

Überblick

Ausblick

EZS2
Hiwi
Studien- und Diplomarbeiten

Entwicklungsumgebung

- Prozessor**
 - ▶ Infineon TriCore 1.3 - TC1796
 - ▶ Motorola PowerPC - MPC565
- Board**
 - ▶ Infineon TriBoard
 - ▶ Phytex phyCore MPC565
- Peripherie**
 - ▶ Serielle Schnittstelle, GPIO, CAN, ADC, DAC
- Betriebssysteme**
 - ▶ Eigenentwicklungen: CiAO, KESO, (eCos)
 - ▶ Industrie: ProOSEK/time, eCos, PXROS, FreeRTOS
- Programmiersprachen**
 - ▶ Assembler, C, C++, Java
- Werkzeuge**
 - ▶ Modellierung: SMC (*State Machine Compiler*)
 - ▶ GNU Tools (GCC, Binutils, GDB, make)
 - ▶ Lauterbach Trace32
 - ▶ Oszilloskop, Funktionsgenerator

Echtzeitsysteme (EZS) 2

Integrierte Lehrveranstaltung, 4SWS

Inhalt

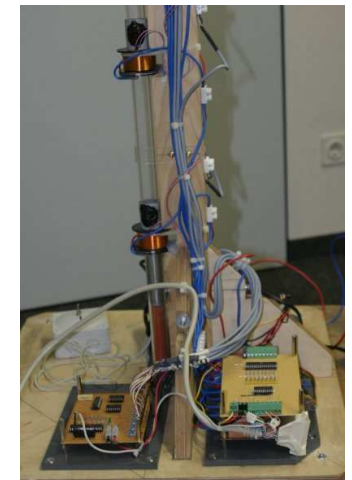
- ▶ ein kompletter Entwicklungszyklus für ein EZS
 1. Anforderungsanalyse
 2. Einarbeitung in die Entwicklungsumgebung
 3. Entwicklung der Komponenten
 4. Testen der Komponenten
 5. Komposition - Integrationsphase
 6. Akzeptanztest

Organisation

- ▶ Bearbeitung der Experimente erfolgt in 3er-Gruppen
- ▶ (benoteter) Schein
- ▶ **keine** Prüfung!

Experiment 1: Hau den Lukas

- ▶ Eisenprojektil in einer Plexiglasröhre
- ▶ wird von Elektromagneten
 - ▶ beschleunigt
 - ▶ gebremst
- ▶ Elektromagneten werden gesteuert
- ▶ Lichtschranken *beobachten* das Projektil
- ▶ verschiedene *Spielarten*
 - ▶ kontinuierlich/schrittweise
 - ▶ anheben/fallen/pendeln
- ▶ mit/ohne Bedienpult



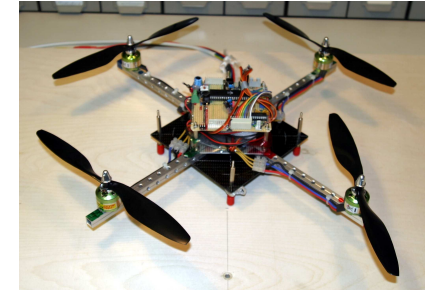
Experiment 2: Generator

- ▶ Regelkreis
 - ▶ Motor treibt Generator an
 - ▶ erzeugte Spannung soll möglichst konstant sein
- ▶ verschiedene Lasten werden zugeschaltet
 - ▶ konstante Lasten
 - ▶ variable Lasten
- ▶ Regler muss die Spannung *nachregeln*
 - ▶ und zwar **rechtzeitig**



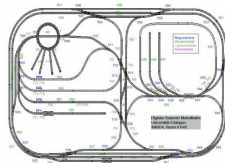
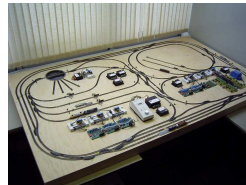
Experiment 3: Quadkopter

- ▶ Fluggerät
 - ▶ 4 unabhängige Rotoren
 - ▶ Steuerung durch
 - ▶ Beschleunigung
 - ▶ Abbremsen
 bestimmter Rotoren
- ▶ Regelkreis
 - ▶ Rückkopplung über Gyrometer
- ▶ Motorsteuerung
 - ▶ dedizierte Schaltung
 - ▶ per Software
- ▶ Fernsteuerung



Experiment 4: Eisenbahn

- ▶ digital gesteuerte Modelleisenbahn
- ▶ Steuerung über den Trix-Bus
 - ▶ serielle Schnittstelle
- ▶ verschiedene *Ausbaustufen*
 - ▶ festes (1) bzw. variables (2) Schienennetz
 - ▶ feste (3) bzw. variable (4) Anzahl von Zügen
 - ⇒ (1) & (3)
 - ⇒ (1) & (4)
 - ⇒ (2) & (3)
 - ⇒ (2) & (4)
- ▶ mit/ohne Bedienpult

Hiwi: Aufpolieren der EZS-Entwicklungsumgebung
Aufgabengebiet

- Portierung
 - ▶ von EZStubs auf den Nintendo DS
 - ▶ sehr ähnlich zum Gameboy Advance
 - ▶ verwendung des Simulators desmume (<http://desmume.org/>)
- Abgabesystem
 - ▶ Archivierung aller Abgaben
 - ▶ *Testing on demand*: bei Abgabe
 - ▶ *Testing Battle*
 - ▶ wessen Testfälle finden die meisten Bugs
 - ▶ wer hat die schnellste Lösung
 - ▶ wer hat die kleinste Lösung
 - ▶ Generierung von Webseiten
 - ▶ Eintragen von Ergebnissen ins Waffel
- Tutorial
 - ▶ Erweiterung
 - ▶ Illustration
- Rechnerübung
 - ▶ Betreuung

Hiwi: Aufpolieren der EZS-Entwicklungsumgebung

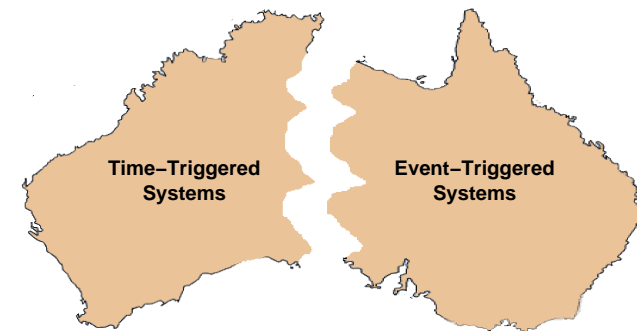
Voraussetzung und Umfang

- Voraussetzung**
- ▶ erfolgreiche Teilnahme an den EZS-Übungen
 - ▶ C/C++ & GNU Tools
 - ▶ HTML & dynamische Generierung von Webseiten
 - ▶ Perl & SQL
 - ▶ Kenntnisse in EZS und BS

- Umfang**
- ▶ x h pro Woche
 - ▶ über y Monate

Atomic Basic Blocks (ABBs)

Abhängigkeiten in Echtzeitsystemen



- ▶ **implizit** sichergestellt
 - ▶ statische Ablaufplanung
- ▶ **explizit** sichergestellt
 - ▶ Schlossvariablen, Semaphore
 - ▶ Nachrichten
 - ▶ ...

Atomic Basic Blocks (ABBs)

Folgen

Fadenabstraktion

- ▶ in taktgesteuerten Systemen: **einfach Ereignisbehandlungen**
- ▶ in vorranggesteuerten Systemen: **komplexe Ereignisbehandlungen**

Portabilität

- ▶ Fadenabstraktionen sind mit der Anwendung verwoben - Fäden ...
 - ▶ sperren Schlossvariablen
 - ▶ versenden Nachrichten
 - ▶ warten auf Signale anderer Fäden
- ▶ oder laufen einfach nur durch (engl. *run-to-completion*)

☞ eine Portierung zwischen Takt-/Vorrangsteuerung ist **sehr schwierig!**

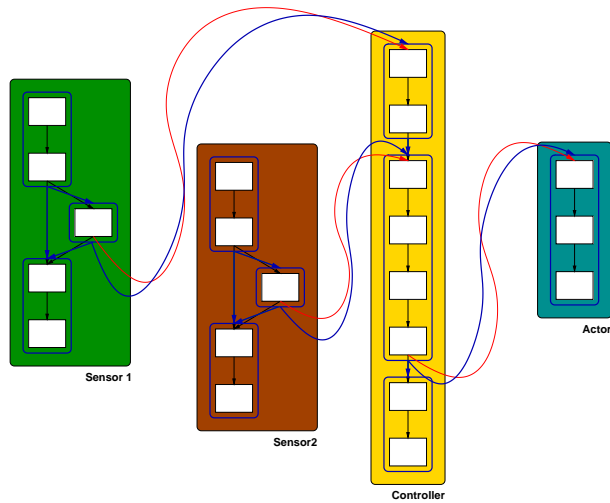
Atomic Basic Blocks (ABBs)

Lösungsidee

- ▶ stelle Abhängigkeiten **unabhängig** von der Fadenabstraktion dar
- ▶ bilde diese Darstellung auf
 - ▶ taktgesteuerte Systeme oder
 - ▶ vorranggesteuerte Systeme ab.
- ▶ fasse Basisblöcke eines CFGs zu sog. **Atomic Basic Blocks** zusammen
 - ▶ ABBs werden durch **verschiedene Abhängigkeitsgraphen** verbunden
 - ▶ Kontrollflussgraphen, Datenflussgraphen, gegenseitiger Ausschluss
 - ▶ ABB-Graphen überspannen **mehrere** Kontrollflüsse
 - ▶ innerhalb eines ABBs existieren **keine** Abhängigkeiten zu anderen Kontrollflüssen

Atomic Basic Blocks (ABBs)

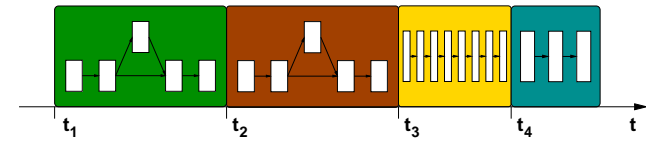
Beispiel



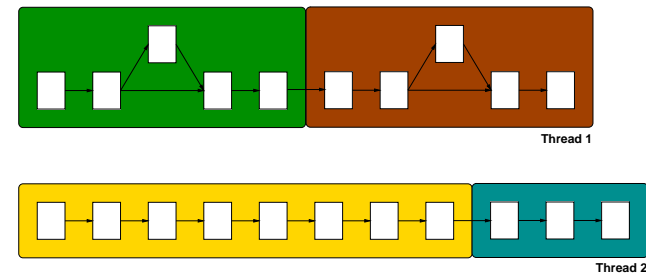
Atomic Basic Blocks (ABBs)

Beispiel

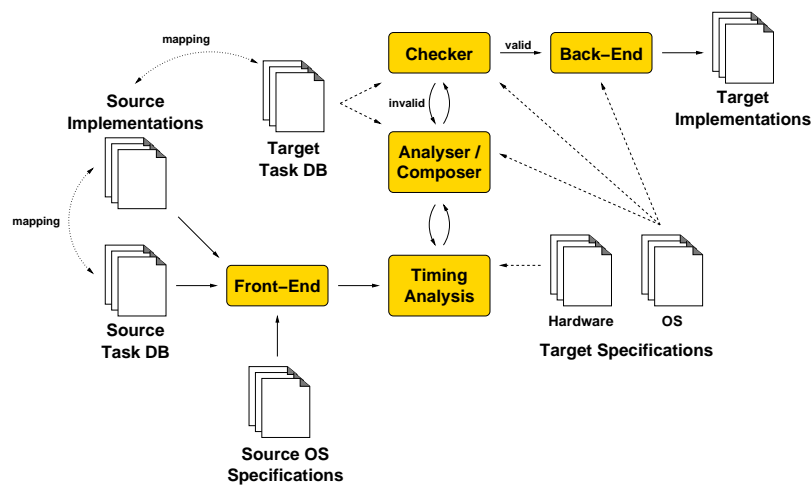
Abbildung auf einen statischen Ablaufplan



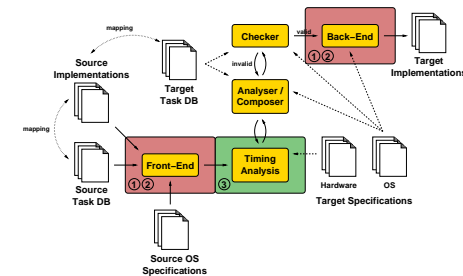
Optimierung in ereignisgesteuerten Systemen



Der Real-Time Systems Compiler (RTSC)

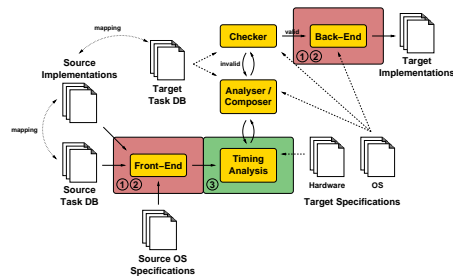


Themen: Diplomarbeiten



1. OSEK OS Front- und Backend für den RTSC
 - ▶ Analyse der OSEK OS API ~> Abhängigkeitsbeziehungen
 - ▶ Codegenerierung ~> Abbildung der Abhängigkeiten auf OSEK OS
2. OSEK ttOS Front- und Backend für den RTSC
 - ▶ Analyse von Abhängigkeiten ~> Annotationen notwendig!
 - ▶ statische Ablaufpläne ~> Abbildung der Abhängigkeiten auf OSEK ttOS

Themen: Studienarbeiten



3. LLVM Backend für TriCore

- ▶ Häufig wird WCET-Information benötigt
- ▶ WCET-Analyse über generierten C-Code ist unbefriedigend

Studien-, Diplom- ... Doktorarbeiten

Forschungs- und Entwicklungsprojekte: Universität, Forschungseinrichtungen, Industrie

weitere Themen im Internet/UnivIS:

<http://www4.informatik.uni-erlangen.de/Theses/>