Echtzeitsysteme
Lehrstuhl Informatik VI - Robotics and Embedded Systems

Echtzeitsysteme
Wintersemester 2013/2014

Dr. Christian Buckl

fortiss GmbH

Lehrstuhl VI Robotics and Embedded Systems

fortiss

Überblick

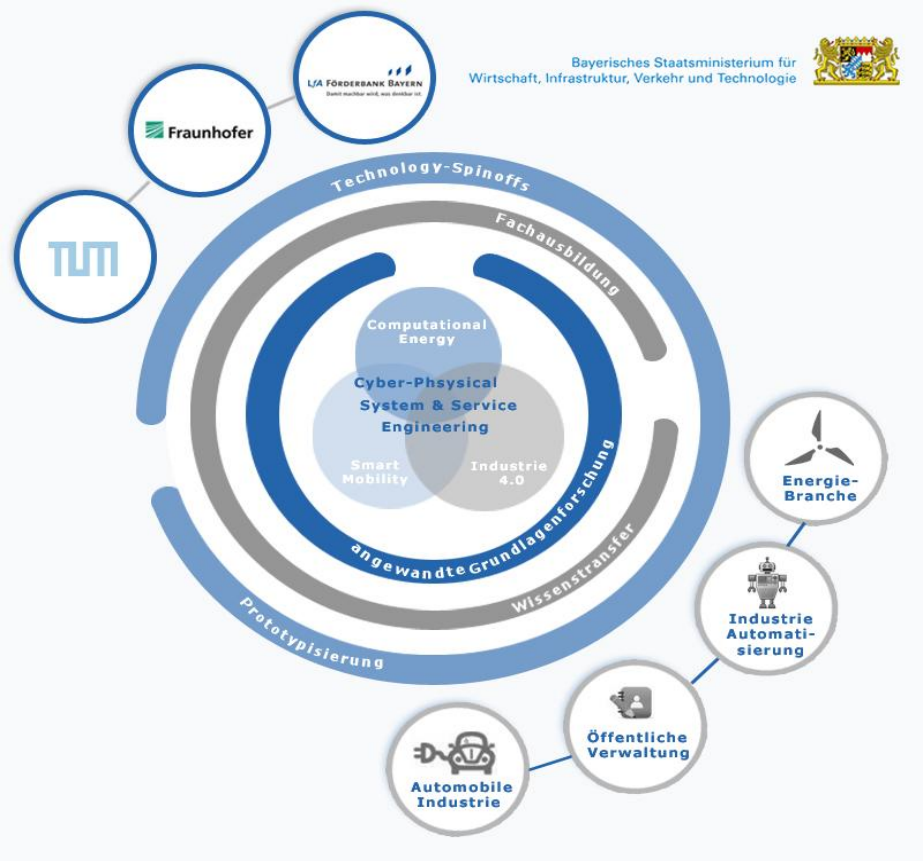
Forschungs- und Transferinstitut assoziiert mit der **Technischen Universität München**
In der Rechtsform einer **gemeinnützigen GmbH**.

Gesellschafter

TUM K.d.ö.R.
Fraunhofer Gesellschaft
LfA Förderbank Bayern

Wissenschaftliche Direktoren

Prof. Dr. Dr. h.c. Manfred Broy
Prof. Dr. Alois Knoll
Prof. Dr. Helmut Krcmar



fortiss

Organisation

DIREKTORIUM

Prof. Dr. Dr. h.c. Manfred Broy (Sprecher)
Prof. Dr. Helmut Krcmar
Prof. Dr. Alois Knoll

GESCHÄFTSLEITUNG

Dr. Harald Rueß
Dr. rer. nat. habil. Bernhard Schätz (Stellv.)

FINANZEN UND PERSONAL

Ines Lenz

FB I

SOFTWARE & SYSTEMS ENGINEERING

Prof. Dr. Dr. h.c. Manfred Broy
Dr. rer. nat. habil Bernhard Schätz

FB II

CYBER-PHYSICAL SYSTEMS

Prof. Dr. Alois Knoll
Dr. rer. nat. Christian Buckl

FB III

INFORMATION SYSTEMS &SERVICES

Prof. Dr. Helmut Krcmar
Dr. Petra Wolf

FB IV

INDUSTRIAL AUTOMATION

N.N.
Dr. Alois Zoitl

FB V

EMBEDDED SYSTEMS SOFTWARE ENGINEERING INSTITUTE

Prof. Manfred Hajek
Werner Burger (Industrie)

EIT ICT LABS

Satellite Co-Location Center Munich

Echtzeitsysteme: Organisation

Team

Dr. Christian Buckl

Philipp Heise

Übungen: Rafael Hostettler, Martin Eder, Manuel Schiller, Hardik Shah, Mohammadali Nasseri

Homepage der Vorlesung mit Folien, Übungsaufgaben und weiterem Material:
http://www6.in.tum.de/Main/TeachingWs20123chtzeitsysteme

Bestandteile der Vorlesung

- Vorlesung:
  - Dienstag 10:15-11:45 Uhr MI HS 2
  - Mittwoch 10:05-10:50 Uhr MI HS 2
  - 6 ECTS Punkte
  - Wahlpflichtvorlesung im Gebiet Echtzeitsysteme (Technische Informatik)
  - Wahlpflichtvorlesung für Studenten der Elektro- und Informationstechnik
  - Pflichtvorlesung für Studenten des Maschinenbau Richtung Mechatronik
- Übung:
  - zweistündige Tutorübung, im Raum 03.05.012
  - Termine werden Anfang nächster Woche angekündigt
  - Beginn: voraussichtlich ab 28.10.2013, Anmeldung ab Anfang über TUMonline (http://www.tumonline.de, siehe Tutorübung zu Echtzeitsysteme)
- Prüfung:
  - Schriftliche Klausur am Ende des Wintersemesters, falls ein Schein benötigt wird.

Grundsätzliches Konzept

- Themen werden aus verschiedenen Blickrichtungen beleuchtet:
  - Stand der Technik in der Industrie
  - Stand der Technik in der Wissenschaft
  - Existierende Werkzeuge
  - Wichtig: nicht die detaillierte Umsetzung, sondern die Konzepte sollen verstanden werden
- Ihre Beteiligung ist uns wichtig: bitte fragen Sie, wenn etwas unklar ist, beteiligen Sie sich an Übungen und wenn für Sie interessant: halten Sie auch selbst einmal die Vorlesung
- Zur Verdeutlichung theoretischer Inhalte wird versucht, Analogien zum Alltag herzustellen. Wichtig: Praktische Aufgaben in der Vorlesung und der Übung
- In jedem Kapitel werden die relevanten Literaturhinweise referenziert
- Zur Erfolgskontrolle werden Klausuraufgaben der letzten Jahre am Ende eines Kapitels diskutiert
- Folien werden in der Regel kurz vor Beginn der Vorlesung auf die Webseite gestellt
- **Wir freuen uns jederzeit über Fragen, Verbesserungsvorschläge und konstruktive Kommentare!**

Informationen zur Übung

- Ziel: Praktisches Einüben von Vorlesungsinhalten
- Übungsaufgaben werden in Gruppen zu zweit am Computer gelöst
- Platz ist begrenzt (8 Computer, 16 Studenten) **Anmeldung erforderlich**
- Übungsaufgaben sind auch auf der Vorlesungsseite verfügbar
- Es werden diverse Aufgaben aus dem Bereich der Echtzeitprogrammierung angeboten, wie z.B. Aufgaben zu Threads, Semaphore, Kommunikation
- Programmiersprache ist überwiegend C, zu Beginn der Übung wird eine kurze Einführung in C angeboten
- Die Anmeldung erfolgt über **TUMonline** (Tutorübungen zu Echtzeitsysteme (IN2060))
- Falls Bedarf an weiteren Terminen besteht, senden Sie bitte eine Mail an Philipp Heise (heise@in.tum.de).
- Die Übungsinhalte sind nicht direkt prüfungsrelevant, **tragen aber stark zum Verständnis bei.**
- **Ihre Rückmeldung ist wichtig, denn sie bestimmt über die Inhalte!**

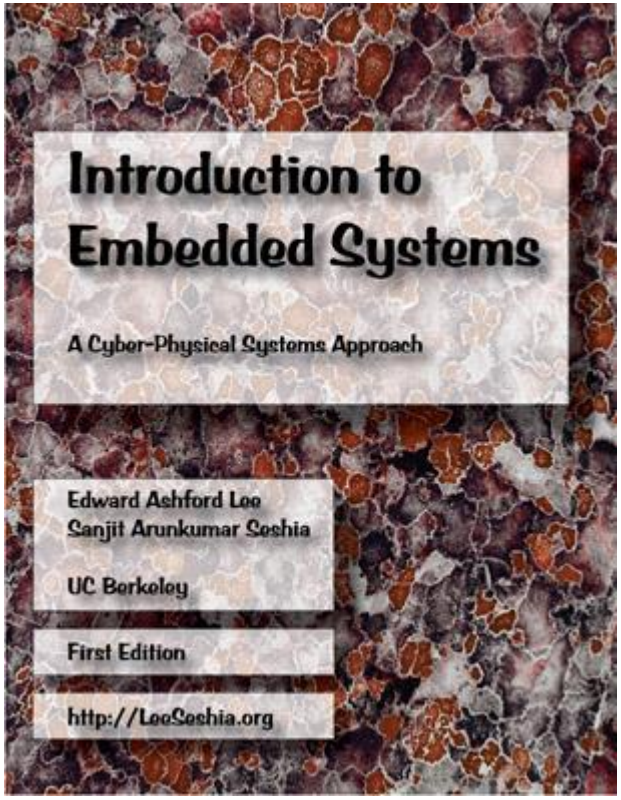
Klausur

- Für Studenten, die einen Schein benötigen, wird am Ende der Vorlesung eine schriftliche Klausur angeboten.
- Stoff der Klausur sind die Inhalte der Vorlesung.
- Die Inhalte der Übung sind nicht direkt prüfungsrelevant, tragen allerdings zum Verständnis des Prüfungsstoffes bei.
- Voraussichtlicher Termin: letzte Vorlesungswoche (Rückmeldung mit Prüfungsamt steht noch aus)
- Voraussichtlich erlaubte Hilfsmittel: keine

Weitere Angebote des Lehrstuhls

- Weitere Vorlesungen: Robotik, Digitale Signalverarbeitung, Maschinelles Lernen und bioinspirierte Optimierung I&II, Sensor- und kamerageführte Roboter
- Praktika: Microcontrollerprogrammierung, Roboterfußball, Industrieroboter, Neuronale Netze und Maschinelles Lernen, Bildverarbeitung, Signalverarbeitung
- Seminare: Sensornetzwerke, Modellierungswerkzeuge, Busprotokolle, Objekterkennung und Lernen, Neurocomputing,
- Diplomarbeiten / Masterarbeiten
- Systementwicklungsprojekte / Bachelorarbeiten
- Guided Research, Stud. Hilfskräfte
- Unser gesamtes Angebot finden Sie unter http://wwwknoll.in.tum.de

Literatur



- Lee, Seshia: Introduction to Embedded Systems
  - Buch deckt die meisten Kapitel der Vorlesung (bis auf Kommunikation) ab
  - Kostenlos online verfügbar unter http://leeseshia.org/

Weitere Literatur

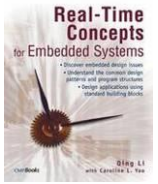
Weitere Literaturangaben befinden sich in den jeweiligen Abschnitten.

Hermann Kopetz: Real-Time Systems (Überblick)

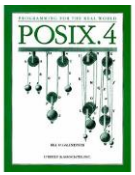
Jane W. S. Liu: Real-Time Systems
(Überblick, Schwerpunkt Scheduling)

Stuart Bennet: Real-Time Computer Control:
An Introduction (Überblick, Hardware)

Alan Burns, Andy Wellings: Real-Time Systems and Programming Languages (Schwerpunkt: Programmiersprachen)



Qing Li, Caroline Yao: Real-Time Concepts for
Embedded Systems (Schwerpunkt: Programmierung)



Bill O. Gallmeister: Programming for the Real-World: POSIX.4
(Schwerpunkt: Posix)

Vorlesungsinhalte

1. Einführung Echtzeitsysteme
2. Uhren
3. Modellierung und Werkzeuge
4. Nebenläufigkeit
5. Scheduling
6. Kommunikation
7. Echtzeitbetriebssysteme
8. Programmiersprachen
9. Fehlertolerante Systeme
10. Spezielle Hardware
11. Regelungstechnik

Weitere Themen können bei Interesse aufgenommen werden. Melden Sie sich einfach nach der Vorlesung oder per Email.

Inhalt I

- Kapitel Einführung (ca. 1 Vorlesungswoche)
  - Definition Echtzeitsysteme
  - Klassifikation
  - Echtzeitsysteme im täglichen Einsatz
  - Beispielanwendungen am Lehrstuhl
- Kapitel Uhren (1 Vorlesungswoche)
  - Uhren
  - Synchronisation von verteilten Uhren

Inhalt II

- Kapitel Modellierung/Werkzeuge (ca. 1-2 Vorlesungswochen)
  - Allgemeine Einführung
  - Grundsätzlicher Aufbau, Models of Computation, Ptolemy
  - Synchrone Sprachen (Esterel, Lustre), SCADE, EasyLab
  - Zeitgesteuerte Systeme: Giotto, FTOS, TTA
- Kapitel Nebenläufigkeit (ca. 3 Vorlesungswochen)
  - Prozesse, Threads
  - Interprozesskommunikation
  - Scheduling Probleme

Inhalt III

- Kapitel Scheduling (ca. 2 Vorlesungswochen)
  - Kriterien
  - Planung Einrechner-System, Mehrrechnersysteme
  - EDF, Least Slack Time, Scheduling mit Prioritäten (FIFO, Round Robin)
  - Scheduling periodischer Prozesse
- Kapitel Echtzeitfähige Kommunikation (ca. 1 Vorlesungswoche)
  - Token-Ring
  - CAN-Bus
  - TTP, FlexRay
  - Real-Time Ethernet

Inhalt IV

- Kapitel Echtzeitbetriebssysteme (ca. 1 Vorlesungswoche)
  - QNX, VxWorks, PikeOS
  - RTLinux, RTAI, Linux Kernel 2.6
  - TinyOS, eCos
  - OSEK
- Kapitel Fehlertoleranz (ca. 2 Vorlesungswochen)
  - Bekannte Softwarefehler
  - Definitionen
  - Fehlerarten
  - Fehlerhypothesen
  - Fehlertoleranzmechanismen

Potentielle zusätzliche Inhalte

- Kapitel: Spezielle Hardware (1 Vorlesungswoche)
  - Digital-Analog-Converter (DAC)
  - Analog-Digital-Converter (ADC)
  - Speicherprogrammierbare Steuerung (SPS)
- Kapitel: Regelungstechnik (ca. 2 Vorlesungswochen)
  - Definitionen
  - P-Regler
  - PI-Regler
  - PID-Regler
  - Fuzzy-Logic

Kapitel 1

Einführung Echtzeitsysteme

Inhalt

- Definition Echtzeitsysteme
- Klassifikation von Echtzeitsystemen
- Echtzeitsysteme im täglichen Leben
- Beispielanwendungen am Lehrstuhl

Definition Echtzeitsystem

Ein Echtzeit-Computersystem ist ein Computersystem, in dem die Korrektheit des Systems nicht nur vom logischen Ergebnis der Berechnung abhängt, sondern auch vom physikalischen Moment, in dem das Ergebnis produziert wird.

Ein Echtzeit-Computer-System ist immer nur ein Teil eines größeren Systems, dieses größere System wird Echtzeit-System genannt.

Hermann Kopetz

TU Wien

Definition Eingebettetes System

Technisches System, das durch ein integriertes, von Software gesteuertes Rechensystem gesteuert wird. Das Rechensystem selbst ist meist nicht sichtbar und kann in der Regel nicht frei programmiert werden. Um die Steuerung zu ermöglichen ist zumeist eine Vielzahl von sehr speziellen Schnittstellen notwendig.

In der Regel werden leistungsärmere Mikroprozessoren mit starken Einschränkung in Bezug auf die Rechenleistung und Speicherfähigkeit eingesetzt.

Resultierende Eigenschaften

➔ zeitliche Anforderungen

- Zeitliche Genauigkeit (nicht zu früh, nicht zu spät)
- Garantierte Antwortzeiten
- Synchronisation von Ereignissen / Daten
- **Aber nicht:** Allgemeine Geschwindigkeit

➔ Eigenschaften aufgrund der Einbettung

- Echtzeitsysteme sind typischerweise sehr Eingabe/Ausgabe (E/A)-lastig
- Echtzeitsysteme müssen fehlertolerant sein, da sie die Umgebung beeinflussen
- Echtzeitsysteme sind häufig verteilt

Zeitlicher Determinismus vs. Leistung

- Konsequenz der Forderung nach deterministischer Ausführungszeit: Mechanismen, die die allgemeine Performance steigern, aber einen negativen, nicht exakt vorhersehbaren Effekt auf einzelne Prozesse haben können, werden in der Regel nicht verwendet:
  - Virtual Memory
  - Garbage Collection
  - Asynchrone IO-Zugriffe
  - rekursive Funktionsaufrufe

Klassifikation von Echtzeitsystemen

- Echtzeitsysteme können in verschiedene Klassen unterteilt werden:
  - Nach den Konsequenzen bei der Überschreitung von Fristen: harte vs. weiche Echtzeitsysteme
  - Nach dem Ausführungsmodell: zeitgesteuert (zyklisch, periodisch) vs. ereignisbasiert (aperiodisch)

Harte bzw. weiche Echtzeitsysteme

- **Weiche Echtzeitsysteme:**
  Die Berechnungen haben eine zeitliche Ausführungsfrist, eine Überschreitung dieser Fristen hat jedoch keine katastrophale Folgen. Eventuell können die Ergebnisse noch verwendet werden, insgesamt kommt es durch die Fristverletzung evtl. zu einer Dienstverschlechterung.
  Beispiel für ein weiches Echtzeitsystem: Video
  Konsequenz von Fristverletzungen: einzelne Videoframes gehen verloren, das Video hängt

- **Harte Echtzeitsysteme:**
  Eine Verletzung der Berechnungsfristen kann sofort zu fatalen Folgen (hohe Sachschäden oder sogar Gefährdung von Menschenleben) führen. Die Einhaltung der Fristen ist absolut notwendig.
  Beispiel für ein hartes Echtzeitsystem: Raketensteuerung
  Konsequenz von Fristverletzung: Absturz bzw. Selbstzerstörung der Rakete

Unterteilung nach Ausführungsmodell

- Zeitgesteuerte Applikationen:
  - Der gesamte zeitliche Systemablauf wird zur Übersetzungszeit festgelegt
  - Notwendigkeit einer präzisen, globalen Uhr) Uhrensynchronisation notwendig
  - Für die einzelnen Berechnungen ist jeweils ein Zeitslot reserviert) Abschätzung der maximalen Laufzeiten (**worst case execution times - WCET**) notwendig
  - **Vorteil**: Statisches Scheduling möglich und damit ein vorhersagbares (**deterministisches**) Verhalten
- Ereignisgesteuerte Applikationen:
  - Alle Ausführungen werden durch das Eintreten von Ereignissen angestoßen
  - Wichtig sind bei ereignisgesteuerten Anwendungen garantierte Antwortzeiten
  - Das Scheduling erfolgt dynamisch, da zur Übersetzungszeit keine Aussage über den zeitlichen Ablauf getroffen werden kann.