University of Erlangen-Nuremberg
Institute for Electrotechnics Engineering

Ausarbeitung eines Praktikums zur Vorlesung Architekturen der Digitalen Signalverarbeitung

Entwurf einer Plattform und Programmierung in VHDL

Daniel Glaser

Lehrstuhl für Technische Elektronik
Prof. Dr.-Ing. Dr.-Ing. habil. Robert Weigel

28. Januar 2007

Übersicht

Übersicht

Übersicht

Übersicht

Übersicht

Überblick

Einführung

Motivation

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware

Einführung

Motivation

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware
- Algorithmen der digitalen Signalverarbeitung

Einführung

Motivation

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware
- Algorithmen der digitalen Signalverarbeitung
- Optimierung von Algorithmen auf quantisierte Signale

Einführung

Motivation

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware
- Algorithmen der digitalen Signalverarbeitung
- Optimierung von Algorithmen auf quantisierte Signale
- Simulationsbeispiele in MATLAB

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware
- Algorithmen der digitalen Signalverarbeitung
- Optimierung von Algorithmen auf quantisierte Signale
- Simulationsbeispiele in MATLAB
- Beispiele realer Hardwareumsetzungen (Blockdiagramme)

Einführung

Motivation

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware
- Algorithmen der digitalen Signalverarbeitung
- Optimierung von Algorithmen auf quantisierte Signale
- Simulationsbeispiele in MATLAB
- Beispiele realer Hardwareumsetzungen (Blockdiagramme)
- Keine Hardware „zum Anfassen“

Einführung

Motivation

Vorlesung: Architekturen Digitaler Signalverarbeitung

- Analoge Konzepte in digitaler Hardware
- Algorithmen der digitalen Signalverarbeitung
- Optimierung von Algorithmen auf quantisierte Signale
- Simulationsbeispiele in MATLAB
- Beispiele realer Hardwareumsetzungen (Blockdiagramme)
- Keine Hardware „zum Anfassen“
- Fehlende praktische Erfahrung der Studenten

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung

Entwurf eines Praktikums

University of Erlangen-Nuremberg
Institute for Electrotechnics Engineering

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel
- Auswahl einer Hardwareplattform

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel
- Auswahl einer Hardwareplattform
- Entwurf eines Praktikumsskripts

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel
- Auswahl einer Hardwareplattform
- Entwurf eines Praktikumsskripts
- $\Rightarrow$ Programmierung der Algorithmen in VHDL

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel
- Auswahl einer Hardwareplattform
- Entwurf eines Praktikumsskripts
- $\Rightarrow$ Programmierung der Algorithmen in VHDL
- Problemanalyse und Vorhersage

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel
- Auswahl einer Hardwareplattform
- Entwurf eines Praktikumsskripts
- $\Rightarrow$ Programmierung der Algorithmen in VHDL
- Problemanalyse und Vorhersage
- Durchführung des Praktikums

Entwurf eines Praktikums

Praktikum: Architekturen Digitaler Signalverarbeitung

- Vorgabenanalyse
- Entwurf eines in sich geschlossenen Gesamtsytems $\Leftarrow$ Hagenberg
- Auswahl geeigneter Konzepte aus der Vorlesung $\Leftarrow$ Hagenberg
- Umsetzung der Algorithmen in MATLAB $\Leftarrow$ SA A. Schedel
- Auswahl einer Hardwareplattform
- Entwurf eines Praktikumsskripts
- $\Rightarrow$ Programmierung der Algorithmen in VHDL
- Problemanalyse und Vorhersage
- Durchführung des Praktikums $\Leftarrow$ zukünftig

Überblick

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:
  - Bandpassfilter (1 je Trägerfrequenz)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:
  - Bandpassfilter (1 je Trägerfrequenz)
  - Hüllkurvendemodulator (1 je Trägerfrequenz)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:
  - Bandpassfilter (1 je Trägerfrequenz)
  - Hüllkurvendemodulator (1 je Trägerfrequenz)
  - Tiefpassfilter (1 je Trägerfrequenz)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:
  - Bandpassfilter (1 je Trägerfrequenz)
  - Hüllkurvendemodulator (1 je Trägerfrequenz)
  - Tiefpassfilter (1 je Trägerfrequenz)
  - Vergleicher (mit Hysterese)

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:
  - Bandpassfilter (1 je Trägerfrequenz)
  - Hüllkurvendemodulator (1 je Trägerfrequenz)
  - Tiefpassfilter (1 je Trägerfrequenz)
  - Vergleicher (mit Hysterese)
- Weitere Schnittstellenmodule

Finden geeigneter Algorithmen für das Gesamtsystem

Gesamtsystem bestehend aus:

- Sender bestehend aus:
  - (Pseudo-)Zufallszahlengenerator (PRN-Schieberegister)
  - Signalgenerator (DDS, CORDIC)
  - Modulator (FSK)
- Empfänger bestehend aus:
  - Bandpassfilter (1 je Trägerfrequenz)
  - Hüllkurvendemodulator (1 je Trägerfrequenz)
  - Tiefpassfilter (1 je Trägerfrequenz)
  - Vergleicher (mit Hysterese)
- Weitere Schnittstellenmodule
- Fehlerkorrekturhilfen

Konzeptdiagramme

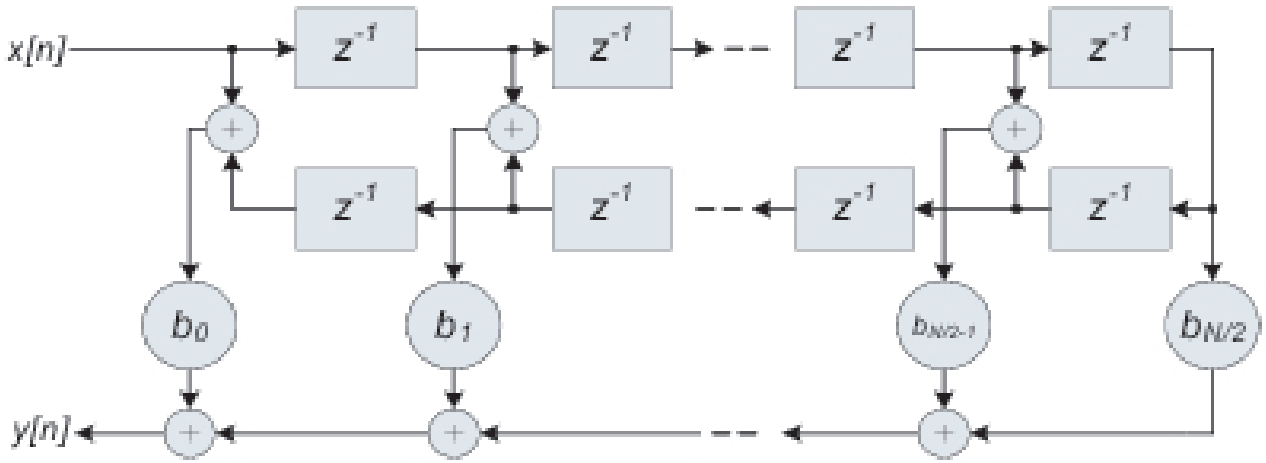


Abbildung: FIR in Linearphasenstruktur

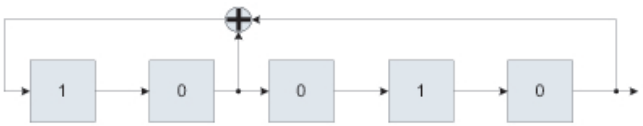


Abbildung: PRN-Register

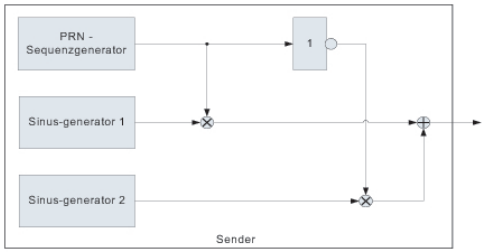


Abbildung: Sender Blockschaltbild

Überblick

Entwicklungsplattform

Möglichkeiten:

- Rahmenbedingungen (aus Konzept)

Entwicklungsplattform

Möglichkeiten:

- Rahmenbedingungen (aus Konzept)
- Kernkomponente FPGA

Entwicklungsplattform

University of Erlangen-Nuremberg
Institute for Electrotechnics Engineering

Möglichkeiten:

- Rahmenbedingungen (aus Konzept)
- Kernkomponente FPGA
- Entwicklungssystem

Entwicklungsplattform

Möglichkeiten:

- Rahmenbedingungen (aus Konzept)
- Kernkomponente FPGA
- Entwicklungssystem
  - Verfügbare Evaluierungsboards

Entwicklungsplattform

Möglichkeiten:

- Rahmenbedingungen (aus Konzept)
- Kernkomponente FPGA
- Entwicklungssystem
  - Verfügbare Evaluierungsboards
  - Design eines eigenen Systems

Entwicklungsplattform

Möglichkeiten:

- Rahmenbedingungen (aus Konzept)
- Kernkomponente FPGA
- Entwicklungssystem
  - Verfügbare Evaluierungsboards
  - Design eines eigenen Systems

$\Rightarrow$ Eigendesign

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile
- Schaltplanerstellung

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile
- Schaltplanerstellung
- Layout (Platzierung und Entflechtung)

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile
- Schaltplanerstellung
- Layout (Platzierung und Entflechtung)
- Aufbau

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile
- Schaltplanerstellung
- Layout (Platzierung und Entflechtung)
- Aufbau
- Fehleranalyse

Entwicklungsplattform

Eigenes System:

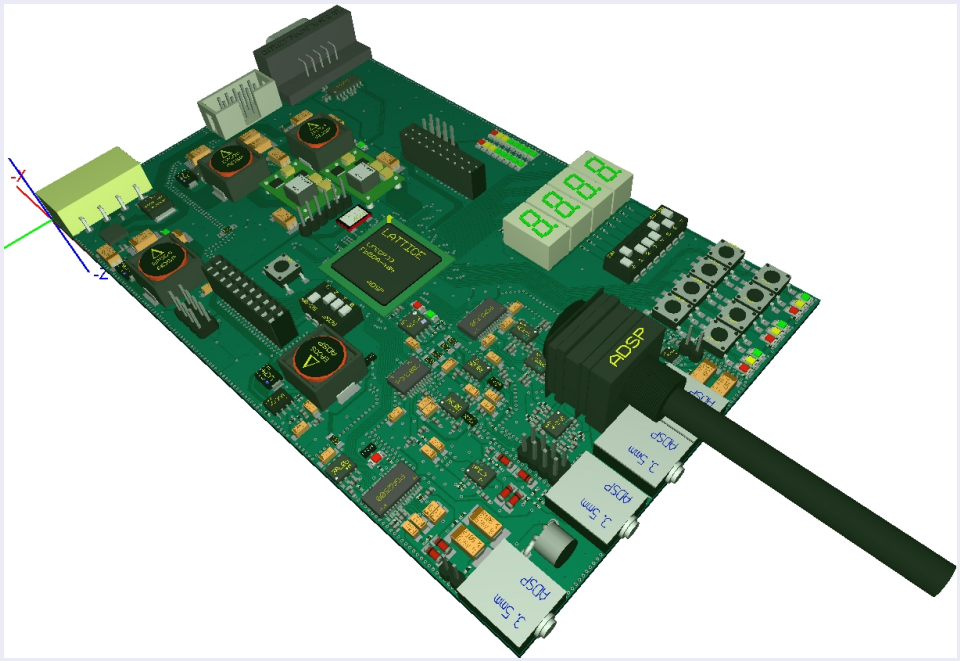
- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile
- Schaltplanerstellung
- Layout (Platzierung und Entflechtung)
- Aufbau
- Fehleranalyse

Entwicklungsplattform

Eigenes System:

- Rahmenbedingungen (2· ≈ 300 €, 15· ≈ 200 €)
- Auswahl geeigneter Bauteile
- Schaltplanerstellung
- Layout (Platzierung und Entflechtung)
- Aufbau
- Fehleranalyse

3D-Modell des SPATES



Praktikumskonzept

Abstraktion der Hardware

- Grundlegend: Starke Abstraktion der Hardware
- Aufbau des Wissens in kleinen Teilen
  - Vorgefertigtes Modul mit kleiner „Leerstelle“
- Auf häufige Anfänger-Fehler direkt hinweisen

Praktikumskonzept

Abstraktion der Hardware

- Grundlegend: Starke Abstraktion der Hardware
- Aufbau des Wissens in kleinen Teilen
  - Vorgefertigtes Modul mit kleiner „Leerstelle“
  - Grobe Vorgabe des inneren Toplevels
- Auf häufige Anfänger-Fehler direkt hinweisen

Praktikumskonzept

Abstraktion der Hardware

- Grundlegend: Starke Abstraktion der Hardware
- Aufbau des Wissens in kleinen Teilen
  - Vorgefertigtes Modul mit kleiner „Leerstelle“
  - Grobe Vorgabe des inneren Toplevels
  - Vorgabe der Ports
- Auf häufige Anfänger-Fehler direkt hinweisen

Praktikumskonzept

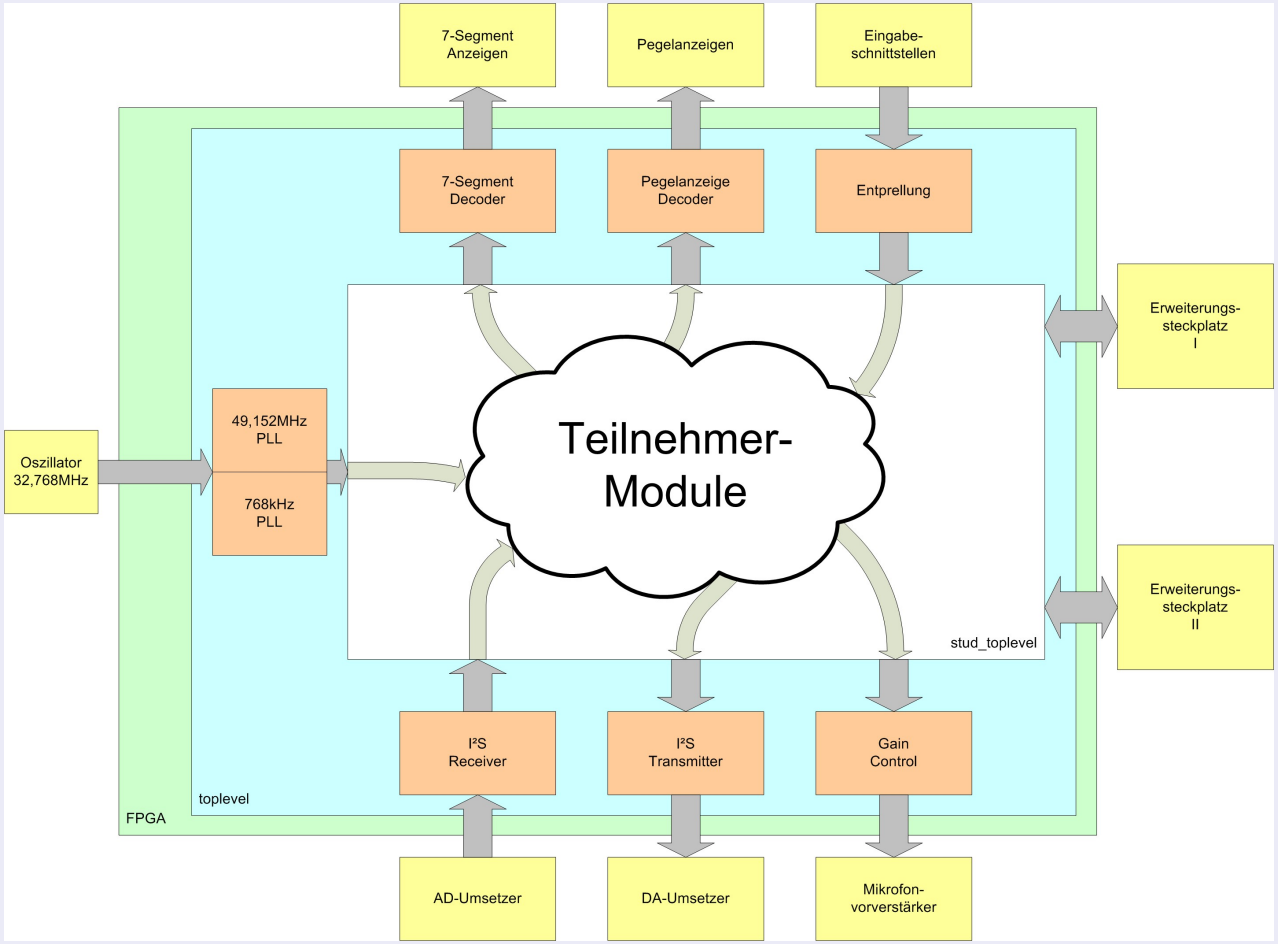
Abstraktion der Hardware

- Grundlegend: Starke Abstraktion der Hardware
- Aufbau des Wissens in kleinen Teilen
  - Vorgefertigtes Modul mit kleiner „Leerstelle“
  - Grobe Vorgabe des inneren Toplevels
  - Vorgabe der Ports
  - Völlig selbständige Programmierung der Aufgabe
- Auf häufige Anfänger-Fehler direkt hinweisen

Abstraktionsmodell

Der Toplevel im Toplevel

Hardwareabstraktion



Überblick

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung
- AD- bzw. DA-Umsetzer (I^2S-Schnittstelle $\Rightarrow$ parallel)

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung
- AD- bzw. DA-Umsetzer (I^2S-Schnittstelle $\Rightarrow$ parallel)
- Vorverstärkerregelung

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung
- AD- bzw. DA-Umsetzer (I^2S-Schnittstelle $\Rightarrow$ parallel)
- Vorverstärkerregelung
- 7-Segmentdecoder

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung
- AD- bzw. DA-Umsetzer (I^2S-Schnittstelle $\Rightarrow$ parallel)
- Vorverstärkerregelung
- 7-Segmentdecoder
- Mikrofonvorverstärkerregelung

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung
- AD- bzw. DA-Umsetzer (I^2S-Schnittstelle $\Rightarrow$ parallel)
- Vorverstärkerregelung
- 7-Segmentdecoder
- Mikrofonvorverstärkerregelung
- Pegelanzeige-Abstraktion

Vorgefertigte Module: Abstraktionmodule

- Initialisierungmodul
- Takterzeugung
- AD- bzw. DA-Umsetzer (I^2S-Schnittstelle $\Rightarrow$ parallel)
- Vorverstärkerregelung
- 7-Segmentdecoder
- Mikrofonvorverstärkerregelung
- Pegelanzeige-Abstraktion
- Schalter-/Tasterentprellung

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)
- Bandpass (Optimal)

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)
- Bandpass (Optimal)
- Demodulator

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)
- Bandpass (Optimal)
- Demodulator
- Tiefpass

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)
- Bandpass (Optimal)
- Demodulator
- Tiefpass
- Signalregenerierung

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)
- Bandpass (Optimal)
- Demodulator
- Tiefpass
- Signalregenerierung
- Hysterese

Studentische Module: Musterlösungen

- Erste Schritte: Multiplexer
- (Pseudo-)Zufallszahlengenerator
- Signalgenerator
- Pegelanzeige (für Fehlersuche)
- Modulator
- Bandpass (Suboptimal)
- Bandpass (Optimal)
- Demodulator
- Tiefpass
- Signalregenerierung
- Hysterese
- Zusammenschaltung

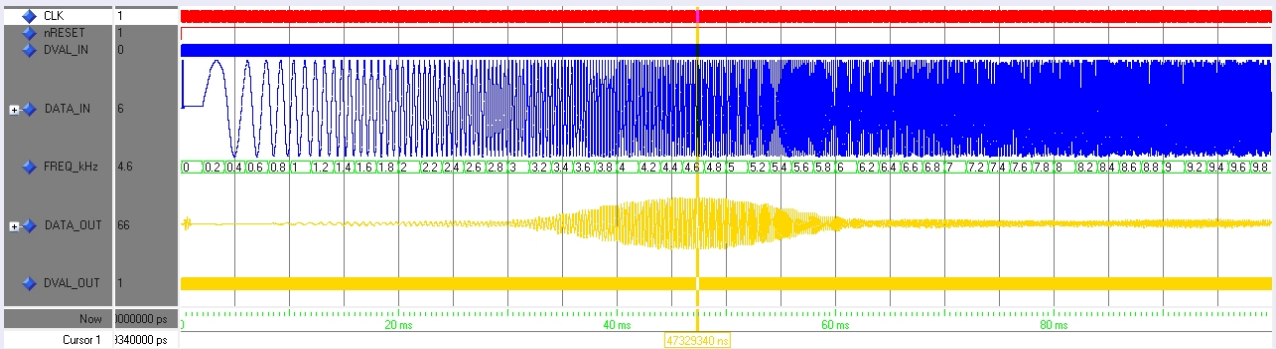
VHDL-Programierung

Verifiaktion: Testbenches

Verifikation

- Alle Module wurden verifiziert
- Können den Studenten als Verifikation dienen

Beispiel: Bandpass (optimal)



Überblick

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation
  - Musterlösungen sind portabel, gut strukturiert, kommentiert

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation
  - Musterlösungen sind portabel, gut strukturiert, kommentiert
  - Testbenches auch für Studenten geeignet

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation
  - Musterlösungen sind portabel, gut strukturiert, kommentiert
  - Testbenches auch für Studenten geeignet
- Hardware reparieren

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation
  - Musterlösungen sind portabel, gut strukturiert, kommentiert
  - Testbenches auch für Studenten geeignet
- Hardware reparieren
- VHDL-Code in Hardware verifizieren

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation
  - Musterlösungen sind portabel, gut strukturiert, kommentiert
  - Testbenches auch für Studenten geeignet
- Hardware reparieren
- VHDL-Code in Hardware verifizieren
- Gesamtsystem verifizieren

Was ist fertig? Was noch zu tun?

Nacharbeit nötig

- MATLAB $\Leftrightarrow$ VHDL aufeinander abgleichen
- VHDL-Programierung erfolgreich
  - Vorgefertigte Module arbeiten hervorragend in der Simulation
  - Musterlösungen sind portabel, gut strukturiert, kommentiert
  - Testbenches auch für Studenten geeignet
- Hardware reparieren
- VHDL-Code in Hardware verifizieren
- Gesamtsystem verifizieren
- **Den ersten Testlauf absolvieren**

Noch Fragen?

Unklarheiten, Detailiertere Informationen?

Vielen Dank für Ihre Aufmerksamkeit!!!