswechsel Zweierkomplement Fixkommadarstellung Gleitkommadarstellung ASCII-Code
Kenntnisse	Speicherlemente Flip-Flops RS D Taktzustabssteuerung Taktflakensteuerung Register parallele Schreibleseregister Schieberegister Parallel-Seriell-Wandlung Seriell-Parallel-Wandlung praxisrelevante Spezifikationen setup time hold time minimum puls width

Besondere Voraussetzungen

keine

Begleitmaterial

elektronische
Vortragsfolien zur
Vorlesung
, elektronische
Übungsaufgabensammlung,
elektronisches
Entwurfswerkzeug für
die Entwicklung in
VHDL, elektronische
Sammlung von Beispiel-
Designs, elektronisches
Tool zur Simulation
eines einfachen Von-
Neumann-Rechners
(SVNR)
, elektronische Tutorials
für Selbststudium
Handhabung des
Entwurfswerkzeugs
Handhabung des
Rechnersimulators
(SVNR)

Separate Prüfung

Nein

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

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

Kenntnisse Zustandsüberförungsdiagramme
Modellierung nach Moore
zu beachtende Eigenschaften
(Determinismus, Vollständigkeit)

Kenntnisse 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
Zustandsüberförungsdiagrammen
in VHDL
Hierarchisches VHDL-Desin
Packages, Components, Portmaps,
Generics

Kenntnisse Programmierbare Bausteine
Aufbau
Grundidee
Technologie
Zellbegriff
CPLD versus FPGA
Entwurfswerkzeug
Spezifikation 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

Kenntnisse 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)

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

Fertigkeiten 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

Fertigkeiten Interpretation und Umwandlung von Codes
Codieren
Decodieren
Fehlererkennung und Fehlerkorrektur

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

Fertigkeiten 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

Fertigkeiten 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)

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

Aufwand Präsenzlehre

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

– Praktikum

Lernziele

Zieltyp	Beschreibung
Fertigkeiten	digitale Systeme entwerfen kommerzielles Entwurfswerkzeug verstehen und einsetzen wesentliche Eigenschaften von Standardkomponenten kennen Hardwarebeschreibungssprache VHDL auf Basis von Design-Pattern kennen und anwenden können
Fertigkeiten	Funktionsweise eines Von-Neuman-Rechners beschreiben
Fertigkeiten	Teilsysteme eines Von-Neumanrechners implementieren
Fertigkeiten	Programmierung einfacher Hochsprachen-Sequenzen in Assembler
Fertigkeiten	komplexere Aufgaben in Kleinteam bewältigen

Besondere Voraussetzungen

keine

Begleitmaterial

elektronische Aufgabenstellung (Problembeschreibung)
, elektronisches Entwurfswerkzeug mit Schematic- und VHDL-Design-Entry
, elektronische Sammlung von Beispiel-Designs (Schematic und VHDL)
, elektronisches Tool zur Simulation eines einfachen Von-Neumann-Rechners (SVNR)
, elektronische Tutorials für Selbststudium
Handhabung des Entwurfswerkzeugs
Handhabung des Rechnersimulators (SVNR)

Separate Prüfung

Nein

Fertigkeiten

- 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
- Zustandsüberförungsdiagramme erstellen
- Wahrheitstabellen erstellen
- Problemlösung mittels Entwurfswerkzeug implementieren, testen und am Zielsystem in Betrieb nehmen
- Spezifikation 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
- Simulation des 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

