Sichere Multifunktionsintegration in Multi-Core basierten Cyber-Physical Systems

Dipl. Inf. Robert Hilbrich

Systemarchitekturen (STAR)

Forschungsleiter Embedded Multicore

Fraunhofer FIRST

FIRST IM ÜBERBLICK

- Gegründet 1983 als GMD-FIRST
- Seit 2001 Institut der Fraunhofer-Gesellschaft
- Gesamthaushalt 2010: 7,9 Mio. Euro
- Externe Finanzierung: 81 %
- Mitarbeiterinnen und Mitarbeiter: 143

Fraunhofer FIRST

ABTEILUNGEN

Künstler: Bertrand Freiesleben

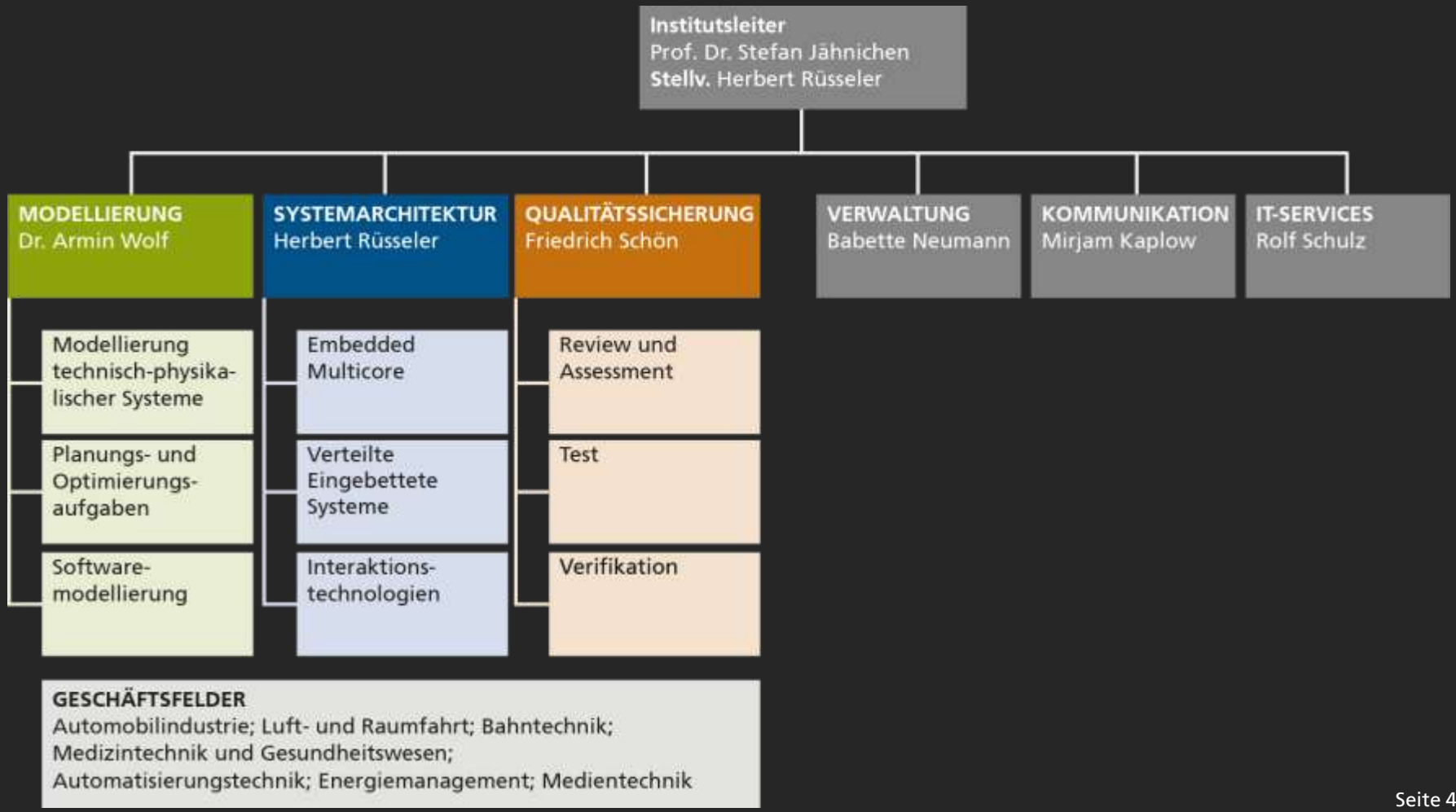
MODELLIERUNG

ARCHITEKTUR

QUALITÄTSICHERUNG

Fraunhofer FIRST

ORGANIGRAMM



Fraunhofer FIRST

Agenda

1. Embedded Systems werden zu Cyber-Physical Systems
2. Single-Cores werden zu Multi-Cores
3. Nutzung von Multi-Cores in Cyber-Physical Systems
4. „*Correctness-by-Construction*“ für die Verteilung
5. „*Correctness-by-Construction*“ konkret:
   - Mapping
   - Scheduling
   - Validierung
   - Timing Analyse
6. Zusammenfassung

Embedded Systems werden zu

Cyber-Physical Systems (CPS)

Fraunhofer FIRST

Cyber-Physical Systems – eine neue Generation eingebetteter Systeme

- Cyber ~ Kybernetes – „Steuermann“
- Physical – physikalischer Prozess
- Helen Gill, National Science Foundation, 2006

Bestandteile („System-of-Systems“ Gedanke)

- Steuergeräte
- Kommunikationsverbindungen
- Feedback-Schleifen (Sensoren und Aktoren)

Herausforderung

- Gemeinsame Betrachtung von Rechentechnik und physikalischen Prozessen

Neue Herausforderungen in der Entwicklung von Cyber-Physical Systems

- Anstieg der Funktionskomplexität und gleichzeitige steigende Anforderungen bzgl. Zuverlässigkeit, Robustheit und Safety
- Heterogenität der Subsysteme
- Eigenschaften physikalischer Prozesse
  - Massive Parallelität
  - Enger Bezug zur Echtzeit
- „Openness“ und Komponierbarkeit
- Qualitätssicherung

Cyber-Physical Systems – eine neue Systemwissenschaft?

- Cyber-Physical Systems sind nicht einfach „neue Geräte“ oder die Ergänzung von Kommunikation zu Embedded Systems
- Sondern:
  - gemeinsame Betrachtung von Kommunikation und Steuerung
  - Kritik an aktuellen Entwicklungsmethoden

- Bisherige Ausrichtung von Systemwissenschaften:
  - Physikalische Prozesse (Signalverarbeitung, …)
  - Digitale Rechentechnik und Berechenbarkeit (Komplexitätslehre, …)

- Hybride Ansätze noch in den Kinderschuhen

[CPS is a] “*new science, which is jointly physical and computational.*"
(Edward A. Lee, 2006)

Single-Core Prozessoren werden zu

Multi-Core Prozessoren

„No one knows how to design a 15 GHz processor, so the other option is to re-train all software developers" [to do parallel programming].
(Prof. D. A. Patterson, Berkeley)

Vom Single-Core zum Multi-Core Prozessor

Software findet immer einen Weg, mehr Leistung der Hardware „sinnvoll“ zu verwenden.

- Mehr Funktionalität
- Mehr Performance
- Wettbewerbsvorteil!

Bisher - Leistungssteigerung bei Single-Core Prozessoren durch:

- Erhöhung der Taktfrequenz
- Erhöhung der Leistung
- Komplexere Logik (~ mehr Transistoren)
- Kleinere Strukturbreiten

Software profitiert automatisch von der Leistungssteigerung.

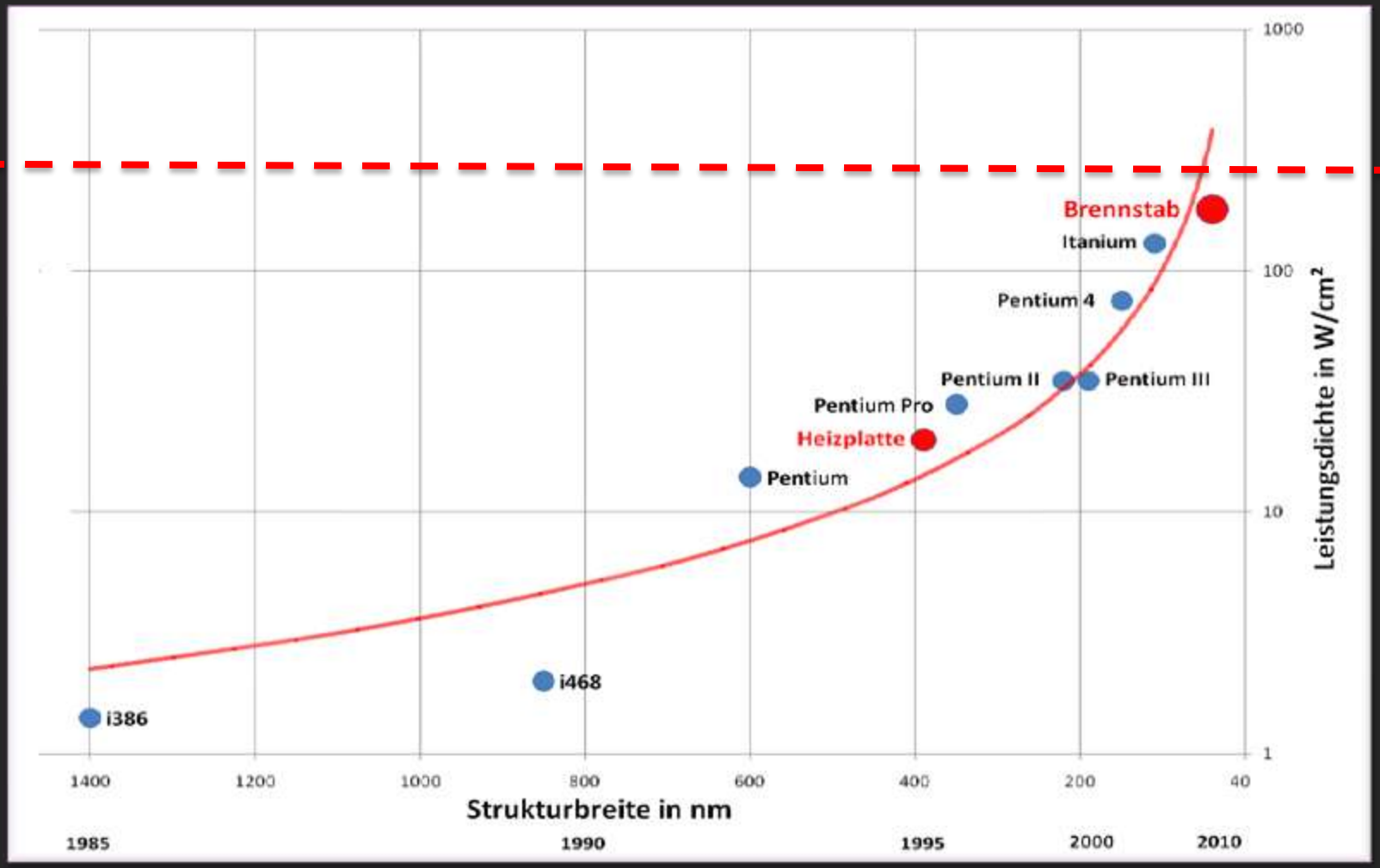
Vom Single-Core zum Multi-Core Prozessor

"*The number of transistors on an integrated circuit increases exponentially, doubling approximately every two years*"

"Moore's Law" (1965, Gordon E. Moore, Intel)

Extrapolation der Single-Core Entwicklung:
Es müssten bereits Prozessoren mit 10 GHz bis 15 GHz erhältlich sein.

Die Power-Wall



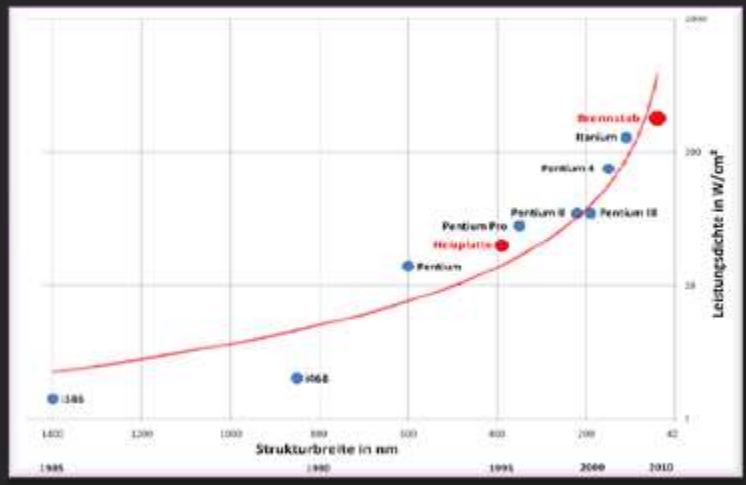
Fraunhofer FIRST

Die Power-Wall

- Überproportionale Steigerung der Leistungsaufnahme
  - 1% höherer Takt führt zu 3% höhere Leistungsaufnahme

- Multi-Core = Mehrere Ausführungseinheiten ("Cores") auf einem Chip

- Aktuell einzige(!) Möglichkeit zur Leistungssteigerung der Software durch "schnellere" Hardware



Fraunhofer FIRST

Multi-Core - ein Déjà Vu?

- Technische Unterschiede
  - Inter-Core Kommunikationslatenz / -bandbreite
  - Art der Speicheranbindung
- Neue Domänen, z.B. Embedded Systems
- Neue Anforderungen, z.B.
  - Echtzeitfähigkeit, Zertifizierbarkeit, …
  - Zuverlässigkeit, Power-Management, …

Herausforderungen der Parallelverarbeitung kommen auch auf Embedded Systems zu!

Multi-Core Prozessoren für Eingebettete Systeme

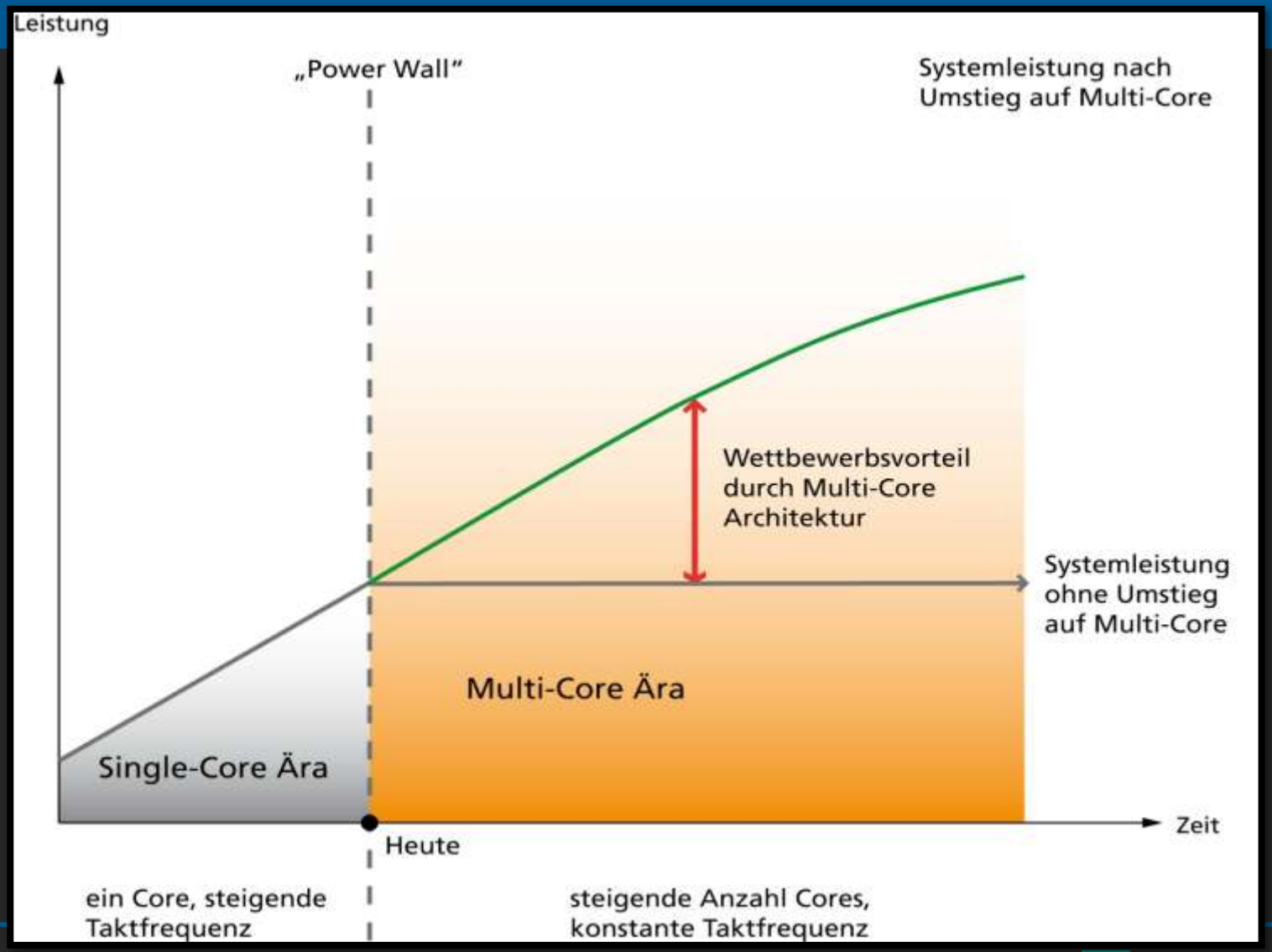
Vorteile von Multi-Core Prozessoren:

- Konsolidierung von Single-Core Systemen
- Weniger Zertifizierungsaufwand*
- Zuverlässige, schnelle Inter-Core Kommunikation
- Reduzierter Platzbedarf, geringeres Gewicht und geringere Leistungsanforderungen ("*reduced SWaP*")
- Geringere Herstellungskosten – Wettbewerbsvorteil!

Aber: Folgen für die Software-Entwicklung:

- Software muss die Parallelität nutzen können und skalierbar sein
- Paradigmenwechsel von der sequenziellen zur parallelen Entwicklung für **jeden Nutzer** von Multi-Core Prozessoren → „Altlasten"?

Parallele Programmierung – ein Paradigmenwechsel



Fraunhofer FIRST

Nutzung von

Multi-Core Prozessoren in Cyber-Physical Systems

Fraunhofer FIRST

Cyber-Physical Systems und Multi-Core Prozessoren

Bisher:

- „Cyber-Physical Systems“ beschreiben Trends & Herausforderungen im Embedded Bereich für die nächsten Jahre
- Multi-Core Prozessoren stellen eine (die einzige?) technologische Plattform für Cyber-Physical Systems dar

Herausforderung für das System Engineering:

- Entwicklung von kommunizierenden, verteilten, Echtzeitsystemen auf Multi-Core Basis

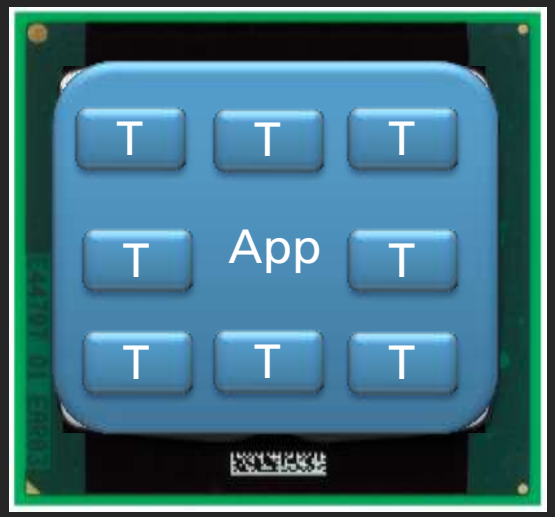
Jetzt:

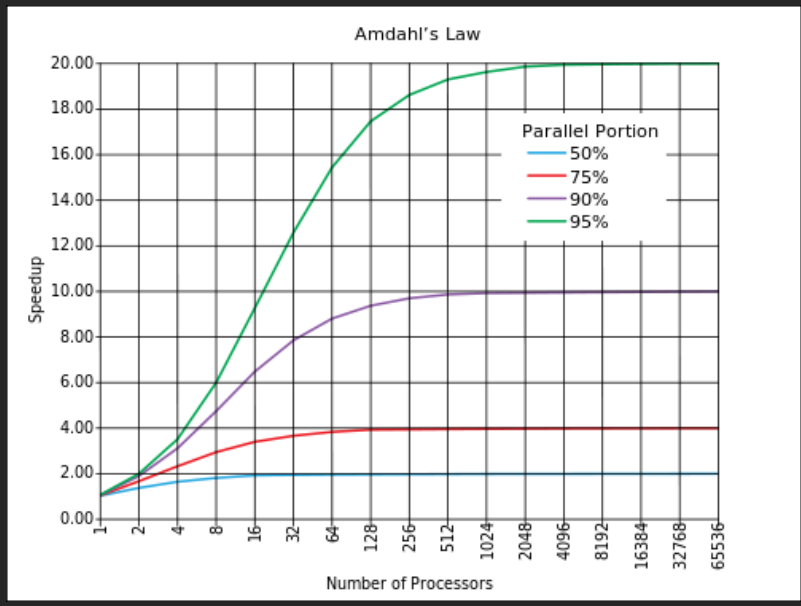
Wie lassen sich Multi-Cores überhaupt sinnvoll nutzen? (Migrationsstrategien)

Strategie 1: „Single Core“ Ansatz

- Geringster Migrationsaufwand
- Software läuft fast ohne Änderungen weiter
- Keine Ausnutzung des Leistungspotentials von Multicores
- Zukünftig sinkende Performance (Many-Core!)

Strategie 2: „1 App – 1 Multi-Core Prozessor“



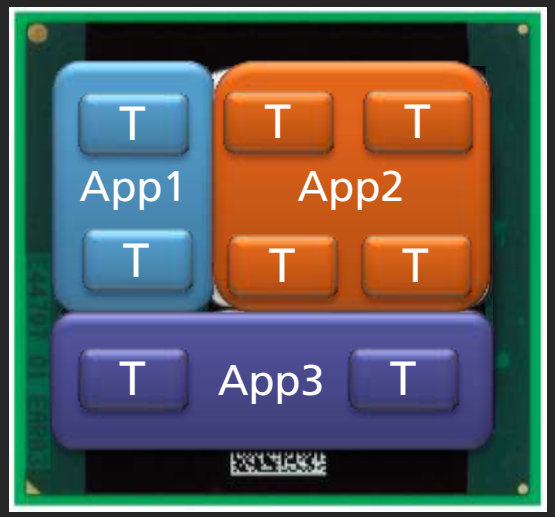


- Nutzung paralleler Rechenkerne
- Etablierte Design-Patterns (insb. für Echtzeit!) funktionieren nicht mehr
- Parallelisierung von Anwendungen ist aufwändig, fehlerträchtig und nicht immer möglich (State-Machines, Regelalgorithmen, ...)
- Skalierbar bei steigender Core-Anzahl?

“The bearing of a child takes nine months, no matter how many women are assigned”
(F. Brooks)

Fraunhofer FIRST

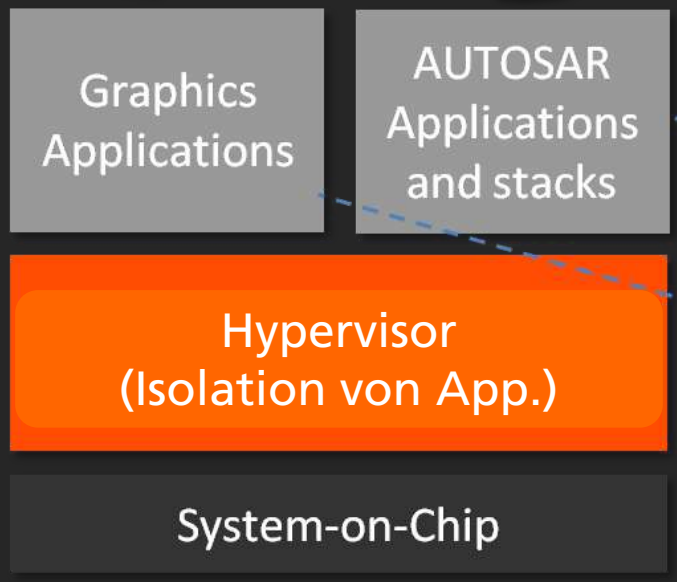
Strategie 3: „Multifunktion-Integration“



- Ausnutzung der Parallelität auf zwei Ebenen
- Beste Nutzung der Potentials von Multi-Core Prozessoren
- Multikritikalität von Applikationen –
  Isolation von Applikationen - Common Cause Fehler?
- Aufwändige Design- und Integrationsphase –
  technisch und organisatorisch(!)

Beispielarchitektur – Multifunktionsintegration mit versch. Kritikalitäten

VirtuOS Projekt: Architekturen für sichere, automotive Softwaresysteme



- **AUTOSAR:** Nutzung existierender AUTOSAR Applikationen und Basis-Software (Kommunikations-Stacks)
- **Graphics:** Zugriff auf Grafiksubsysteme des SOC

Fraunhofer FIRST

Herausforderung bei der Multifunktionsintegration – Sichere Verteilung von Funktionen

Domain-spezifische Trends:

- Standardisierung von Abstraktionsschichten der Hardware
  - Automotive: AUTOSAR
  - Avionik: Integrated Modular Avionics + ARINC 653
- Software wird „verteilbar“

Die neuen Herausforderungen im Systems Engineering für CPS:

- Design-Space Exploration für Hardware-Architekturen
- Verteilung von Applikationen auf Hardware-Architekturen unter Beachtung von:
  - Funktionalen Anforderungen (Sensorik, Kommunikationstechnik, …)
  - Echtzeitanforderungen, Safety- und Zuverlässigkeitsanforderungen
  - Kosten

trotz hoher System-Komplexität!

Entwicklung einer Verteilung mit Hilfe von „Correctness-by-Construction“

Fraunhofer FIRST

Eine Verteilung im „Systems Engineering“

- Verteilung =
  Abbildung zwischen Software und Hardware („Ressourcen“)
  - Optimale Ressourcennutzung
  - Beste Performance
  - Zuverlässigkeit,
  - …

- Örtliche Verteilung: „Mapping“
- Zeitliche Verteilung: „Scheduling“

- Verteilung bestimmt das Systemverhalten(!)
- Sicherheitskritische und zertifizierungsrelevante Komponente
- Hohe Komplexität, da viele Freiheitsgrade und viele Anforderungen

Aus der Praxis heute …

- „Praxis-erprobter“ Ansatz zur Bestimmung einer Verteilung:
  1. Initiale Verteilung
     - Copy-and-Paste aus Vorgänger-Projekten
     - Experten-Wissen
  2. Analyse von Kosten, Timing- und Safety-Eigenschaften
  3. Modifikation der Verteilung, dann Schritt (2)

- *„Timing-by-Accident“*
- Vorgänger-Projekte bzw. Expertenwissen bei neuen Multi-Core Systeme?

- „Analytische Verfahren“ für Cyber-Physical Systems nicht mehr sinnvoll
  - Hohe Komplexität
  - Bestimmung einer initialen Verteilung notwendig

Entwicklung sicherheitskritischer Software

- … war in der Vergangenheit mit ähnlichen Herausforderungen konfrontiert

- Software nach ihrer Erstellung auf Fehler zu analysieren ist sehr aufwändig (Komplexität!)

- Neuer Ansatz: „*Correctness-by-Construction*“ [Hall 2002]
- Idee:
  - Der Prozess der Erstellung muss die „Fehlerfreiheit“ sicherstellen
  - Software wird aus den Anforderungen „konstruiert“
  - Formaler Regelsatz

[Amey 2002], [Bordin 2007], [Chapmann 2006], [Hall 2002], [Resmerita 2011], …

„Correctness-by-Construction“ für die

Sichere Verteilung

Fraunhofer FIRST

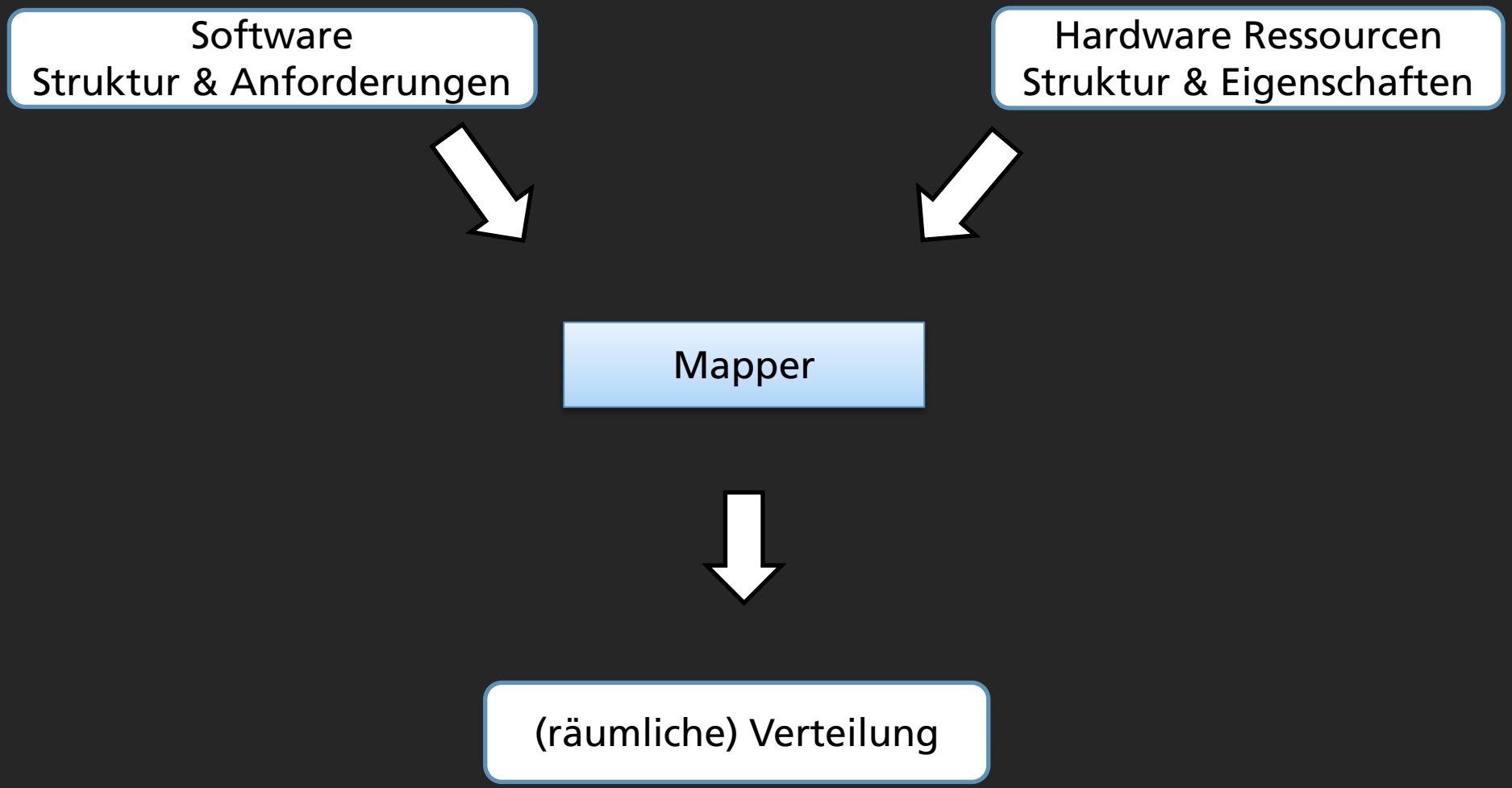
Konstruktion statt Analyse für eine Verteilung

- „Freiheitsgrade“ einer Verteilung sind enorm – nicht mehr sinnvoll manuell zu „durchforsten“
- Klassische Ansätze nicht mehr sinnvoll einsetzbar („timing by accident“)

- „Correctness-by-Construction“ für die Entwicklung einer korrekten Verteilung:
  - Anforderungen explizieren
  - Design-Entscheidungen formalisieren und automatisieren
  - Bewertung von Ergebnissen
  - Garantie der Anforderungserfüllung für konstruierte Ergebnisse

[Hilbrich 2011], [Hilbrich 2012]

Örtliche Verteilung: „Mapper“



„Mapper“ im Detail

Mapper

- Räumliche Verteilung = „Mapping“
  beeinflusst von harten und weichen Kriterien

- Harte Kriterien („Korrektes“ Mapping):
  - Verfügbare Ressourcen (Echtzeit!)
  - Safety-Anforderungen
  - Topologische Anforderungen
- Weiche Kriterien (Besseres Mapping):
  - Kosten
  - Kommunikations-Overhead

- Gleichzeitige Betrachtung und Optimierung von harten und weichen Kriterien

Beispiel: Verteilung von Anwendungen auf komplexe Avionik-Architekturen

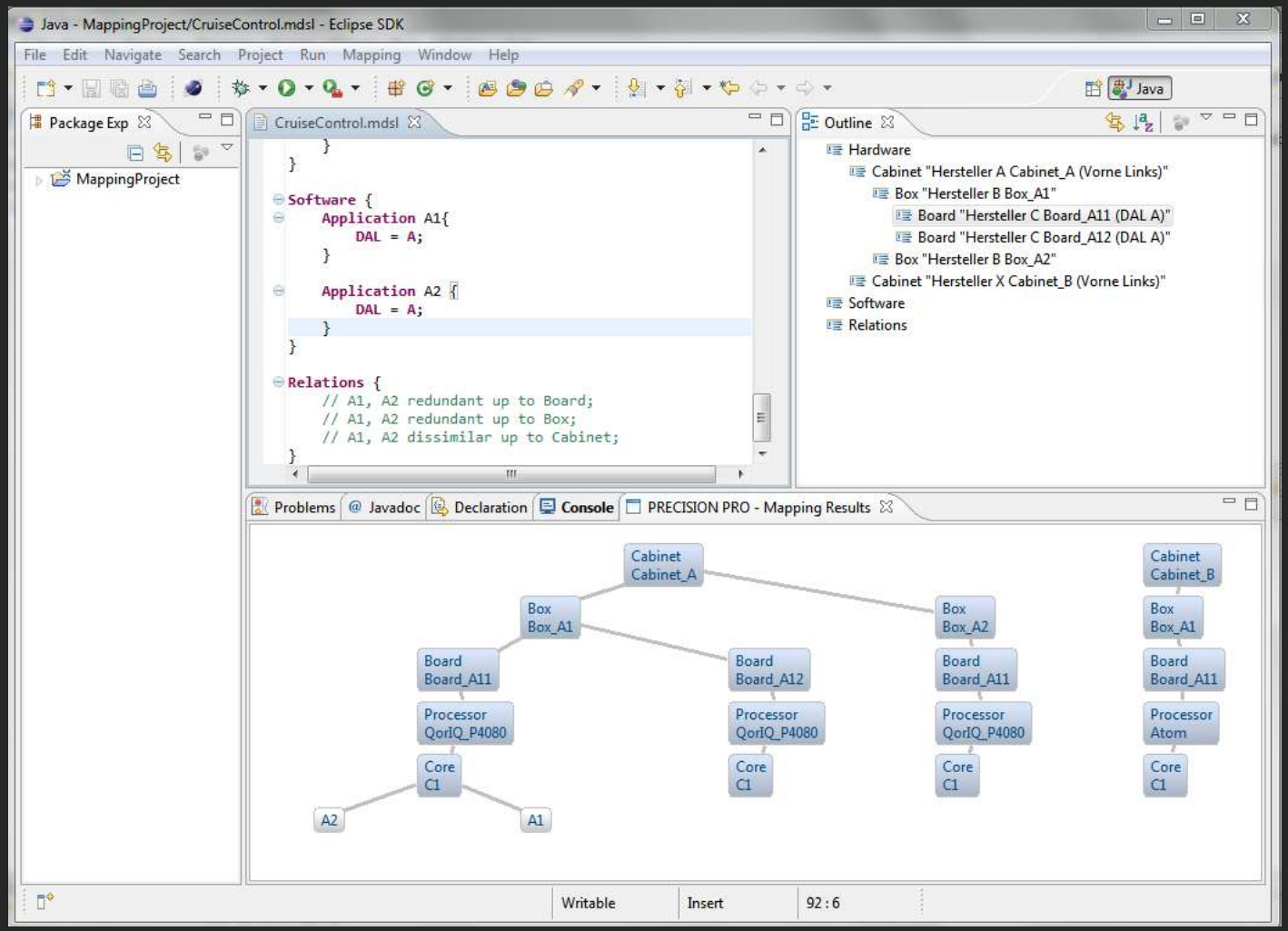
Herausforderung:

- Mehrstufige Avionik Hardware-Architekturen
  - Cabinets, Boxes, Boards, Prozessoren, Cores
- Ca. 1000 Applikationen und ca. 100 Prozessoren
- Effiziente Verteilung finden, so dass
  - Zuverlässigkeitsanforderungen erfüllt werden
  - Ressourcen nicht überbucht werden und
  - die Kommunikation minimiert wird.

Ansatz:

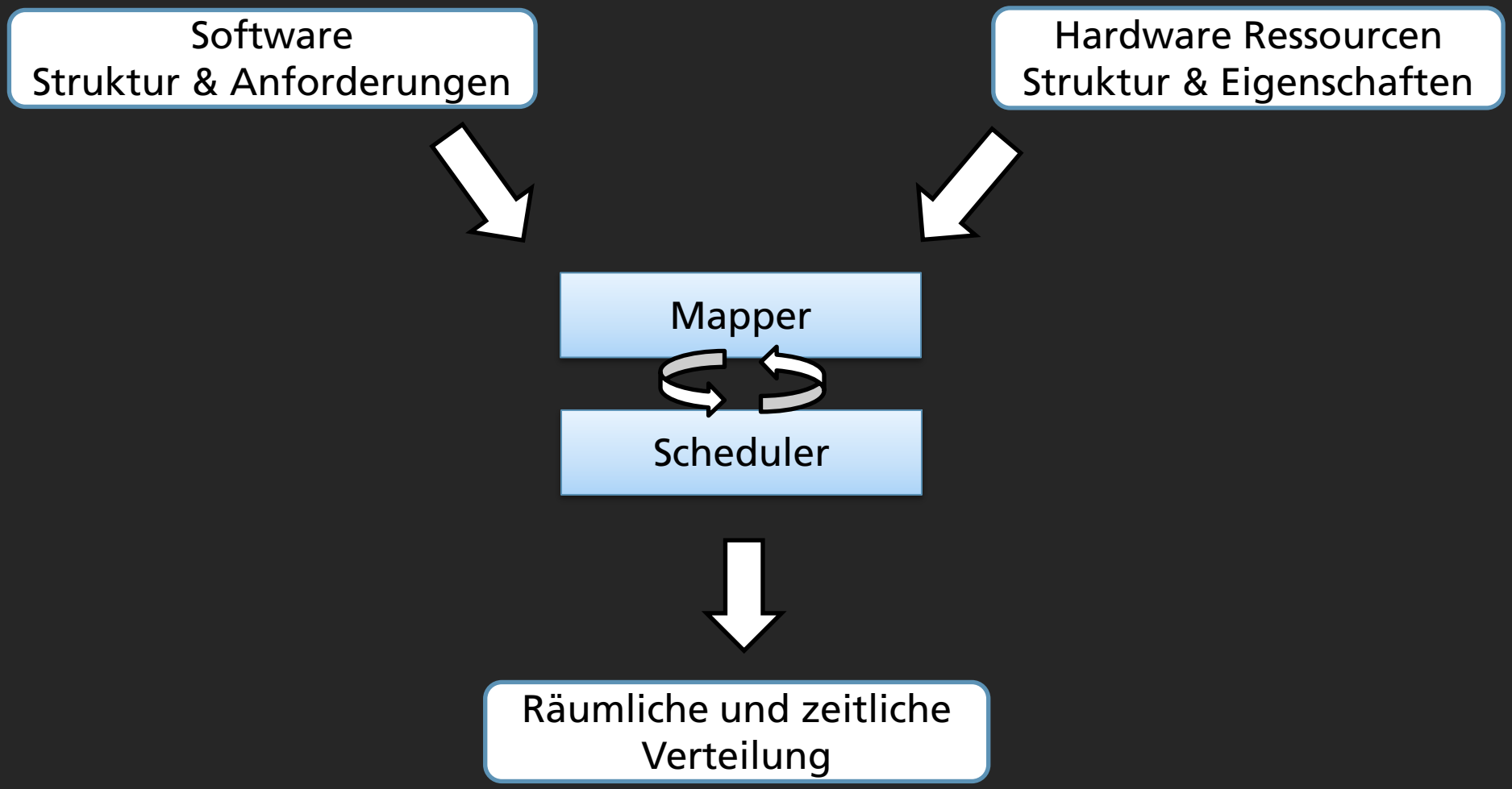
- Spezifikation des Hardware-Angebotes
- Spezifikation der Software-Anforderungen
- Konstruktion einer korrekten Verteilung

„Mapper“ - DEMO



Fraunhofer FIRST

Zeitliche Verteilung: „Scheduler“



„Scheduler“ im Detail

Scheduler

- Ablaufplanung für sicherheitskritische Systeme
- Engineering des vollständigen Laufzeitverhaltens eines Prozessorsystems zum Zeitpunkt der Entwicklung
  - Periodische Tasks
  - Inter-Task Abhängigkeiten
- Statischer, offline Schedule für maximale Vorhersagbarkeit
- Abhängigkeiten zu anderen gemeinsam benutzten Ressourcen („*task synchronisation by static scheduling*“)

Beispiel: “Adaptive Cruise Control”

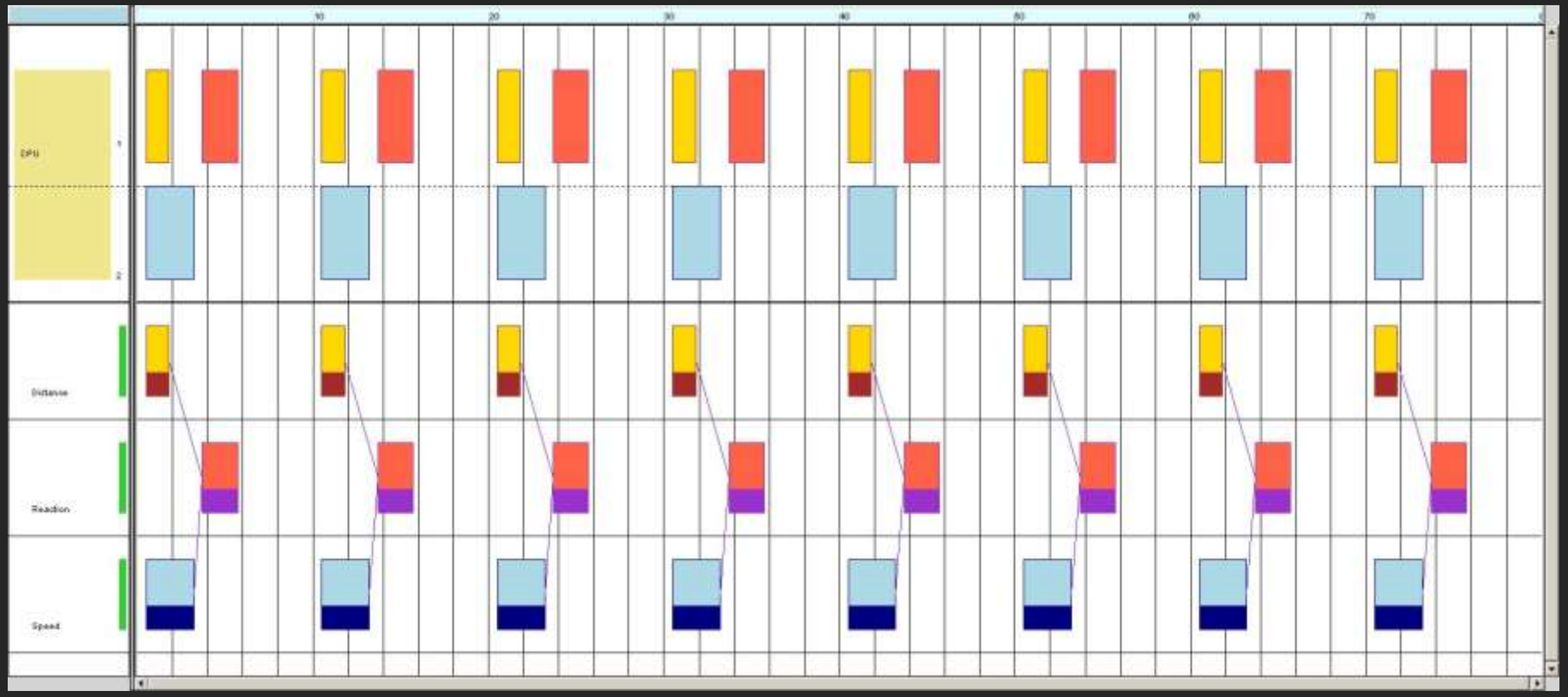
Die (vereinfachte) Funktion bestehe aus 3 Tasks:

- Task 1: “Abstandsmessung"
- Task 2: “Aktuelle Geschwindigkeit"
- Task 3: “Reaktion (Bremsen)"

Notwendige Spezifikation:

- Hardware: Dual-Core Architektur (Core 1, Core 2)
- Software-Abhängigkeiten: Task 3 ('Reaction') benötigt Task 1 und 2
- Timing: Task 1: **1.3ms**, Task 2: **2.7ms**, Task 3: **2.0ms**
- Mapping: Task 1 and Task 3 → Core 1,
  Task 2 → Core 2

“Scheduler” - Demo

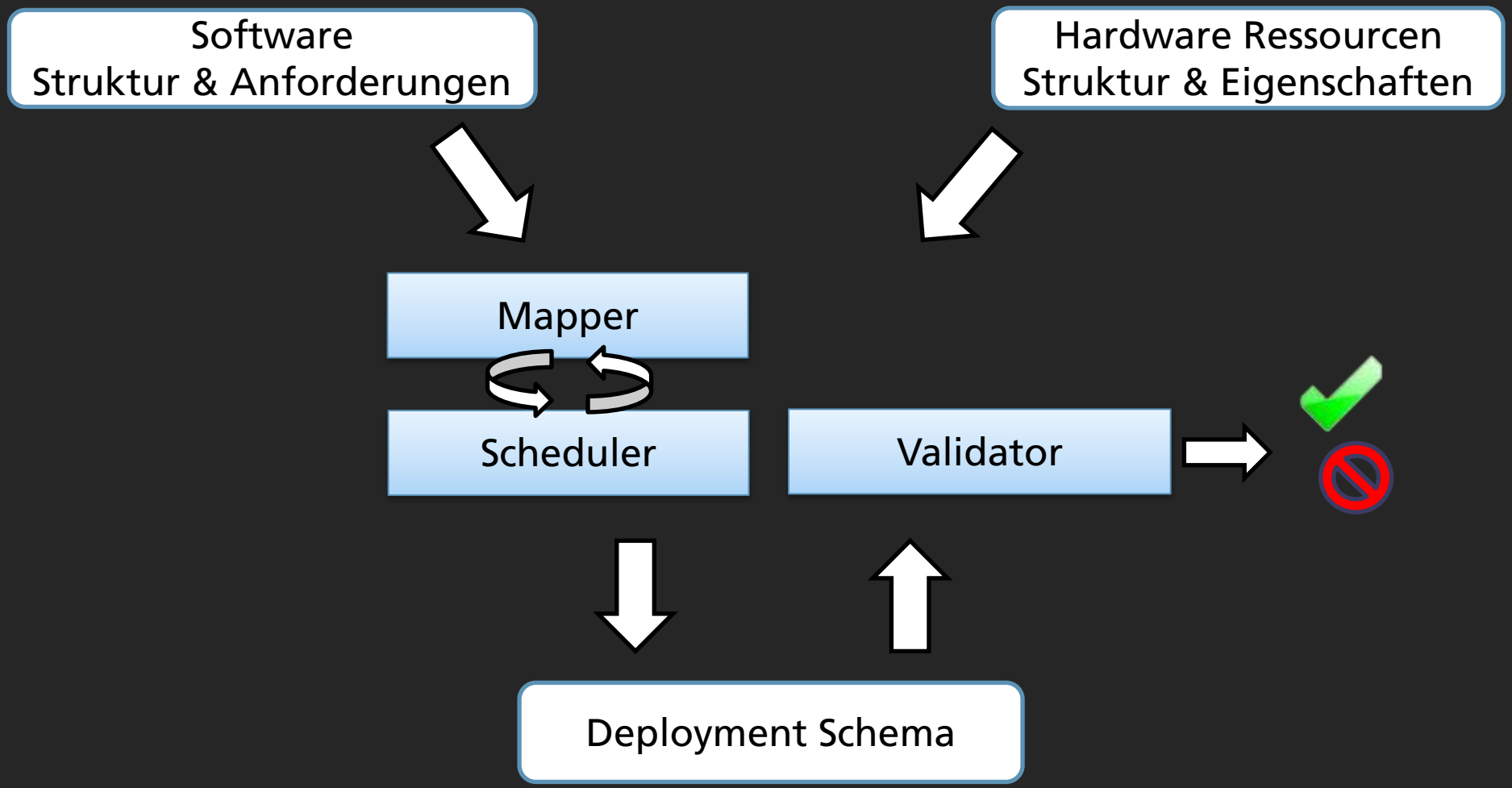


Fraunhofer FIRST

Konstruktion von Schedules

- Kurze Entwicklungszeiten für die Erzeugung von Schedules
  - automatisch und interaktiv (schrittweise, manuell)
- Konstruktion fehlerfreier und ausführbarer Schedules
  - die Berücksichtigung aller Anforderungen wird garantiert
- frühe Validierung des Softwaredesigns und Bewertung von verschiedenen Multi-Core Architekturen durch erfolgreiche/ gescheiterte Konstruktion von Schedules
- Visualisierung und informative Auswertung der aktuellen Schedules
  - Schedules können grafisch mit benutzerdefiniertem Detaillierungsgrad dargestellt werden

Validator

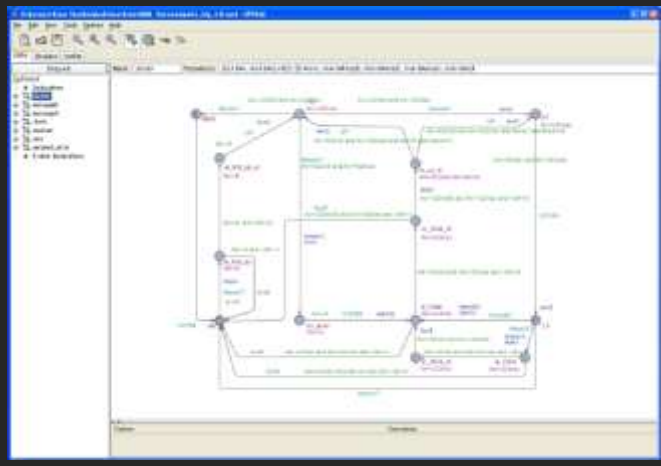


Fraunhofer FIRST

Validator im Detail

Eine dissimilare Technik für die Überprüfung des generierten Schedules.

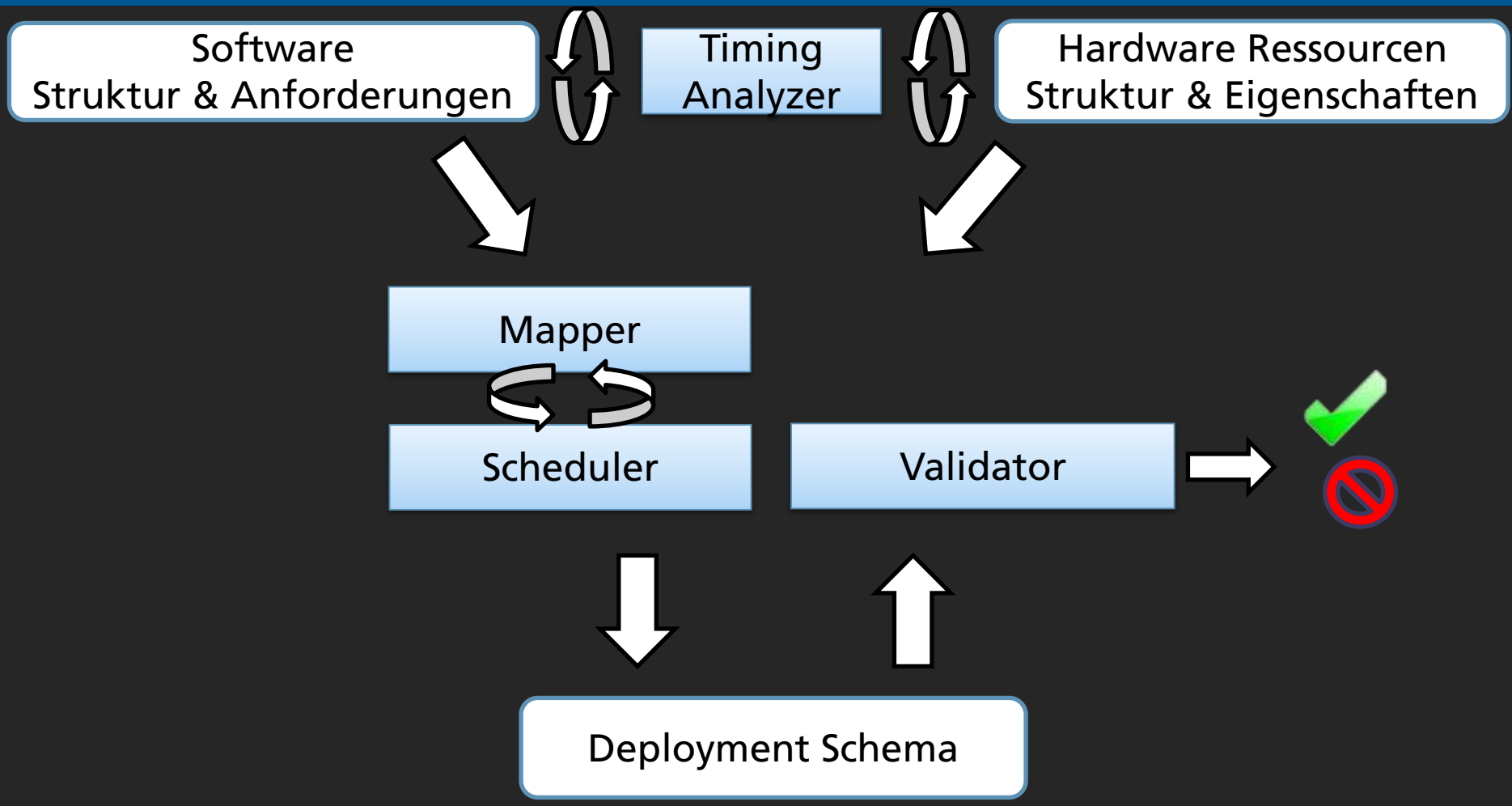
Schedule Validator



- Basiert auf Model Checking mit UPPAAL
- Erstellt automatisch ein formales Modell aus dem generierten Schedule
- Erlaubt die Prüfung von Modell-Eigenschaften
  z.B.:
  "*Ist es immer wahr, dass Task 1 mindestens 1.3ms CPU Zeit bekommt?*"

Fraunhofer FIRST

Empirische Timing Analyse (Work in Progress)



Fraunhofer FIRST

Publikationen und

Zusammenfassung

Fraunhofer FIRST

Zusammenfassung

- Cyber-Physical Systems erhöhen die Komplexität und erfordern neue Entwicklungsverfahren
- Multi-Core Prozessoren bieten mehr Leistung, erfordern aber einen "aktiven Umstieg"
- Herausforderung für das Systems Engineering: Korrekte Verteilung von Software auf Multi-Core Prozessorsysteme
- "Correctness-by-Construction" expliziert und automatisiert Design-Entscheidungen

Q&A

?!

Robert Hilbrich
robert.hilbrich@first.fraunhofer.de
www.first.fraunhofer.de

Fraunhofer FIRST

Publikationen

Robert Hilbrich et al. 2012. **Modeling of Quality Aspects: Real-Time** in The SPES 2020 Engineering-Methodology for software-intensive Embedded Systems – Challenges, Principles, and Application, Klaus Pohl, Reinhold Achatz, Harald Hönninger, Manfred Broy, Springer 2012, pages 113-122 – to appear.

Robert Hilbrich 2012. **How to Safely Integrate Multiple Applications on Embedded Many-Core Systems by Applying the "Correctness by Construction" Principle** in Journal: Advances in Software Engineering, pages 1-24, to appear.

Robert Hilbrich et al. 2011. **Modellbasierte Generierung von statischen Schedules für sicherheitskritische, eingebettete Systeme mit Multicore Prozessoren und harten Echtzeitanforderungen**, Workshop Echtzeit 2011 (Boppard, Deutschland), 03./04.11.2011.

Robert Hilbrich, Matthias Gerlach 2011. **Virtualisierung bei Eingebetteten Multicore Systemen: Integration und Isolation sicherheitskritischer Funktionen**, "Virtualisierung – gestern, heute und morgen" auf der 41. Jahrestagung der Gesellschaft für Informatik.