

Echtzeitsysteme

Ausblick

Lehrstuhl Informatik 4

02. Februar 2011

Gliederung

- 1 Sommersemester
 - Industrievortrag
 - Vorlesungen
- 2 Aktuelle Forschungsarbeiten
 - Aspektorientierte Echtzeitsystemarchitekturen
 - Abschlussarbeiten/Masterprojekte

„Real-Time Aspects in Automation Systems“

Vortragender Dr. René Graf
Siemens AG, Sektor Industry, Vorfeldentwicklung

Arbeitsgebiet eingebettete Systeme, Systemsoftware und
Echtzeitbetriebssysteme in der industriellen
Automatisierung

Zeit Donnerstag, 09.02.2012, 14:00 c.t.

Raum Hörsaal H4, RRZE

- Abstract**
- Teil 1:** Eine kleine Reise durch die (Echt-)Zeit in Automatisierungssystemen. Welche Aufgaben haben welche Anforderungen und Lösungen?
 - Teil 2:** Multi-Core – Single Bus: Der Prozessor ist nicht alles, wenn die Latenz eines Systems bestimmt werden muss.
 - Teil 3:** Transparente Echtzeit unter Linux: Wie kann eine Applikation ohne Änderungen am Code dennoch unter Linux harte Echtzeit im Mikrosekundenbereich erfüllen?

Vorlesung: Verlässliche Echtzeitsysteme

Nicht mehr die Rechtzeitigkeit, sondern die korrekte Funktion steht im Vordergrund

Echtzeitsysteme sind in der Regel sicherheitskritische Systeme
☞ das Wohlergehen von Leib und Leben hat oberste Priorität
~> Fehlfunktionen sind zu vermeiden

Standards & Normen Was schreibt der Gesetzgeber vor?

- Wie muss die Softwareentwicklung aufgebaut sein?
- Was und wie wird begutachtet begutachtet?
- ~> einen „groben Überblick“ vermitteln

Softwareentwicklung ohne „Fehler einzubauen“

- Wie zeigt man die korrekte Funktionsweise von Software?
- ~> statische Analyse, Model Checking, dynamisches Testen, ...

Fehlertoleranz zur Laufzeit

- Wie geht man mit „Bit-Kippen“ um?
- ~> **Fehlertoleranztechniken**

Vorlesung: Verlässliche Echtzeitsysteme (Forts.)

Organisatorisches

Vorlesung

Dozent Fabian Scheler

Wochenstunden 2 Semesterwochenstunden

Raum 01.255-128

Uhrzeit Dienstag, 16:15 - 17:45, Start: 17.04.2012

Übung

Dozenten Martin Hoffmann,
Florian Franzmann,
Isabella Stalkerich

Wochenstunden 2 + 2 Semesterwochenstunden

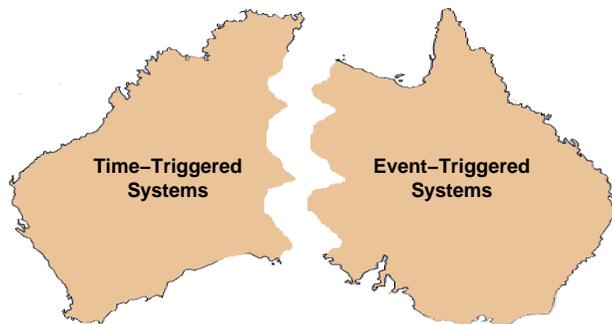
Raum und Zeit siehe UnivIS, nach Vereinbarung

Gliederung

- 1 Sommersemester
 - Industrievortrag
 - Vorlesungen
- 2 Aktuelle Forschungsarbeiten
 - Aspektorientierte Echtzeitsystemarchitekturen
 - Abschlussarbeiten/Masterprojekte

Aspektorientierte Echtzeitsystemarchitekturen

Zeit- und ereignisgesteuerte Echtzeitsysteme sind grundverschieden



... wenn es um die Implementierung von Abhängigkeiten geht

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

Aspektorientierte Echtzeitsystemarchitekturen (Forts.)

☞ Das hat auch Auswirkungen auf die Implementierung von Echtzeitanwendungen!

Fadenabstraktion

- taktgesteuerte Systemen: **einfach Ereignisbehandlungen**
- vorranggesteuerte 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*)

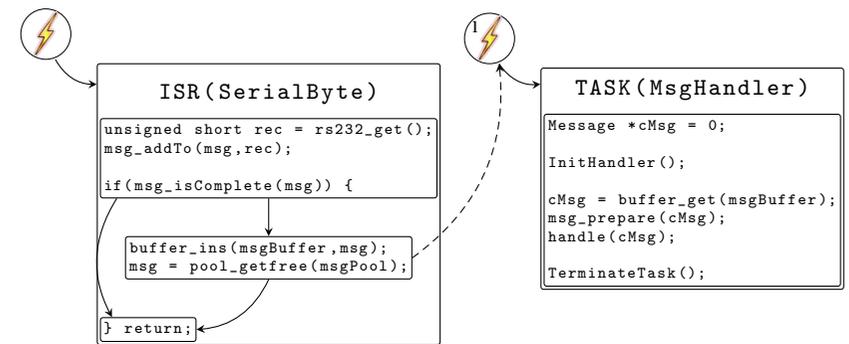
☞ Portierung zwischen Takt-/Vorrangsteuerung ist **sehr schwierig!**

Atomic Basic Blocks (ABBs)

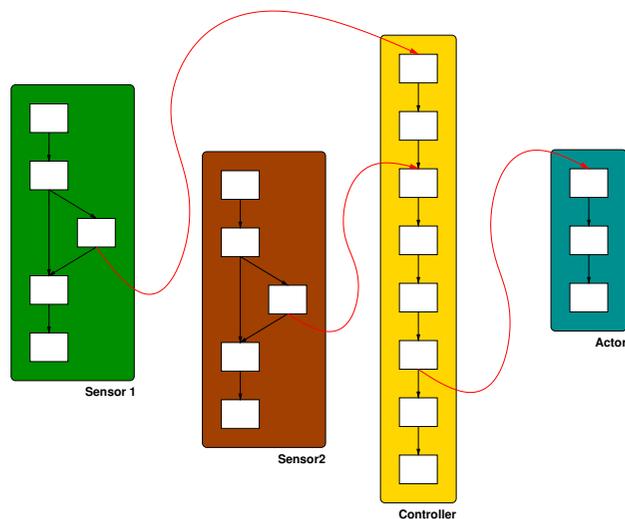
Lösungsidee: Abstrahiere von den Eigenheiten dieser Echtzeitsystemarchitekturen

- stelle Abhängigkeiten **unabhängig** von der Fadenabstraktion dar
- Abbildung auf
 - taktgesteuerte Systeme oder
 - vorranggesteuerte Systeme
- Grundlage: Basisblöcke eines CFGs \rightsquigarrow Atomic Basic Blocks
 - mehrere Grundblöcke werden zu einem ABB zusammengefasst
 - „Basic Blocks“ der erste Namensteil
 - **Abhängigkeiten** verbinden ABBs \rightsquigarrow ABB-Graphen
 - Datenabhängigkeiten, gerichtete und ungerichtete Abhängigkeiten
 - prinzipiell mithilfe der Echtzeitsystemarchitektur implementiert
 - ABB-Graphen überspannen **mehrere** Kontrollflüsse
 - im ABB: **keine** Abhängigkeiten zu anderen Kontrollflüssen
 - das macht sie aus Sicht anderer Kontrollflüsse „atomar“
 - \rightsquigarrow das ist der zweite Teil des Namens

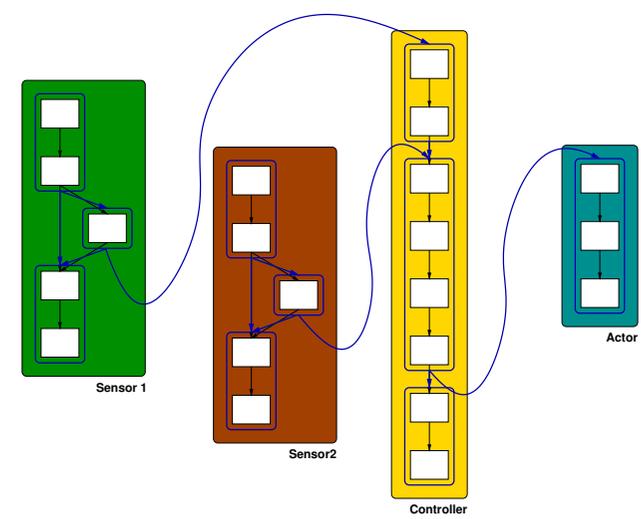
Atomic Basic Blocks (ABBs) — Beispiel



Atomic Basic Blocks (ABBs) — noch ein Beispiel



Atomic Basic Blocks (ABBs) — noch ein Beispiel



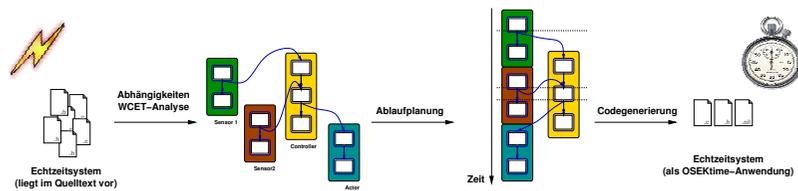
Der Real-Time Systems Compiler (RTSC)

- betriebssystemgewahrer Übersetzer
- vermittelt zwischen zeit- und ereignisgesteuerten Systemen
 - und verwendet dabei ABBs als Zwischendarstellung
- basiert auf der LLVM (Low-Level Virtual Machine)
- Gliederung wie in klassischen Compilern

Front-End Abhängigkeitsgraph erzeugen, WCET-Analyse

Middle-End Transformationen des ABB-Graphen, Ablaufplanung

Back-End Codegenerierung für ein bestimmtes Betriebssystem



Forschungsziele im AORTA-Projekt

Echtzeitbetriebssysteme — Wie sieht die „perfekte“ Schnittstelle aus?

- Semaphore, Mutex, Messages, Flags, Events, ... \leadsto **riesige Auswahl**
 - jedes Betriebssystem bringt seine ganz spezielle Lösung mit
- Welche Abhängigkeiten implementiert man eigentlich damit?
 - Welche Eigenschaften haben die erzeugten Abhängigkeiten?

Ereignisgesteuerte Systeme als Zielplattform — der Weg zurück

- bisher werden nur zeitgesteuerte Zielsysteme unterstützt

Verteilte Systeme bzw. Mehrkernsysteme statt Monoprozessoren

- Verteilung/Ablaufplanung auf mehreren Rechenknoten/-kernen
- Behandlung des Kommunikationssystems

Optimierung übergeordneter Eigenschaften

- z.B. Blockadezeiten durch blockierende Synchronisation

Die „perfekte RTOS-Schnittstelle“

Ausgangspunkt Schnittstellen verschiedener Echtzeitbetriebssysteme

- POSIX (QNX, VxWorks, ...), eCos, Windows CE, AUTOSAR, ...

Fokus: Kontrollflussabstraktion \leadsto Fäden, Unterbrechungen, ...

- Welche Kontrollflussabstraktionen werden angeboten?
- Wie werden sie aktiviert, wie implementieren sie Abhängigkeiten?
 - gerichtete/ungerichtete, synchrone/asynchrone Aufrufsemantik
 - Welche Informationen werden übertragen — Daten, Zeit, Signale

Ergebnis ist eine Studie:

- Welche Abhängigkeiten kann man überall implementieren?
- Gibt es Abhängigkeiten, die nicht implementiert werden können?
- Sind die Schnittstellen orthogonal oder gibt es mehrere Möglichkeiten dieselbe Abhängigkeit zu implementieren?

☞ Untermauernde **Experimente/Beispiele** sind wünschenswert!

RTSC: Multicore Scheduling

- Automobilindustrie: 70 - 100 Steuergeräte je Premium-KFZ

\leadsto **Problem:**

viele Controller \mapsto viele Busse \mapsto

viel Kupfer \mapsto viel Gewicht \mapsto **viel Verbrauch**

\leadsto **Lösung:** Konsolidierung z.B. durch Mehrkernprozessoren, **aber:**

(1) Hardwareanforderungen – I/O

(2) Programmierung von Mehrkernprozessoren ist schwierig

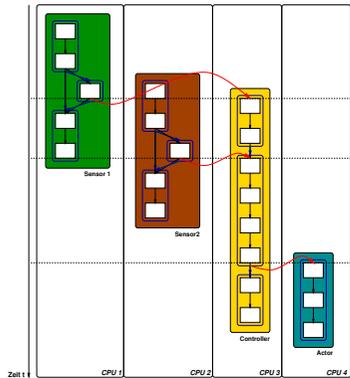
- Idee: zumindest bei (2) könnte der RTSC helfen :-)

- automatisch Abbildung von ABB-Graphen auf
 - Mehrkernsysteme bzw. verteilte Systeme

- Implementierung eines existierenden Algorithmus
 - Peng, Shin und Abdelzaher (1997)
 - statische Allokation von *Modulen* (\approx ABBs) auf Rechenknoten

\leadsto Voraussetzung ist ein Verständnis der Abhängigkeiten!

RTSC: Multicore Scheduling (Forts.)



- Abbildung auf verschiedene Knoten
 - Annahme einer globalen Zeitbasis
- Berücksichtigung von Kommunikation
 - gleicher Knoten
 - ~ gemeinsamen Speicher: Variablen
 - entfernter Knoten
 - ~ Nachrichten: TTEthernet/TTCAN
- Unterstützung von TTEthernet
 - ~ weiteres Thema

TTEthernet – Portierung auf TC1797/WIZnet W5300

TTEthernet zeitgesteuerte Kommunikation auf Ethernet-Basis

- Netzzugangsprotokoll: TDMA ~ Uhrensynchronisation in Software
- Implementierung auf Ebene 2 des ISO/OSI-Schichtenmodells
 - ~ Ethernet-Pakete lesen/schreiben

Entwicklungssystem mit TTEthernet-Switch und 2 Clients vorhanden

- TTEthernet-Protokoll-Implementierung verfügbar
 - ~ 1. Schritt: Inbetriebnahme des TTEthernet-Systems

TriCore als Zielplattform

- mit Betriebssystem (Ciao OS/eCos) oder „bare metal“
- Entwicklung mit GCC, GDB, Lauterbach Trace32
 - ~ 2. Schritt: Inbetriebnahme TriCore

3. Schritt: Portierung des Linux-Treibers

- Elementaroperationen im Linux-Treiber finden und verstehen
 - ~ Transfer in die TTEthernet-Protokollschicht

Studien-/Diplom-/Bachelor-/Master- ... Doktorarbeiten

Forschungs- und Entwicklungsprojekte: Universität, Forschungseinrichtungen, Industrie

weitere Themen im Internet/UnivIS:

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

