Treffen der ASIM/GI-Fachgruppen

„Simulation technischer Systeme“
und
„Grundlagen und Methoden in Modellbildung und Simulation“

München, 20.-21. Februar 2006

Ort

Fachhochschule München, Lothstraße 64, Gebäude R

Veranstalter

ASIM/GI-Fachgruppen und Fachhochschule München

ASIM Mitteilung 99

Informationen zu ASIM und den Fachgruppen:

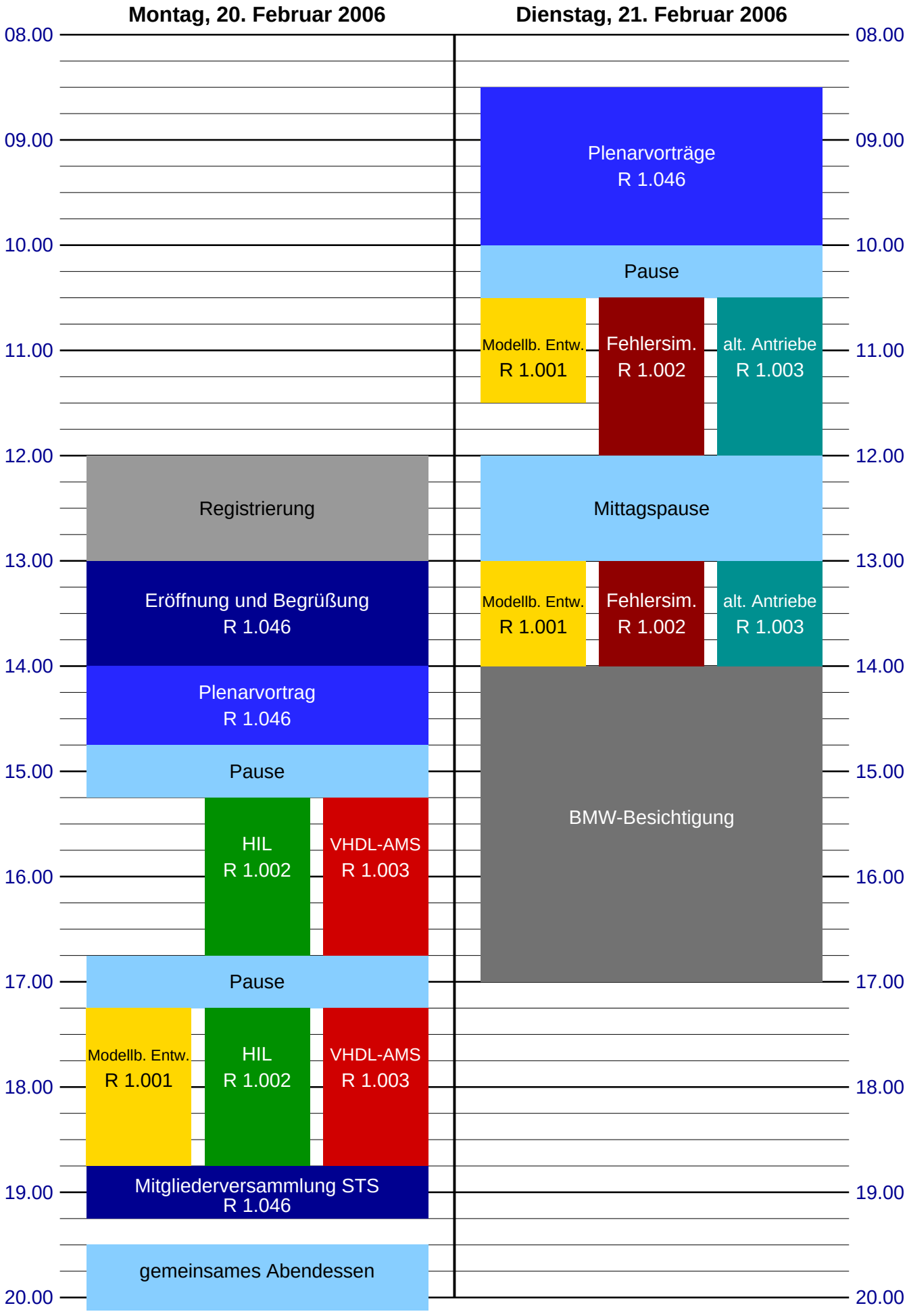
www.asim-gi.org

Ziel des Treffens

Das Treffen soll den Informations- und Erfahrungsaustausch zwischen Fachleuten auf dem Gebiet der Simulation technischer Systeme fördern (Grundlagen, Methoden, Praxisbeispiele, Werkzeuge). In einer begleitenden Ausstellung werden Werkzeuge und Dienstleistungen zur Modellbildung und Simulation vorgestellt.

Stand: 14. Februar 2006

Zeitplan


Montag, 20. Februar 2006
Dienstag, 21. Februar 2006
08.00
09.00
10.00
11.00
12.00
13.00
14.00
15.00
16.00
17.00
18.00
19.00
20.00
Plenarvorträge
R 1.046
Pause
Modellb. Entw.
R 1.001
Fehlersim.
R 1.002
alt. Antriebe
R 1.003
Registrierung
Mittagspause
Eröffnung und Begrüßung
R 1.046
Plenarvortrag
R 1.046
Pause
HIL
R 1.002
VHDL-AMS
R 1.003
BMW-Besichtigung
Pause
Modellb. Entw.
R 1.001
HIL
R 1.002
VHDL-AMS
R 1.003
Mitgliederversammlung STS
R 1.046
gemeinsames Abendessen

Montag, 20. Februar 2006 — Programm

13.00 – 14.00 — **Eröffnung und Begrüßung**
Hörsaal R 1.046 — *Sitzungsleitung: Dr. Ingrid Bausch-Gall*

- Begrüßung und Vorstellung der Fachhochschule München
  Prof. Gerhard Barich, Vizepräsident FH München
- Vorstellung Fachbereich Elektrotechnik und Informationstechnik
  Prof. Dr. Rainer Seck, Prodekan Fachbereich Elektrotechnik, FH München
- Begrüßung durch ASIM
  Prof. Dr. Felix Breitenecker, TU Wien, ASIM-Sprecher
- Simulation in Forschung und Lehre im Fachbereich Elektrotechnik und Informationstechnik
  Prof. Dr. Werner Kohl, FH München

14.00 – 14.45 — **Plenarvortrag**
Hörsaal R 1.046 — *Sitzungsleitung: Dr. Ingrid Bausch-Gall*

- Ausführliche Fehlersimulation zur Absicherung von Robustheit und Verfügbarkeit mechatronischer Systeme
  Dr. Andreas Junghanns, DaimlerChrysler AG, Berlin

14.45 – 15.15 — **Pause**

15.15 – 16.45 — **Parallelsitzungen**

HIL (Hardware-in-the-Loop)-Simulation, Fahrzeugtechnik

Hörsaal R 1.002 — *Sitzungsleitung: Markus Stöbe*

- Modellierung und HIL-Simulation eines automatisierten Schaltgetriebes
  Dr. Clemens Schlegel, Schlegel Simulation GmbH, Freising
- Entwicklung von Fahrerassistenzsystemen im FASlab des DLR
  Markus Stöbe, DLR, Braunschweig
- Simulatorstudien - ein Werkzeug zur Bewertung und Optimierung von Fahrerassistenzsystemen
  Martin Brünger-Koch, DLR, Braunschweig

Workshop - Einsatz und Nutzen von VHDL-AMS im Automotive Umfeld

Hörsaal R 1.003 *Sitzungsleitung: Heinz-Theo Mammen*

- Einführung in VHDL-AMS
  Dr. Klaus Panreck, Hella KG, Lippstadt
- Thermische Modellierung elektrischer Leitungen mit Computer-Algebra-Simulation und VHDL-AMS – Theoretische Grundlagen und praktische Anwendungen
  Stefan Braun, SmartCAE, München;
  Manfred Klinkenberg, Yazaki EDS GmbH, Köln
- Sprachstandardisierung für die Systemsimulation mit VHDL-AMS – Praxisbeispiel DC/DC-Wandler
  Dr. Thomas Lang, BMW AG, München
- Anwendung mehrdimensionaler Netzwerkmodelle in VHDL-AMS
  Dr. Joachim Haase, FhG-IIS/EAS, Dresden

16.45 – 17.15 **Pause**

17.15 – 18.45 **Parallelsitzungen**

Modellbasierte Entwicklung, Prozess

Hörsaal R 1.001 *Sitzungsleitung: Dr. Walter Commerell*

- Systemsimulation im Entwicklungsprozeß aus der Sicht von Systems Engineering
  Prof. Dr. Werner Kohl, FH München
- Durchgängiger modellbasierter Entwicklungsprozeß mit Autocodegenerierung
  Rainer Moser, T-Systems GEI GmbH, Leinfelden-Echterdingen
- Die Veränderung der Simulationsinfrastruktur von Simulink für die Modellierung physikalischer Systeme
  Steve Miller, The Mathworks GmbH, Ismaning

HIL (Hardware-in-the-Loop)-Simulation, Luftfahrt

Hörsaal R 1.002 *Sitzungsleitung: Peter Saager*

- Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR
  Peter Saager, DLR, Braunschweig
- ATTAS - Der fliegende Simulator des DLR
  Thomas Heinecke, DLR, Braunschweig

- Entwicklung von Echtzeit-Experiment-Funktionen für den Flugversuchsträger ATTAS
  Martin Gestwa, DLR, Braunschweig

VHDL-AMS Vendor Session

Hörsaal R 1.003 *Sitzungsleitung: Heinz-Theo Mammen*

- Präsentation des Werkzeuges Simplorer
  Olaf Hädrich, Ansoft GmbH & Co. KG, München
- Präsentation des Werkzeuges SyAD
  Markus Pistauer, CISC Semiconductor Design+Consulting GmbH, Klagenfurt
- Präsentation des Werkzeuges Smash
  Dirk Dammers, Dolphin Integration GmbH, Duisburg
- Präsentation des Werkzeuges System Vision
  Thomas Heurung, Mentor Graphics GmbH, München
- Präsentation des Werkzeuges SABER
  Andre Jennert , Synopsys GmbH, Aschheim/Dornach

18.45 – 19.15 **Mitgliederversammlung Fachgruppe „STS“**

Hörsaal R 1.046 *Dr. Achim Wohnhaas*

- Bericht des Sprechers
- Wahl der Fachgruppen-Sprecher

ab 19.30 **gemeinsames Abendessen**

Löwenbräukeller (Nymphenburger Str. 2, am Stiglmaierplatz).
Kurzer Fußweg oder Strassenbahn.
Raum: „Dachauer Stuben“.

08.30 – 10.00 **Plenarvorträge**
Hörsaal R 1.046 *Sitzungsleitung: Dr. Ingrid Bausch-Gall*

- Simulation: Durch Abstraktion zum Systemverständnis
  Dr. Thomas Lang, BMW AG, München
- XML-Datenschnittstellen für Simulations- und Berechnungsprogramme
  Hansjörg Kapeller, arsenal research, Wien

10.00 – 10.30 **Pause**

10.30 – 12.00 **Parallelsitzungen**

Modellbasierte Entwicklung, Anwendungen

Hörsaal R 1.001 *Sitzungsleitung: Prof. Dr. Felix Breitenecker*

- Thermische Simulation einer Scheibenbremse für Nutzfahrzeuge
  Stephan Pitzing, Knorr-Bremse Systeme für Nutzfahrzeuge GmbH, München
- Sensitivitätsberechnung zur Toleranzanalyse und Fehlerlokalisierung analoger Schaltungen und Systeme
  Dr. Peter Schwarz, FhG-IIS/EAS, Dresden

Fehlersimulation, Fehlertests, Teil 1

Hörsaal R 1.002 *Sitzungsleitung: Franz Pirker*

- Modellierung von elektrischen und mechanischen Fehlern bei elektrischen Antrieben
  Franz Pirker, arsenal research, Wien
- Testen von Software: Ein Nachweis der korrekten Steuergeräte-Funktionalität?
  Manfred Melzig, Hella KG, Lippstadt
- Durchgängige Open-Loop-Testverfahren für Kfz-Elektronik im Labor- und Fahrversuch
  Dr. Gerd Baumann, Michael Brost, Prof. Dr. Hans-Christian Reuss, FKFS, Stuttgart

Simulation alternativer Antriebe, Teil 1

Hörsaal R 1.003 *Sitzungsleitung: Sven-Erik Pohl*

- Modellierung von Brennstoffzellen für die mobile Anwendung
  Nicola Bundschuh, TT, DLR Stuttgart, TTI GmbH/TGU Sim-BEL

- Implementierung eines Stoffmodells für die Modellierung von Brennstoffzellen in Modelica und Simulink: Ein Vergleich
  Jörg Ungethüm, Institut für Fahrzeugkonzepte, DLR, Stuttgart
- Methodik der impedanzbasierten Modellierung für Batterien
  Julia Schiffer, Institut für Stromrichtertechnik und Elektrische Antriebe, RWTH Aachen

12.00 – 13:00 **Mittagspause**

13.00 – 14.00 **Parallelsitzungen**

Modellbasierte Entwicklung, Methoden
Hörsaal R 1.001 *Sitzungsleitung: Dr. Achim Wohnhaas*

- Avatare in technischen Simulationen
  Prof. Dr. Dietrich Wloka, Universität Kassel
- Einsatzmöglichkeiten der Computer-Algebra-Simulation in der Fahrzeugentwicklung
  Stefan Braun, SmartCAE, München

Fehlersimulation, Fehlertests, Teil 2
Hörsaal R 1.002 *Sitzungsleitung: Dr.-Ing. Peter Schwarz*

- Fehlersimulation elektronischer Schaltungen (digital und mixed-signal)
  Prof. Dr. Bernd Straube, FhG-IIS/EAS, Dresden
- Fehlersimulation mit SPICE
  Herbert Müller, Thomatronik GmbH, Rosenheim

Simulation alternativer Antriebe, Teil 2
Hörsaal R 1.003 *Sitzungsleitung: Sven-Erik Pohl*

- Simulation von elektrischen Antrieben im automotiven Bereich mit der Smart Electric Drives (SED) Library in Dymola (Doppelvortrag)
  Johannes v. Gragger, Dragan Simic, arsenal research, Wien

14.05 – ca. 17.00 **BMW-Besichtigung**

Bustransfer für die angemeldeten Teilnehmer zur Besichtigung des Münchner Stammwerks der BMW AG.

Die Beiträge werden nach dem Treffen auf der ASIM Homepage (www.asim-gi.org) als PDF-Dateien zum Herunterladen bereitgestellt.

Die Teilnahmegebühr schließt Erfrischungen und Imbisse während des Fachgruppentreffens ein.

Aussteller

- Ansoft Germany GmbH (www.ansoft.de)
- arsenal research (www.arsenal.ac.at)
- BAUSCH-GALL GmbH (www.bausch-gall.de)
- CISC Semiconductor Design+Consulting GmbH (www.cisc.at)
- Dolphin Integration GmbH (www.dolphin-integration.com)
- Mentor Graphics GmbH (www.mentor.com/germany)
- Schlegel Simulation GmbH (www.schlegel-simulation.de)
- SmartCAE (www.smartcae.de)
- Synopsys GmbH (europe.synopsys.com/germany)
- Thomatronik GmbH (www.thomatronik.de)

Fragen, Anregungen und Wünsche an:

- Dr. Ingrid Bausch-Gall, BAUSCH-GALL GmbH,
  asimMuenchen@bausch-gall.de
- Dr. Achim Wohnhaas, T-Systems GEI GmbH,
  Sprecher der Fachgruppe „STS“,
  Achim.Wohnhaas@t-systems.com
- Dr. Peter Schwarz, FhG-IIS/EAS Dresden,
  Sprecher der Fachgruppe „GMMS“,
  Schwarz@eas.iis.fhg.de
- Prof. Dr. Werner Kohl, FH München,
  Kohl@ee.fhm.edu

DAIMLERCHRYSLER

Ausführliche Fehlersimulation zur Absicherung von Robustheit und Verfügbarkeit mechatronischer Systeme

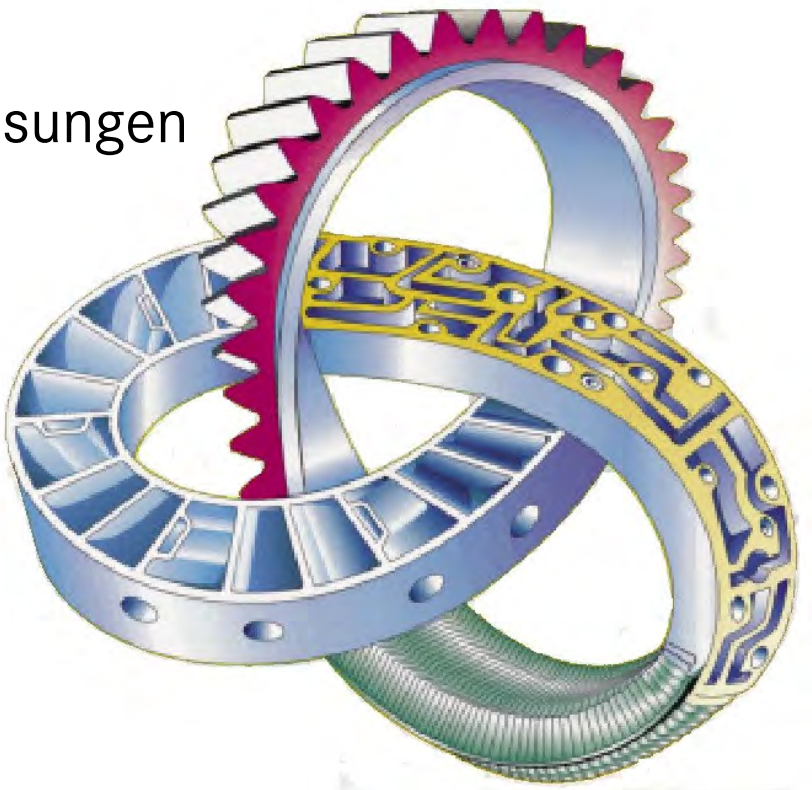
Dr. Andreas Junghanns, DaimlerChrysler Forschung und Technologie

Herausforderungen in der Produkt- und Systementwicklung

Innovationen sind:

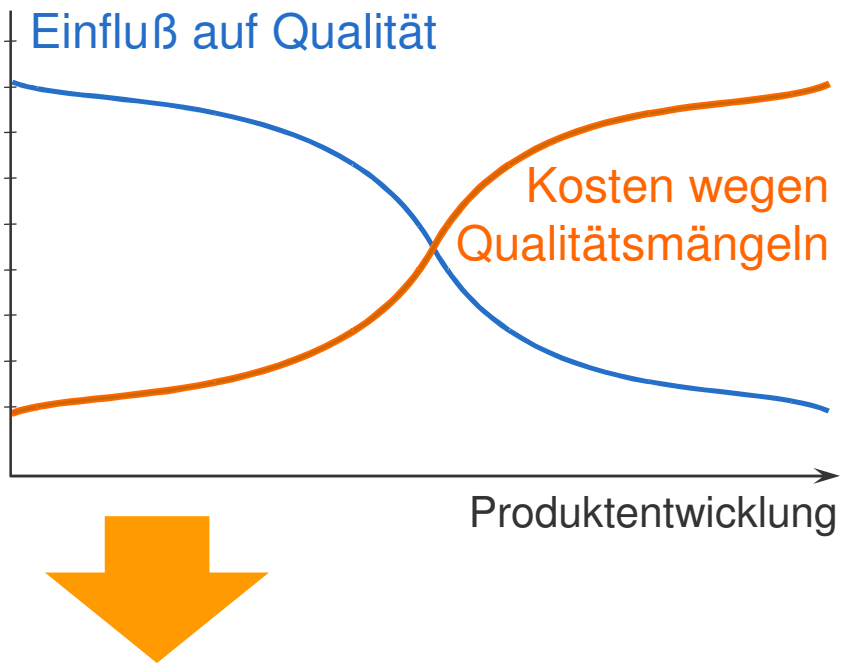
integrierte MEC-HYD-EE-**SW** Lösungen

- Fahrzeugkomplexität ↑
- Entwicklungszeit ↓
- simultanes, domänen-übergreifendes Engineering ↑
- Kombinatorik der Funktionsbedingungen ↑



Produktqualität und Gesamtkosten

Audi-Studie: Fehlerbehebung in Null-Serie 10x teurer als in Musterphasen



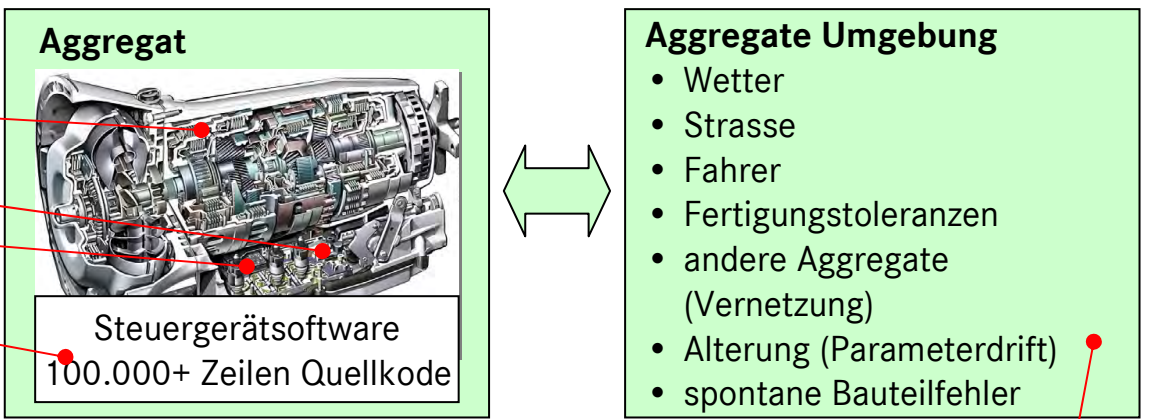
“Front Loading” von Qualitätssicherung

Integrierte Methoden der Produkt- und Systementwicklung

Herausforderung Mechatronikentwicklung

Verteilte Entwicklung

- Mechanik
- Hydraulik
- Elektronik
- Software



Ansatz

Front-Loading: frühe Entwurfsbewertung

- fundierte Bewertung nur rechnergestützt (Simulation) möglich
- Kombinatorik
  "funktioniert unter Laborbedingungen" reicht nicht
- Randbedingung: Wirtschaftlichkeit
  Spezifikation 1000er Test-Szenarien ist keine Lösung: Abdeckung + Kosten

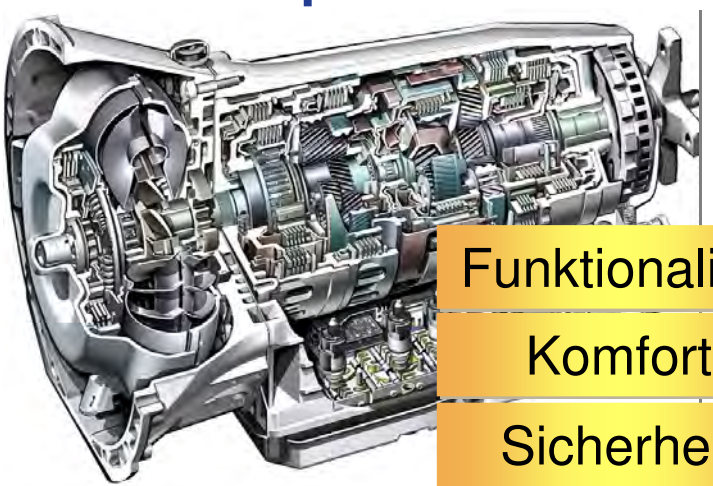
- vage bekannt
- Zufall
- Kombinatorik

Disziplinübergreifende Funktionsqualität

Ziel

Disziplinübergreifende Absicherung der Funktionsqualität
für Mechanik, Hydraulik, Elektrik und Software

Funktionsqualität



- Funktionalität
- Komfort
- Sicherheit
- Robustheit

- Validierung für alle Betriebsabläufe
- z.B. Schaltqualität & Verlustleistung
- Sicherheitskritische Ausfälle (FMEA)
- Fehlerreaktion & Parametertoleranz

Disziplinübergreifende Funktionsqualität

SPIEGEL ONLINE

„Allein in diesem Frühjahr legten … insgesamt 2000 Fahrer in 500 S-Klasse-Modellen rund 8 Millionen Kilometer zurück, um … mögliche Fehlerquellen zu orten und abzustellen."

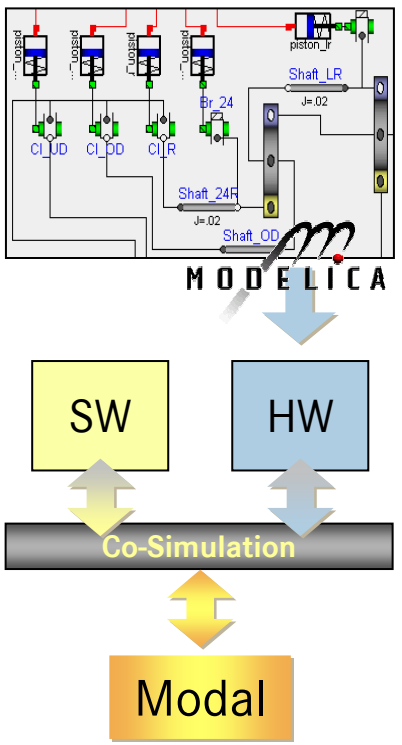
Der Spiegel im Gespräch mit Dieter Zetsche, 23.09.2005
Mercedes S-Klasse: Hier fährt der Chef noch selbst

WELT am SONNTAG

„Zu meinen Zielen gehört es, daß Qualität nicht erst durch Kontrollen am Ende sichergestellt wird.
… es beginnt sogar noch **früher** – in der Entwicklung."

Dieter Zetsche in
Welt am Sonntag , 15.01.2006

Virtuelle Absicherung der Funktionsqualität



Physikalische Modellierung der Hardware

wieder verwendbar, interdisziplinär, standardisiert

Co-Simulation von Hardware und Software

auf dem PC des Ingenieurs

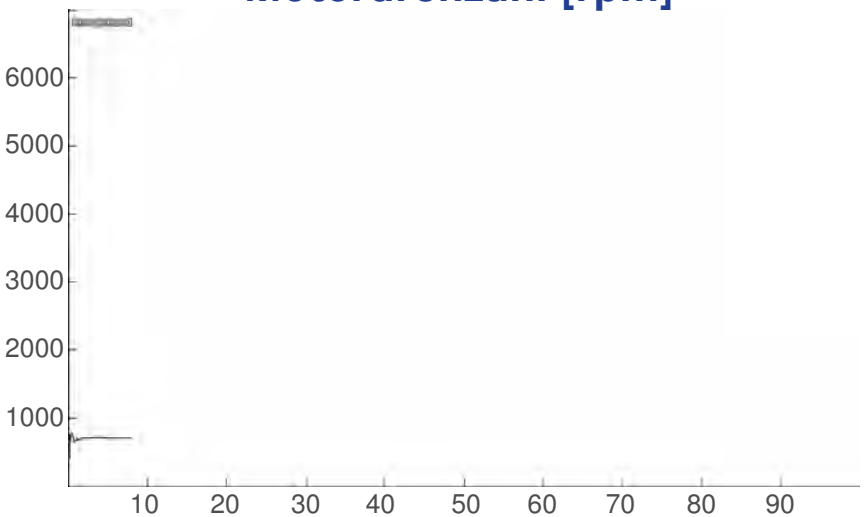
Szenariengenerierung & -auswertung

automatisch, geringer Aufwand, hohe Abdeckung, unvorhergesehene Fälle, auch Hardwareausfälle

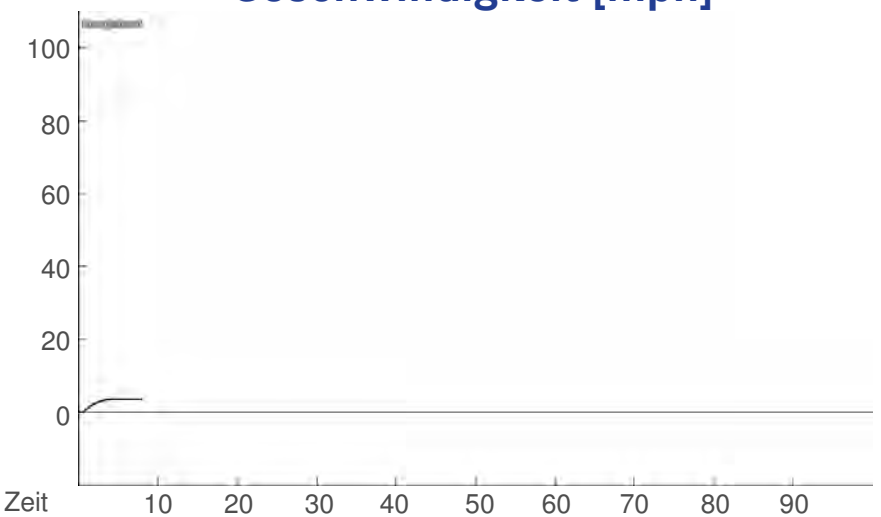
Funktionsanalyse | Sensitivitätsanalyse | Ausfallanalyse

300 von 15000 automatisch untersuchten Szenarien, 3 kritische Szenarien gefunden

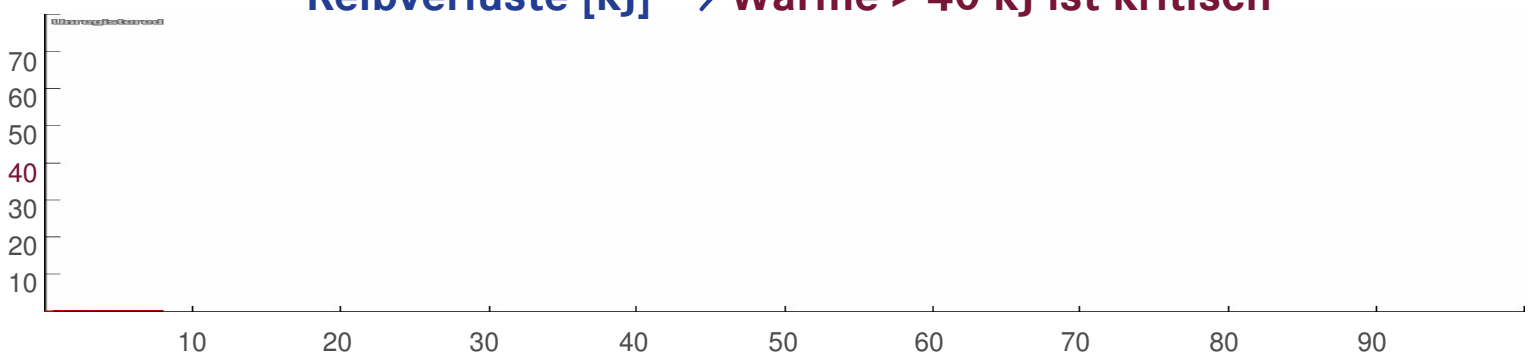
Motordrehzahl [rpm]



Geschwindigkeit [mph]



Reibverluste [kJ] → Wärme > 40 kJ ist kritisch



Kritischer Beispielfall

Ergebnisbericht über allen Szenarien

Problems	Failures		Gears		Scenario
alarms	failures	inserted in	current	target	reference
heat = very high, lossTime > 2.5s, state = fighting	solenoidUD = disconnected	G4	G6	G6	scen509

Detailbericht pro Szenario

arsenal research
Ein Unternehmen der Austrian Research Centers.

XML Daten-Handling

XML-Datenschnittstellen für Simulations- und Berechnungsprogramme

21.02.2006

H. Kapeller, A. Haumer, U. Reisenbichler, C. Kral

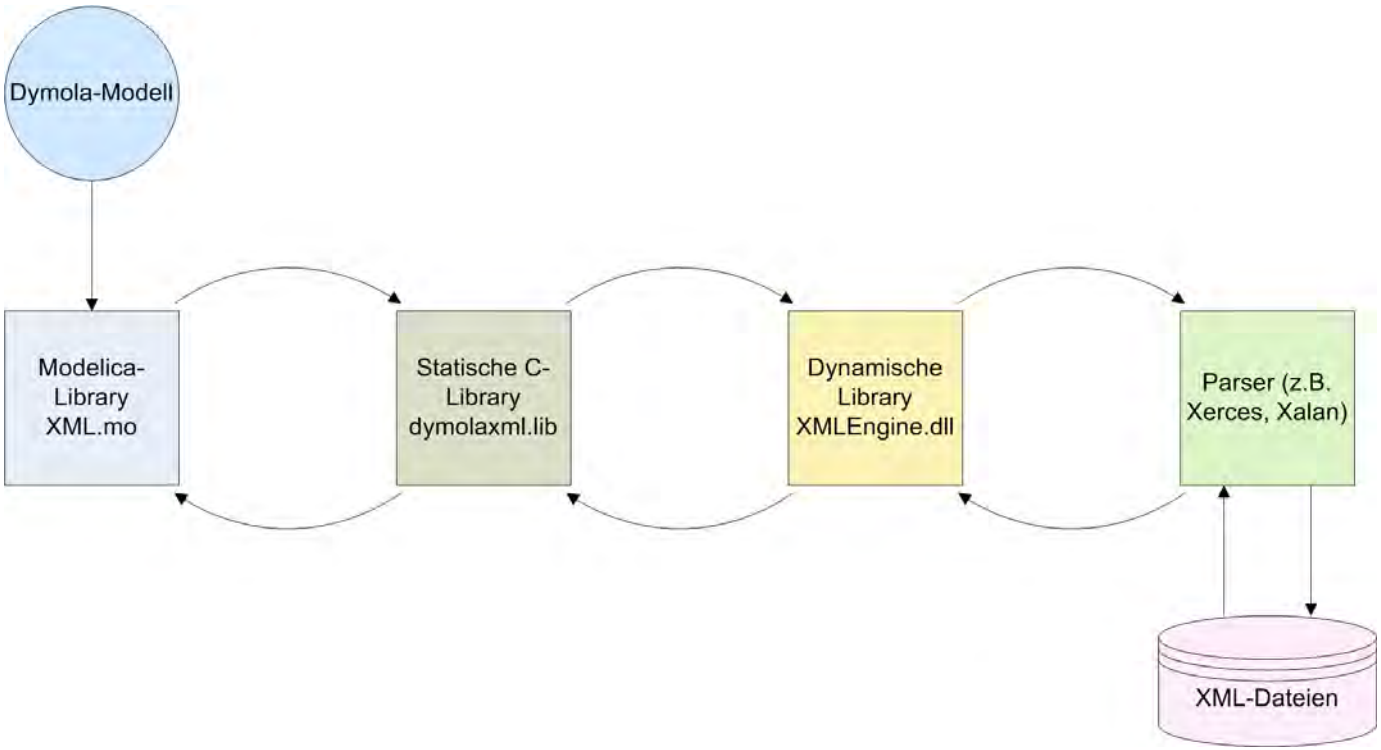
XML Daten-Handling – Überblick

- Motivation
- Elemente
- Datenzugriff mit DymolaXML
- XPath
- XML-Engine
- Beispiel
- Zusammenfassung

Motivation

- Trennung von Programmcode (Modellen) und Daten (Parametern)
- Leicht bedienbares Set von Funktionen für die Modellentwicklung
  - Benutzer kann bequem aus verschiedenen Parametersätzen wählen
- Standardisiertes und flexibles Datenformat für den gemeinsamen Datenzugriff
  - Einlesen von Parameterwerten, die in Zusatzprogrammen (Preprocessing) berechnet wurden
  - Abspeichern von Zuständen (Zustandsvariablen)
  - Einlesen von Zuständen (Zustandsvariablen) zum Setzen von Anfangsbedingungen
- XML: standardisiertes Datenformat
- XPath: standardisierte Syntax zum Abruf von Daten

Elemente


Dymola-Modell
Modelica-Library XML.mo
Statische C-Library dymolaxml.lib
Dynamische Library XMLEngine.dll
Parser (z.B. Xerces, Xalan)
XML-Dateien

Elemente

- Integration in die Entwicklungs- und Simulationsumgebung *Dymola*
  - Dymola Library:

    benutzt das *Modelica standard external function interface* für den Zugriff auf Daten aus externen XML-Datenquellen
- XML-Engine
  - Hauptelement für die Datenabfrage
  - Steuert alle Komponenten für den Datenzugriff
  - Speichert Abfrageresultate für späteren Abruf
  - Integrierte Datenumwandlung
- XML-Parser
  - Wohlgeformtheit (wellformatted) und Gültigkeit (validated) der XML-Dokumente
  - Zugriff auf die in XML-Dateien gespeicherten Daten mittels XPath

Datenzugriff mit DymolaXML

- Parser-Initialisierung
  - ParserHandle = createDOMParser(ParserName)
- Laden einer XML-Datei in den Speicher
  - DocumentHandle = loadDocument(ParserHandle, FileName)
- Zugriff auf Einzelergebnisse mittels *XPath-Expression*
  - getInt(DocumentHandle, XPath)
  - getIntFromContext(DocumentHandle, XPathContextNode, Xpath)

Beide Funktionen sind für verschiedene Datentypen verfügbar (Real, Boolean, String) und geben den ersten Treffer zurück

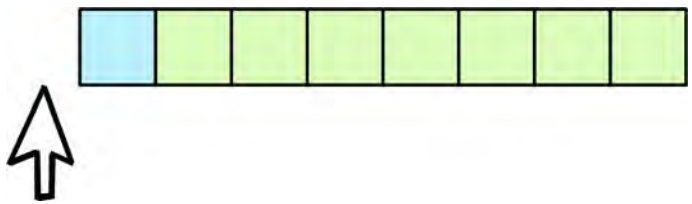
Datenzugriff mit DymolaXML (2)

- Erzeugung von *ResultSets* mit *XPath-Expressions*
  - ResultSetHandle = getXPathResultSet (DocumentHandle, XPath)
  - ResultSetHandle = getXPathResultSetFromContext(DocumentHandle, XPathContextNode, XPath)
- Zugriff auf *ResultSet*
  - getNumberOfResults(ResultSetHandle)
  - getIntAt(ResultSetHandle, Index)

    Diese Funktion ist für verschiedene Datentypen verfügbar (Real, Boolean, String)

Datenzugriff mit DymolaXML (3)

- Navigation im *ResultSet*
  - Der Result-Pointer zeigt anfänglich vor das **erste** Ergebnis

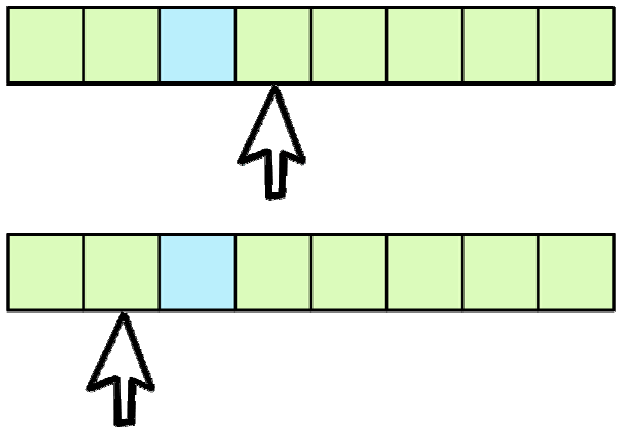


  - Bewegen des Result-Pointers:
    moveBeforeFirst(ResultSetHandle) bzw.
    moveAfterLast(ResultSetHandle)

Datenzugriff mit DymolaXML (4)

- Navigation im *ResultSet*
  - Datenzugriff und Bewegen des Result-Pointers:

    getNextInt(ResultSetHandle), getPreviousInt(ResultSetHandle),



    getFirstInt(ResultSetHandle) bzw. getLastInt(ResultSetHandle)

  Alle Funktionen sind für verschiedene Datentypen verfügbar (Real, Boolean, String)

Datenzugriff mit DymolaXML (5)

- Weitere Auswertung eines *ResultSets*
  - reevaluateResultSet(ResultSetHandle, XPath)
  - reevaluateResultSetFromContext(ResultSetHandle, XPathContextNode, XPath)
- Integrierte Datenumwandlung
  - Ist nur möglich, wenn ein XML-Parser initialisiert wurde
  - StrToInt(String), IntToStr(Int)
  - StrToReal(String), RealToStr(Real)
  - StrToBoolean(String), BooleanToStr(Boolean)

Fehlerbehandlung

- XML-Engine
  - Gibt *Integer error codes* zurück
  - Die entsprechenden Klartext-Fehlermeldungen können abgerufen werden
- package XML.mo
  - Stellt eine Standard-Fehlerbehandlung zur Verfügung
  - Individuelle Reaktion auf Fehler ist möglich
    z.B. Setzen von Standardwerten bei fehlenden Daten

XPath

- Vom Standard abweichende Implementierung
  - Ausschließlich Endknoten können abrufbare Werte beinhalten
  - Knotennamen werden durch ihren Wert ersetzt
  - Knoten-Attribute sind nicht abrufbar
  - XML-Namensraumkonvention (XML-namespace) wird nicht unterstützt
- Quellen
  - http://www.w3.org/TR/xpath
  - http://www.w3schools.com/xpath/default.asp

XML-Engine

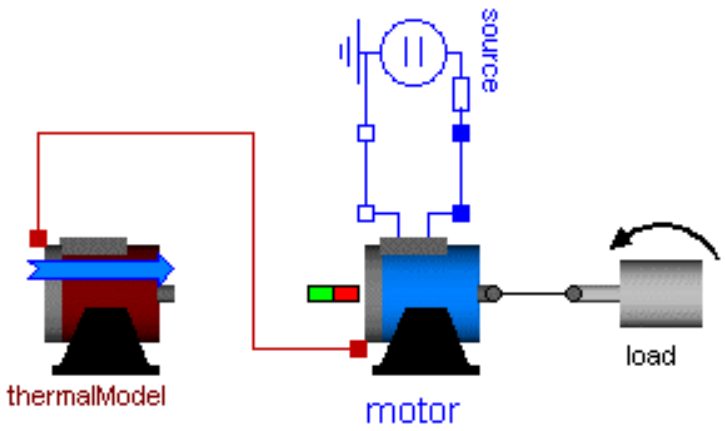
- Ermöglicht Laden und die Anwendung verschiedener XML-Parser
  - Die Test-Implementierung verwendet die *xerces-C API* über das *DLL-Interface Xerces.dll* [http://xml.apache.org/xerces-c/]
  - Mehrere XML-Parser können unabhängig voneinander implementiert werden, wenn die Schnittstellendefinitionen eingehalten werden
- Jeder Parser kann mehrere Dokumente laden und verwalten
- Aus jedem XML-Dokument können mehrere ResultSets erzeugt werden
- Grundsätzlich gibt es keine Einschränkung bzgl. der Daten
  - z.B. Verknüpfung zu SQL Datenbanken möglich, ...

Installation

- Unter Dymola
  - XML.mo: Dymola XML-Bibliothek
  - dymolaxml.h: C Header-File → \Dymola\Source
  - dymolaxml.lib (VC++) oder libdymolaxml.a (GCC):
    statische C Bibliothek → \Dymola\bin\lib\ (VC++) bzw. \Dymola\bin\lib\egcs\ (GCC)
- XML-Engine
  - XMLEngine.dll: → \Windows\System32
- Parser
  - Xerces.dll: → \Windows\System32
  - xerces-c_2_6.dll, Xalan-C_1_9.dll, XalanMessages_1_9.dll: → \Windows\System32

Beispiel

- Interdisziplinäres Modell
  - Elektrische Quelle (Batterie)
  - Gleichstrommotor (elektromechanisches Modell)
  - Modell der thermodynamischen Vorgänge in der Maschine
  - Mechanische Last



Ausblick

- Wünschenswert
  - Auswahloptionen (drop-down) aus XML-Datei abrufbar
  - Auswahloptionen (drop-down) für die Parameter des *top model*
  - drop-down Menü im Simulationsfenster
- Zukünftige Verbesserungen
  - Überladen von Funktionen (function overloading) / generische Funktionen
  - Arbeiten mit Feldern (arrays)
  - Arbeiten mit Tabellen (tables)

Zusammenfassung

- Leistungsfähiges Produkt
  - **flexibel + portabel + umfassend + laufende Weiterentwicklung +Support**
    $\Rightarrow$ kommerzielle Version
- Infos unter: Modelica@arsenal.ac.at
- Weitere Entwicklung
  - Implementierung dynamischer Felder (Arrays) in Dymola
  - Function overloading / generische Funktionen in Modelica / Dymola
  - Funktionen zum Aufbau eines XML-Baums
    (dzt. Schreiben nur mit XML-Template möglich)
  - DTD-Prüfung (Document Type Definition)

Vielen Dank für Ihre Aufmerksamkeit!

modelica@arsenal.ac.at

www.arsenal.ac.at

fachhochschule münchen

munich university of applied sciences

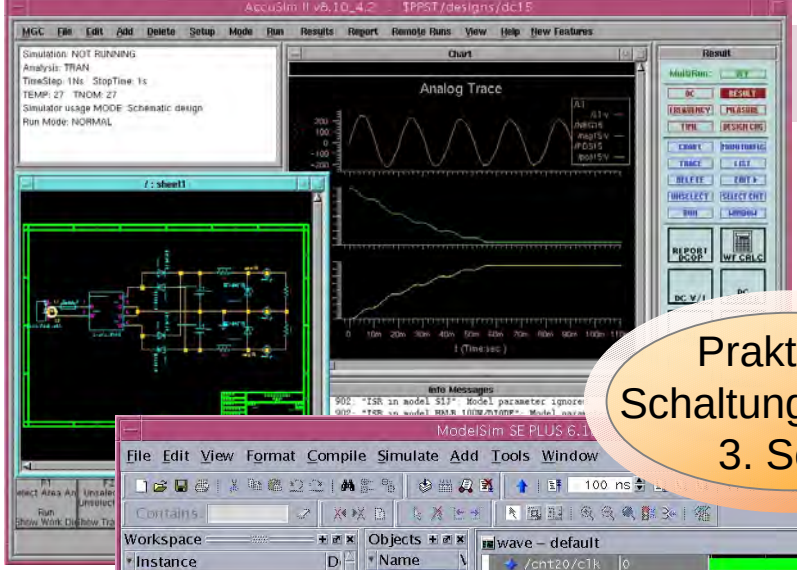
Simulation in Forschung und Lehre

Fachbereich
Elektrotechnik und Informationstechnik

Prof. Dr.-Ing. Werner Kohl
Kohl@ee.fhm.edu

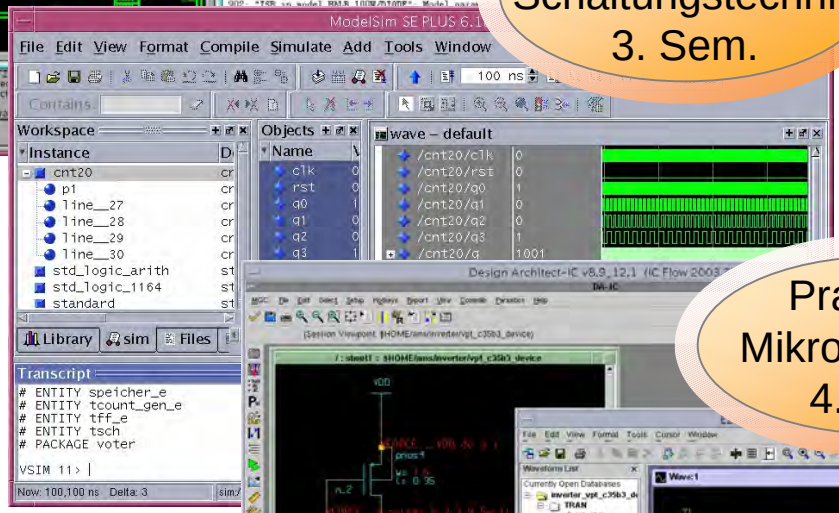
- **Simulation im Studium hat hohen Stellenwert**
  - beginnt bereits im 1. Semester Schaltungssimulation PSPICE
  - im 4. Semester größere Tool-Landschaft erreicht
  - verstärkte Beschäftigung im Hauptstudium 5. - 8. Semester
  - Modellierungstechnik im Masterstudiengang Systems Engineering
- **Lehre**
  - Im Vordergrund steht Methodenschulung
  - Toolschulung ist sekundär
- **Lizenzgebühren**
  - Teilweise gut abgedeckt durch Programme (EUROCHIP oä.)
  - Hohe Gebühren belasten knappes Budget
  - Wartungsverträge erzwingen jeweils neueste Versionen

Tool-Landschaft



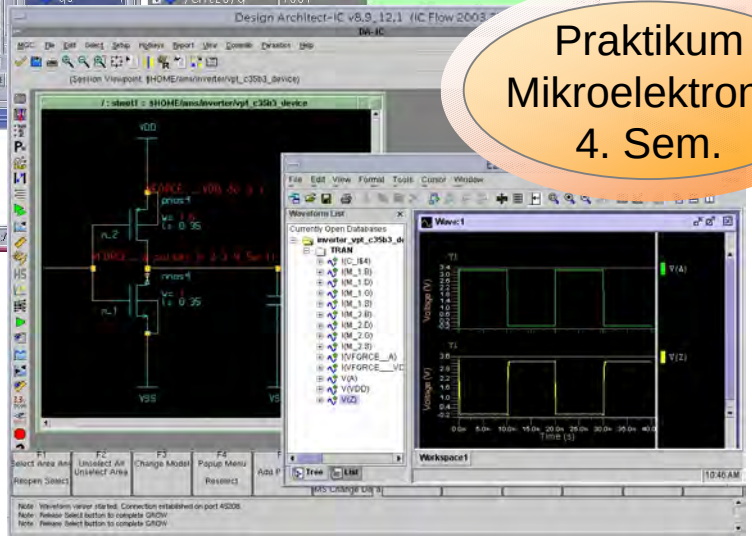
Accusim

Praktikum
Schaltungstechnik
3. Sem.

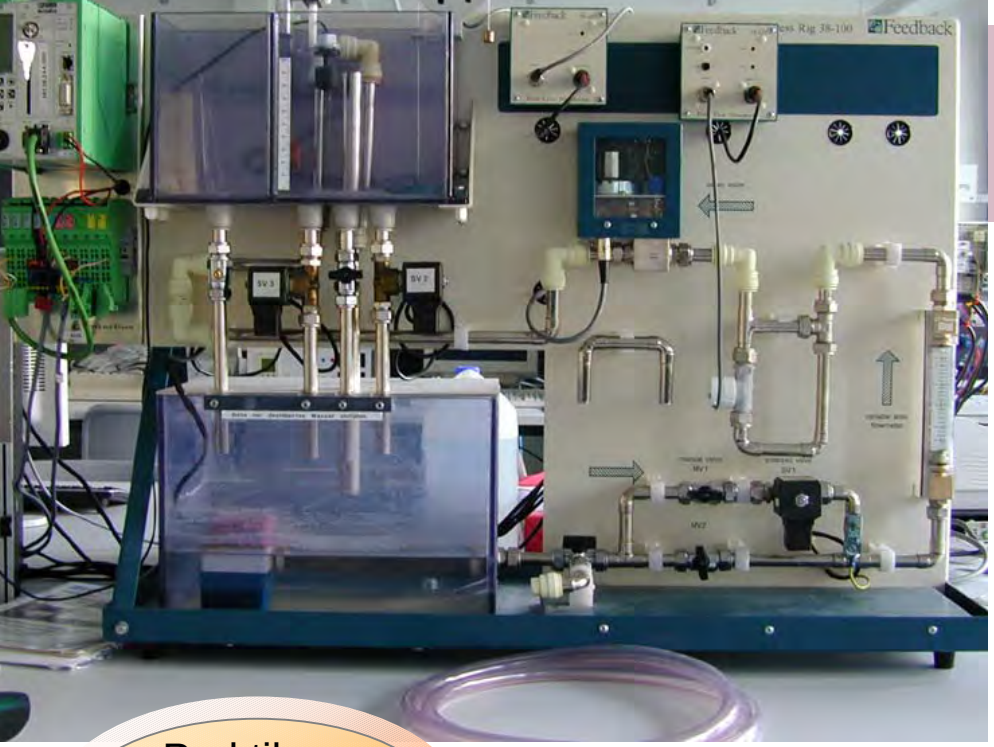


Modelsim
VHDL/Verilog/System-C
Simulator

Praktikum
Mikroelektronik
4. Sem.



Analog Simulator Eldo



Füllstandsregelung Wasserbehälte

Weitere:
Lageregelung Hubschraubermodell
Temperaturregelung Halogenlampe
Drehzahlregelung Motor/Generatorsatz
Positionsregelung Laufkatze
Untersuchung Zweifachregelung

Praktikum
Regelungstechnik
4. / 5. Sem.

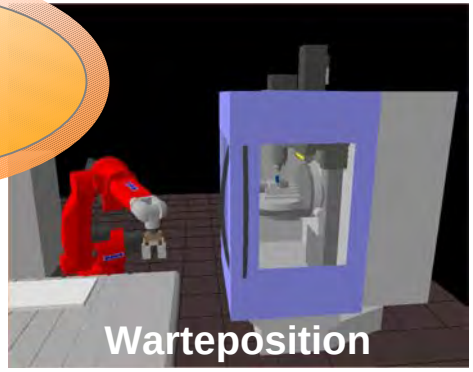
Analoge Regelungen realer Regelstrecken werden unter Verwendung eines PC's als Regler ausgeführt.

- Reglerstruktur > SIMULINK.
- C-Code Generierung des Reglers > Math. Works

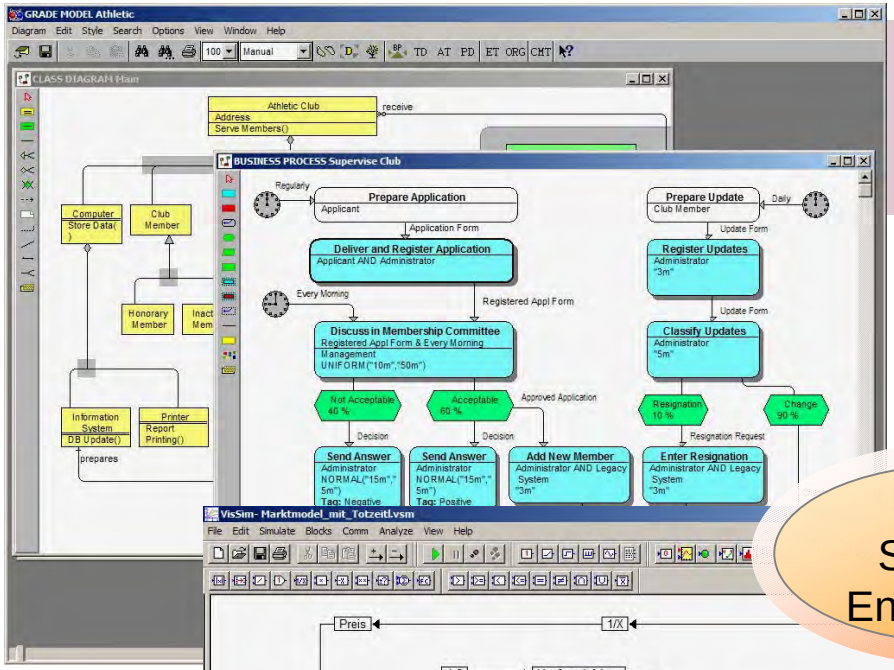
Simulationssoftware IGRIP der Fa. Deneb/Delmia

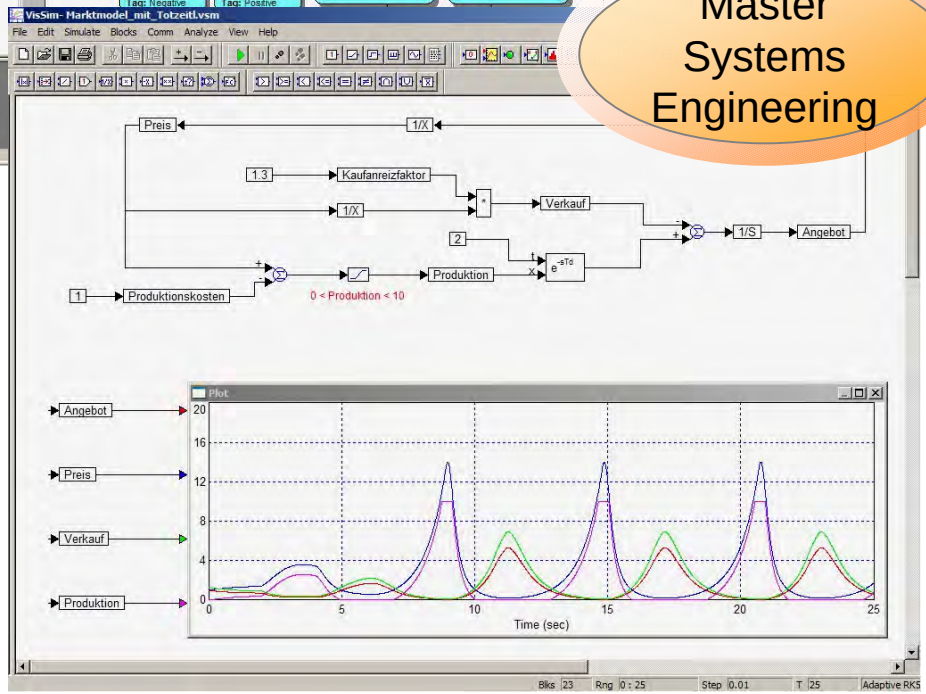
zur Erstellung, Simulation und Optimierung der Bewegungsabläufe von Geräten in Werkzellen und Werkstraßen.

Praktikum
Robotik
5. Sem.



Systems Engineering
Profs. Hettich, Dr. Geisweid





Master Systems Engineering

GRADE

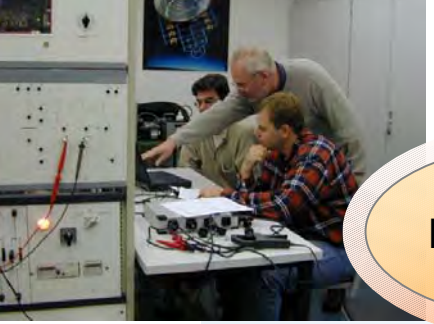
Geschäftsprozessmodellierung

VisSim

Modellierung, Simulation dynamischer Systeme

Simulation in der Forschung

- **In der Regel Überlappung von Lehre und Forschung**
- **Anwendungsbezogen**
- **Simulation in der Forschung deckt breites Spektrum ab**
  - Solaranlagen
  - Künstliche Blase
  - HF-Antennen



Praktikum
K&R Energie
5. / 7. Sem.

Simulation regenerativer Energiesysteme:

Grundlagen Simulationstechnik, Aufbau von Simulationsmodelle
verschiedene Simulationsprogramme, Projektarbeit

Laborausstattung u.a.

Software zur Simulation regenerativer Energiesysteme
(PVSOL, PVSYST, INSEL)

Software zur Simulation elektrischer Energieversorgungsnetze
(NEPLAN oder DIGSILENT)

SolEm

ist eine eigenentwickelte Software zur Simulation von netz-gekoppelten PhotoVoltaic Systemem - „Spielwiese für Ideen“

3D Solarwelt
Prof. Dr. Becker

Schatten5.avi

Künstliche Blase
Prof‘s Wassermann, Dr. Jocham

Diese Dame ziehen wir aus,

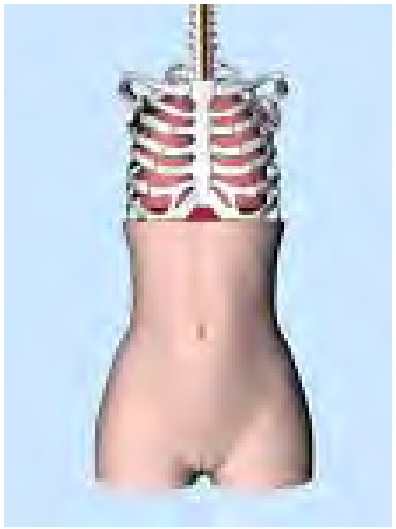
zu wissenschaftlichen Zwecken

wir nennen das dann Animation,
wir machen das mit CINEMA 4D,
im Labor für Sensorik der FHM und
an der Medizinischen Univ. Lübeck

Künstliche Blase
Prof‘s Wassermann, Dr. Jocham

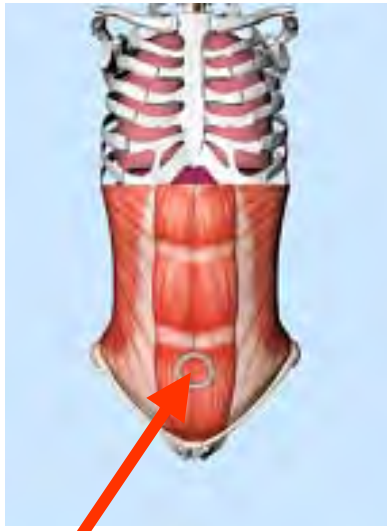
wir ziehen sie aus,

bis auf die Haut



bis aufs Fett

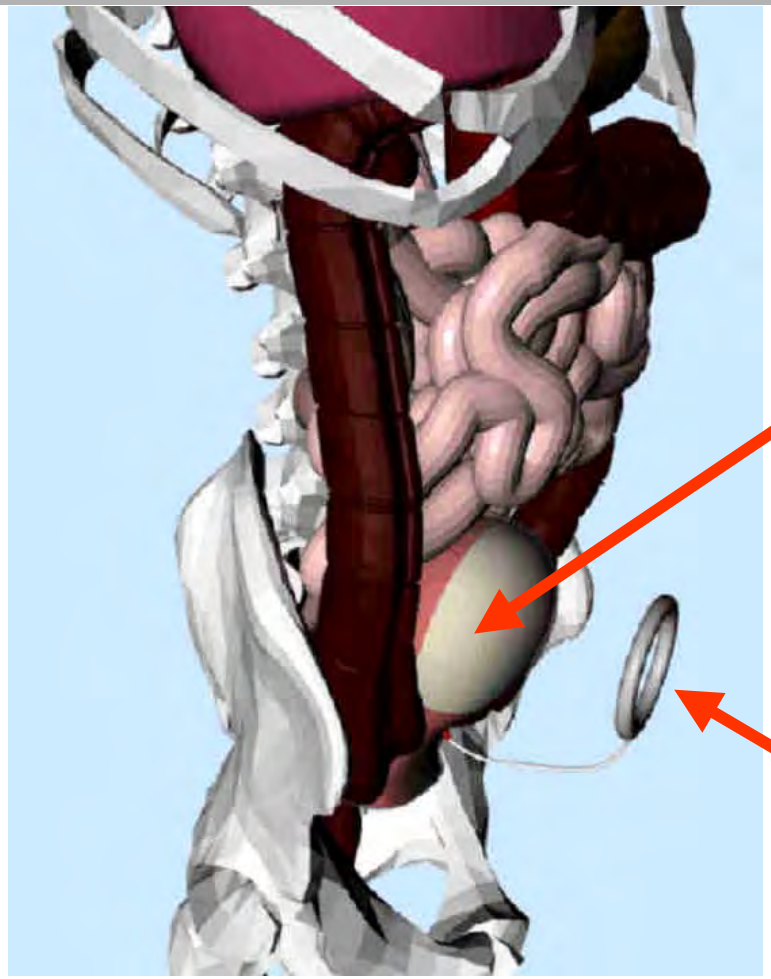
bis auf die Muskeln



Wir wollen die optimale Lage der Induktionsspul simulieren

Künstliche Blase
Prof‘s Wassermann, Dr. Jocham

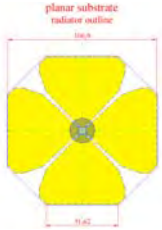
... weiter gilt es, die optimale Lage der Kunstblase zu simulieren



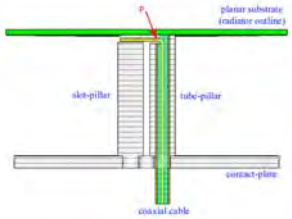
Simulation GSM Test Antenne
Prof. Dr. Fasold

Simulation with Microwave Studio

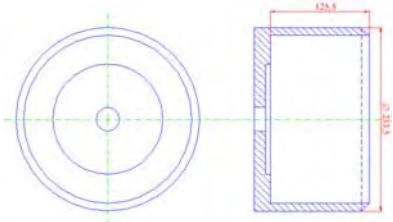
- **Radiator Outline**

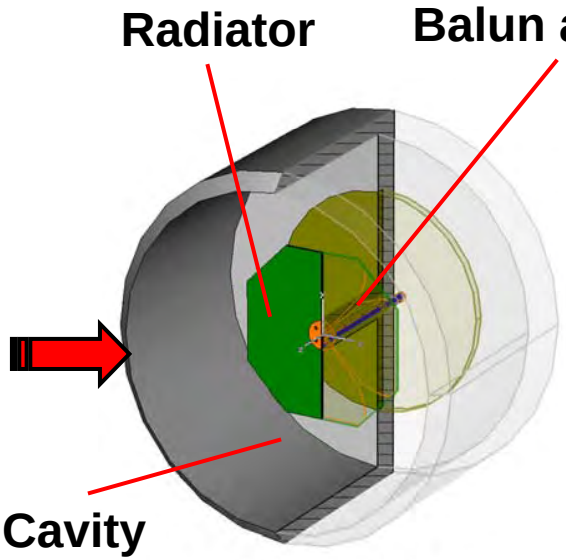


- **Balun and Feeding**

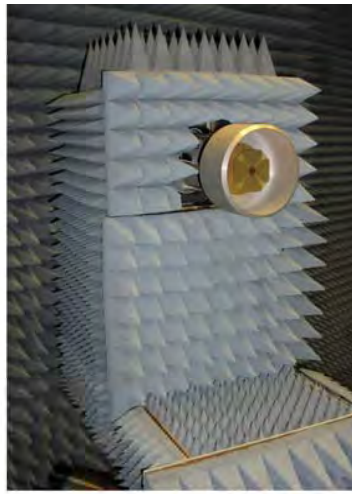


- **Cavity**





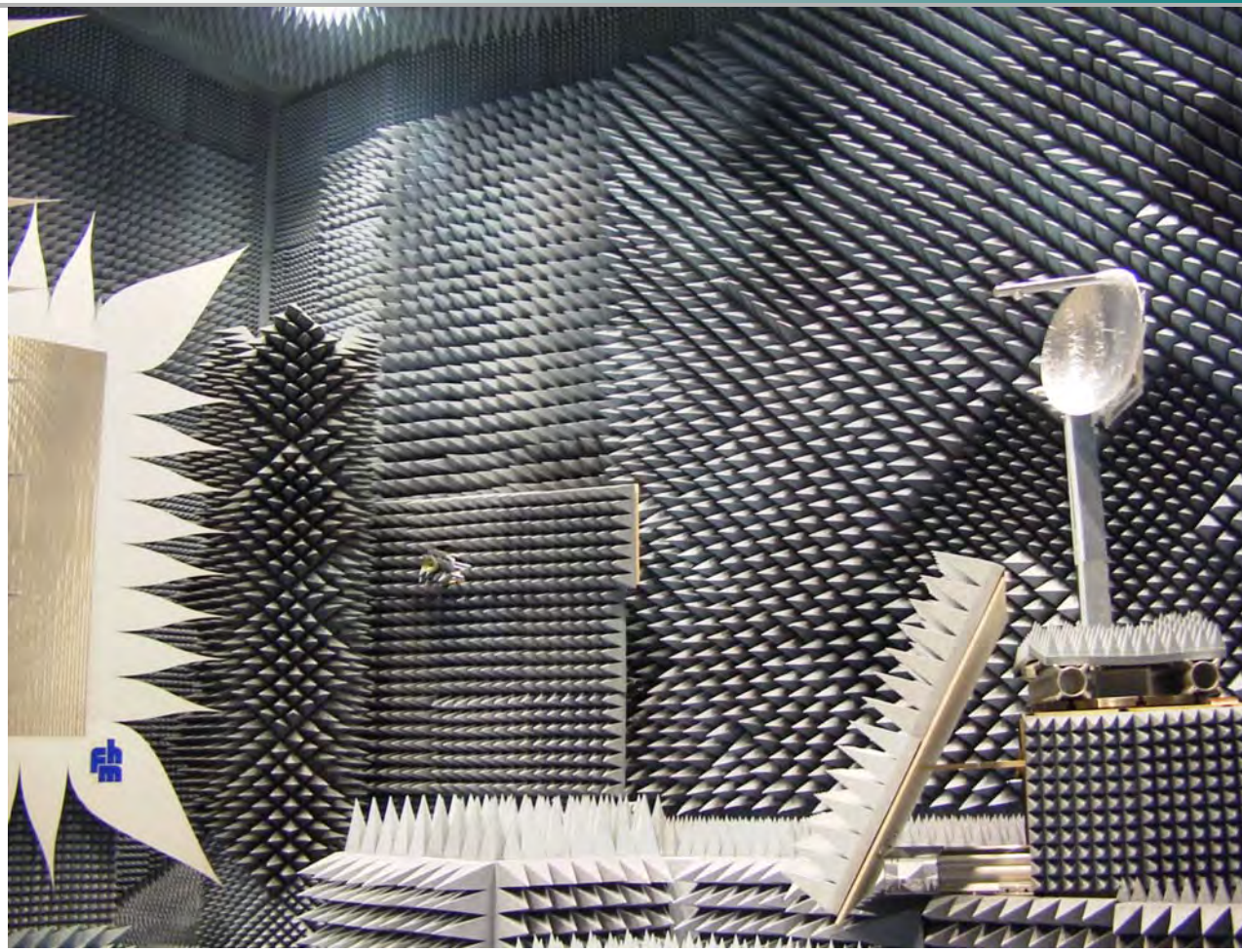
Simulation model (CST)



Photography (CCR)

Total view

HF-Messungen an skalierten Modellen
Prof. Dr. Fasold



Compact Range Messanlage: Messung eines skalierten Satellitenmodells

Ausbau der Forschungsaktivitäten

- **Intensivierung von drittmittelwirksamen Forschungsaktivitäten mit Unternehmen**
- **Verbesserter Zugang zu Förderprogrammen durch Unterstützung im Bereich Drittmittelbeschaffung (erfolgreichste bayerische Fachhochschule bei BMBF- Forschungsprojekten)**
- **Stärkung der Auftragsforschung und des Technologie- und Wissenstransfers**

Simulation: durch Abstraktion zum Systemverständnis.

ASIM Fachtagung, München, 20.-21.Februar 2006
Dr. Thomas Lang, BMW Group

BMW Group

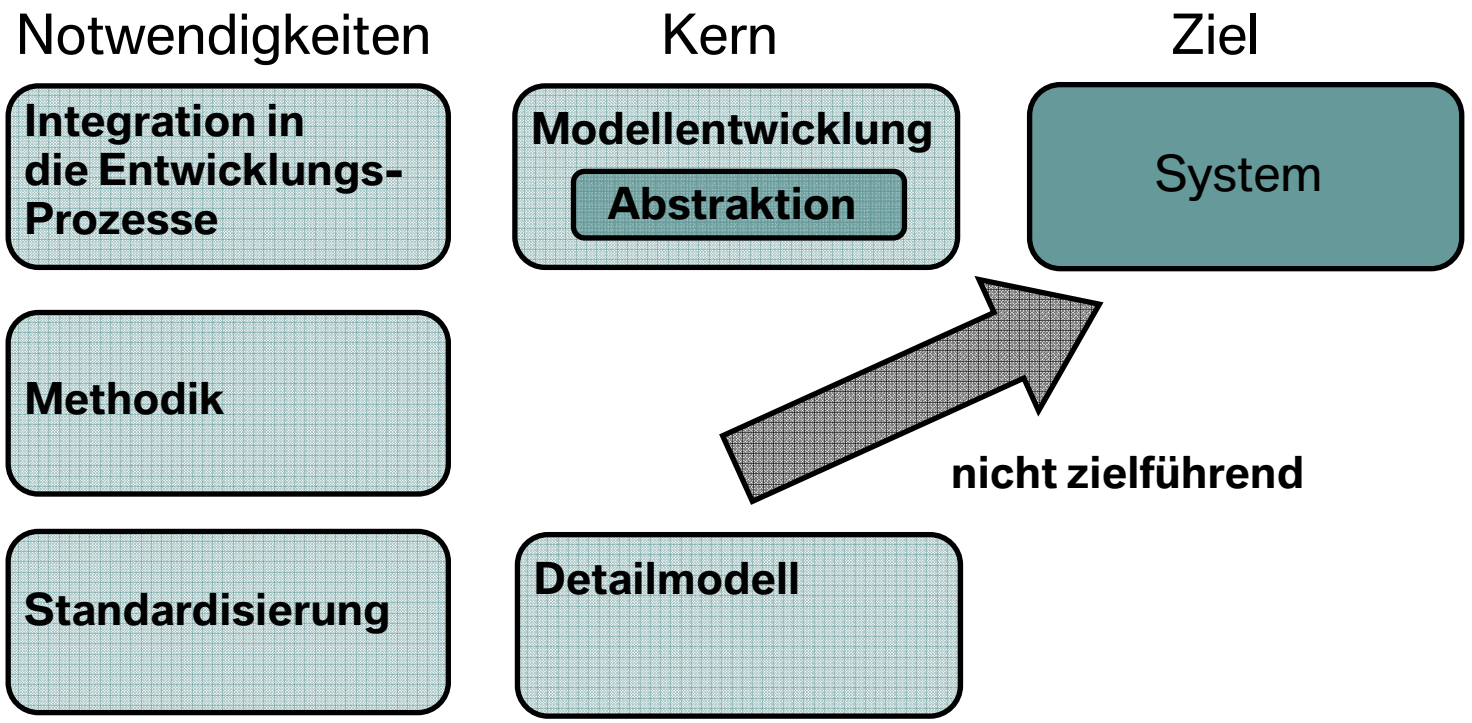
Simulation: Durch Abstraktion zum Systemverständnis.

Inhaltsverzeichnis.

Simulation: Durch Abstraktion zum Systemverständnis.

Einleitung (1) - Bisherige Vorgehensweise.

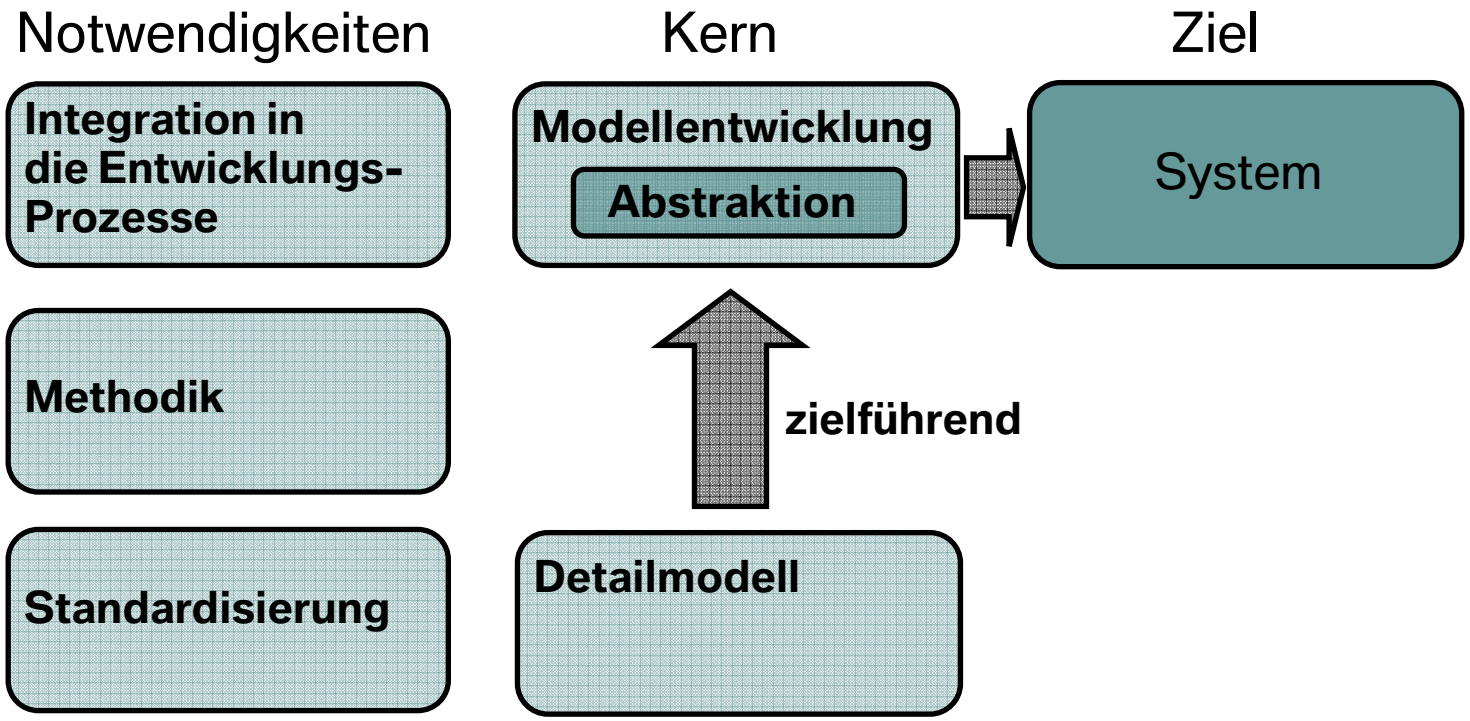
- Die Simulation geht weit über die Modellentwicklung hinaus.



Simulation: Durch Abstraktion zum Systemverständnis.

Einleitung (2) - Zukünftige Vorgehensweise.

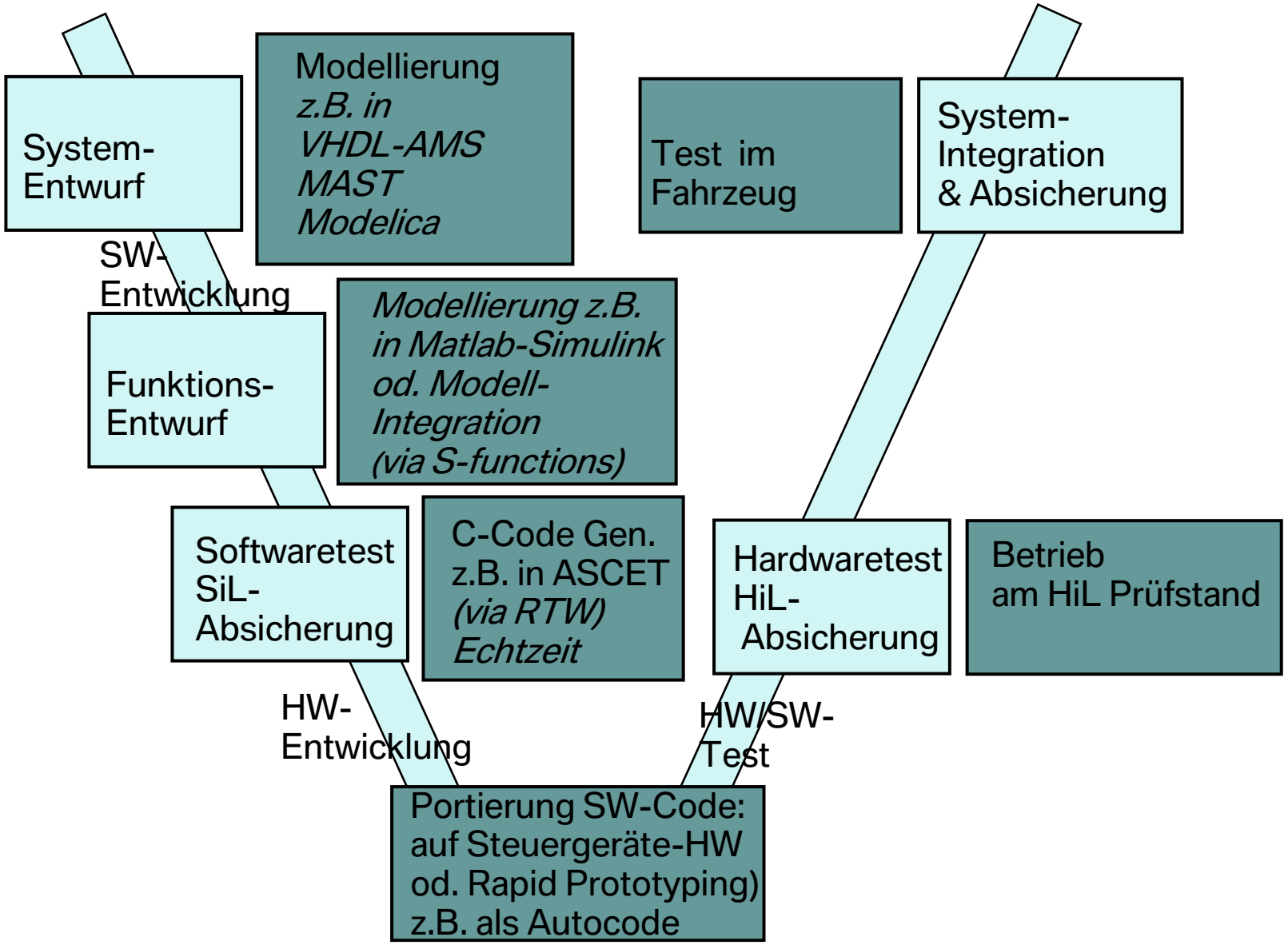
- Die Simulation geht weit über die Modellentwicklung hinaus.



Vorhandenes System analysieren -> abstrahieren -> neues System verstehen.

Simulation: Durch Abstraktion zum Systemverständnis.

Von der Insellösung zur Integration (1).



Simulation: Durch Abstraktion zum Systemverständnis.
Von der Insellösung zur Integration (2).

- Die Simulation ist fester Bestandteil der System- und Produktentwicklung vom Konzept bis zur Serie.

- Simulationen in unterschiedlicher Abschnitten der Entwicklung haben andere Inhalte und Ziele. In der Praxis werden während eines Entwicklungsprozesses verschiedene Tools eingesetzt.

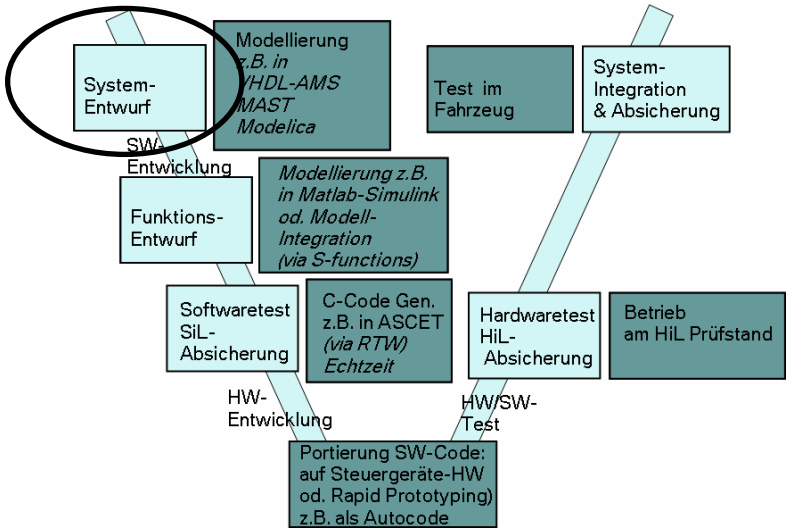
- Modelle müssen aufeinander aufbauen und weiterentwickelt werden können. Im Falle der Steuergeräteentwicklung entsteht z.B. aus der virtuellen Modellierung auf dem PC ein lauffähiger Steuergerätecode.

- Diese Vorgehensweise erfordert eine abgestimmte **Methodik** in der Modellentwicklung und Simulation.
  Die verwendeten Tools dürfen keine Insellösung bilden, sondern müssen sich durch standardisierte Schnittstellen in eine Prozesskette integrieren lassen.

Simulation: durch Abstraktion zum Systemverständnis.

Anforderungen an das Simulationsergebnis im Systementwurf.

- Bewertung der möglichen Systemarchitekturen.
- Dimensionierung der Systemkomponenten.
- Optimierungspotential des Systems erkennen.
- Grenzen des Systems erkennen.
- Vollständigkeit der Systembeschreibung prüfen.



Simulation: Durch Abstraktion zum Systemverständnis.
Anforderungen an die Methodik (1).

- Die Methodik unterstützt die genannten Anforderungen der Prozessdurchgängigkeit in der Simulation.

- Die durchgängige Methodik wird durch konsistente Verfahrensanweisungen und **„Design rules“** in der Modellierung erreicht.

- Physikalische Grundmodelle und Signalblöcke müssen durchgängig verwendet werden können, z.B. um die Echtzeitfähigkeit der Modelle am Ende des Entwicklungs-prozesses mit geringem Aufwand zu erreichen.
  Dies führt zur Verwendung eines eingeschränkten (normierten) „Befehlssatzes“.

- Durch Sprach-**Standardisierung** kann die Methodik wirkungsvoll unterstützt werden.
  Eine Standardisierung der Schnittstellen von Tool zu Tool ist in diesem Zusammenhang ebenfalls sinnvoll.

Simulation: Durch Abstraktion zum Systemverständnis.
Anforderungen an die Methodik (2).

Beispiel des Inhalts von Design rules:

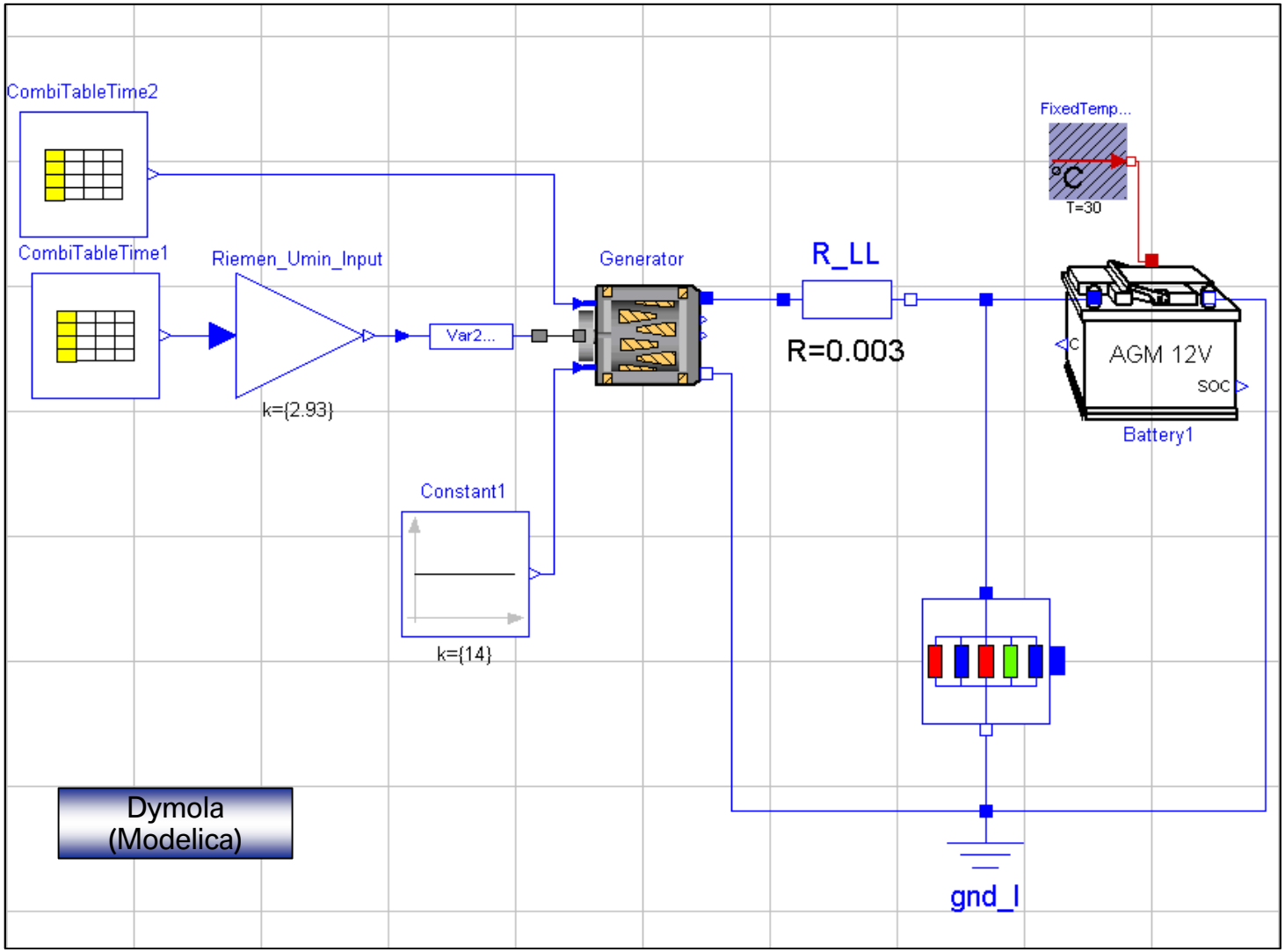
- Verwendungszweck von Modellen.
- Eigenschaften der angeforderten Modelle.
  - Basismodelle
  - Erweiterte Modelle
  - Systemmodelle
- Eigenschaften der geforderten Modellliersprache(n).

Simulation: Durch Abstraktion zum Systemverständnis.
Anforderungen an die Standardisierung.

- Austauschbarkeit des Modellcodes. Förderung der Zusammenarbeit z.B. zwischen Halbleiterhersteller, Systemlieferant und Fahrzeughersteller.
- Lastenhefte müssen nicht für ein bestimmtes Tool „interpretiert“ werden. „Ausführbares Lastenheft“.
- Weitgehende Unabhängigkeit von einem bestimmten Tool.
- Über standardisierte Schnittstellen können Modelle in alle Tools innerhalb der Prozesskette eingebunden und damit weiterverwendet werden.
- Wiederverwendbarkeit der Sprachkonstrukte und Grundmodelle im Prozess. Stichwort „Rückwärtskompatibilität“.

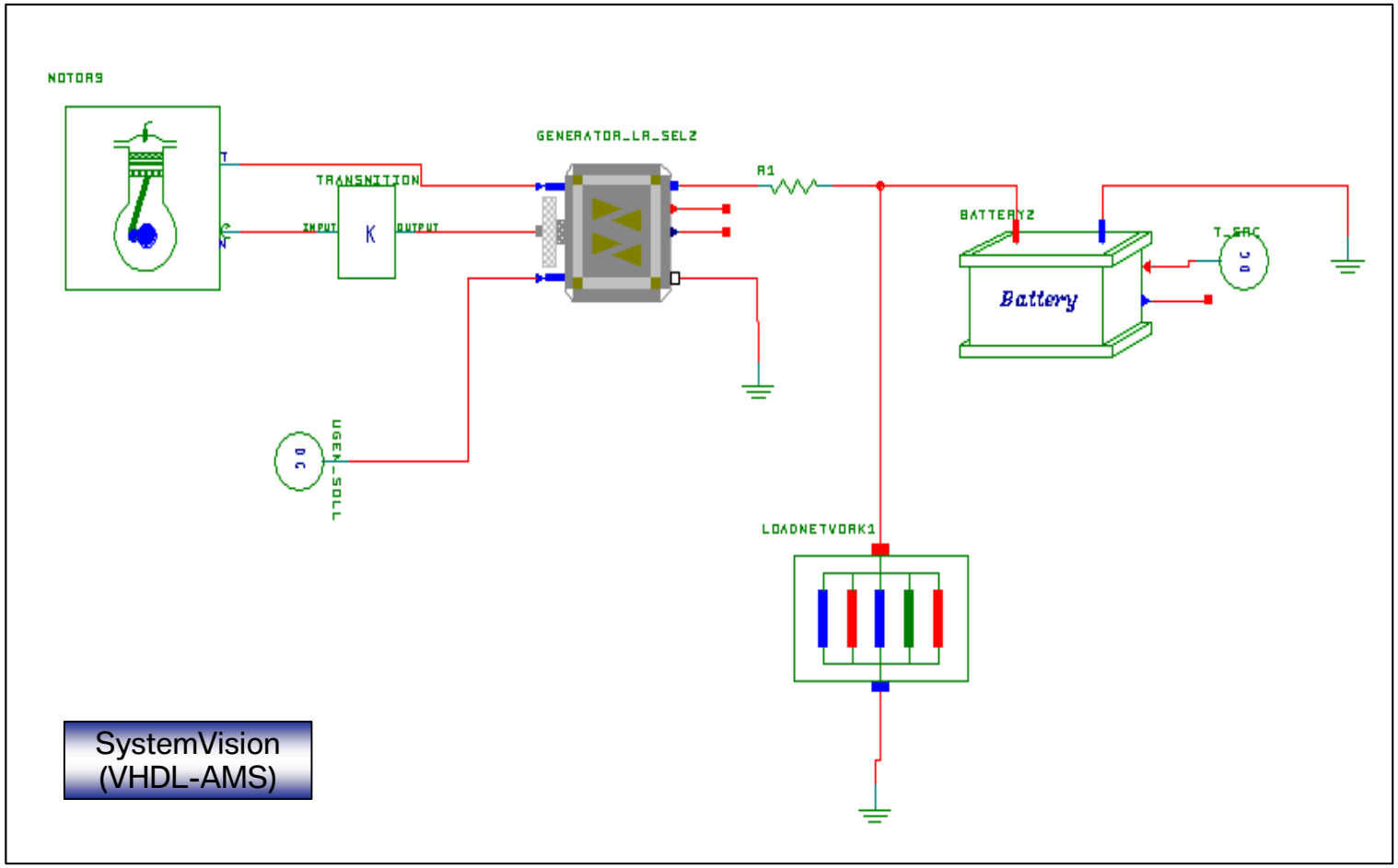
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Beispiel Bordnetzmodell in Modelica.



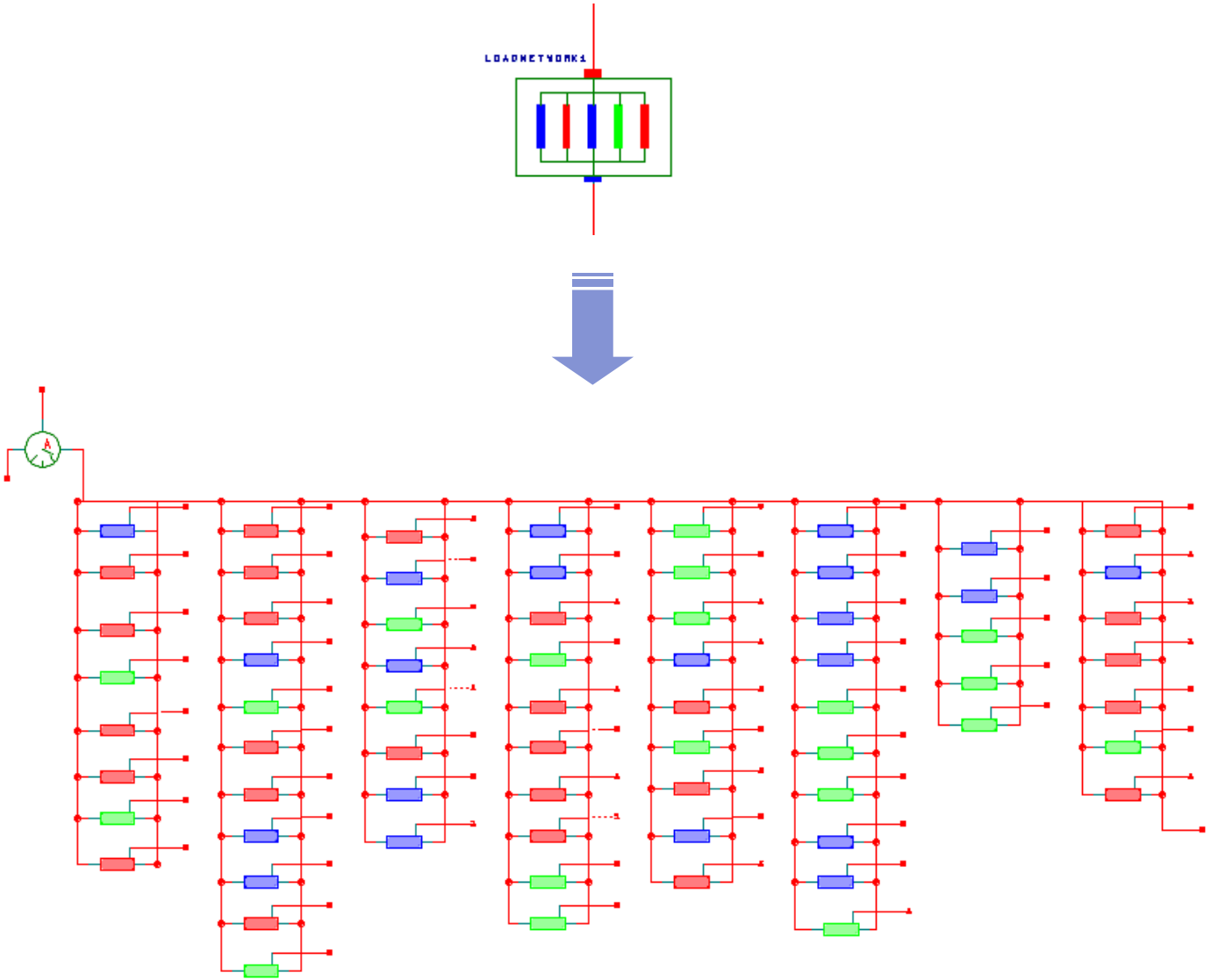
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Beispiel Bordnetzmodell in VHDL-AMS.



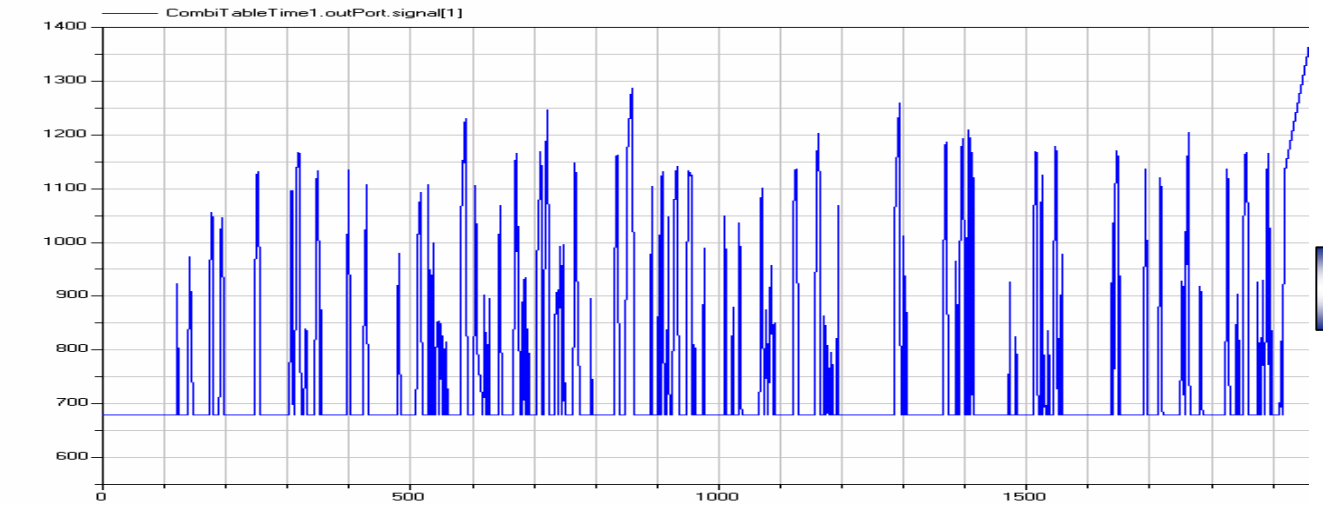
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Beispiel Bordnetzmodell in VHDL-AMS. Detaillierung der Verbrauchermodelle.



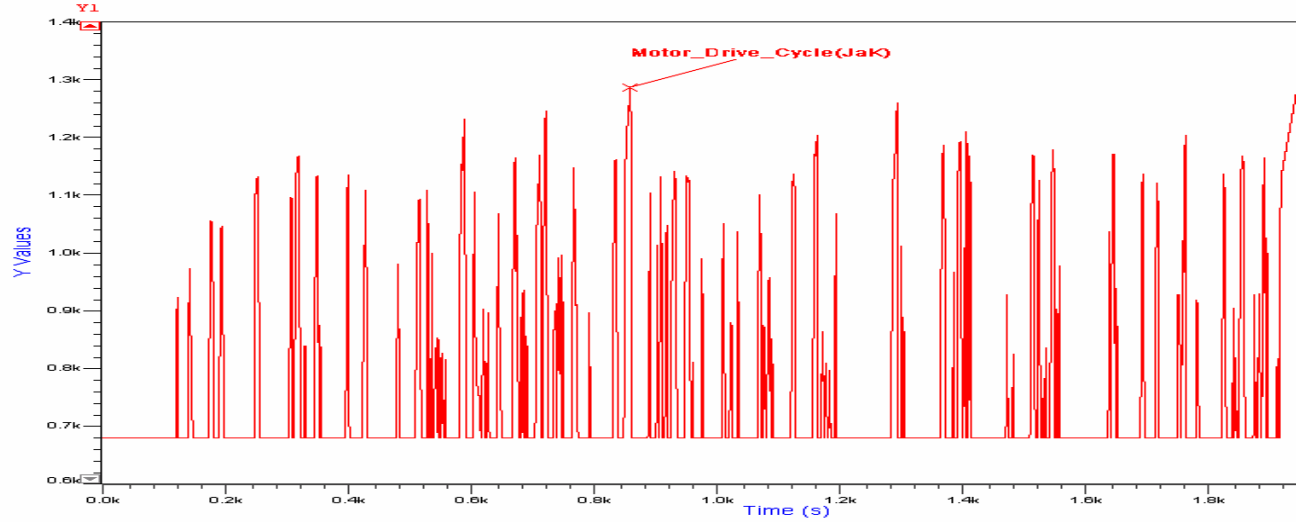
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Vergl. der Simulationsergebnisse (1).



Dymola
(Modelica)

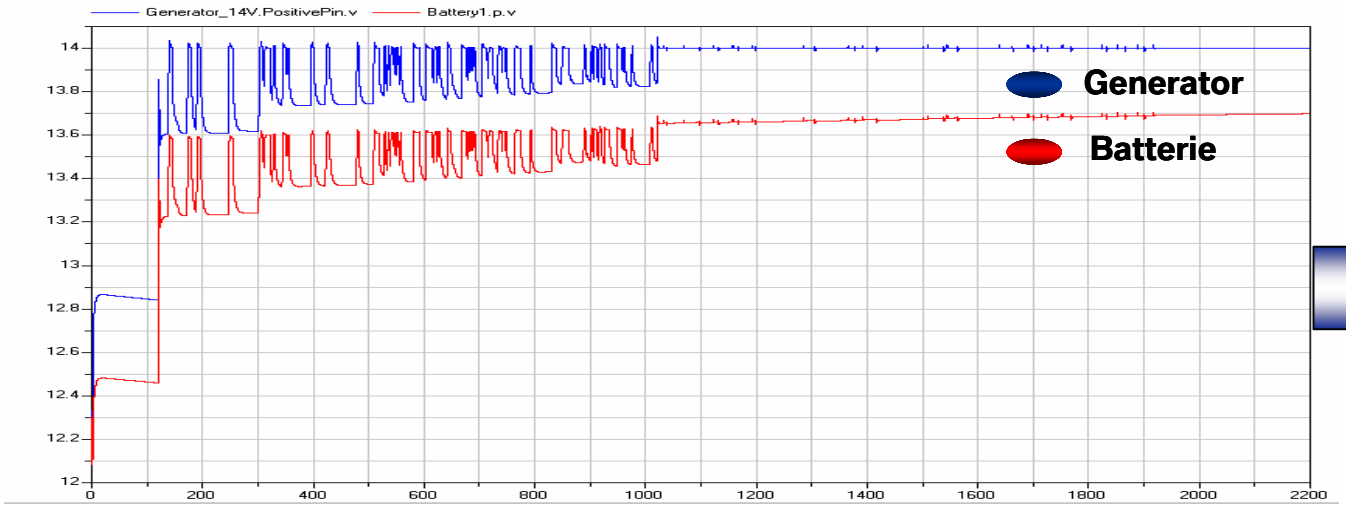
Motordrehzahl



SystemVision
(VHDL-AMS)

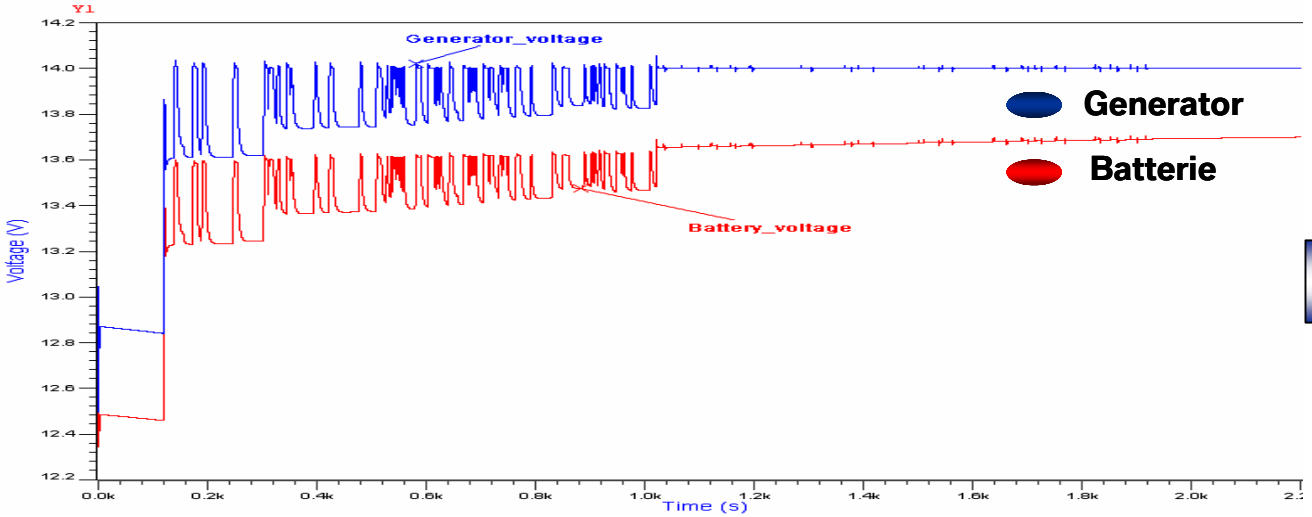
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Vergl. der Simulationsergebnisse (2).



Dymola
(Modelica)

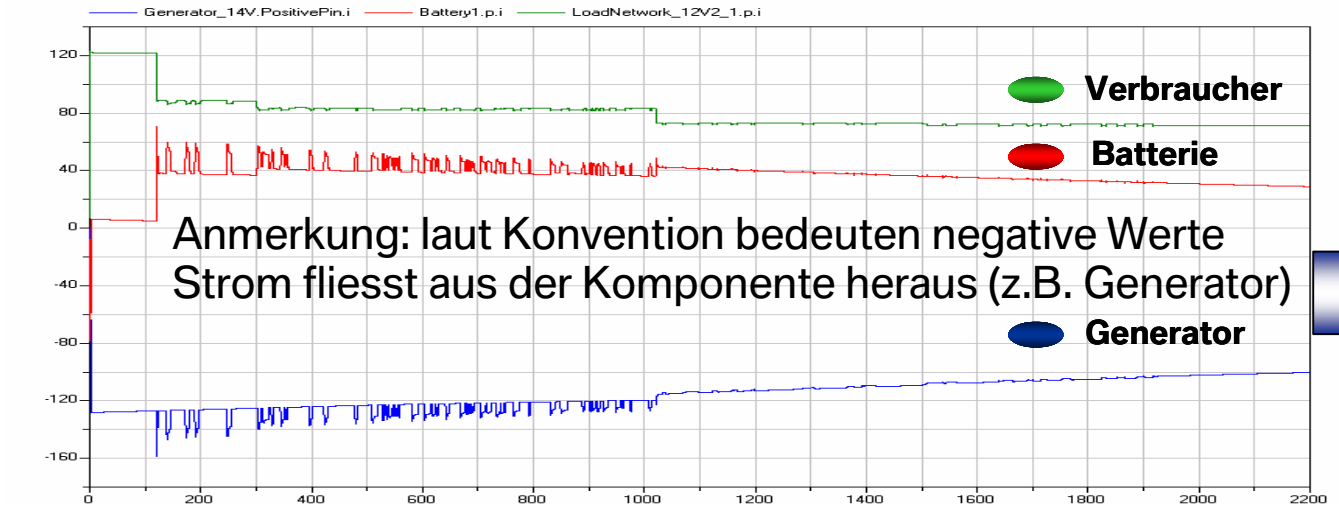
Spannungsverläufe



SystemVision
(VHDL-AMS)

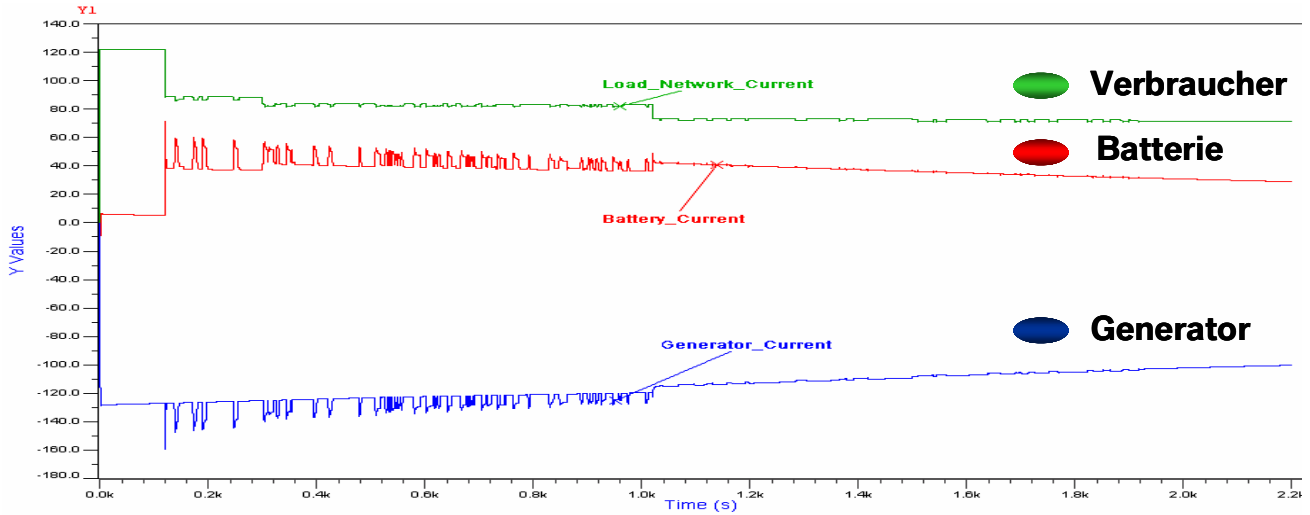
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Vergl. der Simulationsergebnisse (3).



Dymola (Modelica)

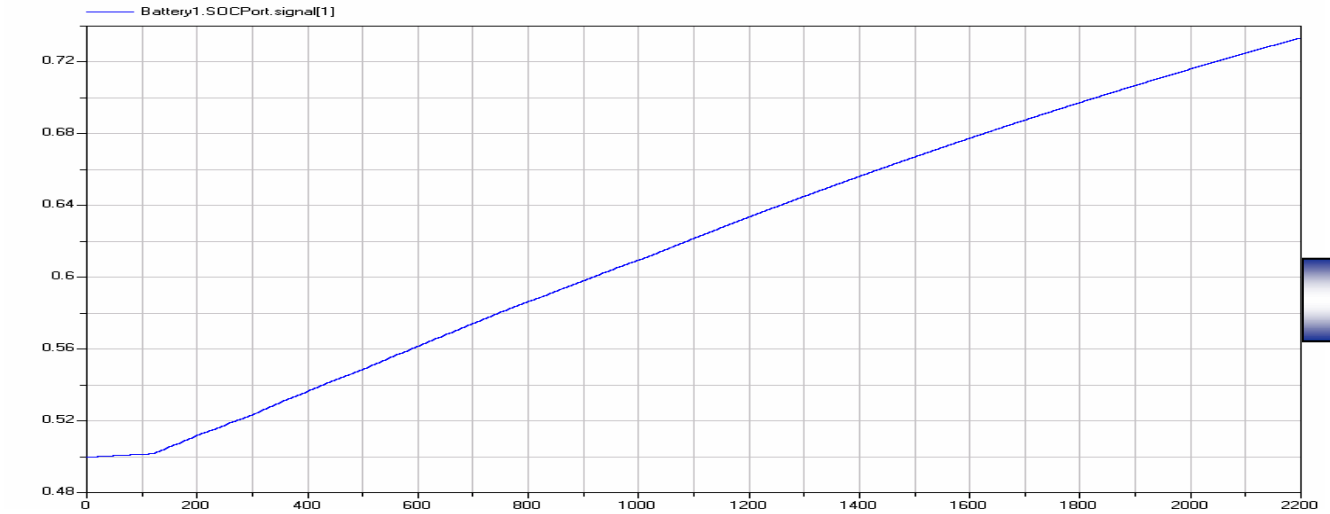
Stromverläufe



SystemVision (VHDL-AMS)

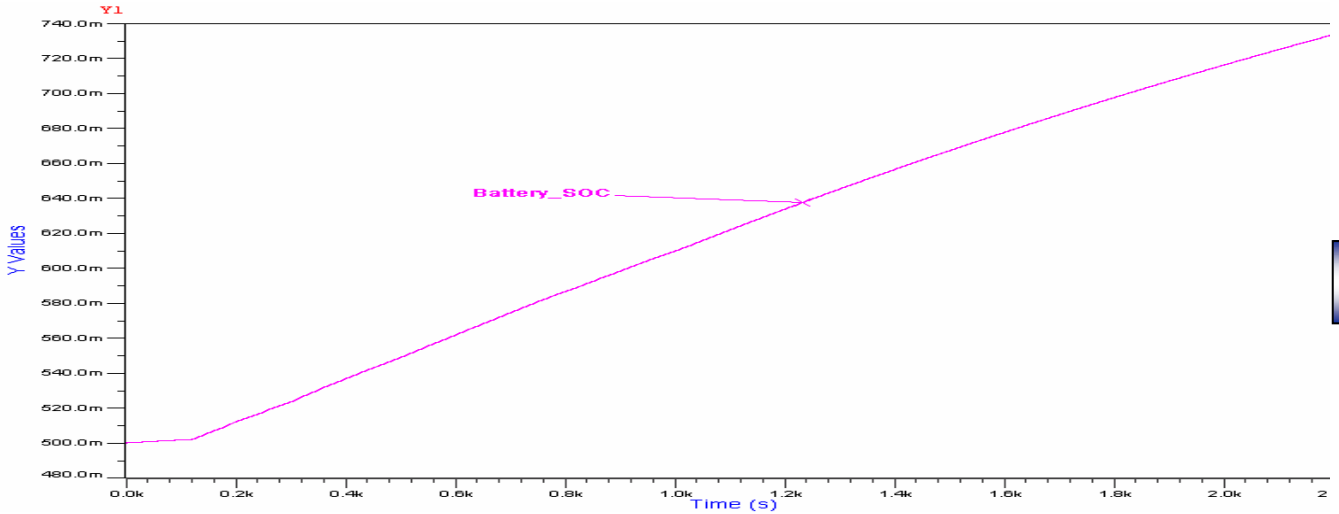
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Vergl. der Simulationsergebnisse (4).



Dymola
(Modelica)

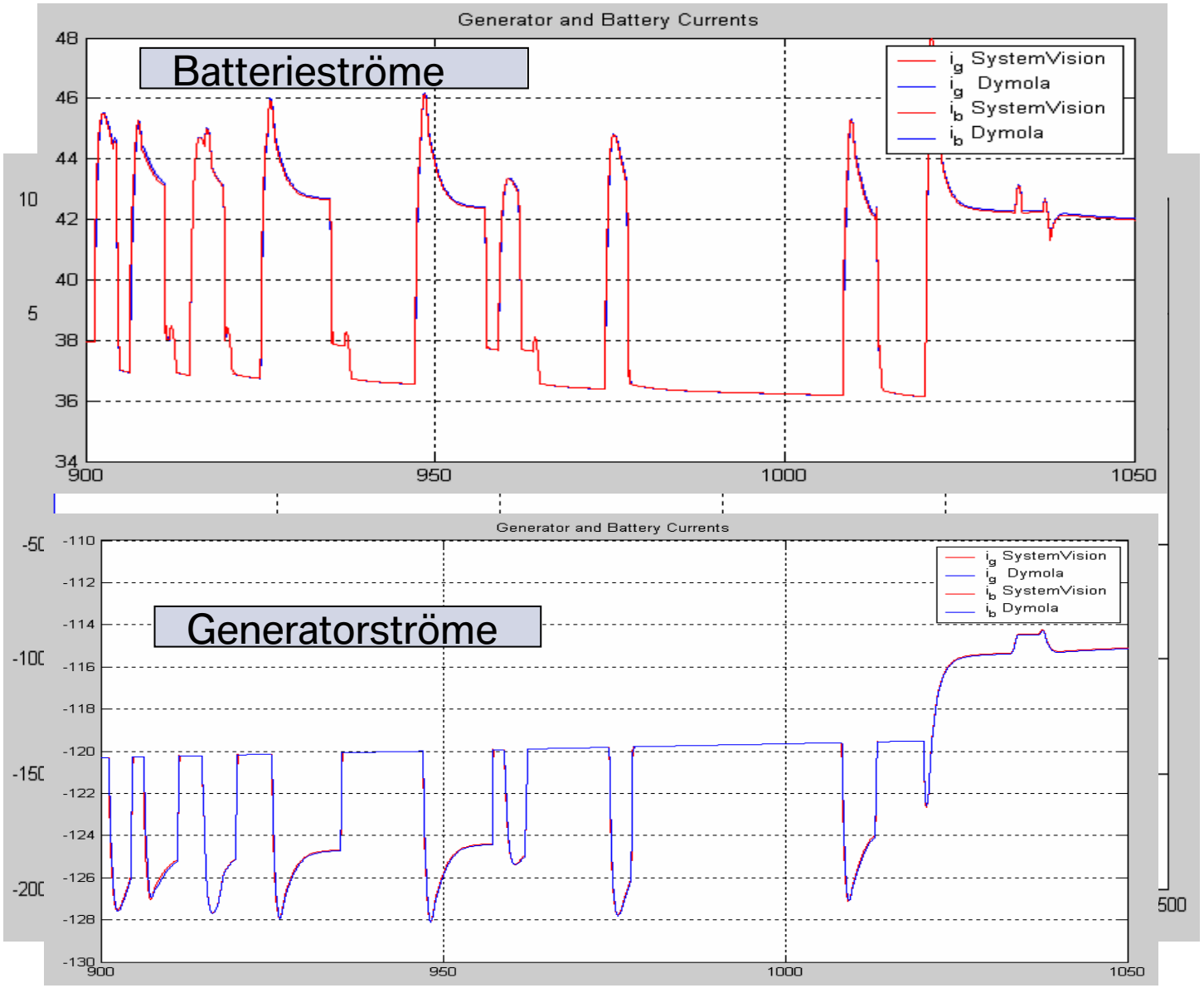
Batterie State of Charge (SOC)



SystemVision
(VHDL-AMS)

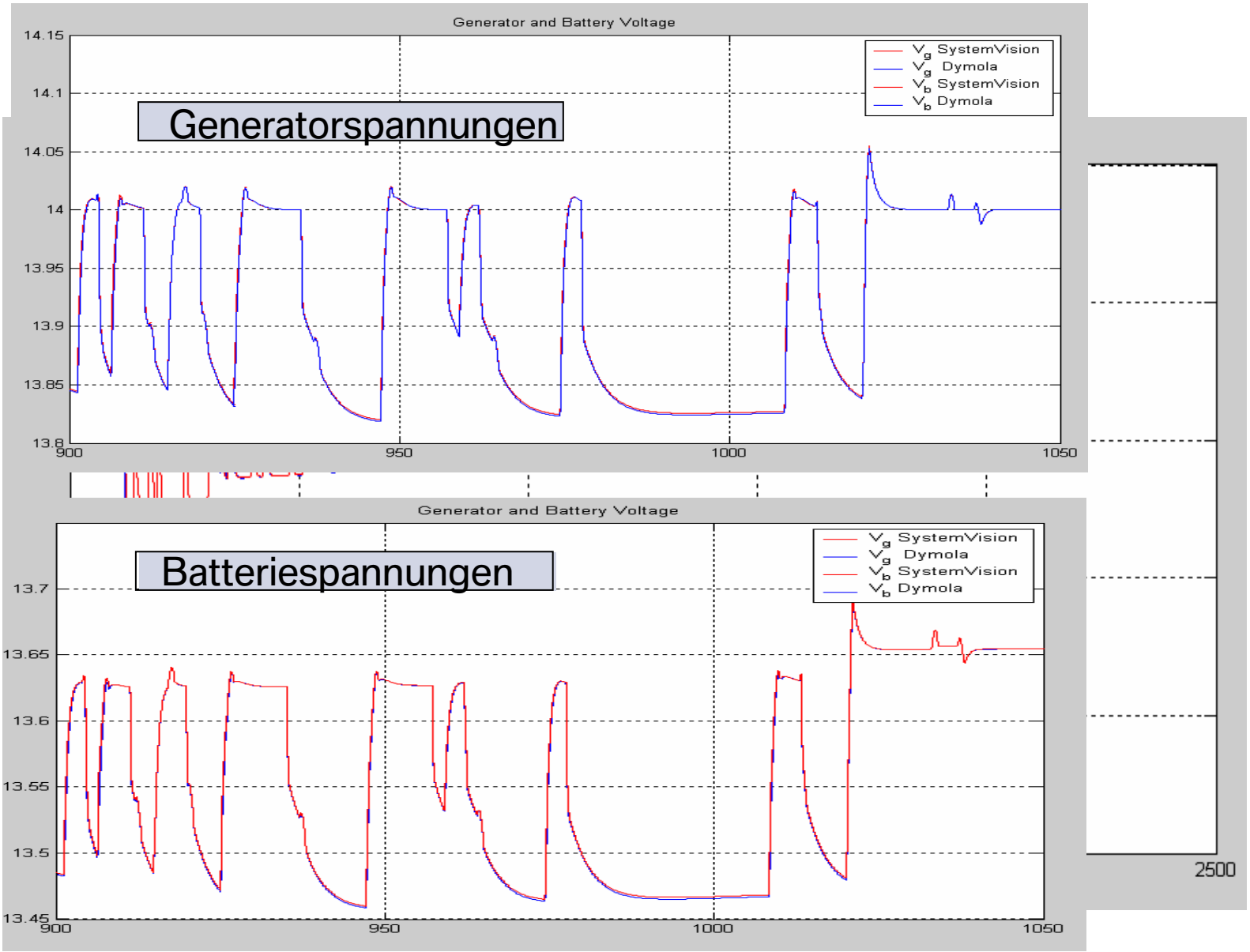
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Vergl. der Simulationsergebnisse (5).



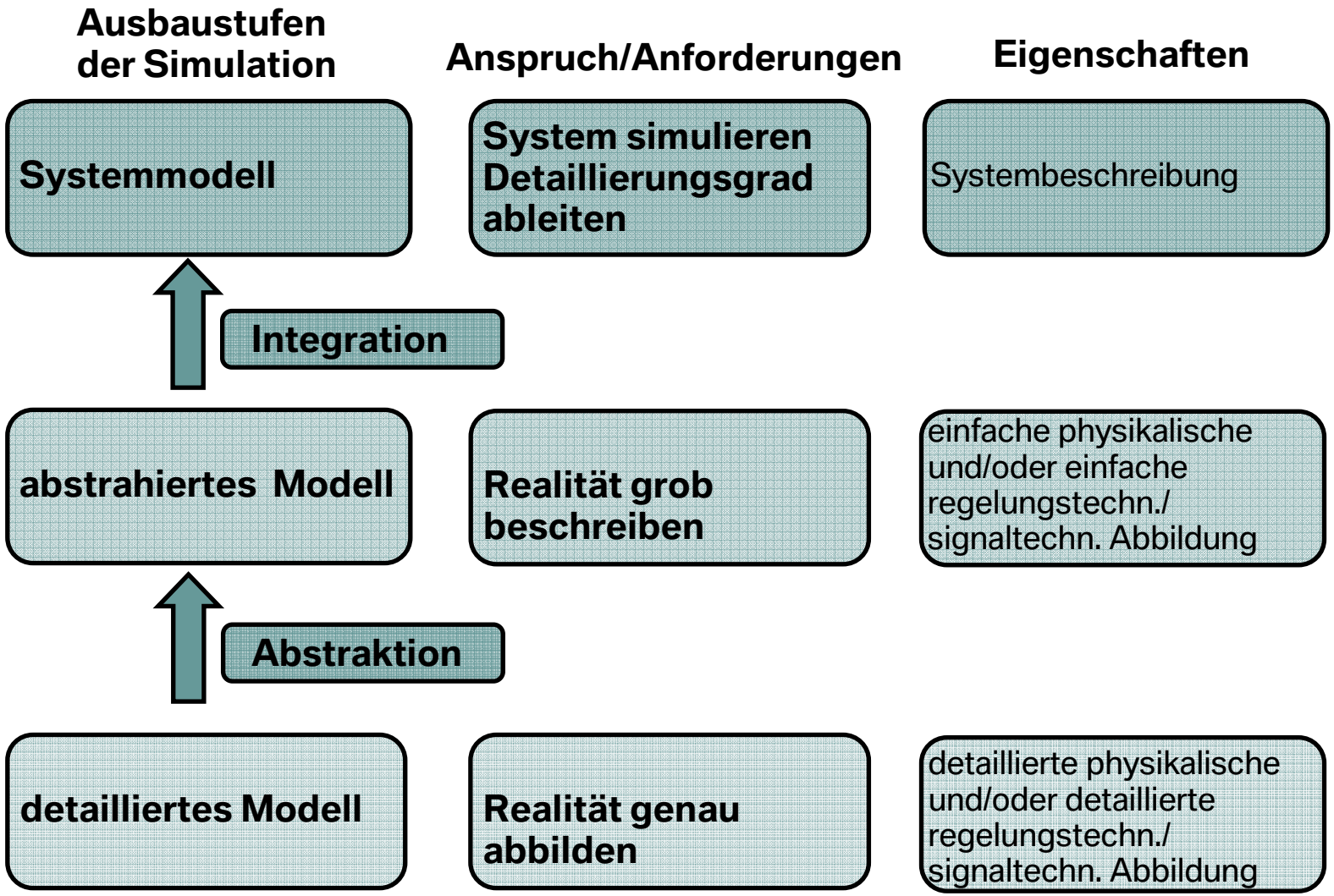
Simulation: Durch Abstraktion zum Systemverständnis.

Standardisierung – Vergl. der Simulationsergebnisse (6).



Simulation: Durch Abstraktion zum Systemverständnis.

Prozesskette des Systementwurfs aus Modellsicht.



Simulation: Durch Abstraktion zum Systemverständnis.

Modellabstraktion am Beispiel Generator.

- Generisches Modell erstellen.
- Parameterreduktion.
- Physikalische Eigenschaften reduzieren.

Detaillierte Eigenschaften

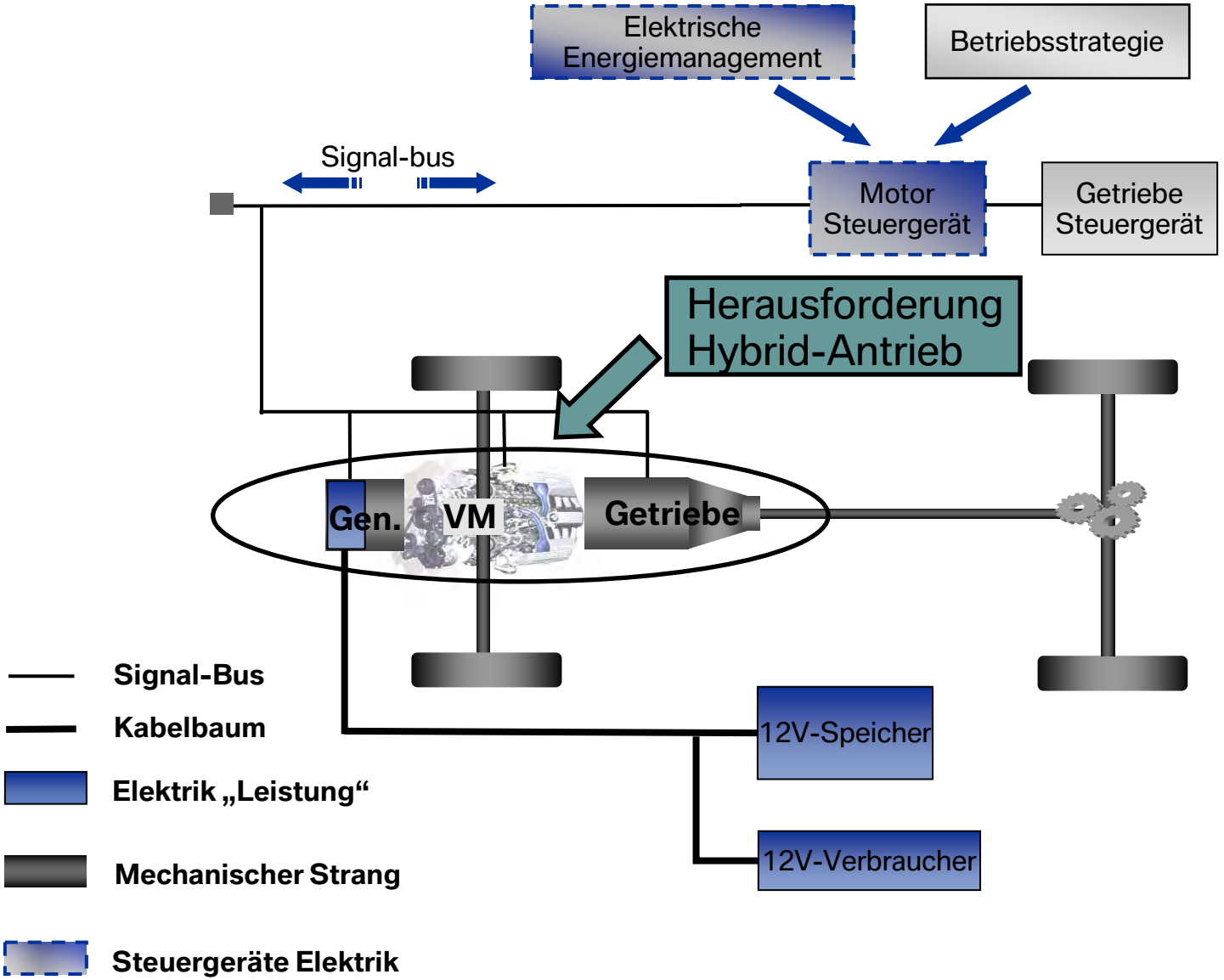
- Parametrierung
  - realer Generator
- Stromanstiegsgeschwindigk.
- Ansprechzeit (response time)
- load response time
- Thermisches Netzwerk
- Maximalstromkennlinie (3D)
  - als f(nMotor,U,T)
- Generator Auslastung
- Wirkungsgrad
- Bus-Signale
  - Sollspannungsvorgabe
  - Auslastungssignal
  - weitere Signale

Abstrahierte Eigenschaften

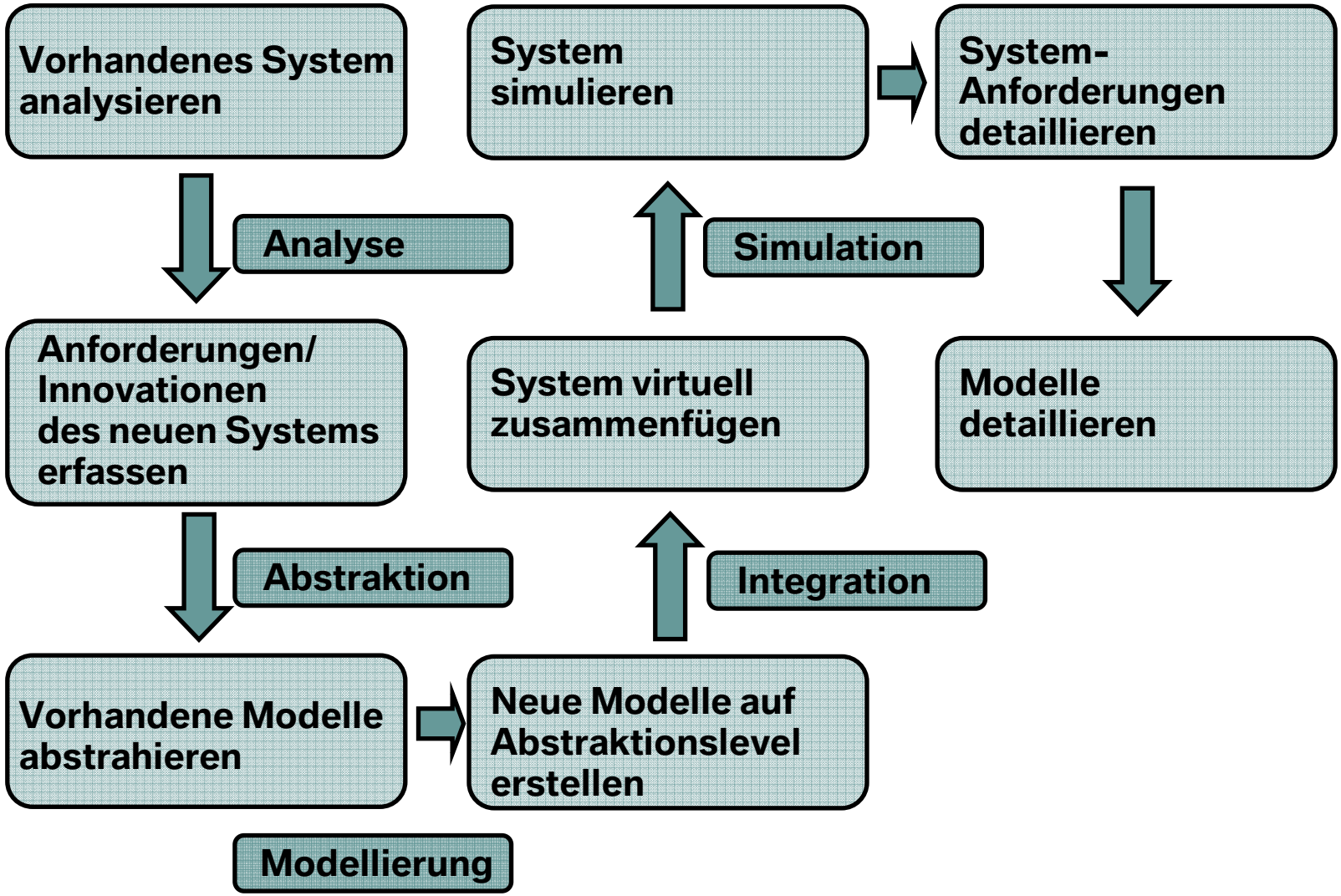
- Parametrierung
  - Zieleigenschaften
- Entfall
- Entfall
- Entfall
- Entfall
- Maximalstromkennl. (2D)
  - f(nMotor,T)
- Generator Auslastung
- Entfall
- Bus-Signale
  - Sollspannungsvorgabe
  - Auslastungssignal
  - Entfall

Simulation: Durch Abstraktion zum Systemverständnis.

Abstraktion am Beispiel Fahrzeugentwicklung.



Simulation: Durch Abstraktion zum Systemverständnis.
Gesamtheitliche Prozesskette des Systementwurfs.



Simulation: Durch Abstraktion zum Systemverständnis.
Ausblick/Vision.

- Die Simulation wird ihren Anteil im Entwicklungsprozess stetig ausbauen.

- Zukünftig werden sich die Tools noch stärker vernetzen müssen, um eine Durchgängigkeit und damit eine kosten- und zeitoptimierte Simulation darstellen zu können.

- In einigen Disziplinen ist die Simulation bereits teilautomatisiert. Es wird notwendig sein, Simulationsverfahren stärker zu automatisieren um schneller belastbare Ergebnisse zu erzeugen.

- Die Modellbibliotheken müssen in Form eines Baukastens stetig ausgebaut werden, sie bilden die Basis für die Qualität der Simulationsergebnisse.

- Standardisierung der Sprachen, der Methoden und des Modell-austausches werden das Fundament der Simulation bilden.

Simulation: Durch Abstraktion zum Systemverständnis.
Zusammenfassung.

- Die Simulation ist wesentlicher Bestandteil der Systemgestaltung von der Konzeptphase bis zum Test und Absicherung.
- Die Tools und Ziele der Simulation sind dabei sehr unterschiedlich.
- Hohe Anforderungen an die Prozesskette, Methodik und Standardisierung werden gestellt. Diese sind zukünftig noch stärker zu verfolgen.
- Durch Abstraktion kann ein Systemverständnis erreicht werden. Erst dann ist es sinnvoll, Modelle zu detaillieren um den Anspruch einer genauen Beschreibung der Realität zu erfüllen.

Simulation: Durch Abstraktion zum Systemverständnis.

- **Herrn Kemayou danke ich für seine Unterstützung in der Ausarbeitung der Folien.**

- **Vielen Dank für Ihre Aufmerksamkeit !**

Simulatorstudien - ein Werkzeug zur Bewertung und Optimierung von Fahrerassistenzsystemen

Dipl.-Ing. M. Brünger-Koch, ASIM 2006

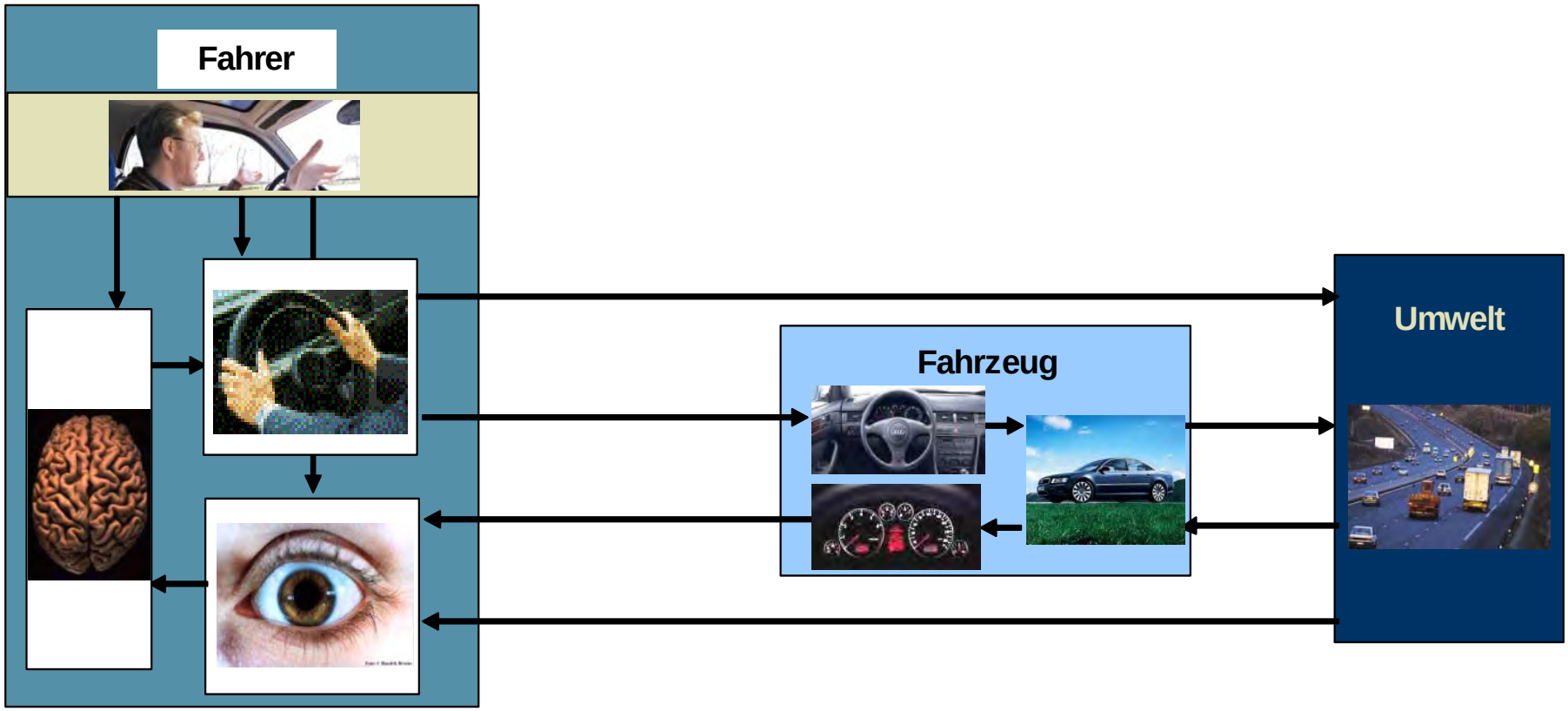
Deutsches Zentrum
DLR für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft


... und wer fährt?
BG
DVR

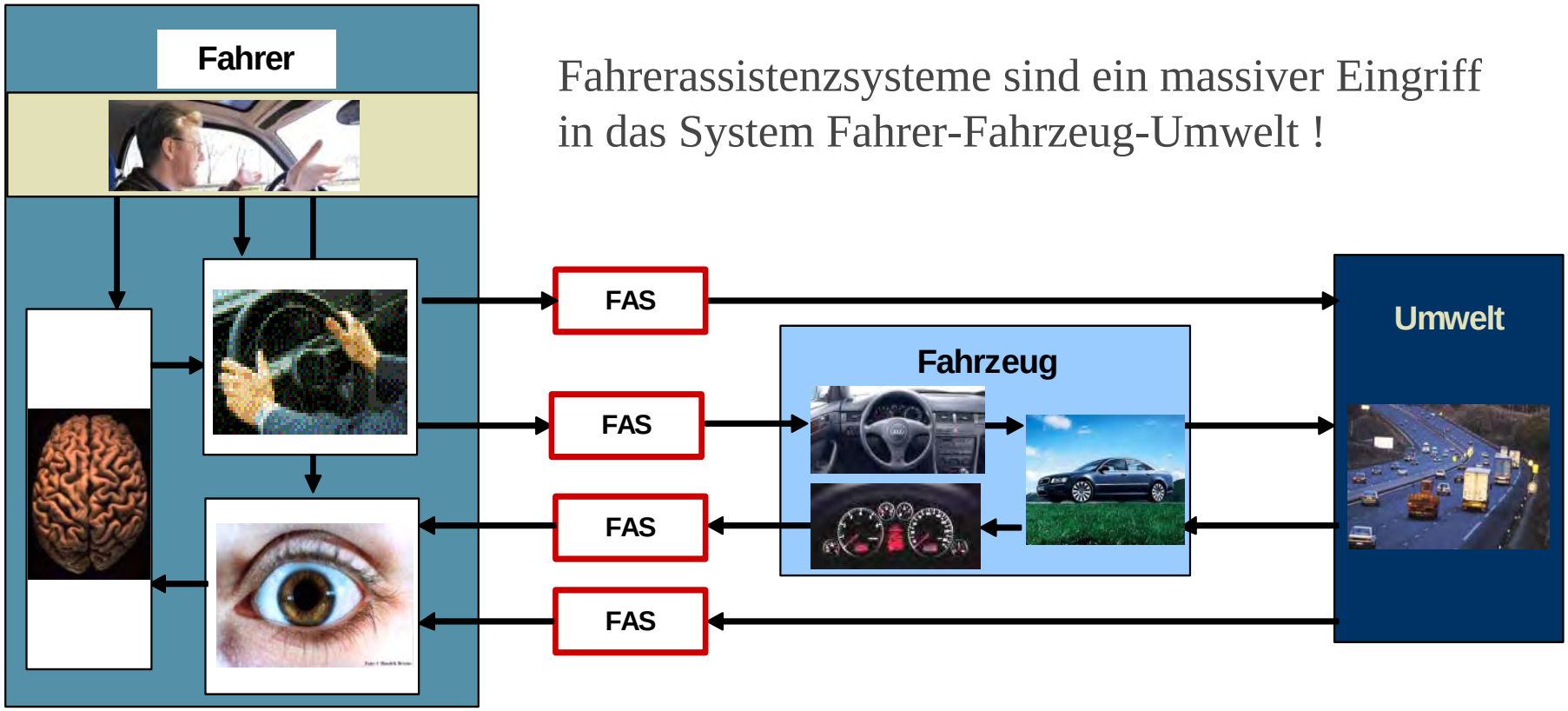
Agenda

- Fahrerassistenzsystem
- Simulation vs. Realität
- FAS / Validitäts-Studien
- Zusammenfassung

System Fahrer-Fahrzeug-Umwelt

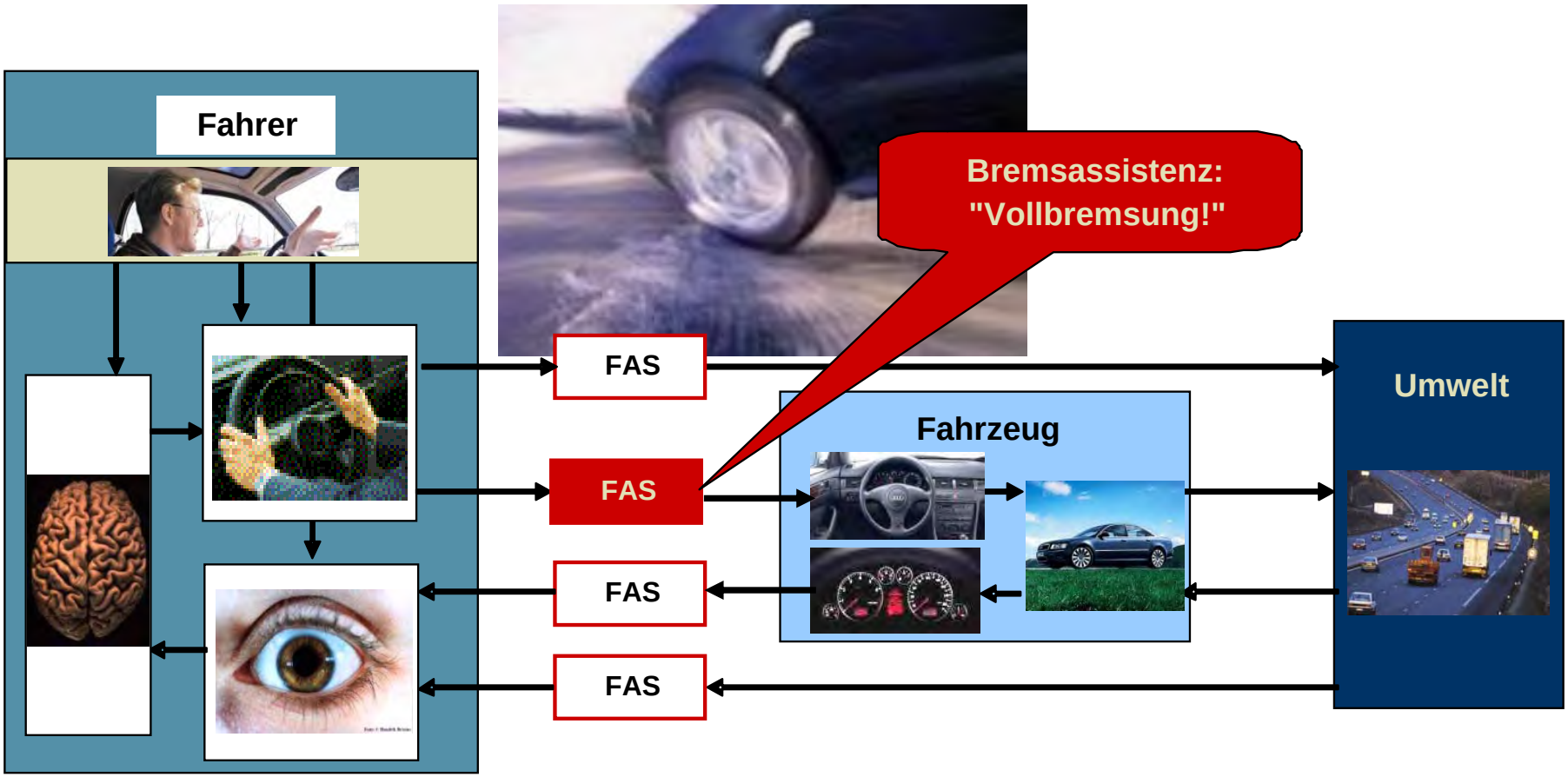


Fahrerassistenzsysteme (FAS)

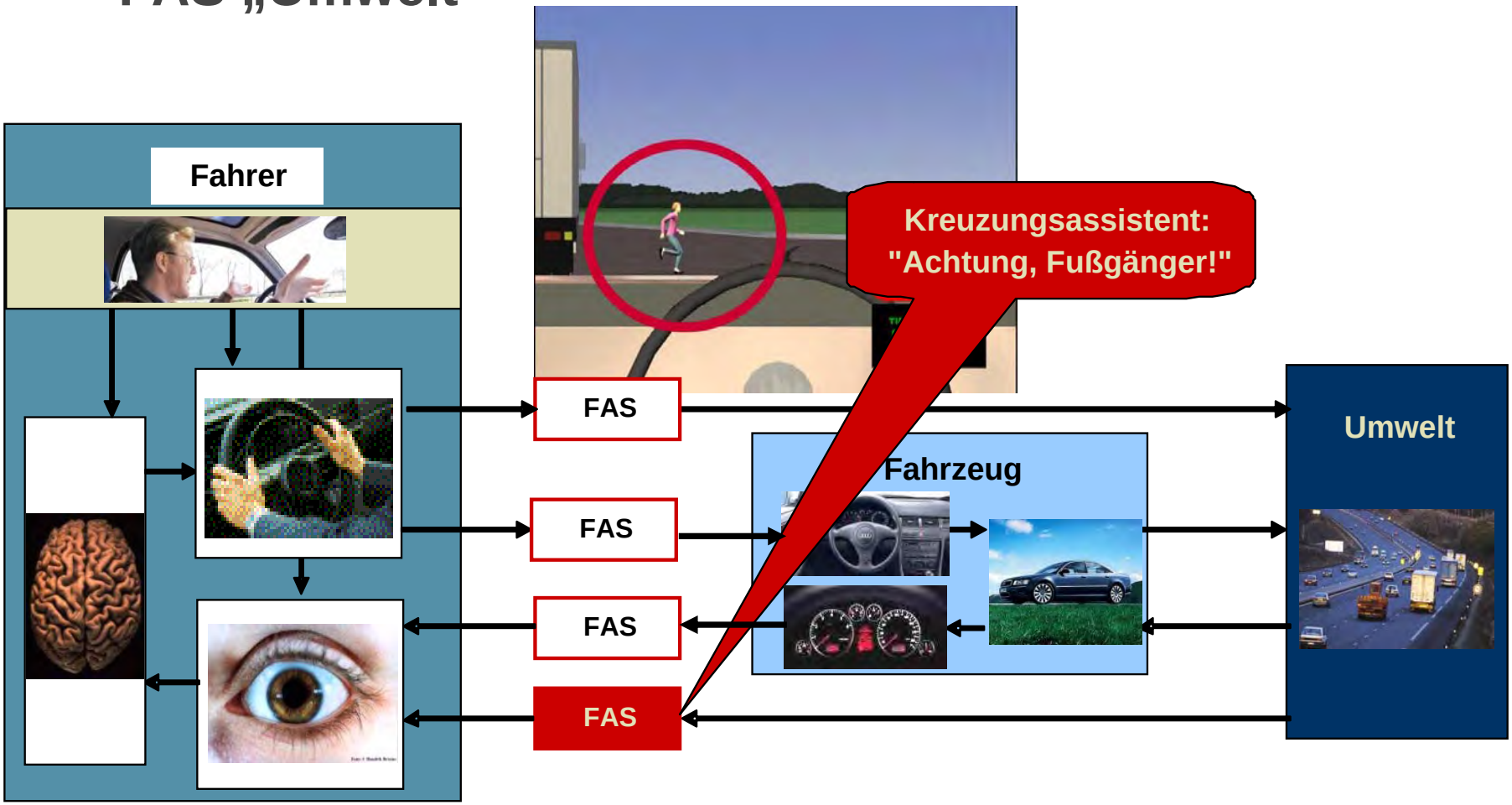


Fahrerassistenzsysteme sind ein massiver Eingriff in das System Fahrer-Fahrzeug-Umwelt !

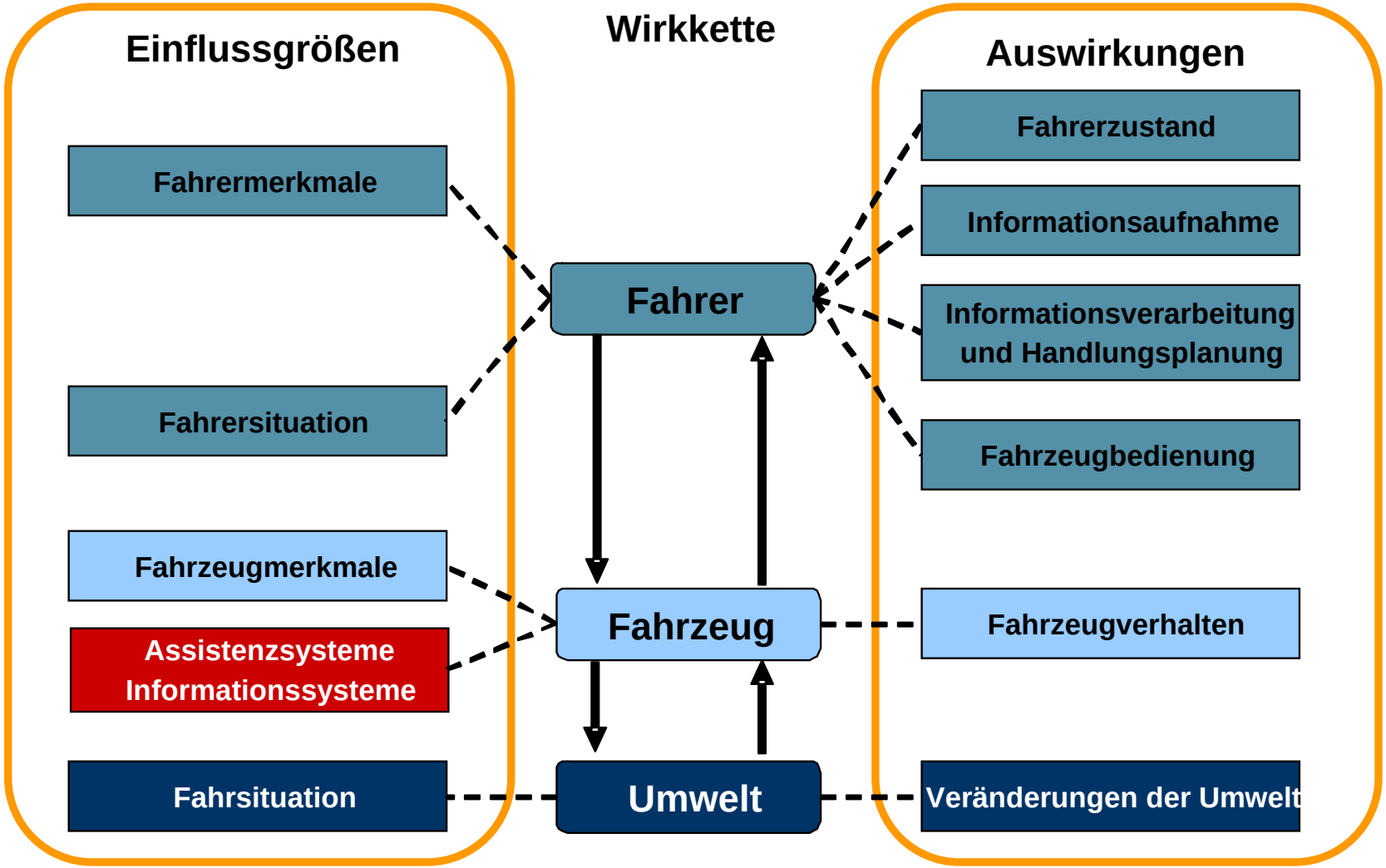
FAS „Fahrzeugbedienung“



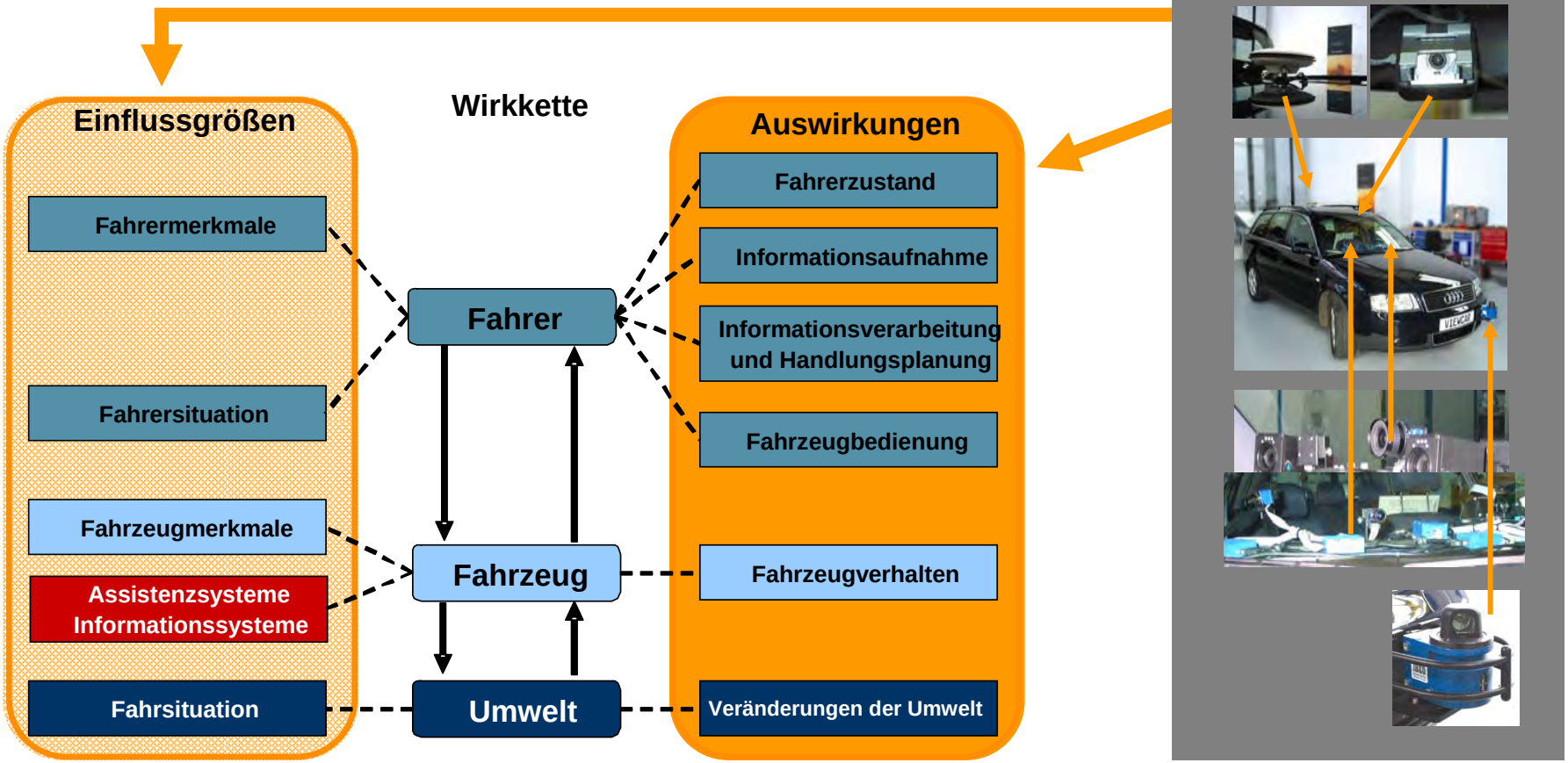
FAS „Umwelt“


Fahrer
Kreuzungsassistent:
"Achtung, Fußgänger!"
FAS
FAS
FAS
FAS
Fahrzeug
Umwelt

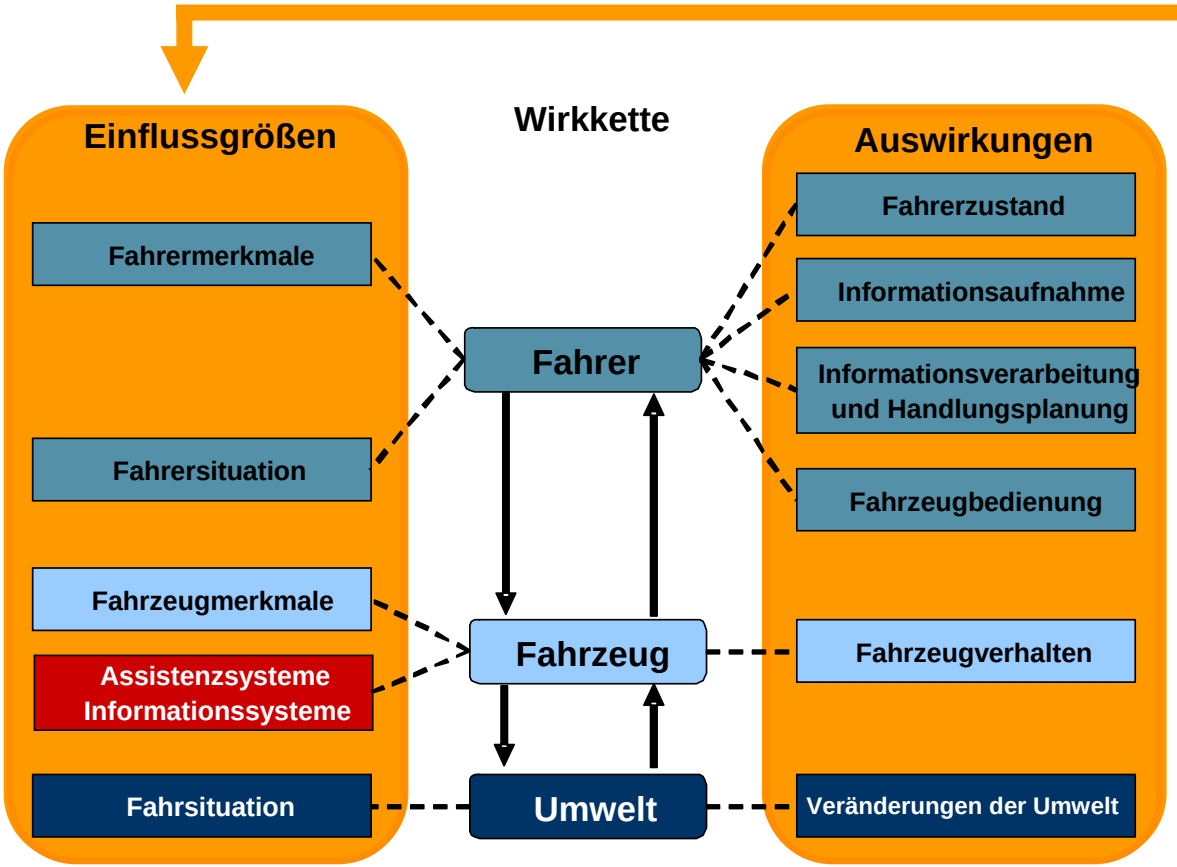
Modellvorstellung



ViewCar: Auswirkungen



VRLab: Einflüsse



Konzept Übertragbarkeit

- Ziel:
  - Vergleichbare Wirkungen für Einflussgrößen
  - Bestimmung der Übertragungsfunktion
- Weg:
  - Variation wesentlicher Einflussgrößen
  - Ausgewählte Stufen im Realverkehr

	ViewCar	VR-Labor
Wach	x	x
Schläfrig	x	x
Müde		x

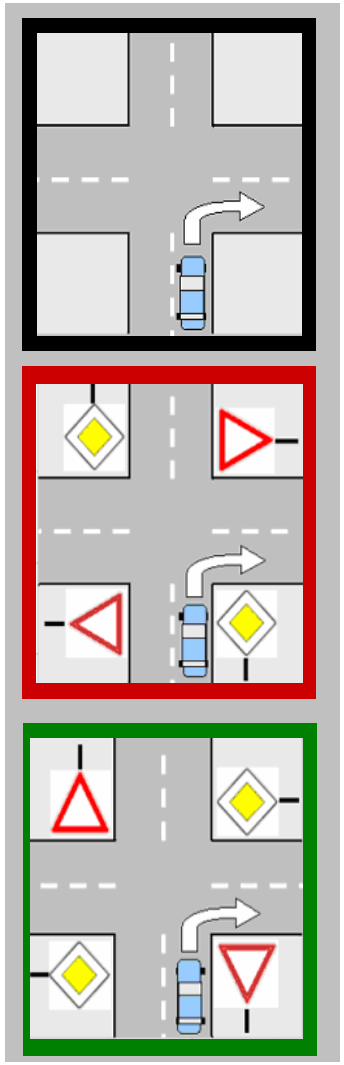
	ViewCar	VR-Labor
Trocken	x	x
Nass	x	x
Eis		x

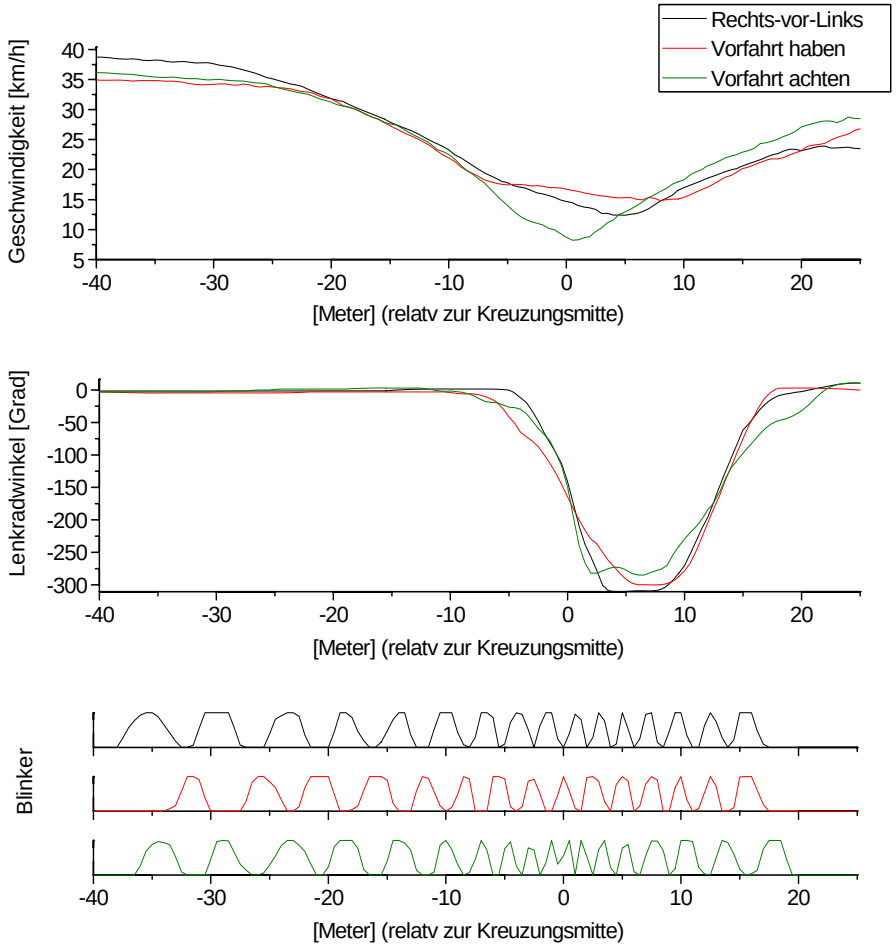
	ViewCar	VR-Labor
Hell	x	x
Dunkel	x	x
Nebel		x

	ViewCar	VR-Labor
Ohne FAS	x	x
Information	x	x
Unterstützung		x

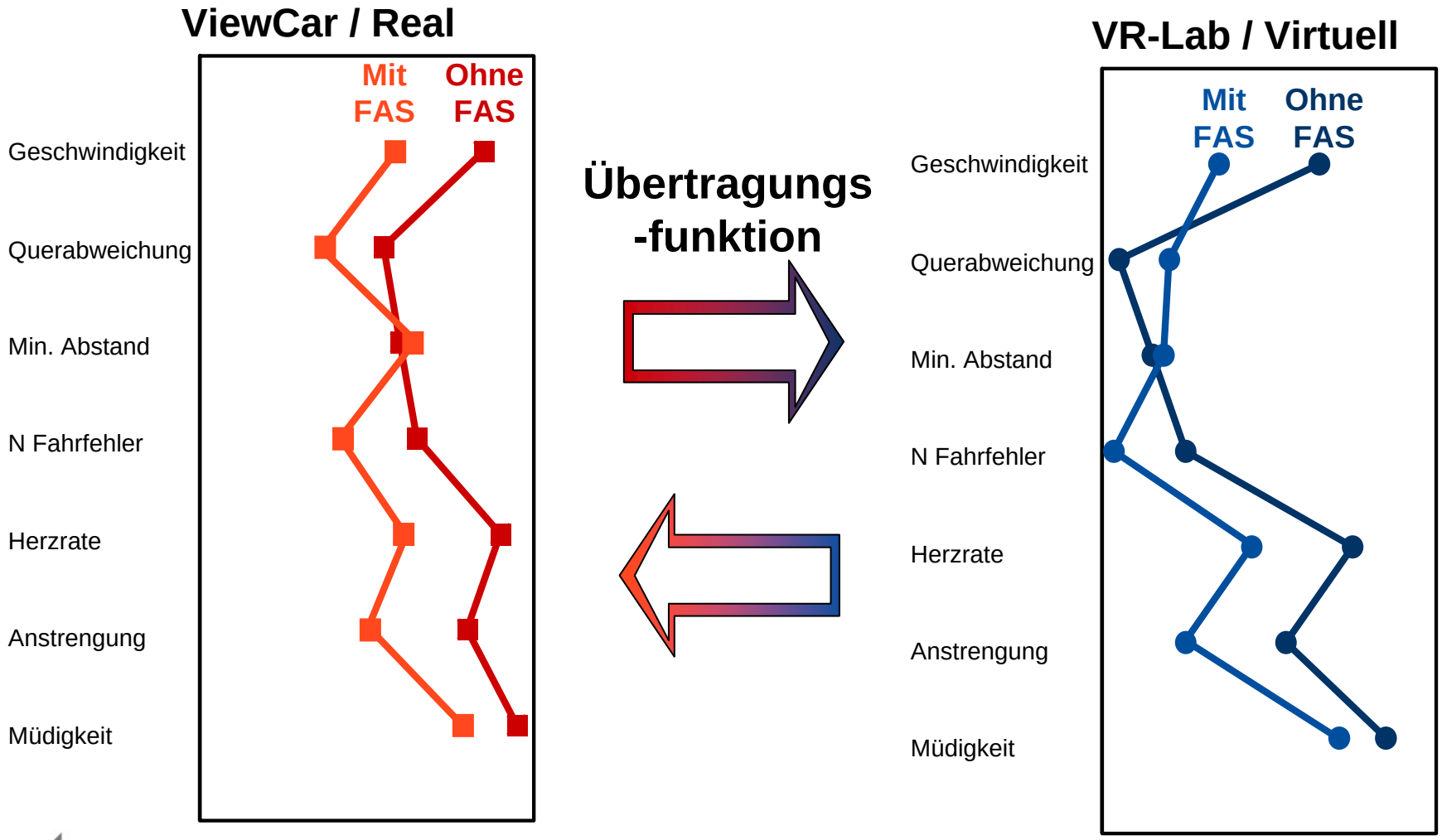
Beispiel Kreuzungsfahrten

- Rechts Abbiegen
  - Rechts-vor-Links
  - Vorfahrt haben
  - Vorfahrt-Achten





Beispiel Effekt Vorfahrtsregelung

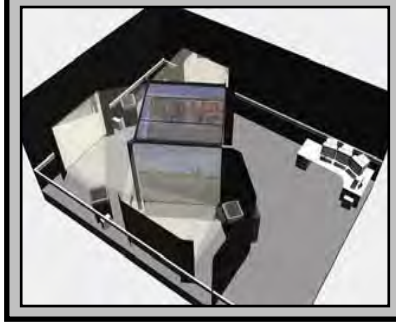


Realitätsnähe/Komplexität

- **Realer Verkehr**

- **Reale Prototypen**

- **Virtuelle Welten**



- **Virtuelle Komponenten**

Kosten

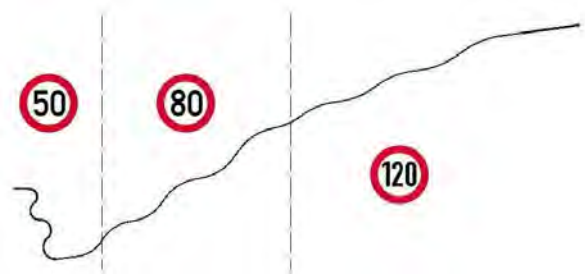
Validität

- Kenngrößen für Validität
  - Fahrdaten (Virtuell vs. Real)
  - Subjektive Bewertung
  - Simulatorkrankheit

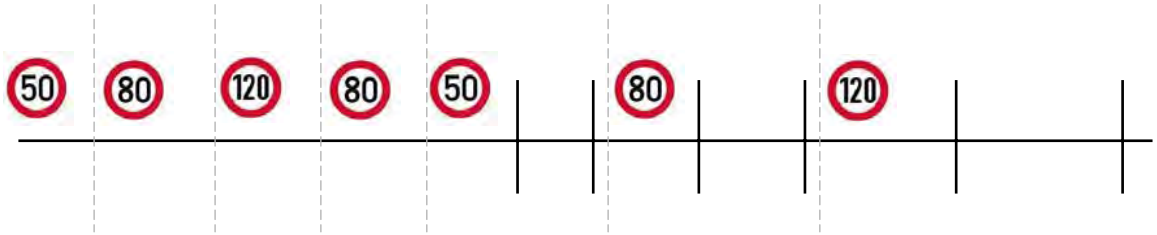
- Erhöhung der Validität durch…
  - Hohe visuelle Immersion
  - Reale Komponenten
  - Realistisches Motion Cueing

Experimental Design
Strecken S2 und S3

- Fahrmanöverspezifische Strecken
  - Kurvenfahrt (S2)

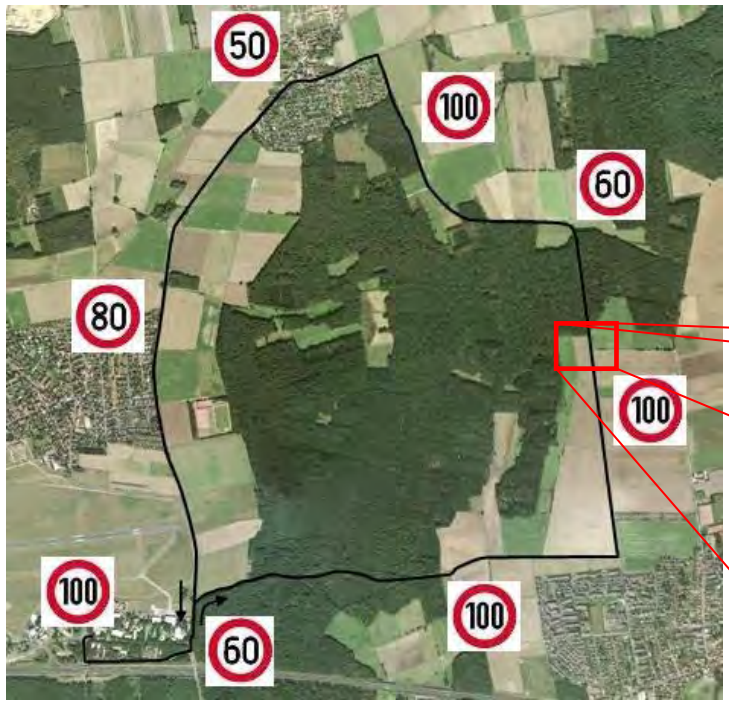


  - Bremsen / Beschleunigen (S3)



Experimental Design
Strecken S1 und R

- Virtuelles vs. reales Fahren
- Erhebung objektiver Daten



Virtuell
(S1)

Real
(R)

Experimental Design
Parametervariation

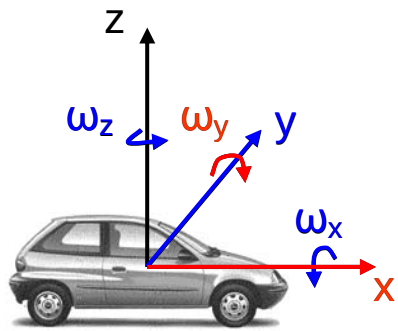
- Beschleunigungsabteile bei verschiedenen Fahrmanövern
  - Kurvenfahrt :

    a_y, w_x, w_z
  - Bremsen:

    a_x, ω_y

$$\bar{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

$$\bar{\omega} = \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix}$$



- Hier: keine gemeinsamen Komponenten (DoF).

  ⇒ Unterschiedliche Parameter für das Tuning

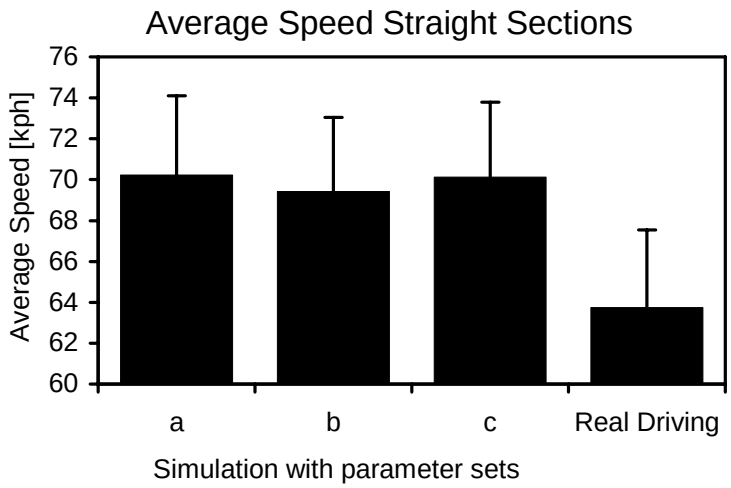
Fahrmanöver	***Parameter Set-up***		
	a	**b**	**c**
Kurvenfahrt			
Bremsen			

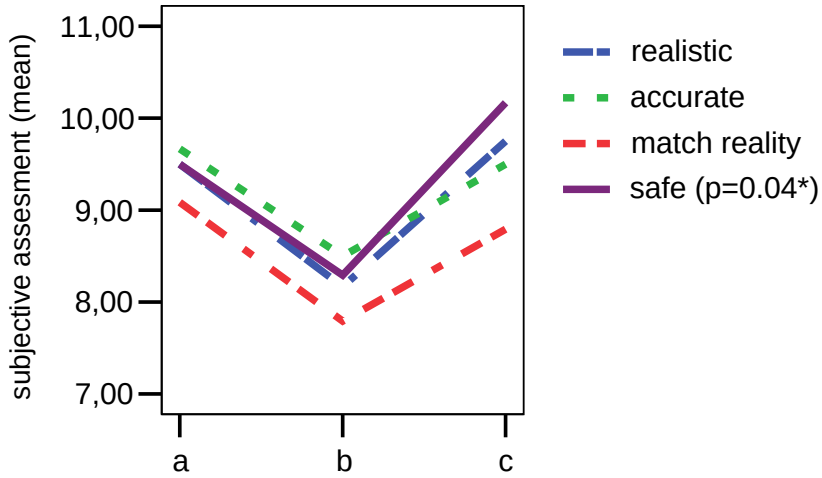
1: Original Tuning
2: DLR Tuning

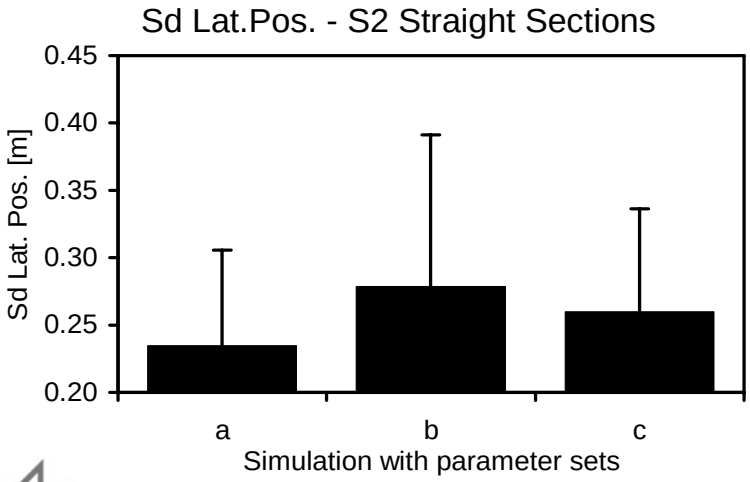
Experimental Design
Auswirkung der Parameter Sets

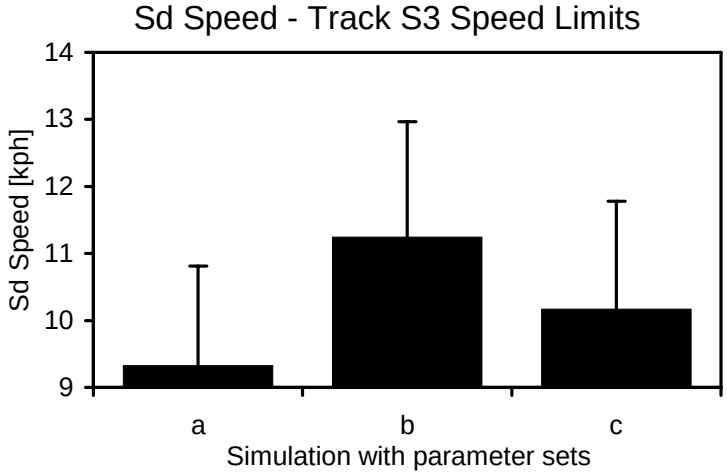
- Set c vs. b

Ergebnisse


Average Speed Straight Sections
Average Speed [kph]
76
74
72
70
68
66
64
62
60
a
b
c
Real Driving
Simulation with parameter sets

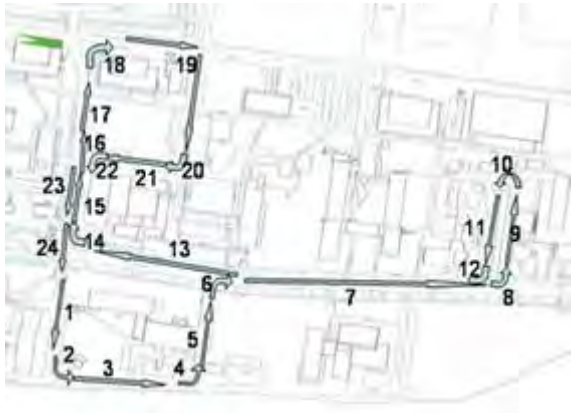

subjective assesment (mean)
11,00
10,00
9,00
8,00
7,00
a
b
c
parameter
realistic
accurate
match reality
safe (p=0.04*)


Sd Lat.Pos. - S2 Straight Sections
Sd Lat. Pos. [m]
0.45
0.40
0.35
0.30
0.25
0.20
a
b
c
Simulation with parameter sets

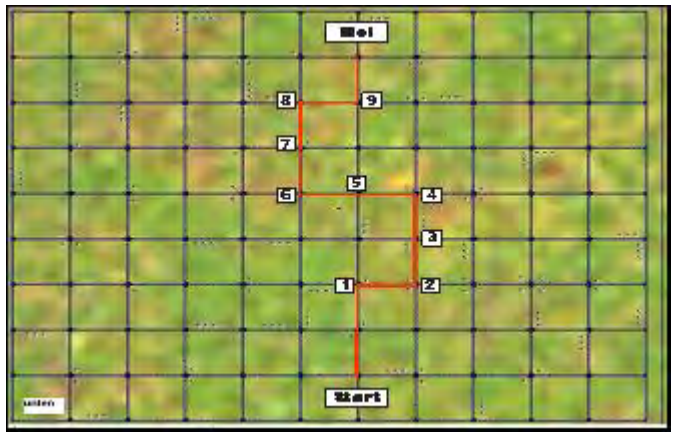

Sd Speed - Track S3 Speed Limits
Sd Speed [kph]
14
13
12
11
10
9
a
b
c
Simulation with parameter sets

Weitere durchgeführte Studien

- Verhalten an Kreuzungen:
  - Einfluss von Kreuzungsmerkmalen und Fahrerabsicht
  - Vergleich Real- und Simulatorfahrten



- Navigation:
  - Optimale Gestaltung der Sprachausgabe
  - Einfluss des Zeitpunkts der Ausgabe



Zusammenfassung

- Hintergrund für neue FAS:
  - Untersuchung des Fahrerverhaltens und von Fahrfehlern zur Ableitung von Anforderungen an Unterstützung
- Entwicklung und Bewertung von FAS:
  - Fahrtests in allen Stadien der Entwicklung
  - Frühe Berücksichtigung von Fahrerwünschen und Vermeidung von Problemen
  - Berücksichtigung sicherheitskritischer Situationen
- Methode:
  - Simulation: kritische Situationen, HIL-Test
  - Prototypen - Fahrtests: Realität
- **Fazit:**
  - **Die Entwicklung neuer Fahrerassistenzsysteme benötigt die geschickte und effektive Kombination von Simulation und Realfahrten**


Ich hab' ganz vergessen, den Tank zu kontrollieren!
Nicht zu fassen! Ächz!
Und du willst ein intelligentes Auto sein?! Pfui!
Und was ist mit dir? Anscheinend bist du zu blöd, um die Tankanzeige abzulesen!

Entwicklung von Experiment-Funktionen für den Flugversuchsträger ATTAS

M. Gestwa, D. Leißling, J.-M. Bauschat

(martin.gestwa | dirk.leissling | michael.bauschat)@dlr.de

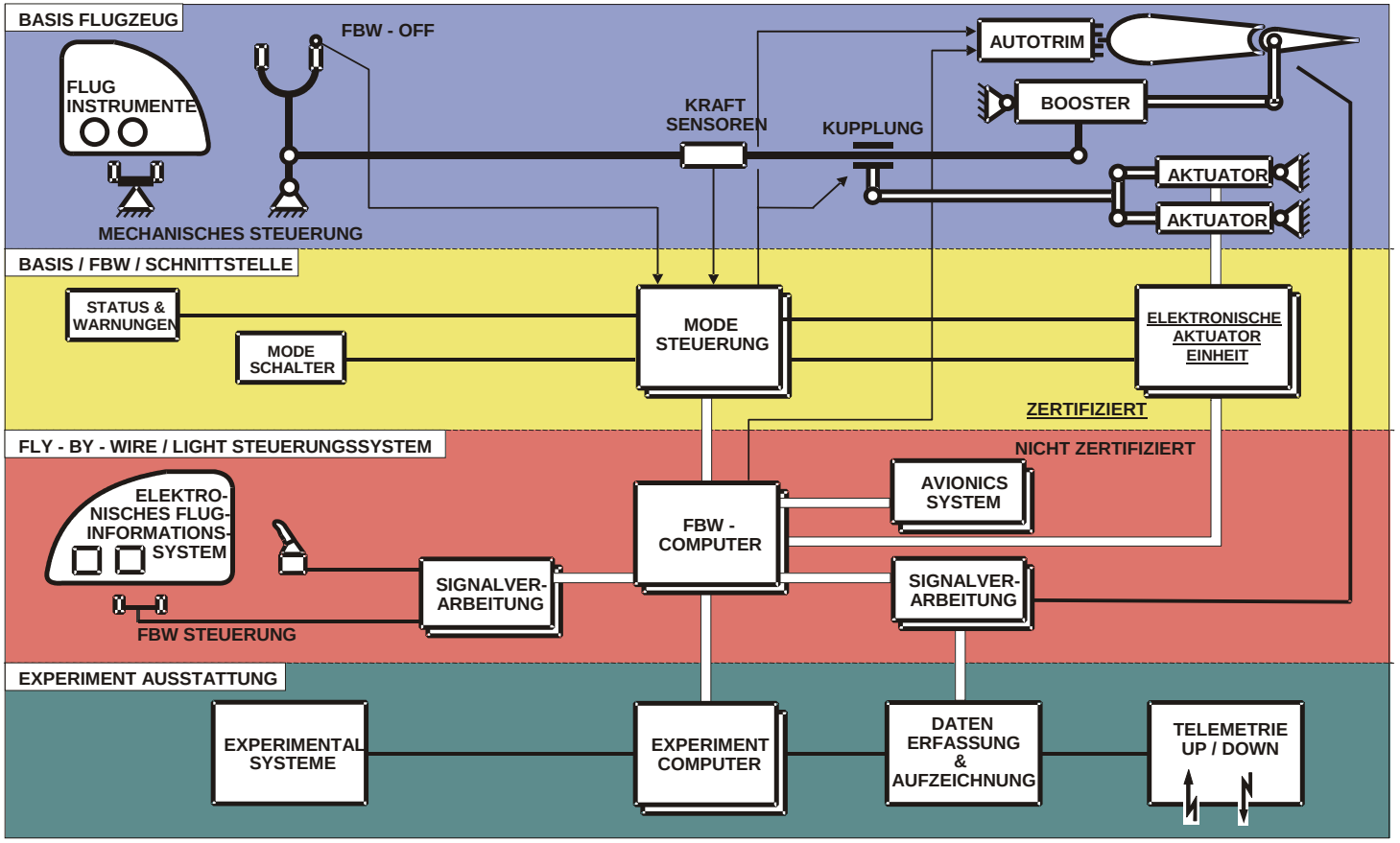
Deutsches Zentrum
DLR für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft

Treffen der ASIM/GI-Fachgruppen STS und GMMS am 20./21.02.2006 in München

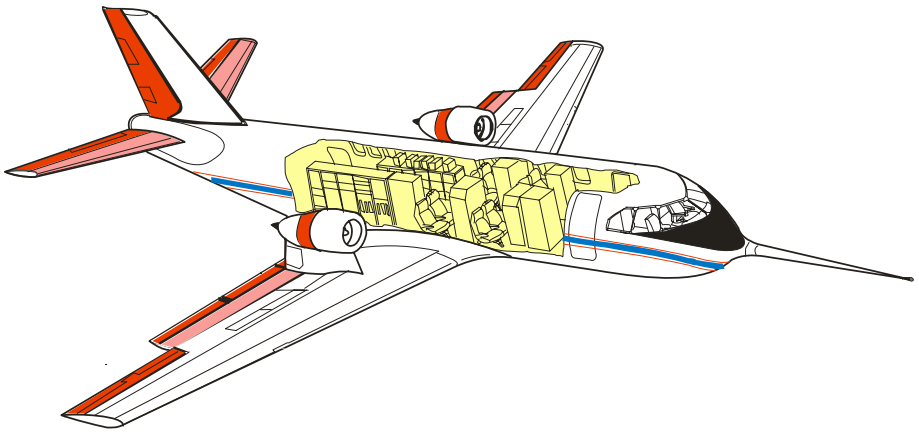
Vortragsgliederung

- Flugversuchsträger ATTAS
- ATTAS Software-Entwicklungsumgebung
- Entwicklung von Experiment-Funktionen
- Anwendungen

Konzept des ATTAS Flugsteuerungssystems

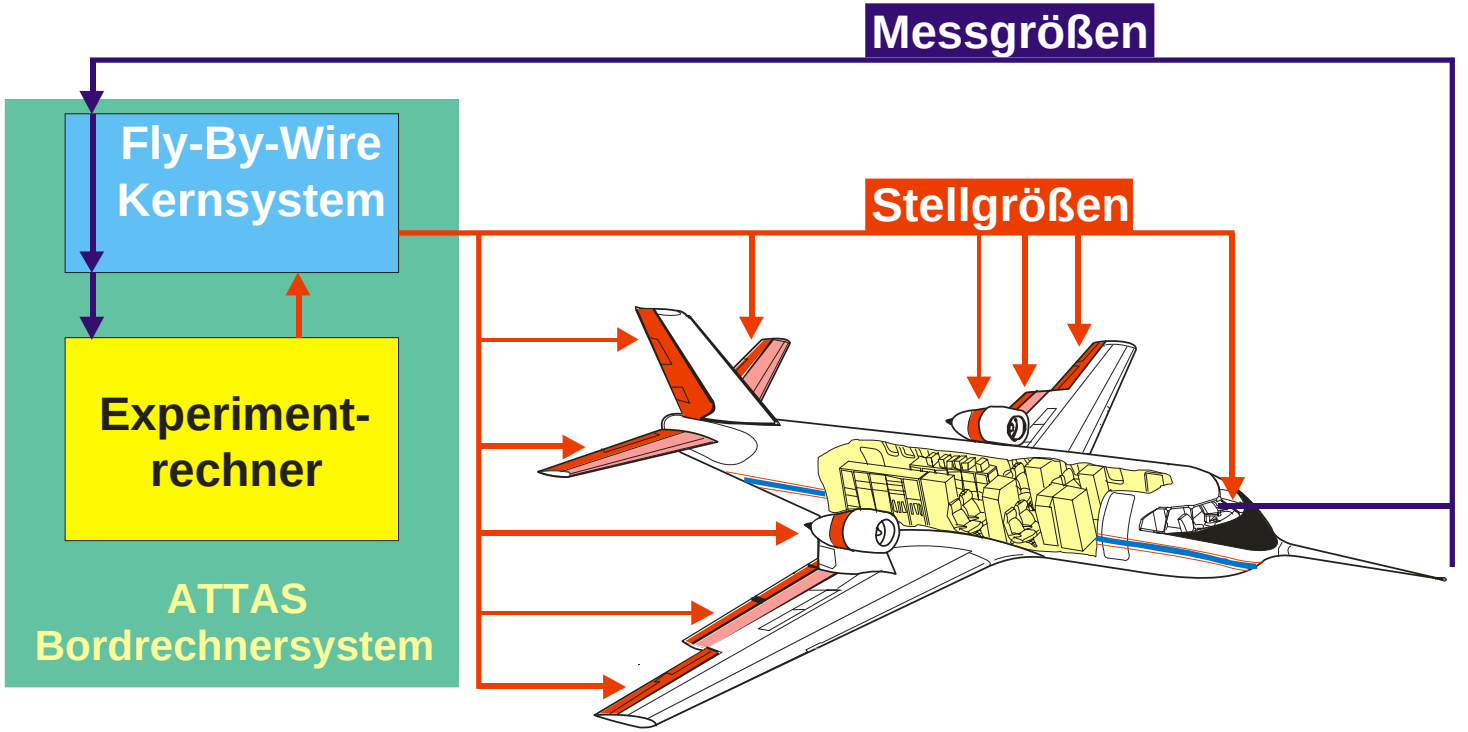


DAS ATTAS Bordrechnersystem



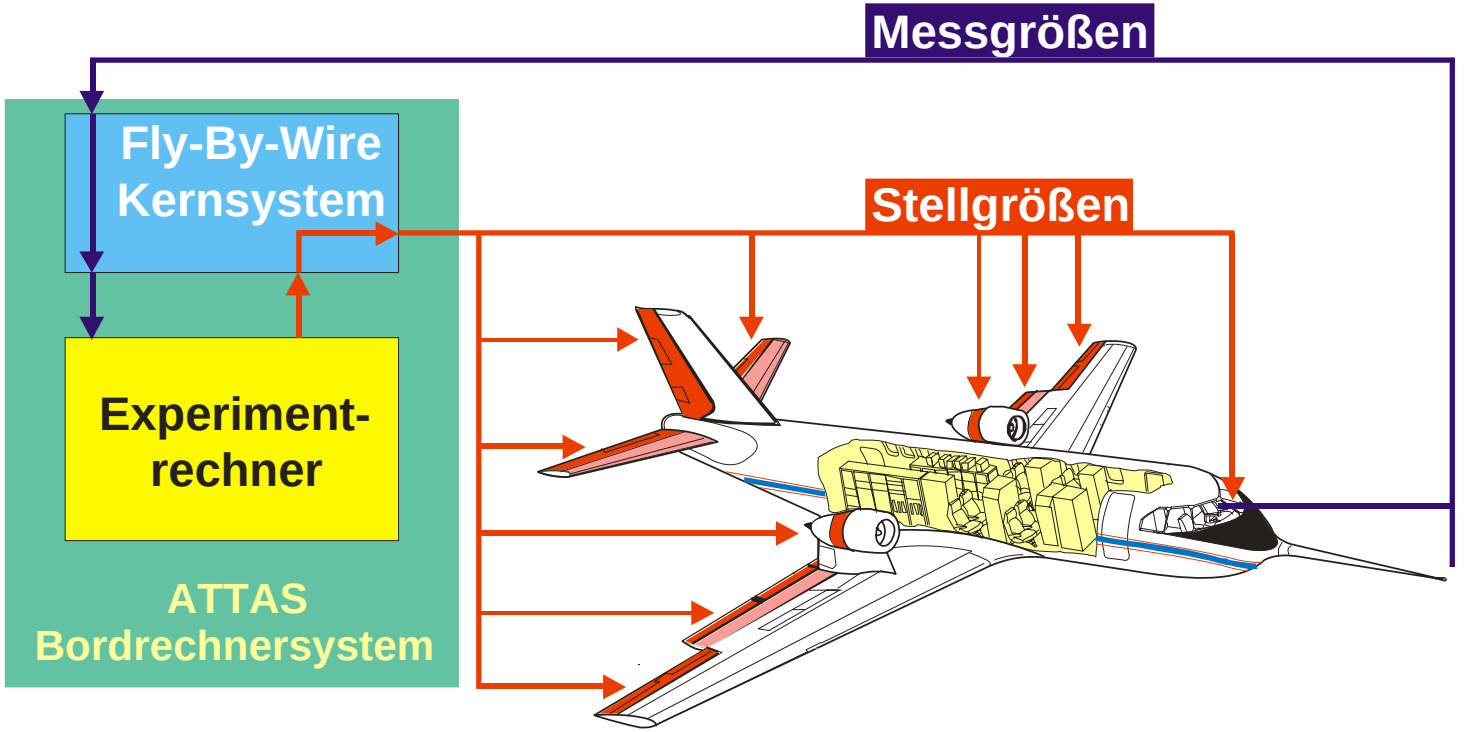
Basismodus

DAS ATTAS Bordrechnersystem



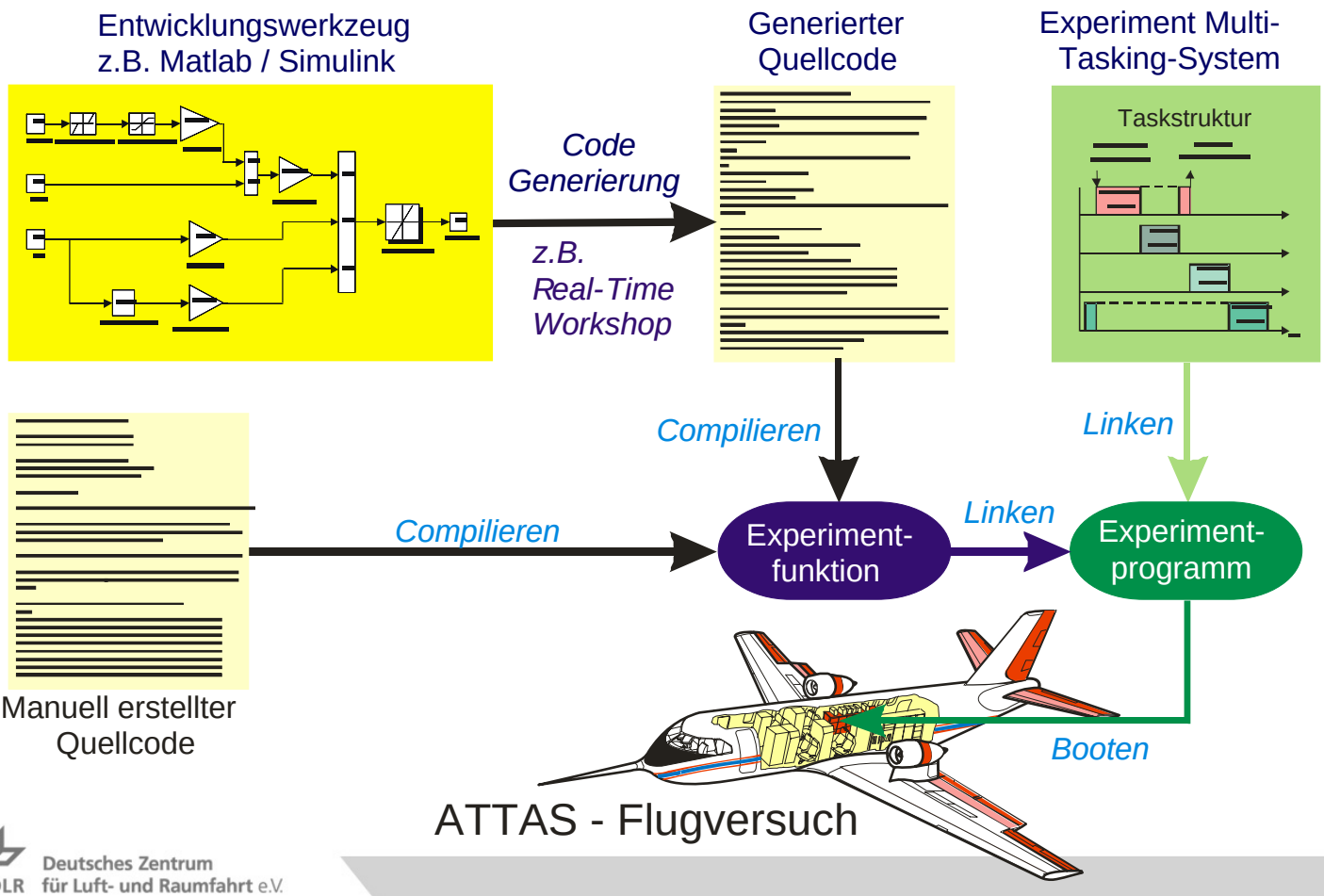
Fly-by-Wire-Modus

DAS ATTAS Bordrechnersystem



Simulationsmodus

Prinzip der Heterogenen Software-Entwicklung



ATTAS-Coder - Ein modifizierter Real-Time Workshop

Anforderungen an die Code-Generierung:

1. Alle Simulink-Blöcke müssen unterstützt werden
2. Der generierte Quellcode muss ein in sich geschlossenes Modul sein
3. Verschiedene Simulink-Modelle müssen parallel nutzbar sein
4. Pro Modell-Bibliothek immer nur eine Instanz des zugehörigen Simulink-Modells

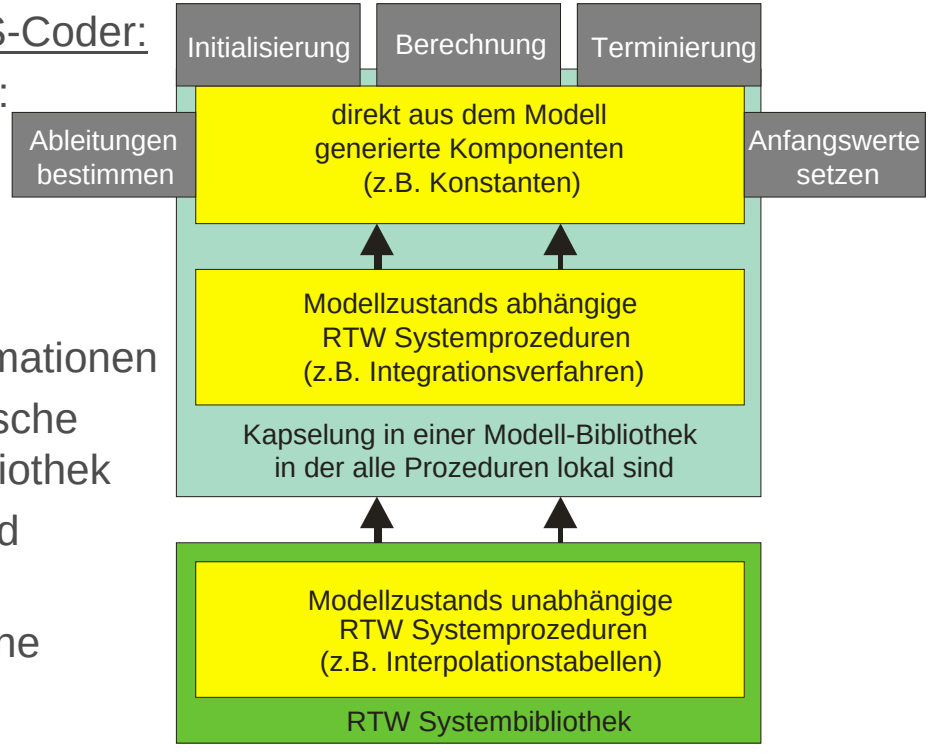
Alle RTW Standard Coder erfüllen diese Anforderungen nicht

$\Rightarrow$ Entwicklung des ATTAS Coders

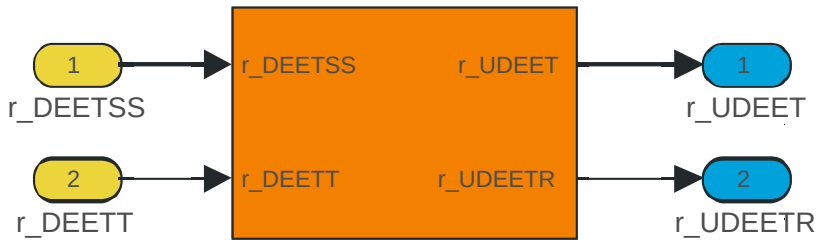
ATTAS-Coder - Ein modifizierter Real-Time Workshop

Modifikationen für den ATTAS-Coder:

- Teilung in zwei Bibliotheken:
  - vom Zustand des Modells abhängige
  - vom Zustand des Modells unabhängige
- Kapselung der Modell-Informationen
- typische Simulations-technische Schnittstelle der Modell-Bibliothek
- Zugriff auf Anfangswerte und Ableitungen
- Pro Modell-Bibliothek nur eine Instanz



ATTAS Coder – Generierte Schnittstelle des Modells

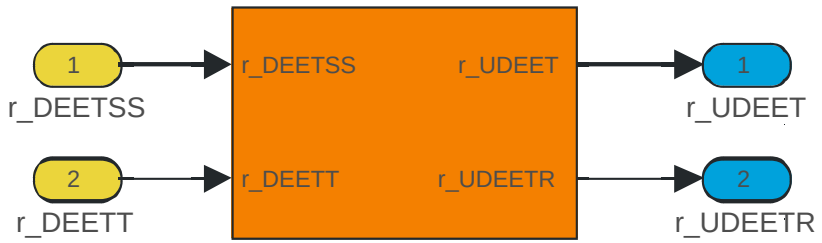


Umsetzung der In- und Outports:

- eindimensionale Felder

```
r_Inport(1) = …       // Wert für Inport r_DEETSS
r_Inport(2) = …       // Wert für Inport r_DEETT

<Aufruf der Berechnungsfunktion des Modells>

… = r_Outport(1)      // Wert vom Outport r_UDEET
… = r_Outport(2)      // Wert vom Outport r_UDEETR
```

ATTAS Coder – Generierte Schnittstelle des Modells



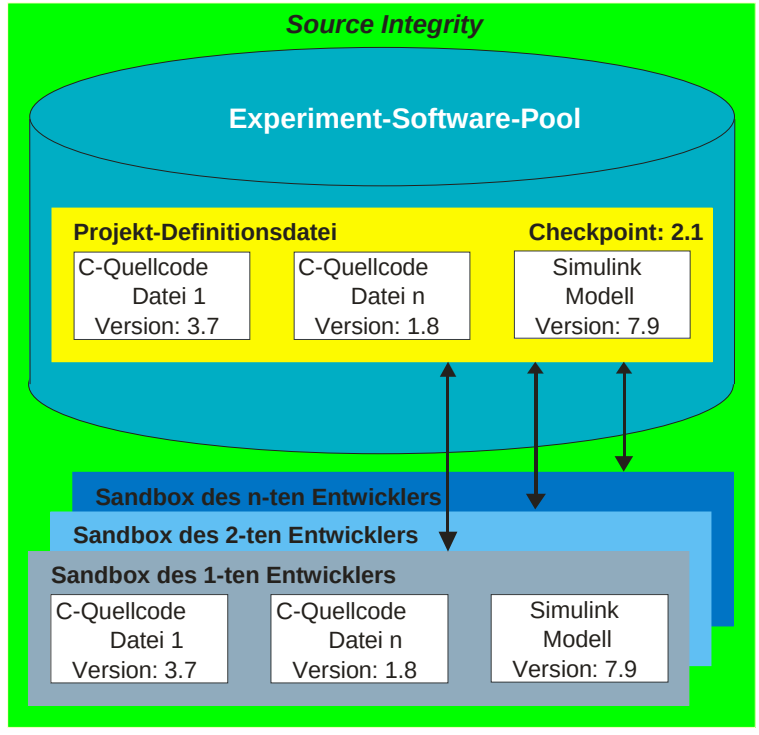
Umsetzung der In- und Outports:

- eindimensionale Felder
- Strukturen

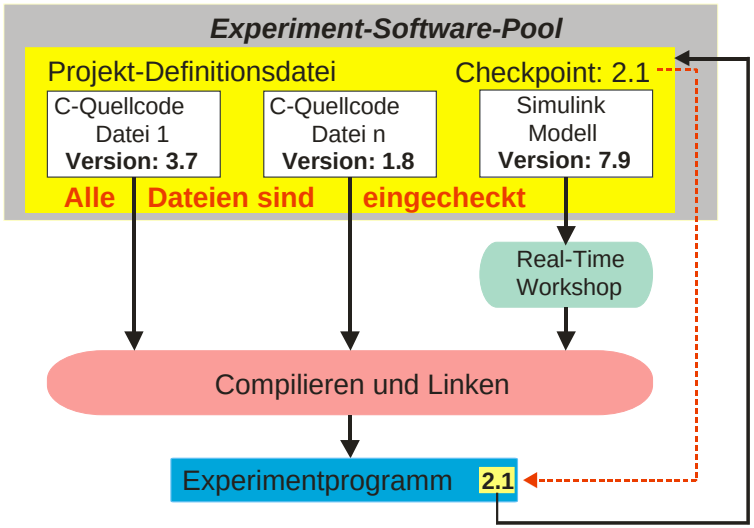
```
Inport.r_DEETSS = …     // Wert für Inport r_DEETSS
Inport.r_DEETT  = …     // Wert für Inport r_DEETT

<Aufruf der Berechnungsfunktion des Modells>

… = Outport.r_UDEET     // Wert vom Outport r_UDEET
… = Outport.r_UDEETR    // Wert vom Outport r_UDEETR
```

Der ATTAS Experiment-Software-Pool (ESP)

- Der ESP verwaltet den Entwicklungszyklus jeder Quelldatei
- Die Organisation des ESP basiert auf einem Projekt-basiertem Ansatz
- Jedes Projekt besitzt ein eigene Projekt-Definitionsdatei, welche alle benötigten Quelldateien beinhaltet
- Die Projekt-Definitionsdatei wird ebenfalls im ESP verwaltet
- Ein Checkpoint ist eine Version der Projekt-Definitionsdatei
- Jeder Entwickler kann sich seinen eigenen Arbeitsbereich (Sandbox) erzeugen
- Die Kommunikation zwischen der Sandbox und der zugehörigen Projekt-Definitionsdatei übernimmt *Source Integrity*



Freigabe der Flugversuchs-Software

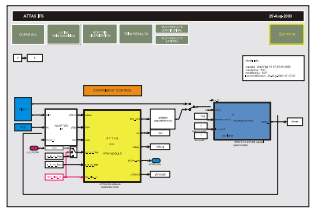


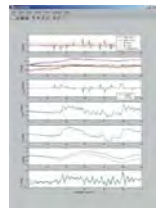
- Alle Quelldateien der getesteten Experimentfunktion sind im ESP eingecheckt.
- Das Projekt erhält eine Checkpoint.
- Die Checkpoint-Kennung wird in das binär-ausführbare Experimentprogramm eingebrannt.

⇒ Herstellung einer eineindeutige Beziehung zwischen dem binär-ausführbarem Experimentprogramm und allen Quelldateien, die für seine Generierung benötigt werden.

Entwicklungsprozess von Experimentfunktionen

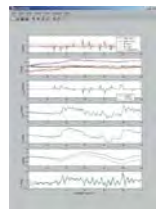
Entwicklung in der Offline Simulation





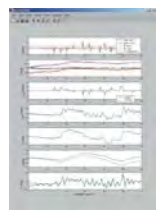
Code Generation

Testen in der Bodensimulation



Abgenommenes Bit-identisches Programm

Interner Flugversuch

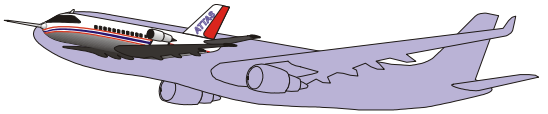


1. Fehlerbeseitigung
2. Einarbeitung von Änderungswünschen der Piloten und des Flugversuchsingenieurs
3. Berücksichtigung von Erkenntnissen aus dem Flugversuch

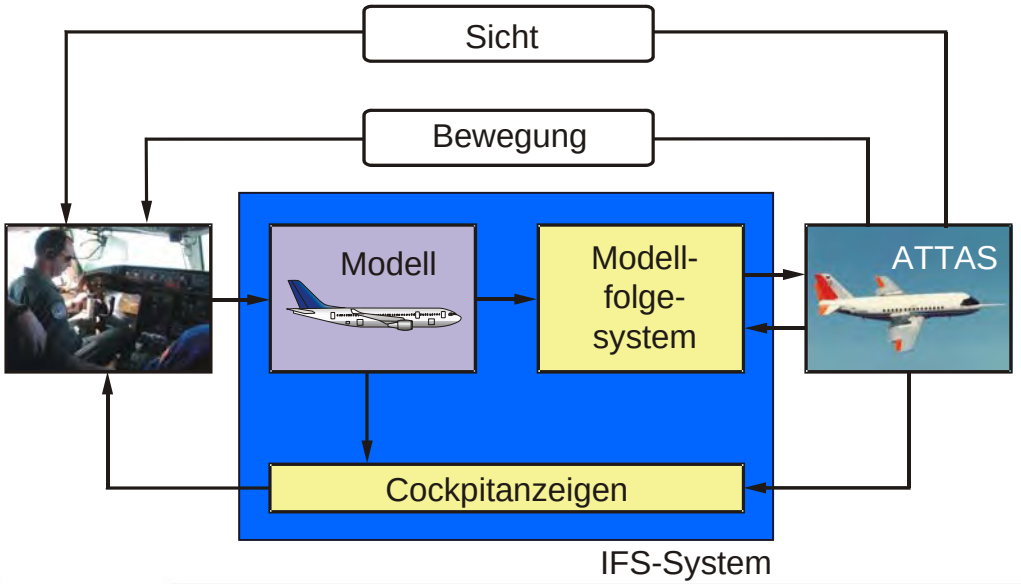
Freigegebene Version

Offizielle Flugversuche

Prinzip der In-Flight Simulation



- Ermöglicht die Abbildung des Flugverhaltens eines modellbasierten virtuellen Flugzeuges auf ein spezielles Trägerflugzeug (Modellfolgeregelung)
- Vermittelt reale und somit uneingeschränkte Sicht- und Bewegungseindrücke
- Führt zu realistischer Arbeitsbelastung für den Versuchspiloten



Anwendungen

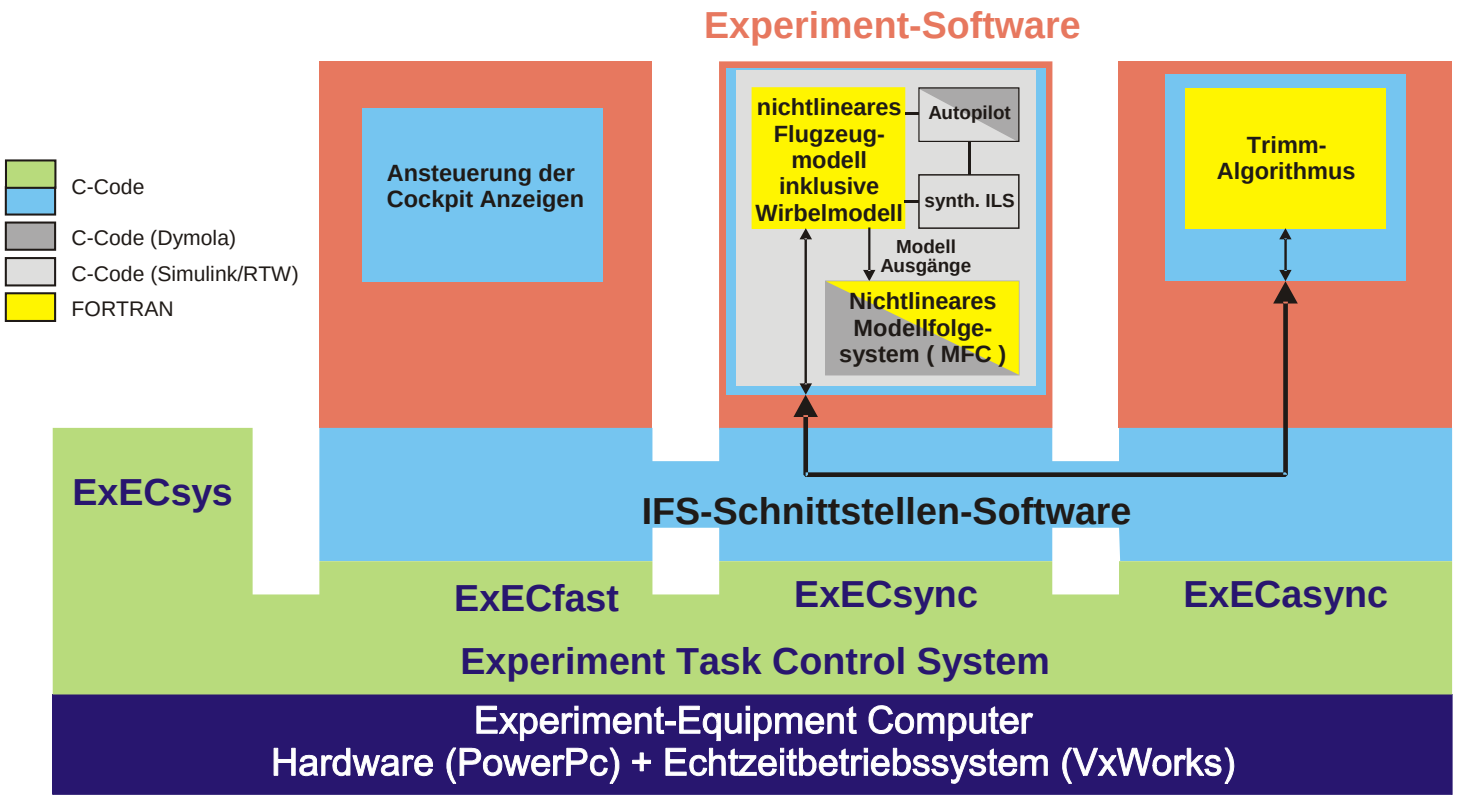
In-Flight Simulation eines Regional-Jets: • Flugeigenschaftsuntersuchung • Unterstützung notwendiger Nachweisführung im Rahmen des bevorstehenden Zulassungsprozesses	2002 Flüge: 8 Flugstunden: ~ 17

Simulierte Einflüge in Wirbelschleppen: • IFS von Wirbelschleppeneinflügen • Validation von Ergebnissen aus Bodensimulatoren, z.B. ZFB • Identifizierung der Gefahrenbereiche	2003 (→ 2006) Flüge: 5 Flugstunden: ~ 10

Ausbildung von Flugversuchsingenieuren: • modifiziertes VFW 614-Modell • Reglerentwicklung und Durchführung von Flugversuchen	jährlich seit 2000 pro Kampagne Flüge: 4 Flugstunden: ~ 8

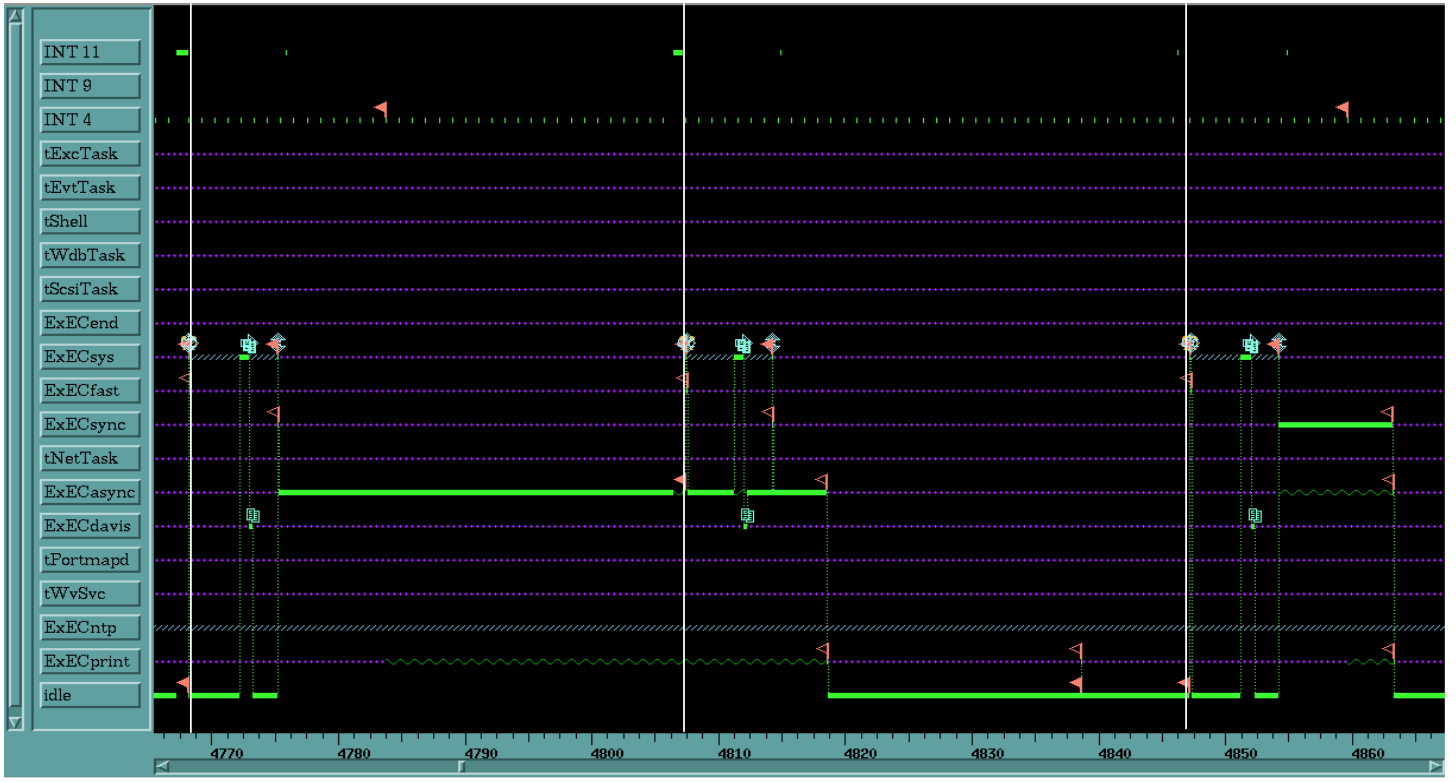
Simulierte Einflüge in Wirbelschleppen

- Software-Struktur -



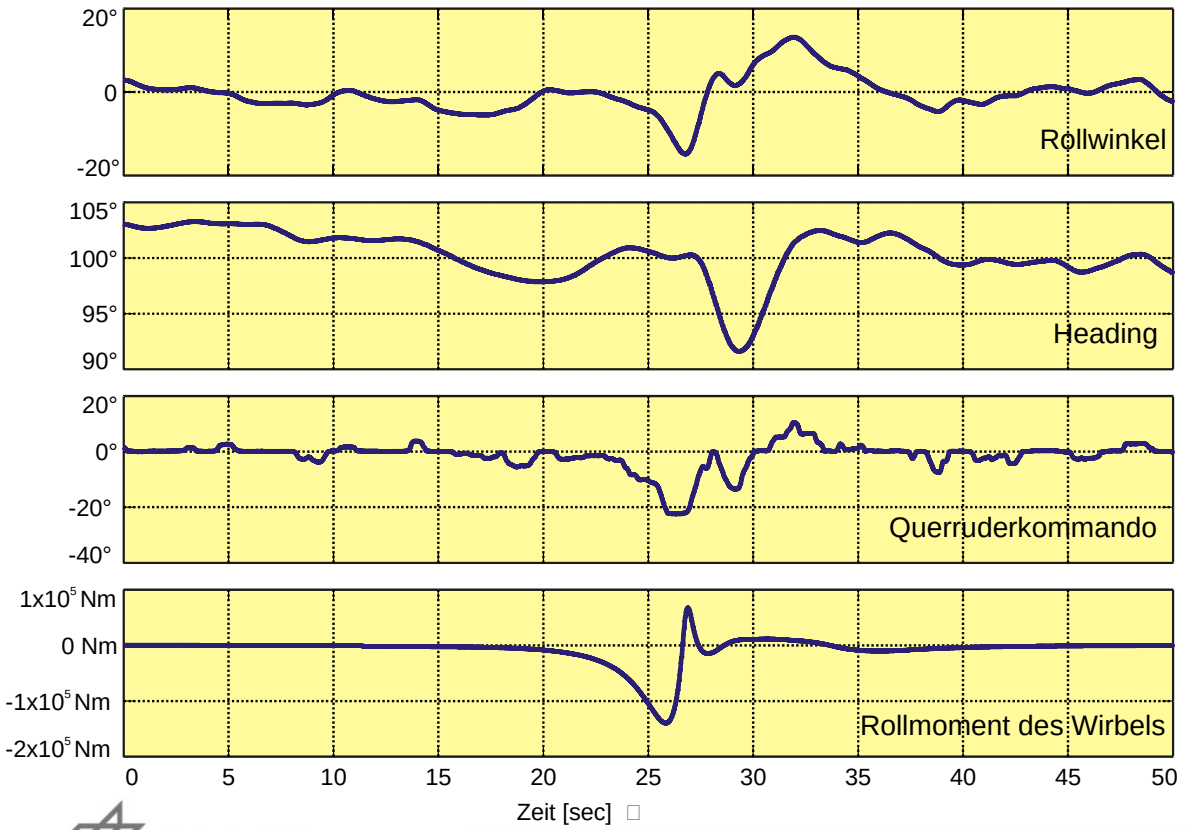
Simulierte Einflüge in Wirbelschleppen

- Ausführung -



Simulierte Einflüge in Wirbelschleppen

- Flugversuchsergebnisse -



Phasen:

1. Vorwirbel
2. Wirbel
   - Erzeuger: A321
   - Alter: 50 sec
3. Kurskorrektur
4. Stabilisierung

Zusammenfassung

- ATTAS-Flugsteuerungssystem
- ATTAS-Bordrechnersystem
- Prinzip der heterogenen Software-Entwicklung
- ATTAS-Coder
- Software-Konfigurationsmanagement
- Entwicklungsprozess von Experiment-Funktionen
- Prinzip der In-Flight Simulation
- Simulierte Einflüge in Wirbelschleppen

ATTAS – der fliegende Simulator des DLR
Technische Ausstattung

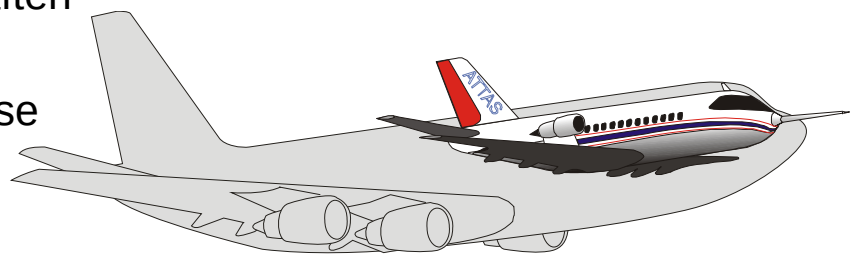
Thomas Heinecke

Deutsches Zentrum
DLR für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft

ATTAS – der fliegende Simulator des DLR

In Flight Simulator – Was ist das eigentlich?

- Ein echtes Flugzeug
- nimmt im Flug die Flugeigenschaften anderer Flugzeuge an
- oder simuliert z.B. Umwelteinflüsse wie Böen, Wirbelschleppen...



- Der Pilot hat einen echten Bewegungseindruck des Simulationsobjekts, wie es in Boden gebundenen Simulatoren nicht möglich ist.
- Echte Sicht!
- Die Arbeitsbelastung ist echt!

ATTAS –
Advanced **T**echnologies **T**esting **A**ircraft **S**ystem

- Kooperation VFW / MBB und DFVLR seit 1981
- Umbau einer VFW 614 zum Flugversuchsträger und In-Flight-Simulator
- Entwicklung und Einrüstung einer (Versuchs-) Fly-by-Wire-Steuerung
- FBW-Erstflug 1986, seitdem kontinuierliche Weiterentwicklungen
- weltweit einziges Flugzeug, das eine vollständige nichtlineare In-Flight-Simulation ermöglicht

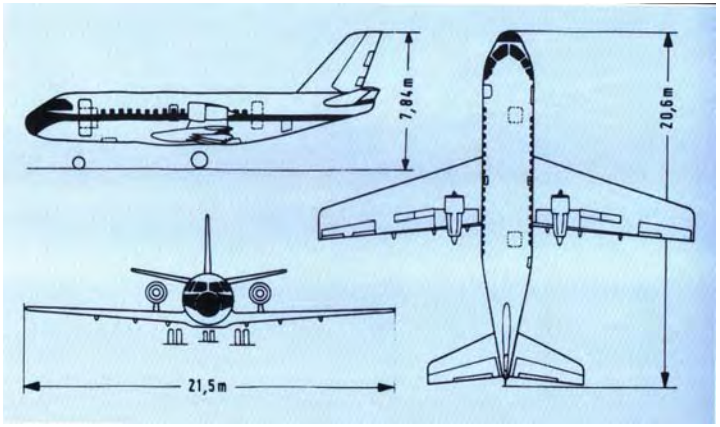
Gliederung

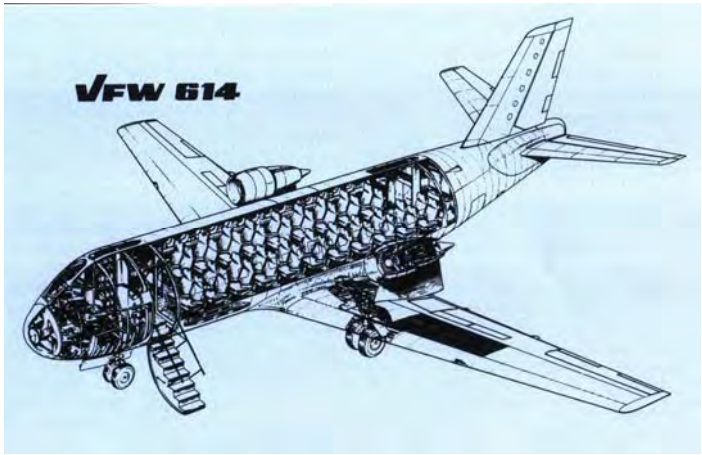
- Einleitung
- Die VFW 614 – Basis für ein einmaliges Forschungsflugzeug
- Von der VFW 614 zum ATTAS - Die Modifikationen
- Das Sicherheitskonzept
- Die Entwicklungsumgebung
- Ein schönes Beispiel

Geschichte der VFW 614

- **V**ereinigte **F**lugtechnische **W**erke
- Projekt No. 4 im Jahr 61
- Erstflug 14. Juli 1971
- Leisestes Strahlverkehrsflugzeug der Welt!
- Nur 19 fertig gestellte Exemplare
- Einstellung des Programms Dezember 1977

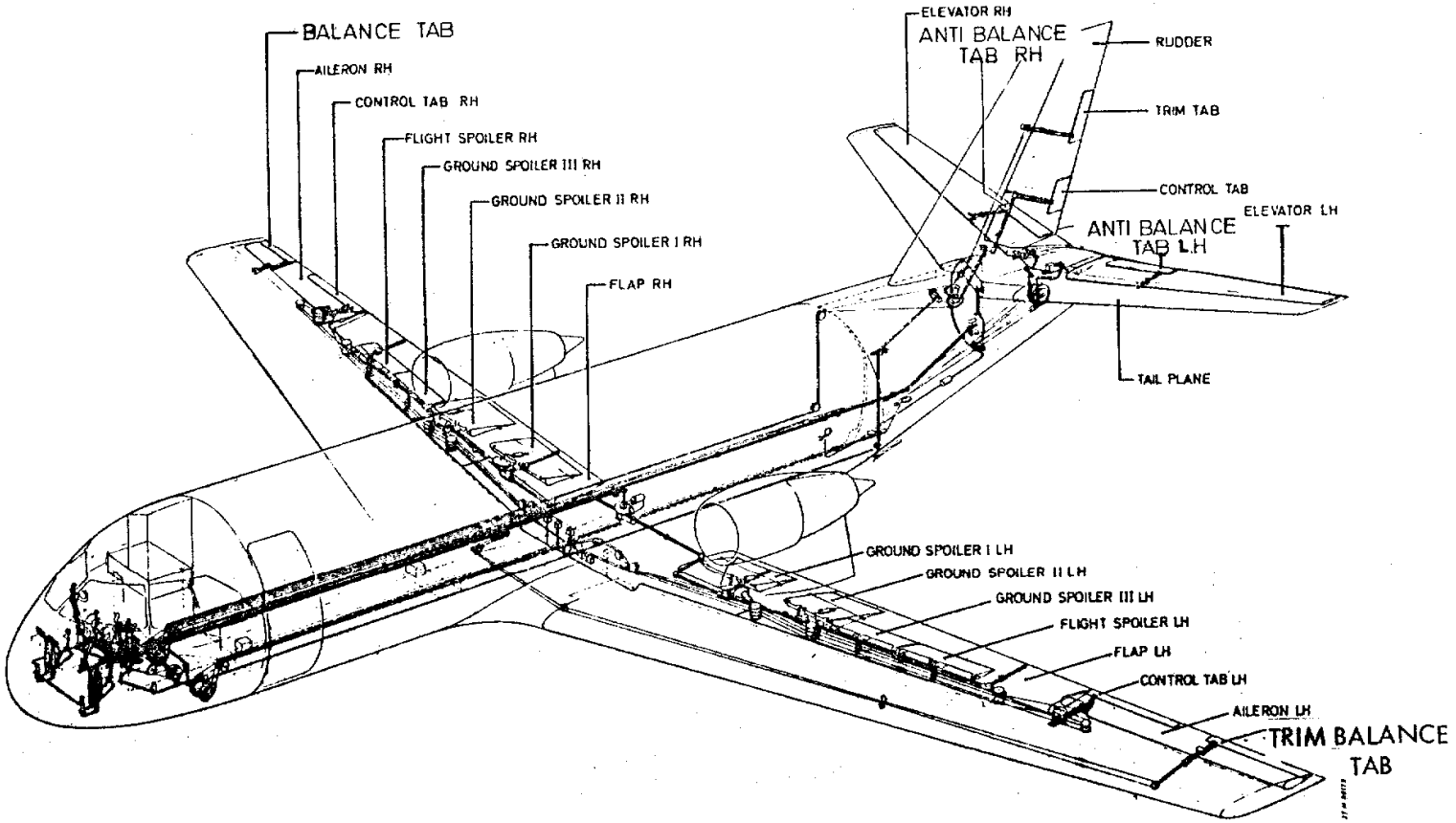
Technische Daten der VFW 614





- bis zu 44 Sitzplätze
- Reichweite ca. 1200 km
- Startgewicht ca. 20 t
- Reisegeschwindigkeit 720 km/h
- Flughöhe max. 7600 m
- Triebwerke 2x Rolls Royce M45-H
- Flugsteuerung mechanisch, z.T. hydraulisch unterstützt

VFW 614 - mechanische Flugsteuerung



VFW 614 - Cockpit

VFW 614 - Kabine

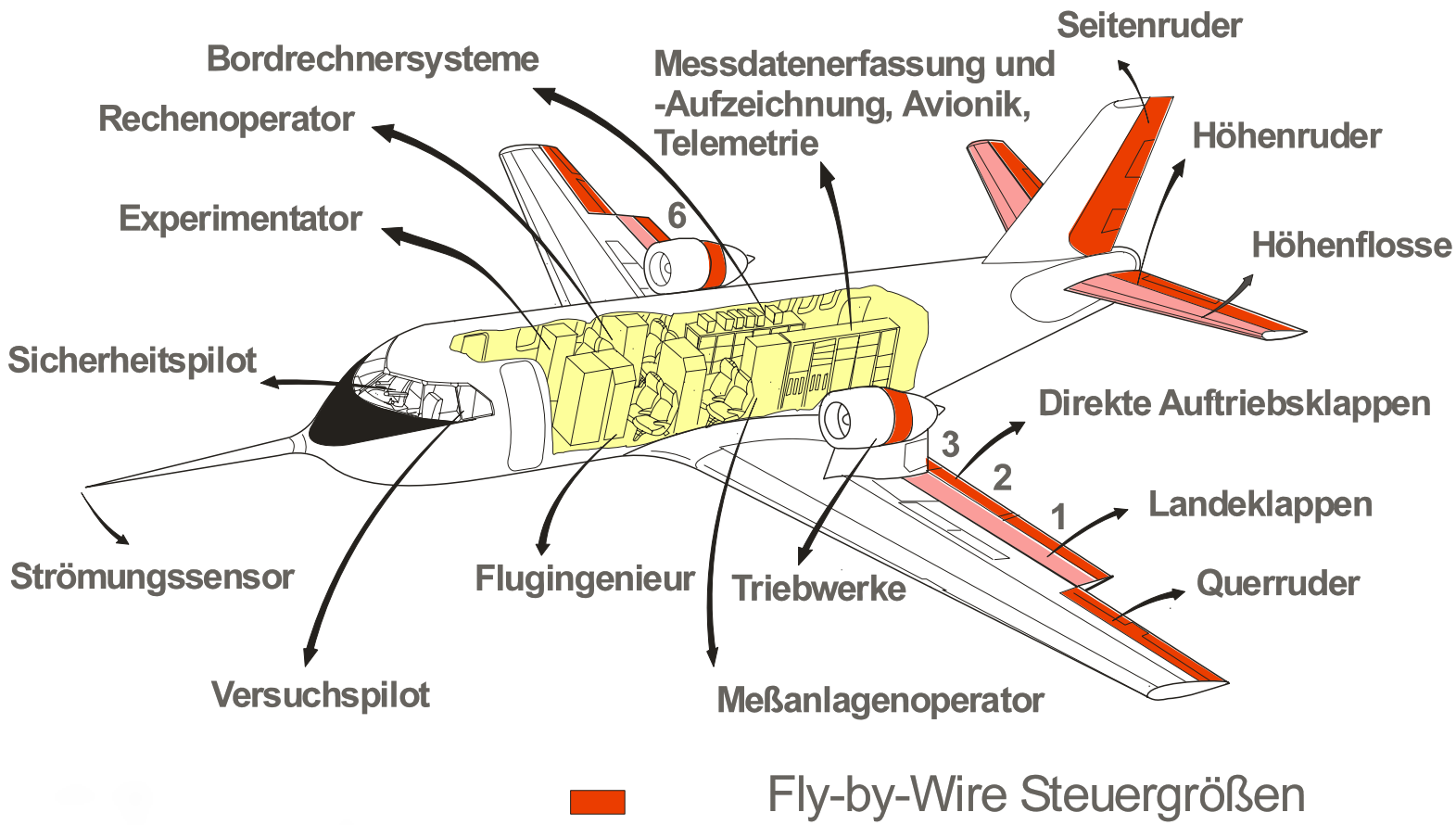
Probesitzen in der komfortabel eingerichteten Fluggastkabine eines in Originalgröße angefertigten Modells.

VFW 614 - Kabine

VFW 614 – ATTAS Kabine

Von der VFW 614 zum ATTAS

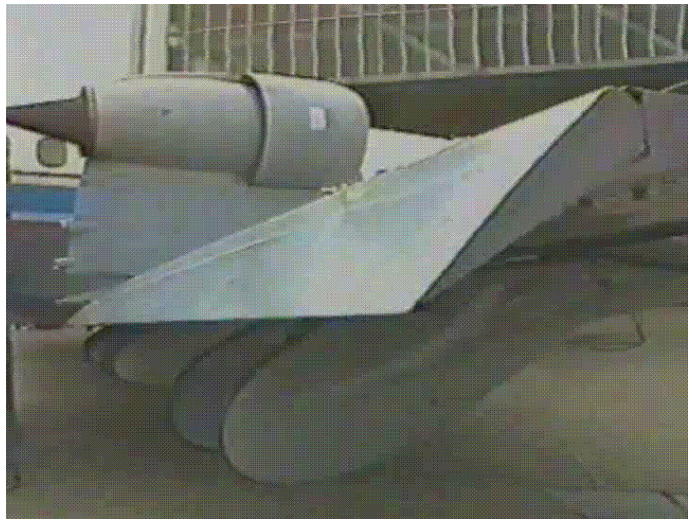
Die Modifikationen



Von der VFW 614 zum ATTAS

Die Modifikationen

- DLC-Klappen – Direkte Auftriebssteuerung
- Stellrate bis 90°/s



ATTAS Cockpit

Sidestick

FbW-Steuersäule

Das Sicherheitskonzept

oder: Warum dürfen die ihre Flugsteuerung ändern?

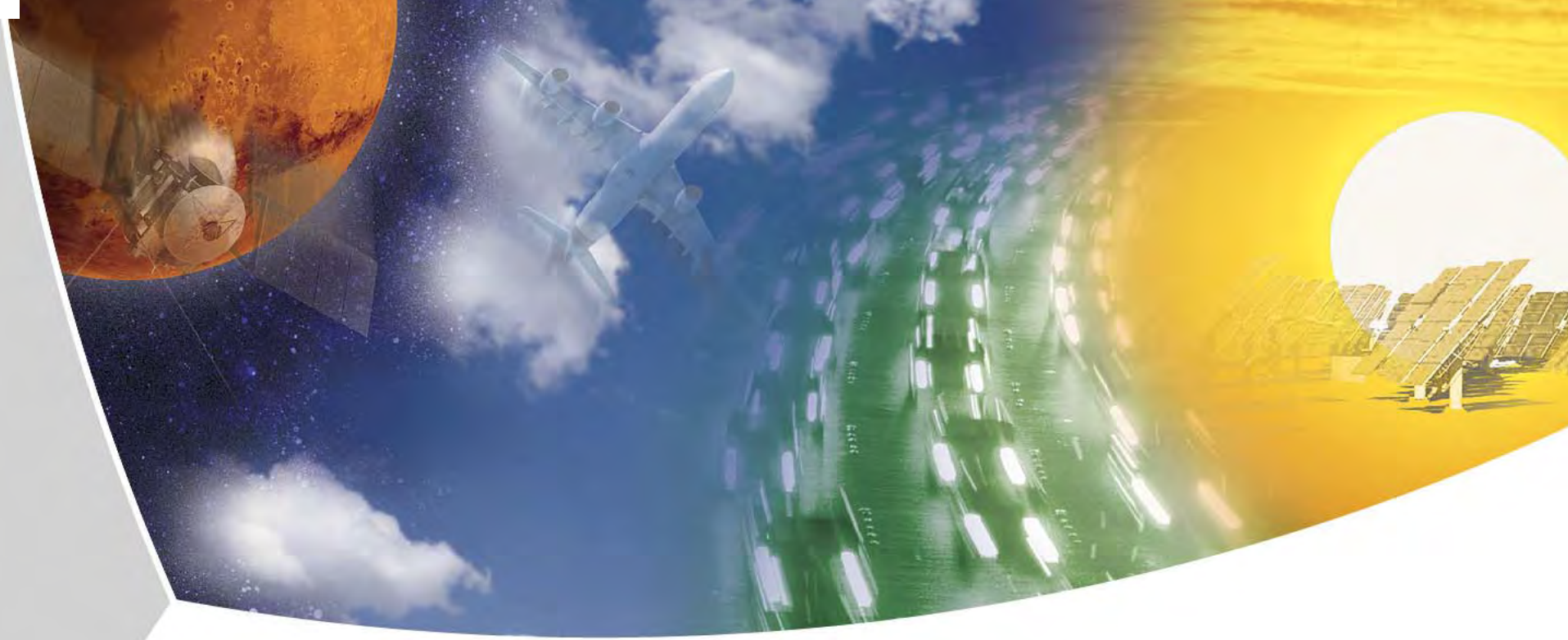
- Fehler in der elektrischen Steuerung sind erlaubt!
- Der Sicherheitspilot überwacht den gesamten Flugversuch.
- Die mechanische Verbindung zwischen Stellflächen und Bedienorganen ermöglicht das Erfühlen der FbW-Kommandos.
- Der Sicherheitspilot schaltet das Versuchssystem ggfs. durch Knopfdruck oder Überdrücken ab und fliegt mechanisch weiter.
- Zusätzlich kann die Stellrate der Aktuatoren begrenzt werden.

Ein Ausflug in unser Computermuseum

- Entwicklungssystem für Fly-by-Wire-Software:
  - DataGeneral MV20000 und MV8000
  - ROLM MSE/14

- Simulator zur Validierung der Software:
  - Simulation der VFW 614
  +
  - Verwendung identischer Computer wie im Flugzeug für die Fly-by-Wire-Software

ENDE



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Peter Saager

Institut für Flugsystemtechnik

DLR-Braunschweig

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

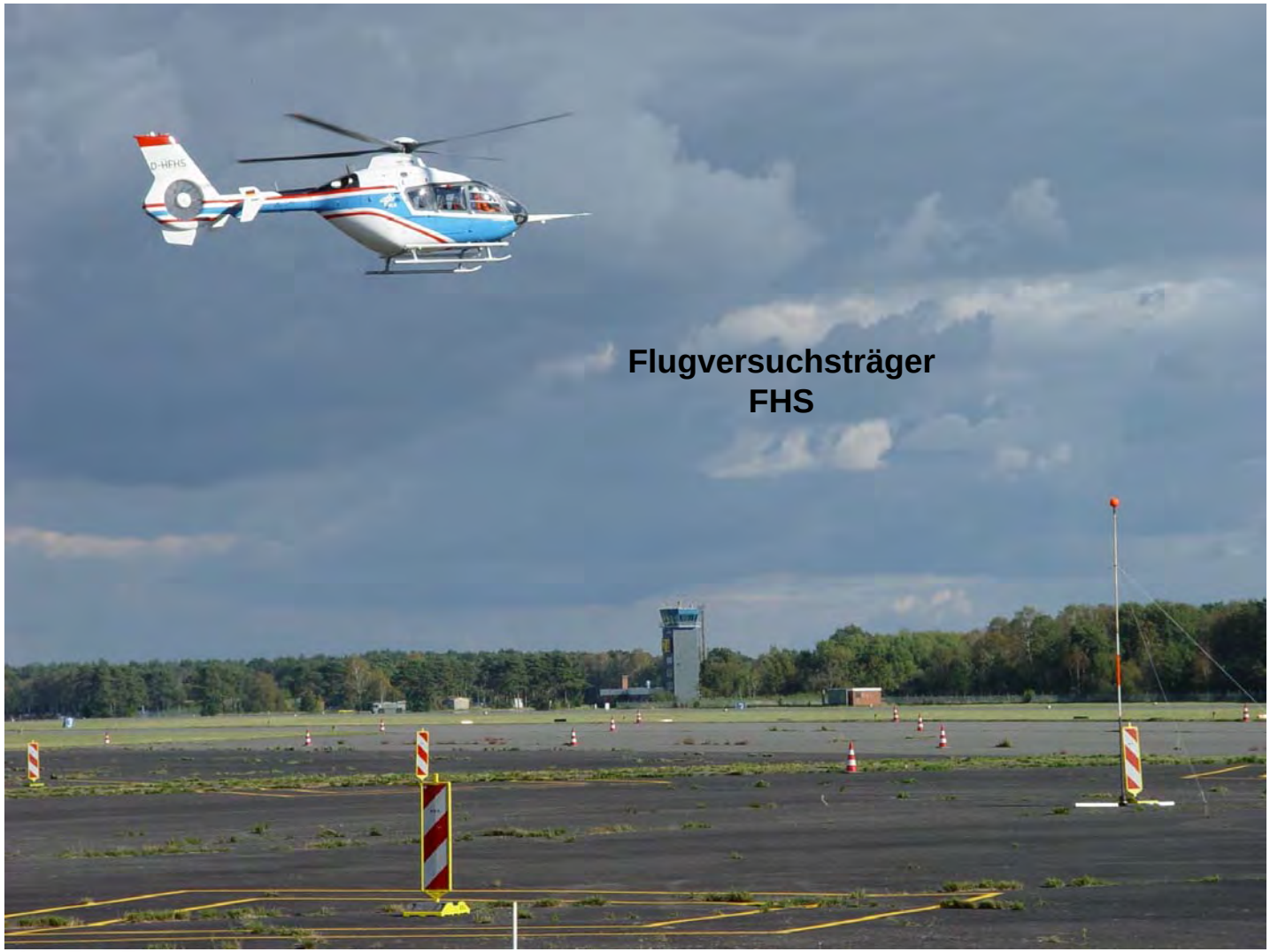
Übersicht

- Einleitung
- Vorstellung des Flugversuchsträgers FHS
- Aufgaben und Ziele
- Architektur
- Aufgaben der Bodensimulation
- Struktur der Bodensimulation
- Spezielle Komponenten der Bodensimulation
- Integration von Systemkomponenten
- Zusammenfassung

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Flugversuchsträger BO105

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Sicherheitskonzepte

- Effiziente Zertifizierungs-Verfahren
- System-Zuverlässigkeit
- Flugsicherheit

Technologische Konzepte

- Kosten Reduzierung über die gesamte Nutzungszeit
- Autonomes Fliegen
- 24Std. Allwetter-Tauglichkeit

Mensch-Maschine Interface

- Entlastung des Piloten
- Erweiterung des Flugbereichs
- Leichte Bedienung

In-flight Simulation

- Entwicklungsrisiko minimieren
- Modellfolgeregelung verbessern
- Verbesserung des Flugverhaltens
- Pilotentraining

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Modulares Experimental System

- Programmierbare Displays
- Erweiterte Sensorik
- Hohe Rechnerkapazität
- Modulare Avionik
- Flexible Experiment Schnittstellen

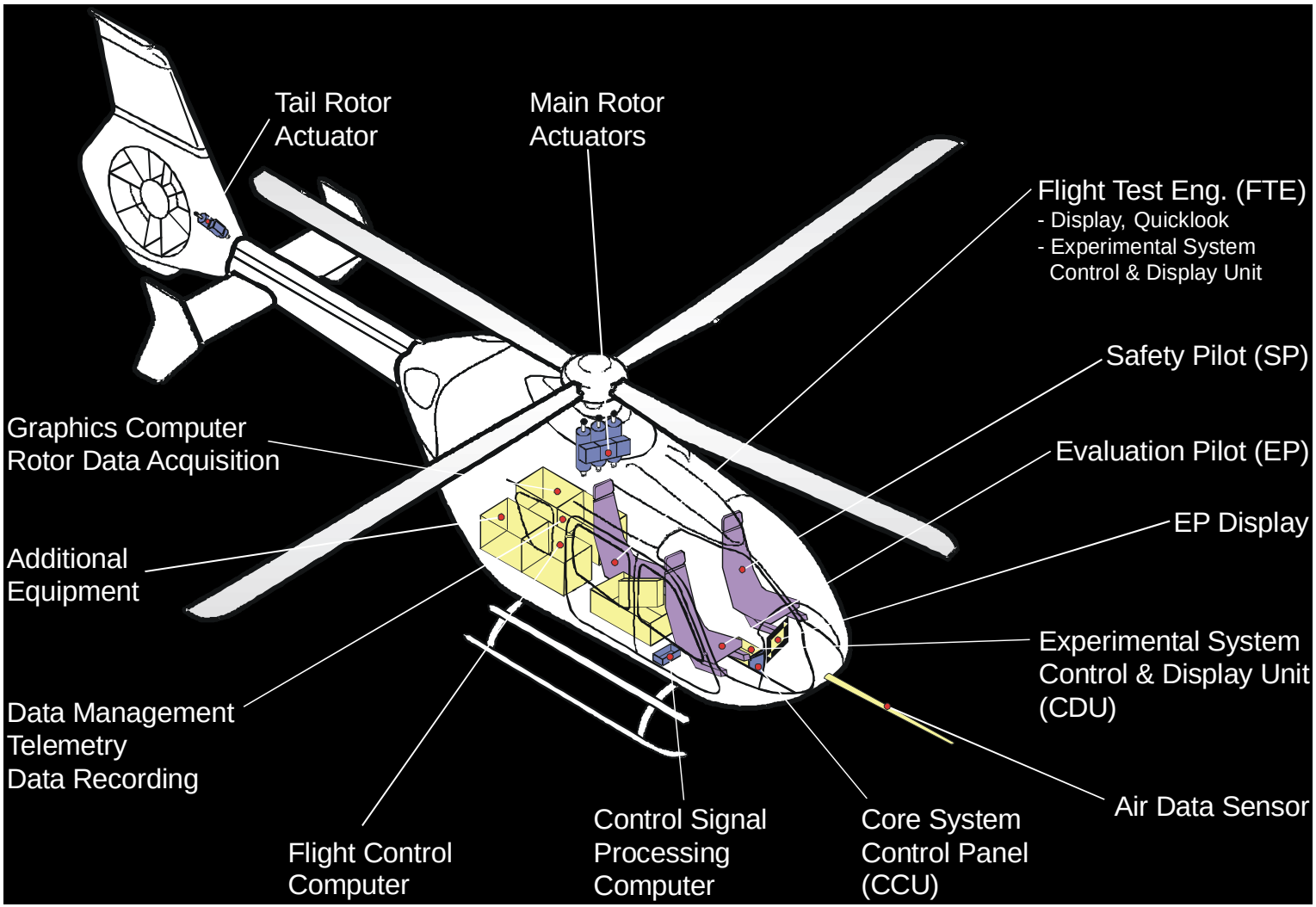
Boden-Anlagen

- System-Simulator
- Bodenstation mit Telemetrie
- On- und Offline Datenanalyse

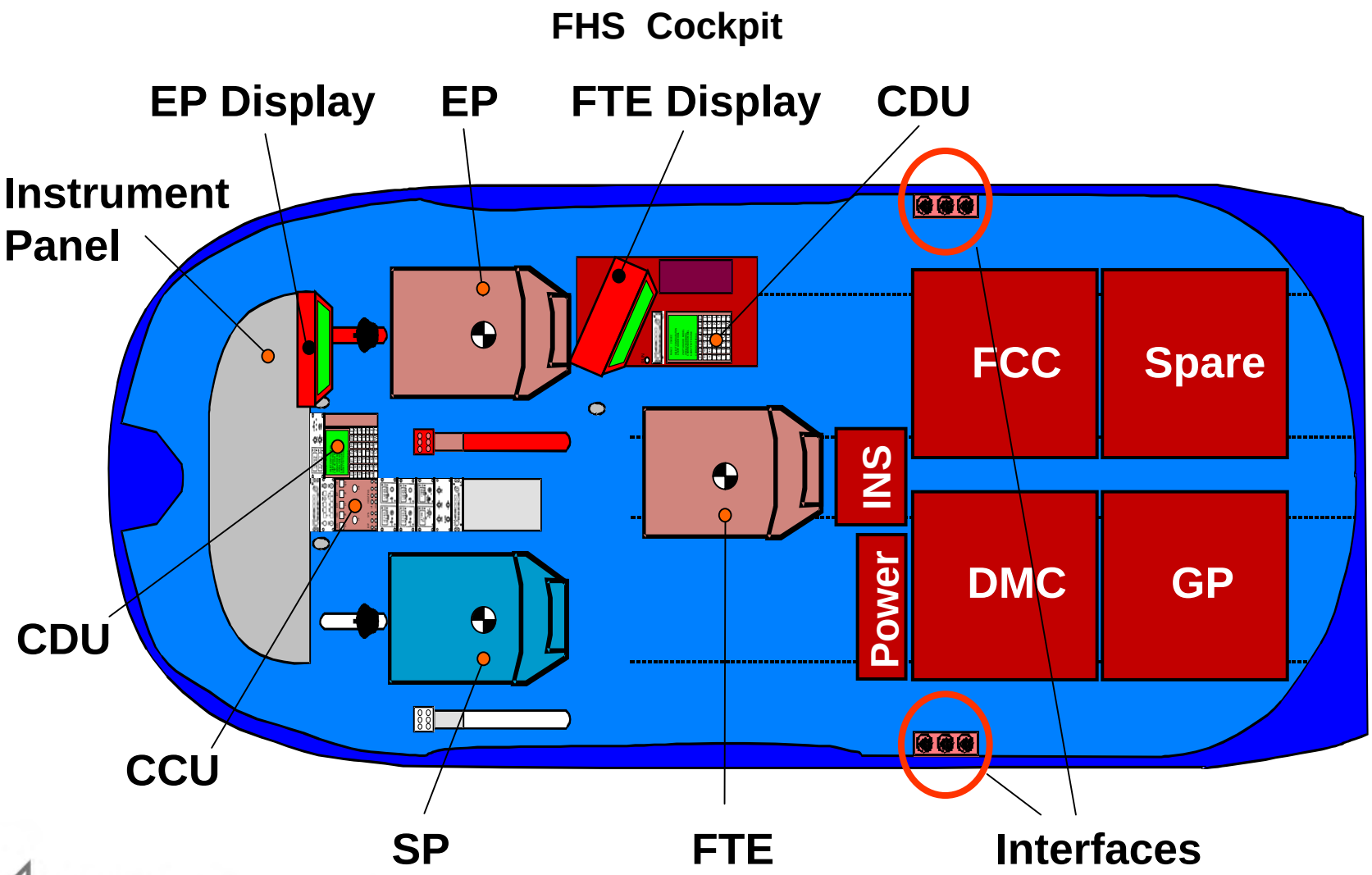
Digitales Flight Control System

- Lichtleiter Technologie
- Uneingeschränkte Autorität, hohe Bandweite
- Spezielle Aktuatoren
- Konzipiert für Zulassung

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Aufgaben des Bodensimulators

• Flugversuchsvorbereitung für den Flugversuchsträger FHS

- Entwicklung von Flugversuchssoftware
- Integration und Test von Experimentsoftware interner und externer Kunden
- Integration und Test von Experiment-Hardware
- Vorbereitung der Versuchspiloten

• Bereitstellung von Simulationsumgebungen für unterschiedliche Anwendungen

- Simulation diverser Hubschraubertypen
- Simulation und Test spezieller Soft- und Hardware-Komponenten
- Testbed für die Weiterentwicklung der Simulationstechnik
- Ausbildung von wissenschaftlich technischem Personal
- Entwicklung innovativer Techniken und Verfahren

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

VMEbus basiertes Interface-System

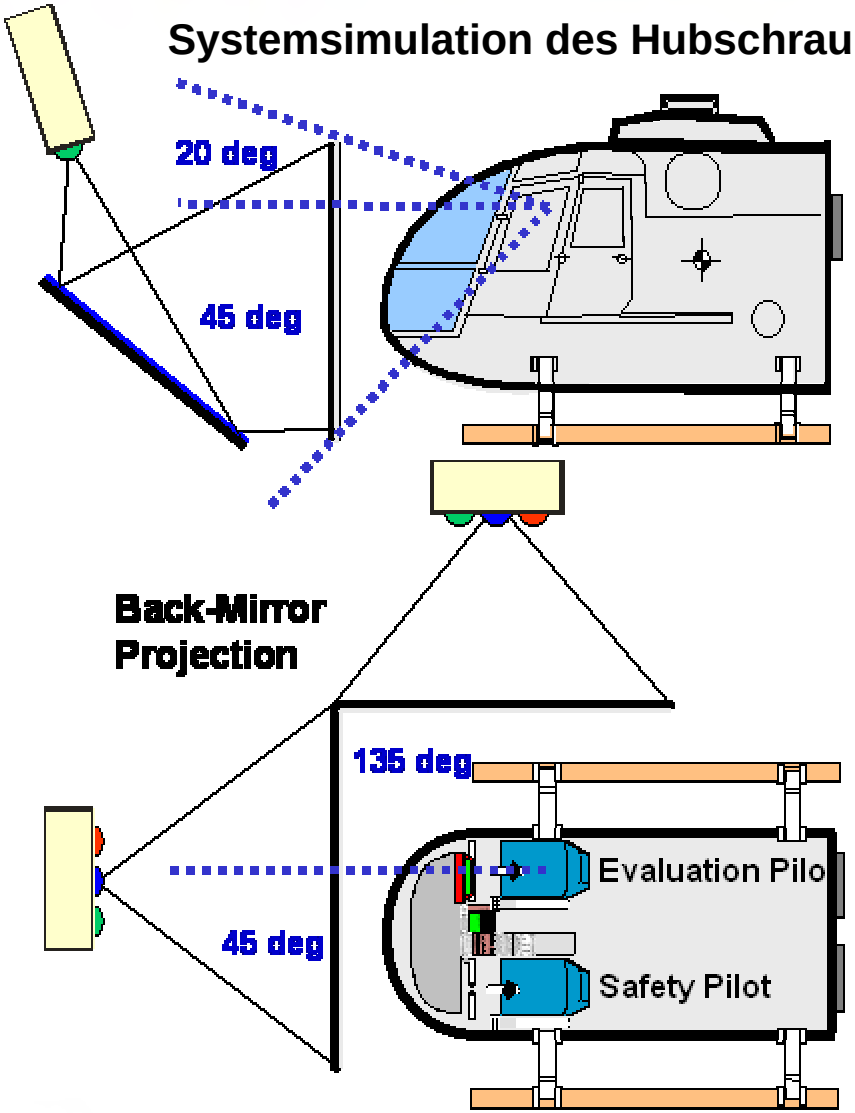
Hardware

- 4 CPUs 68060
- ARINC 429
- MILbus
- Reflective Memory

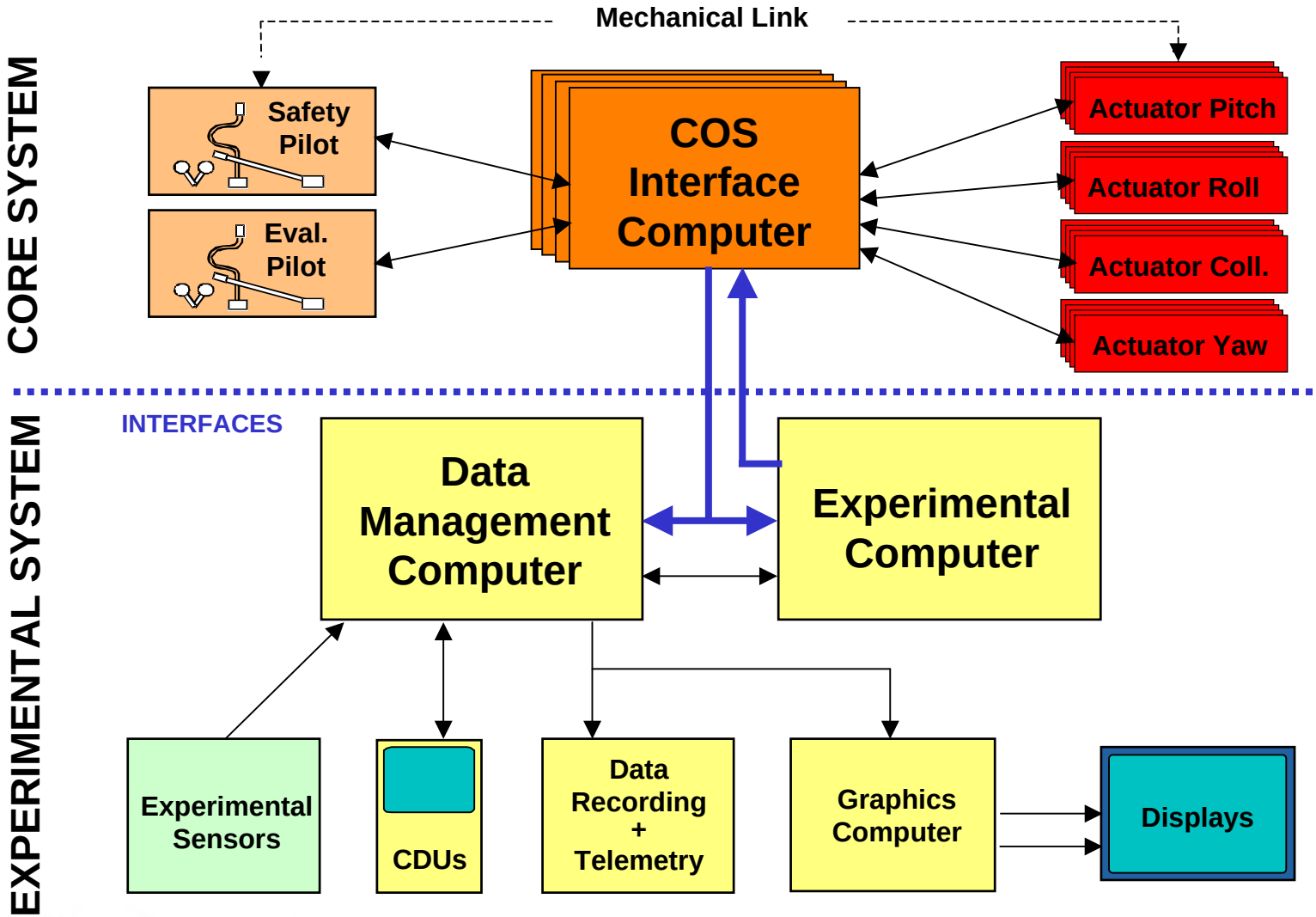
Software

- Datenaufbereitung
- Datenaufzeichnung
- Daten-Monitoring
- Systemstatus anzeigen
- Schnittstellen bedienen (I/O)
- Simulation von Subkomponenten (COS)

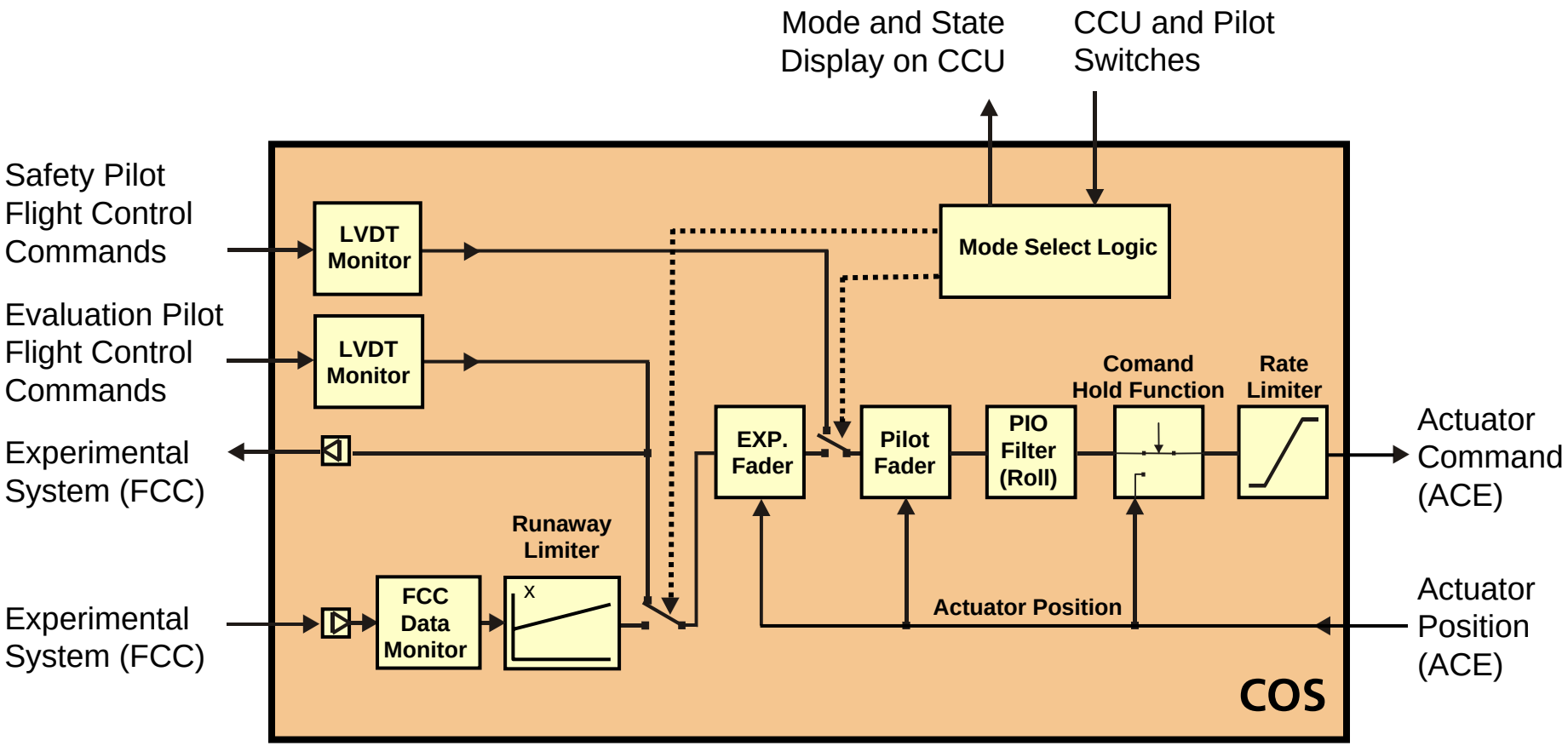
Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

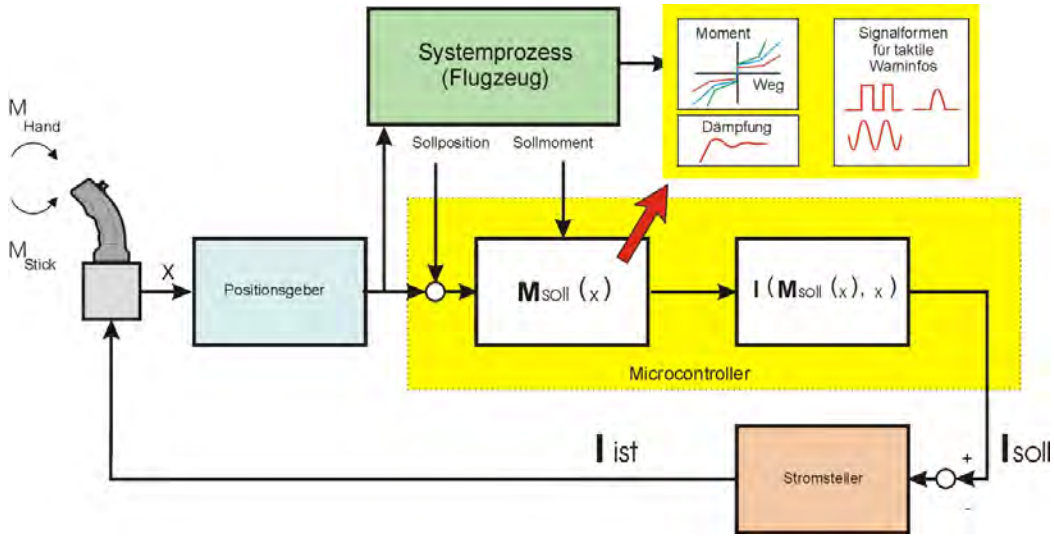


Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Beispiel Sidestick-Integration



Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Prototyping Software iObjects

Software-Produkt (DLR) zur flexiblen und schnellen Erstellung von Displays und Instrumentenanzeigen

- plattformunabhängig
- OpenGL basiert
- objektorientiert
- eigene Script-Sprache, Textural
- direkte Interpretation, kein Compilieren
- Flexible Parametrisierung

- iObjects Library
- iObjects Render Engine
- iObjects Datalogger

- Handbuch

Systemsimulation des Hubschrauber-Flugversuchsträgers FHS des DLR

Zusammenfassung

Anforderungen

- Exakte Nachbildung realer Schnittstellen
- Sehr genaue Modellierung der Einzelkomponenten
- Ständige Anpassungen an das reale Basis-System
- Ständige Anpassungen an das reale Versuchssystem
- Integrationsmöglichkeit von spezieller Hard- u. Software für Tests
- Zweckmässiger Aufbau des Simulators

Benefits

Vorbereitung von Flugversuchen
- GeeigneteTestumgebung für Soft- und Hardware-Komponenten, die im Flugversuch eingesetzt werden sollen
- Vorbereitung und frühzeitige Integration der Testpiloten
- Vorbereitung und Einbeziehung des Flugversuchsingenieurs

Testbed für Simulationstechnik
- Simulation und Test von Basiskomponenten im Gesamtsystem
- Erprobung und Einsatz neuer Technologien und Verfahren
- Ausbildung wissenschaftlichen Personals

Entwicklung von Fahrerassistenzsystemen im FASLab des DLR

Dipl.-Inform. Markus Stöbe

Deutsches Zentrum
für Luft- und Raumfahrt e.V.
in der Helmholtz-Gemeinschaft

Das Deutsche Zentrum für Luft- und Raumfahrt e.V. in der Helmholtz-Gemeinschaft

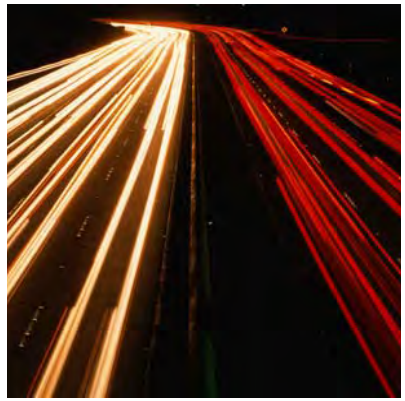
Forschungsbereiche

- Luftfahrt
- Raumfahrt
- Energie
- Verkehr

Das DLR in Zahlen

- Gesamtbudget:
  2004 1.194 Mio. Euro
  2005 1.168 Mio. Euro
- Wissenschaftliche Kompetenz:
  über 5.100 MitarbeiterInnen
  davon 2.300 WissenschaftlerInnen



Institute und wissenschaftliche Einrichtungen des DLR
Standorte

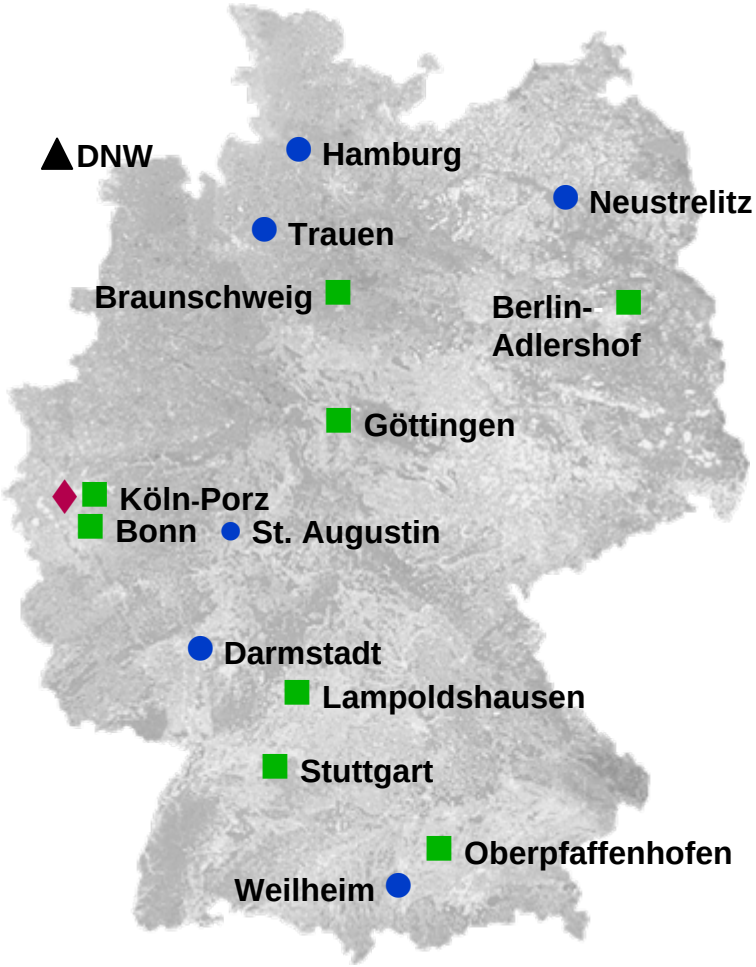
31 Institute und Einrichtungen in

- 8 Standorten
- 6 Außenstellen

Außenbüros in Brüssel, Paris und Washington.

Beteiligung des DLR an:

- European Transonic Wind Tunnel (ETW)
- Deutsch-Niederländische Windkanäle (DNW)



Institut für Verkehrsführung und Fahrzeugsteuerung

Sitz:	Braunschweig
Seit:	März 2001
Leitung:	Prof. Dr.-Ing. Karsten Lemmer
Mitarbeiter:	Momentan 61 Mitarbeiter aus verschiedenen wissenschaftlichen Bereichen

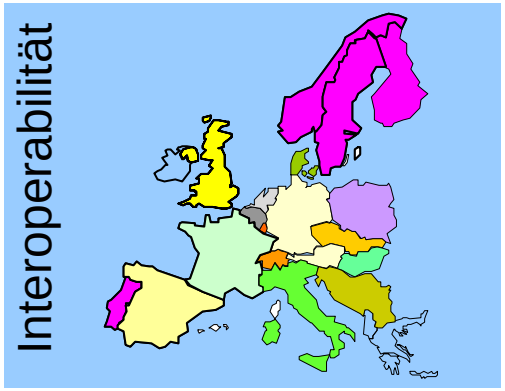
Aufgabenspektrum

- Grundlagenforschung
- Erstellen von Konzepten und Strategien
- Prototypische Entwicklungen

Forschungsgebiete

- Automotive
- Bahnsysteme

Themenfelder im Bereich Bahnsysteme



Betriebsführung Fernverkehr

- Betriebliche und technische Interoperabilität
- Sicherheitsbetrachtung
- Ortungsverfahren
- Migrationsszenarien

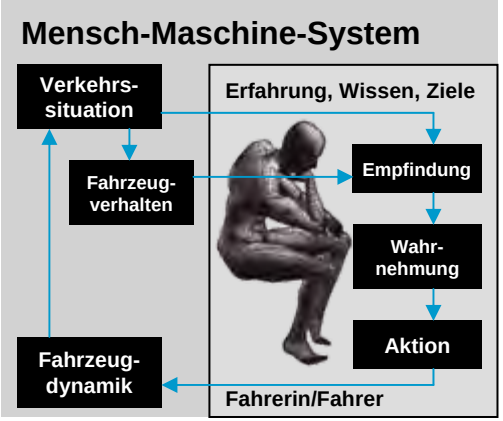
Betriebsführung Nah- und Regionalverkehr

- Optimierung der Betriebsführung
- Wirtschaftliche und betriebliche Bewertung von Betriebsverfahren
- Migrationsszenarien

Validation und Erprobung

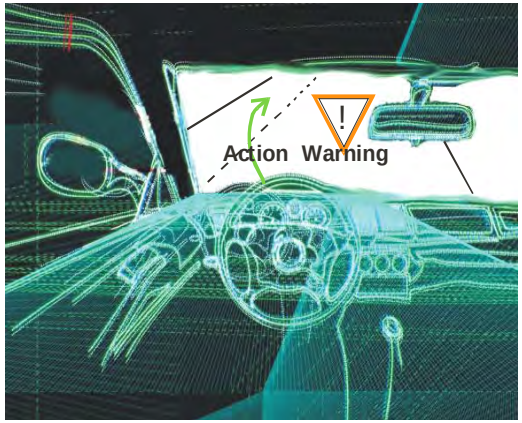
- Automatisiertes und optimiertes Testen
- Bewertung von Systemen, Subsystemen und Komponenten
- Funktionale Tests und Interoperabilitätstests

Themenfelder im Bereich Automotive



Grundlagen der Fahrerassistenz

- Fahrermodellierung
- Systemarchitektur
- Fahrer-Umwelt: Verkehrsführung



Angewandte Fahrerassistenz

- Menschzentrierte Gestaltung
- Mensch-Maschine-Schnittstelle
- Anpassung an Fahrer und Situation

Erprobung der Fahrerassistenz

- Fahrerassistenz-systemelabor
- Entwicklung von Untersuchungs-methodiken

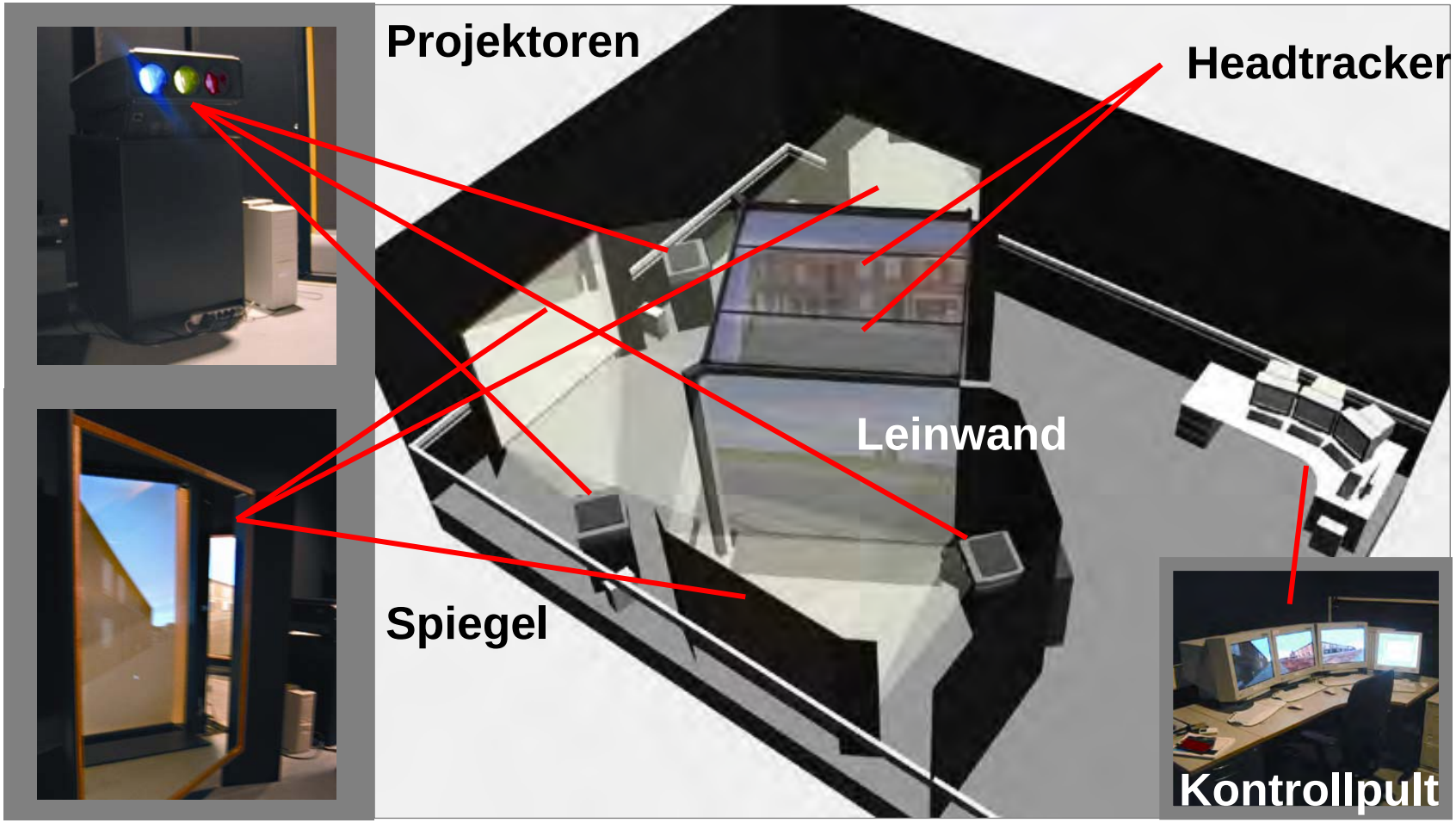
Forschungsinfrastruktur Bereich Automotive

Das Messfahrzeug ViewCar

- Messfahrzeug zur Analyse von Wahrnehmungsprozessen und Verhalten des Fahrers im Straßenverkehr
- Sensorik zur Messung und Aufzeichnung
  - des umgebenden Verkehrs,
  - der Fahrerblickrichtung,
  - der Bedienung des Fahrzeugs durch den Fahrer und
  - des resultierenden Fahrzeugverhaltens
- Ermöglicht Untersuchungen
  - zur Modellierung des Fahrerverhaltens und
  - zur Akzeptanz und Sicherheit von Assistenzsystemen

Forschungsinfrastruktur Bereich Automotive

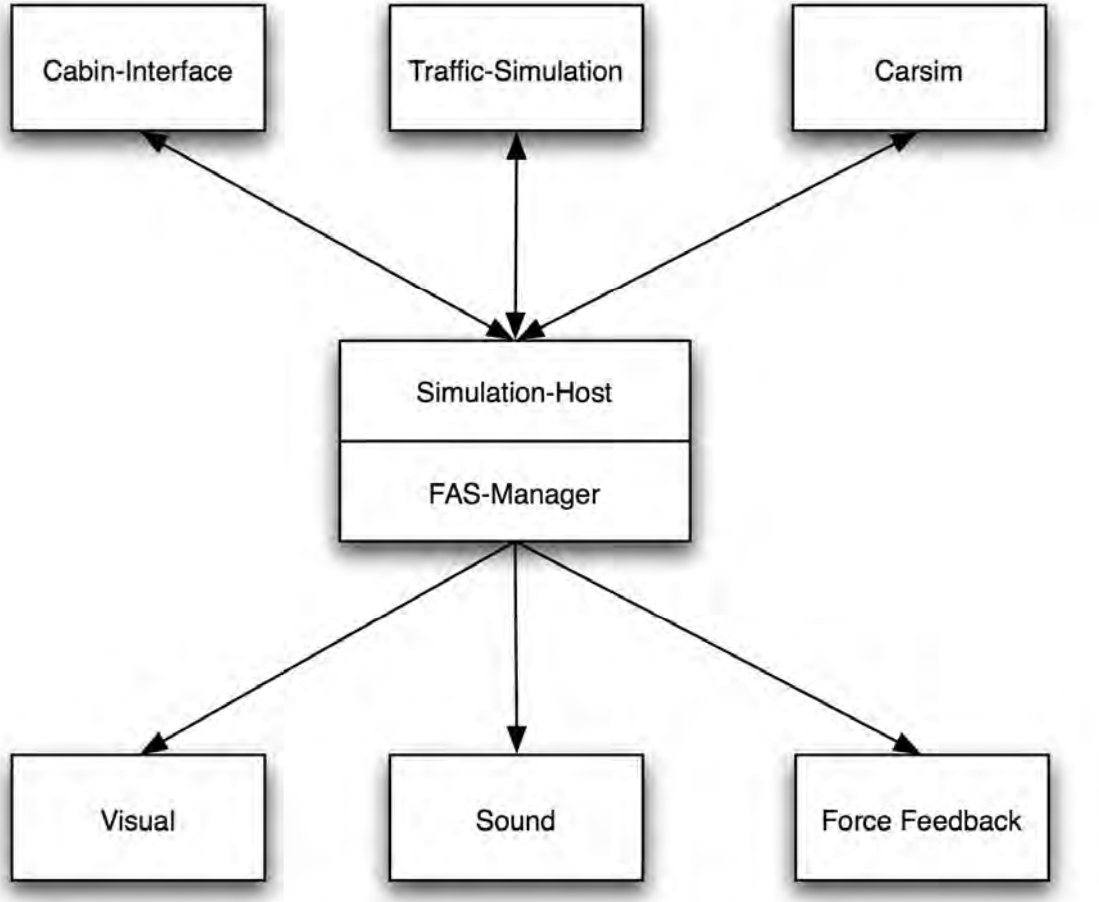
Virtual Reality Labor



Forschungsinfrastruktur Bereich Automotive

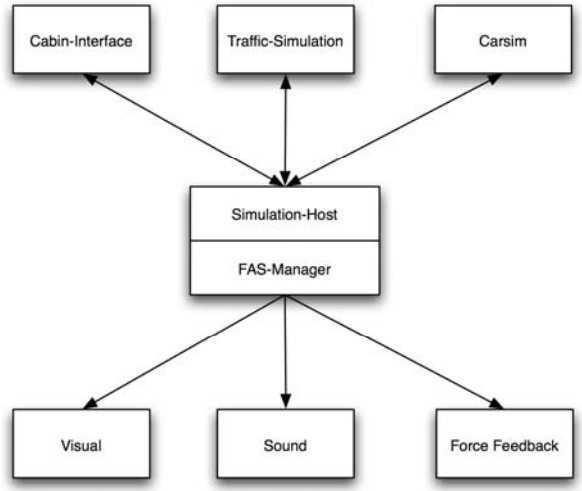
Dynamischer Fahrsimulator

Systemarchitektur im FASLab
Überblick I



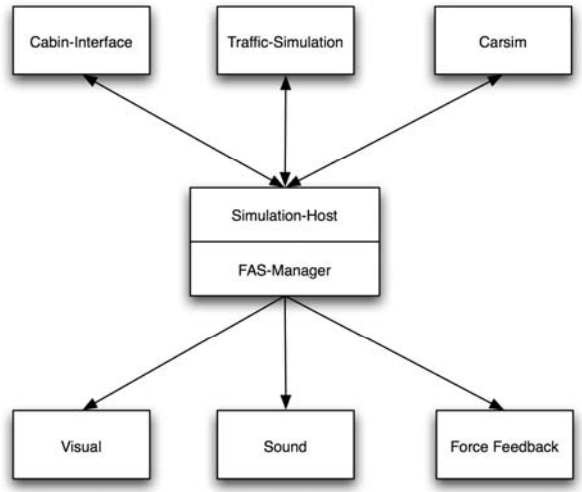
Systemarchitektur im FASLab
Überblick II

- modular aufgebaut
- Dienst-basiert
- zentrale Instanz zur Datenhaltung/-speicherung
- erweiterbare Schnittstellen zur Anbindung von Assistenzsystemen
- Kommunikation via CAN-Bus oder UDP
- skalierbar vom Desktop-System bis hin zum Fahrsimulator mit Motionsystem
- HIL-Tests über XPC-Target-Systeme möglich



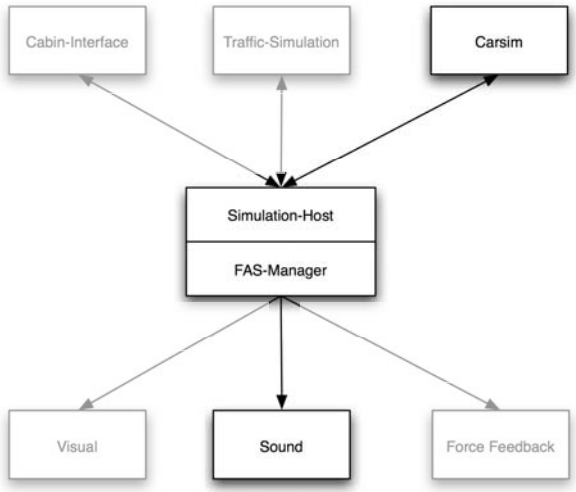
Systemarchitektur im FASLab
Dienste im Detail

- jedes Modul bietet einen oder mehrere Dienste an
- jeder Dienst ist weiter in einzelne Komponenten unterteilt
- jedes Modul kann einen Dienst oder einzelne Dienstkomponenten abbonieren
- der Simulations-Host gibt Auskunft, welche Komponenten evtl. für einen Start der Simulation noch fehlen
- Vorteil: Dienste können erweitert/ersetzt werden, ohne die Abbonenten anzupassen

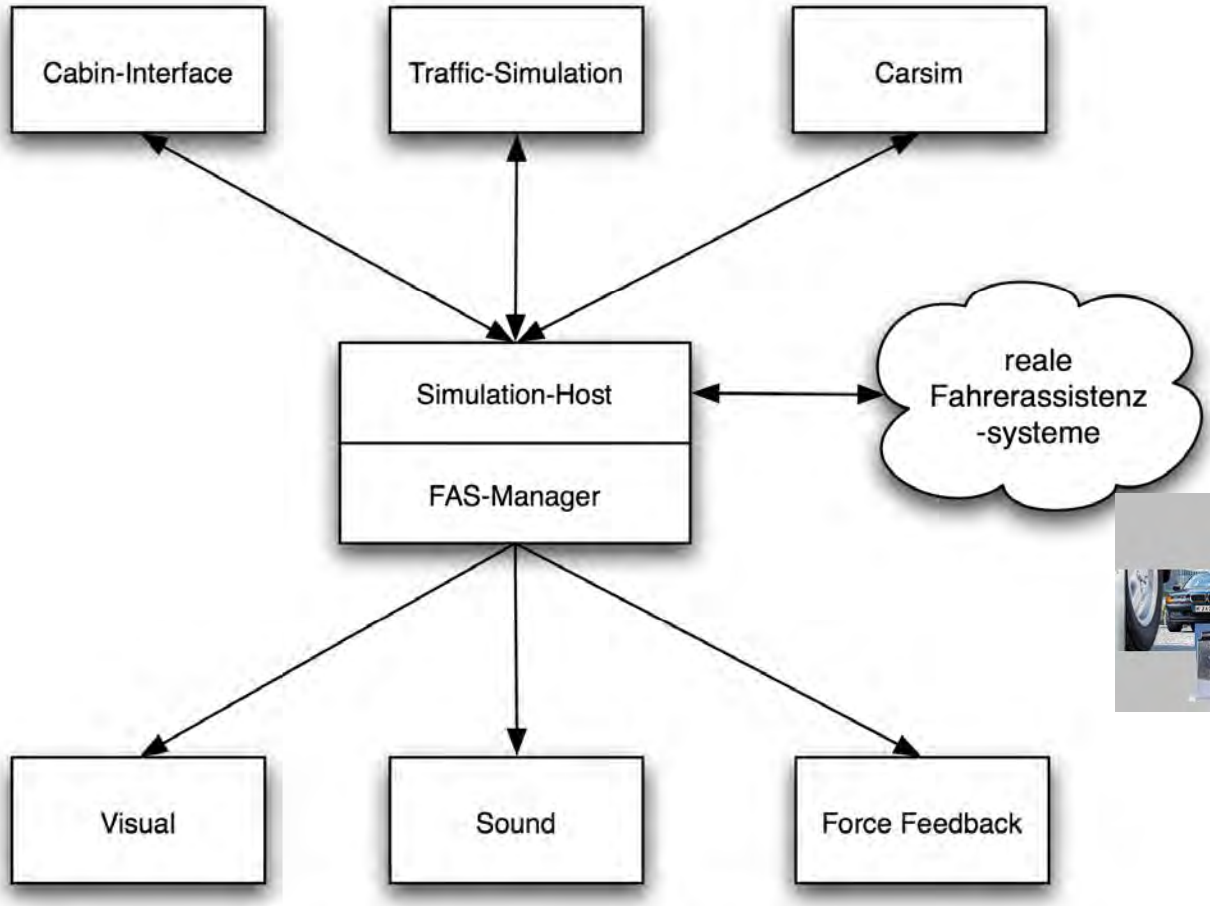


Systemarchitektur im FASLab
Dienste am Beispiel Fahrdynamik

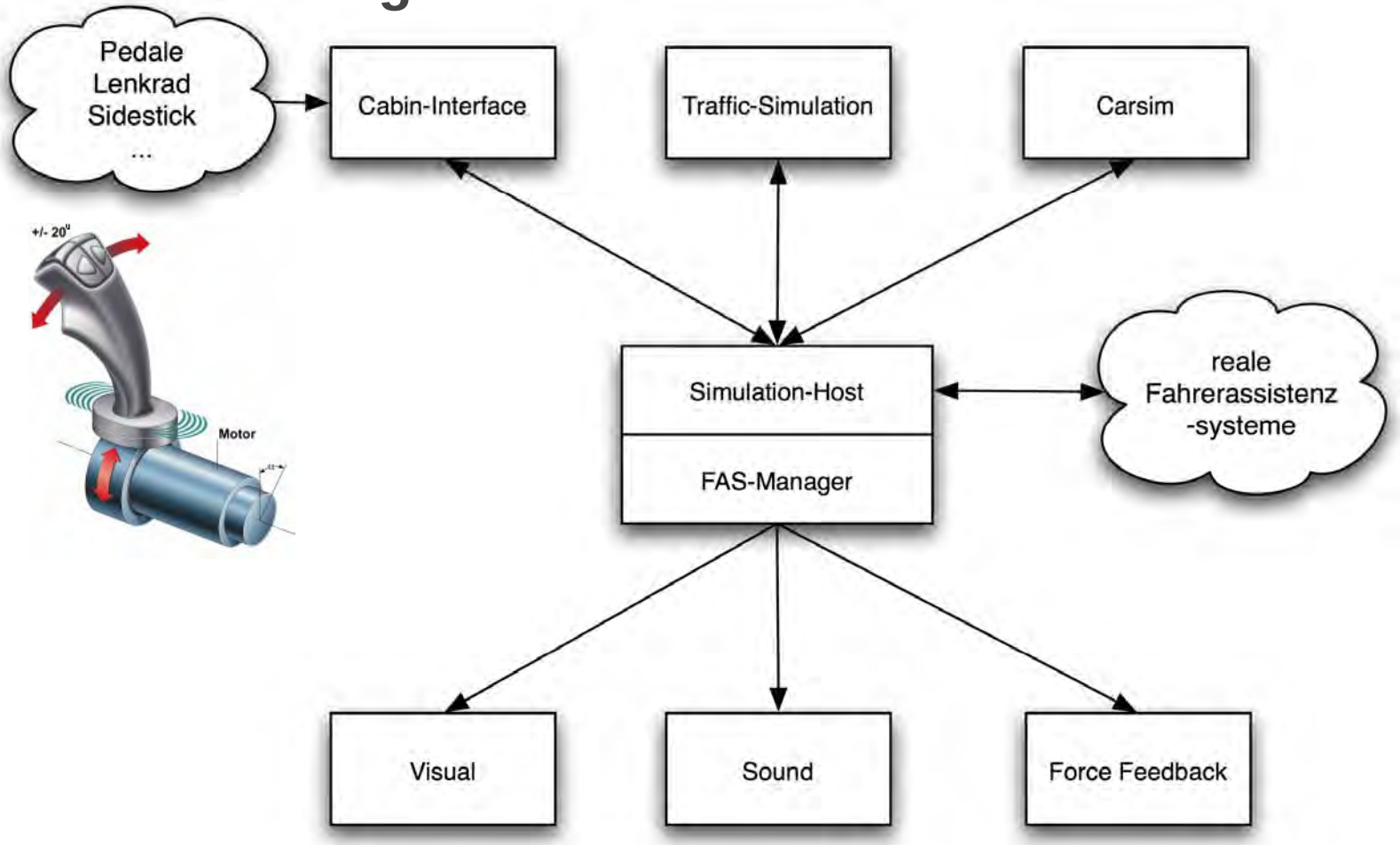
- Carsim bietet den Dienst „Fahrdynamik“ an
- Komponenten sind unter anderem Drehzahl, Geschwindigkeit, Winkelbeschleunigungen, Verbrauch
- Die Soundsimulation abboniert davon lediglich Drehzahl und Geschwindigkeit
- Sobald neue Daten für diese Komponenten durch den Dienst bereit gestellt werden, bekommt der Abbonent diese Daten zugestellt
- tauscht man die Fahrdynamiksimulation aus, kann die Soundsimulation unverändert bleiben, solange die benötigten Dienstkomponenten weiter verfügbar sind



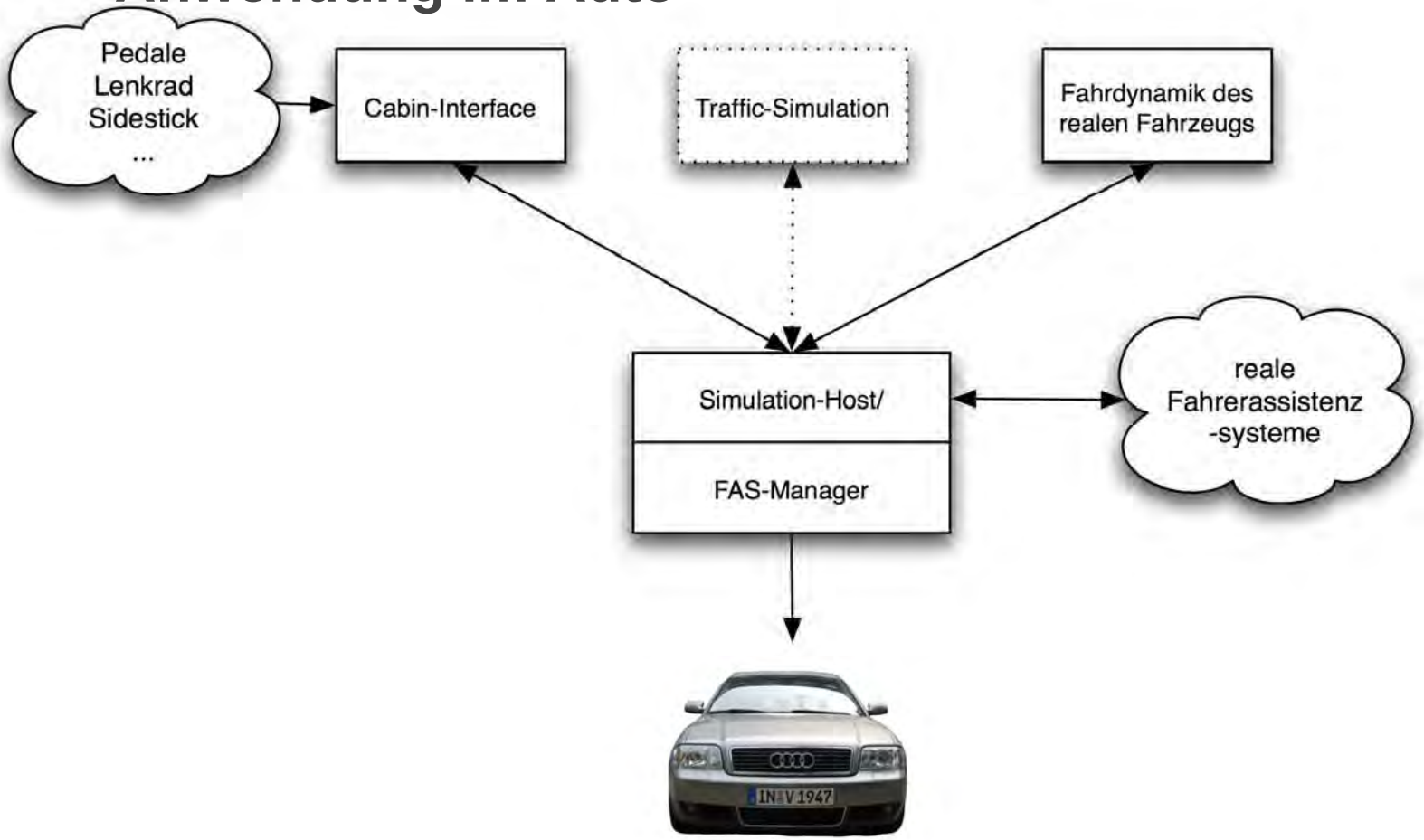
Systemarchitektur im FASLab
Anbindung von Assistenzsystemen



Systemarchitektur im FASLab
Anbindung realer Hardware

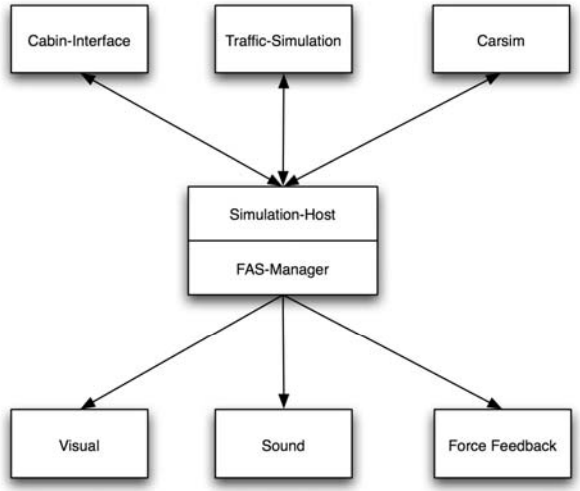


Systemarchitektur im FASLab
Anwendung im Auto

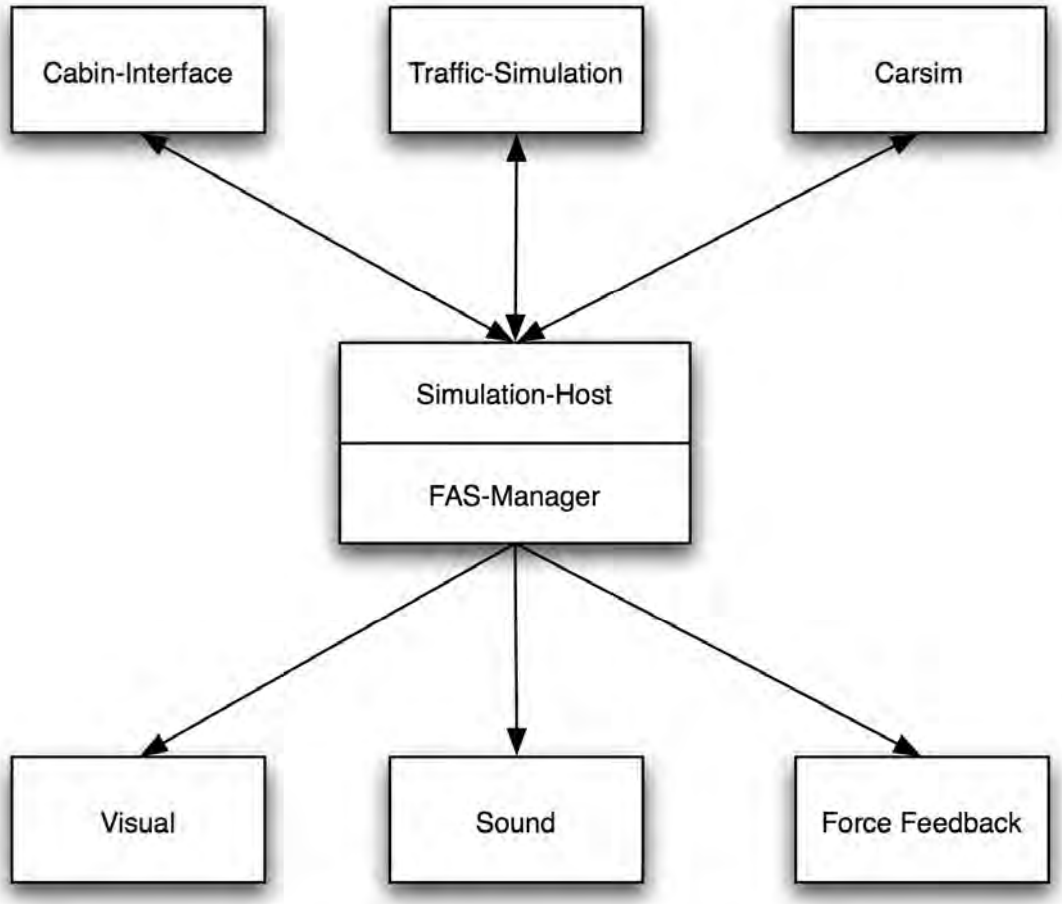


Systemarchitektur im FASLab
Zusammenfassung

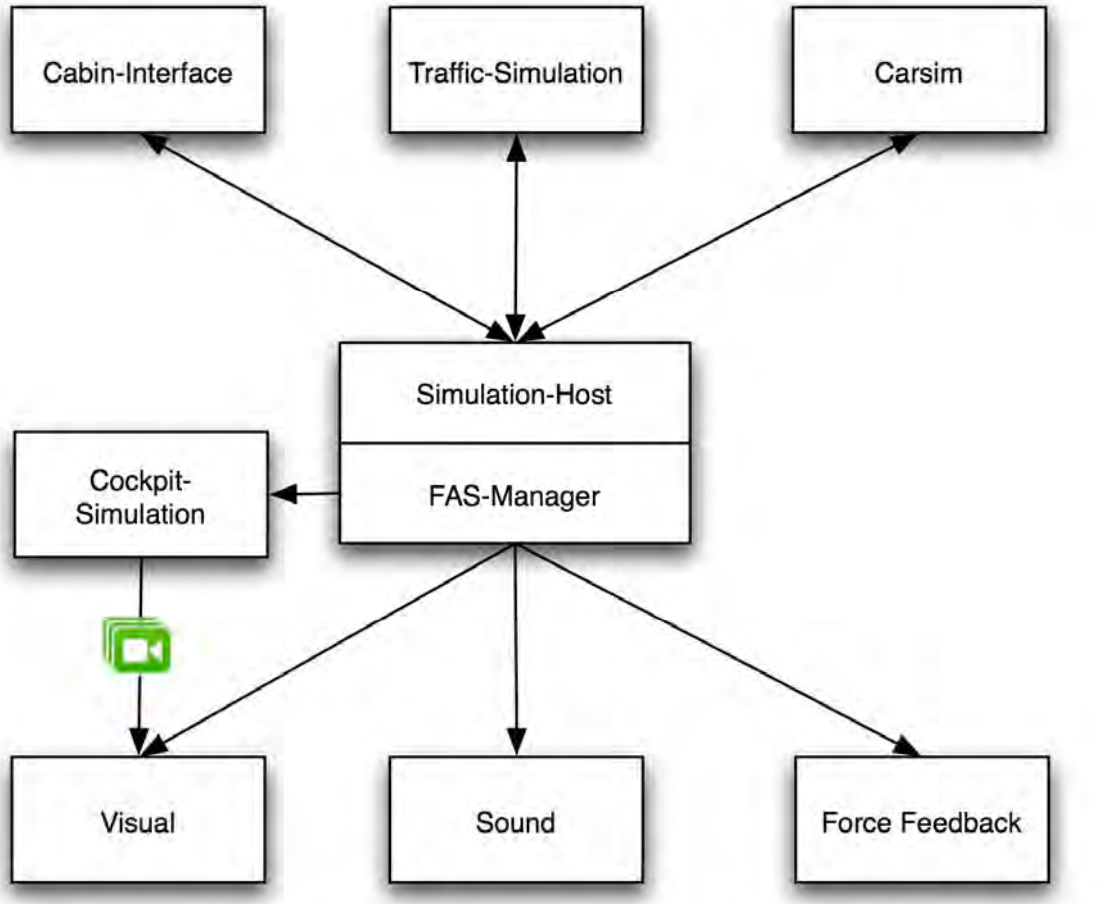
- Durch die modulare Struktur können Module und Dienste ausgetauscht oder erweitert werden
- Ein Dienst kann wahlweise durch Hard- oder Software angeboten werden
- FAS können als Simulation oder Steuergerät eingebunden werden
- Die Software kann sowohl im Simulator als auch im Fahrzeug laufen. So ergibt sich eine durchgängige Entwicklungskette von der Simulation bis hin zur Realität



Systemarchitektur im FASLab
Ausblick: Erweiterung um eine Cockpit-Simulation



Systemarchitektur im FASLab
Ausblick: Erweiterung um eine Cockpit-Simulation



Ausblick: Erweiterung um eine Cockpit-Simulation iObjects

Ausblick: Erweiterung um eine Cockpit-Simulation
Die Konfiguration der Streams erfolgt grafisch

Ausblick: Erweiterung um eine Cockpit-Simulation

- Das Erzeugen der Videostreams ist unabhängig von der verwendeten Software, so sind auch Tools wie z.B. Altia nutzbar
- Die Videostreams können sowohl im CAVE als auch im Fahrsimulator oder im realen Fahrzeug genutzt werden
- Die Ausgabe erfolgt entweder als Videotextur (VR-Labor) oder auf einem Touchscreen (Fahrsimulator oder Fahrzeug)
- Mittels Touchscreen können auch Eingaben erfolgen

Systemarchitektur im FASLab

Danke für Ihre Aufmerksamkeit.
Fragen? Bemerkungen?

schlegel simulation

Modellierung und Hardware-in-the-loop Simulation eines automatisierten Schaltgetriebes

Clemens Schlegel, Schlegel Simulation GmbH, Freising

cs@schlegel-simulation.de

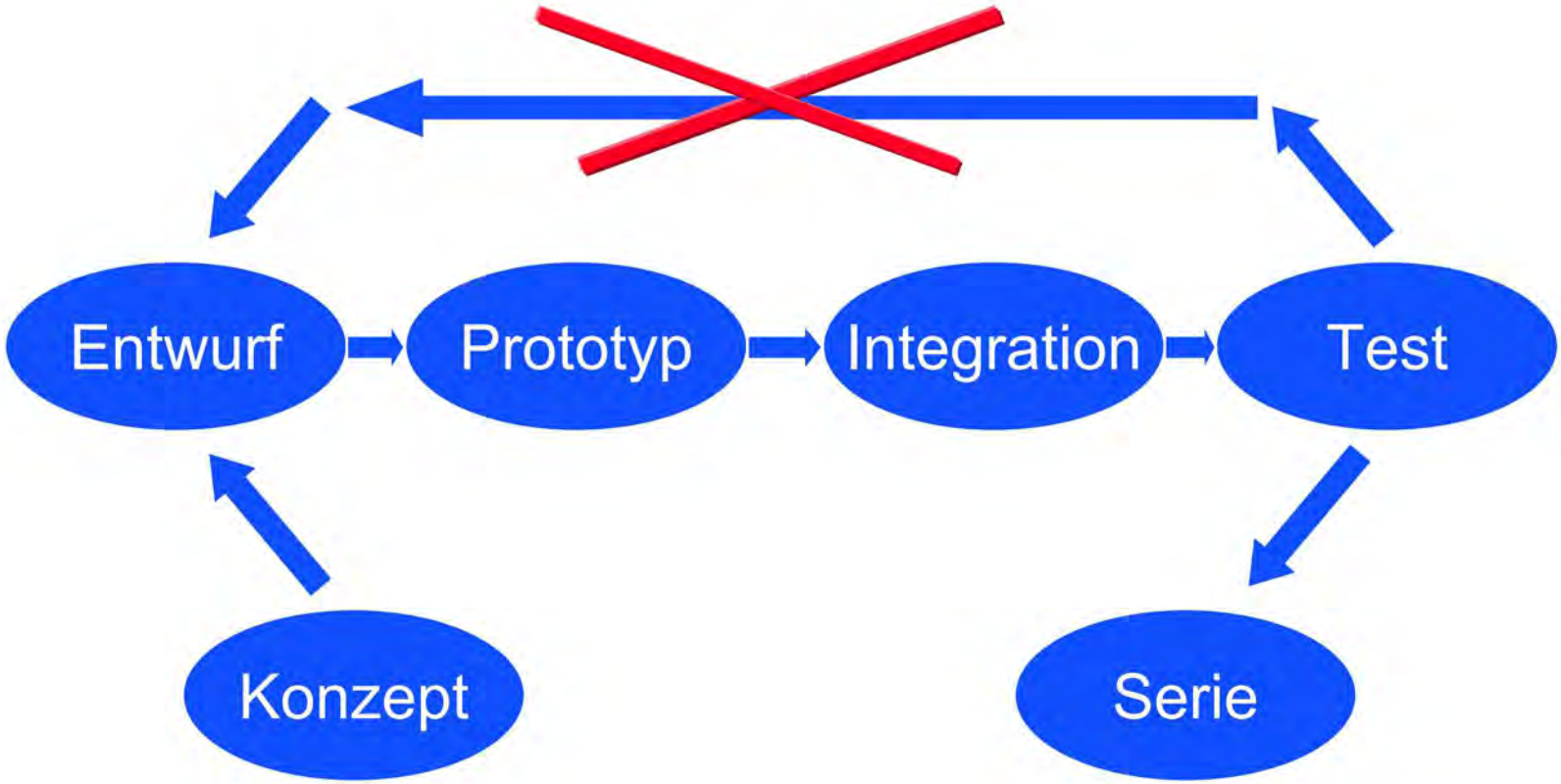
Übersicht

schlegel simulation

1. Warum Simulieren ?
2. Wie Modellieren ?
3. Automatisiertes Schaltgetriebe H-SMG
   - Getriebemechanik
   - Aktuatorik / Sensorik
4. Modellierung des Antriebstranges
5. Modellierung des Getriebes
6. Modellierung von Reibung
7. HIL Prüfstand
8. Simulationsergebnisse
9. Zusammenfassung

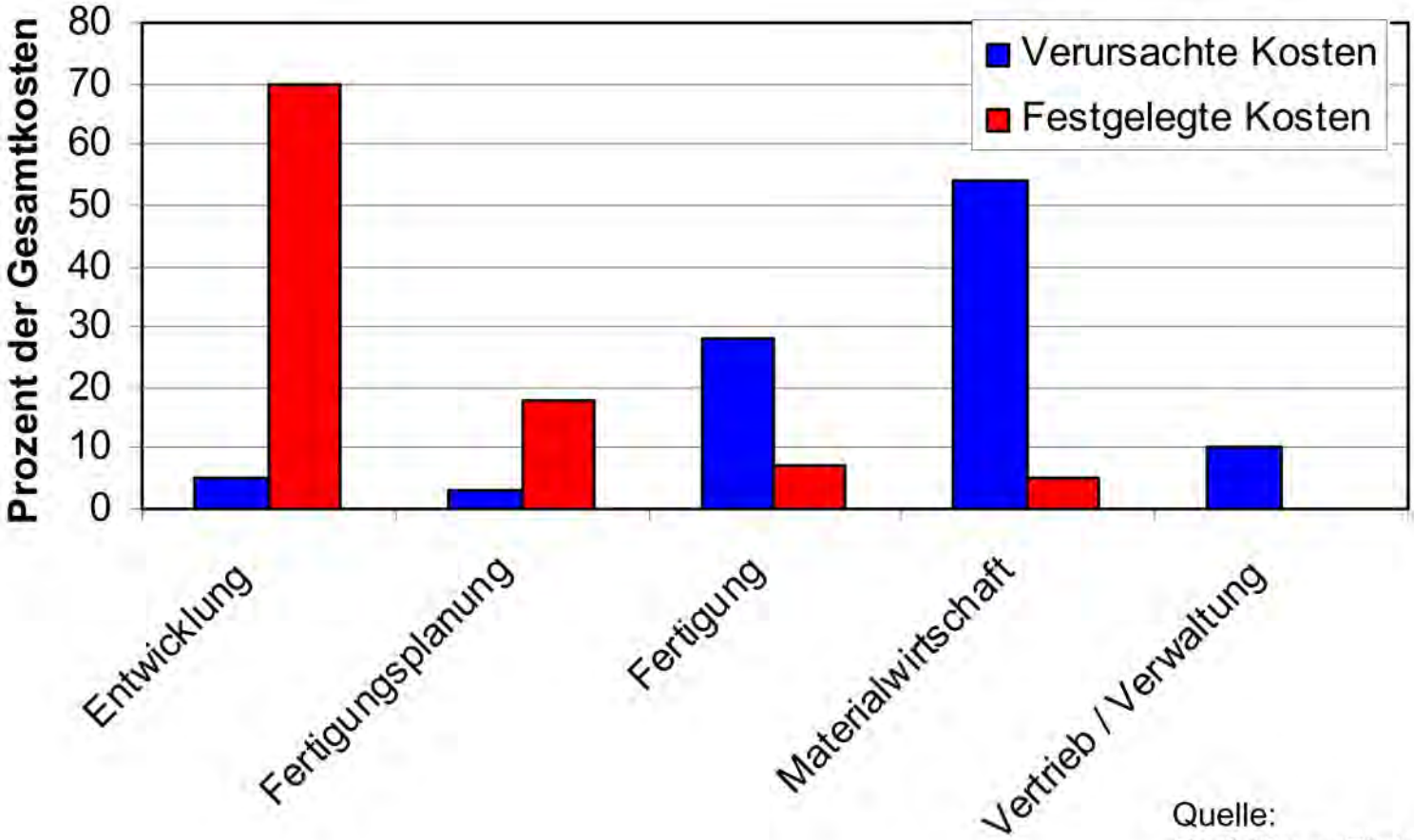
Warum Simulieren (1): Zeit

schlegel simulation



Warum Simulieren (2): Kosten

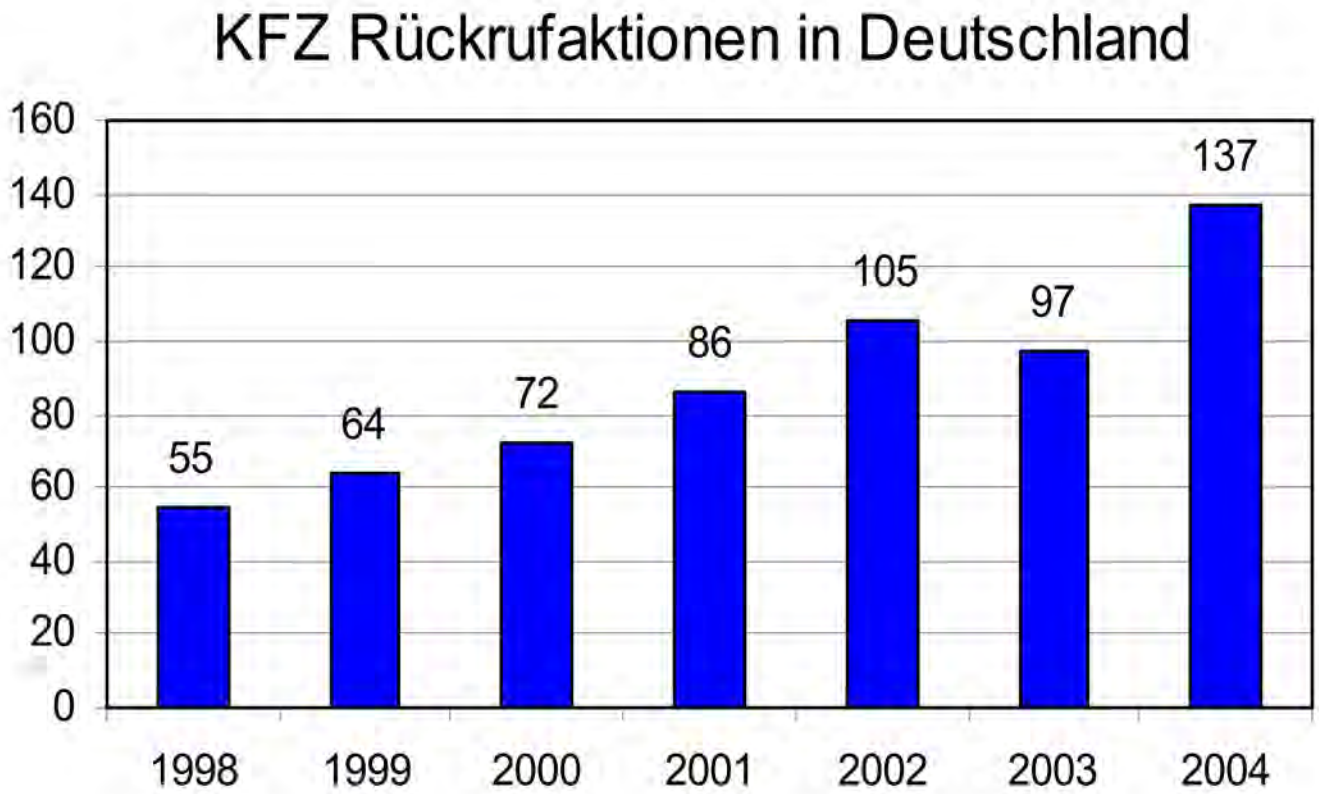
schlegel simulation



Quelle:
Prof. Klaus Ehrlenspiel, 1992

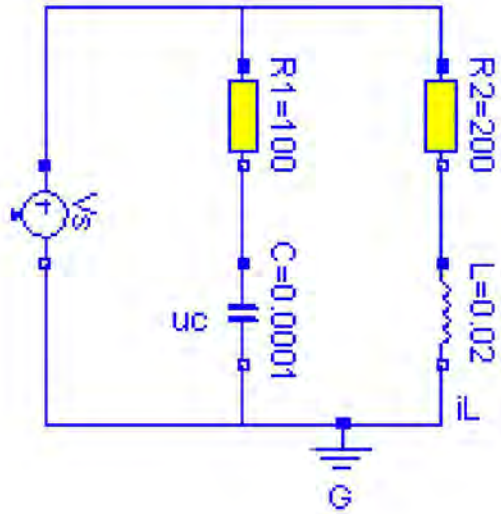
Warum Simulieren (3): Qualität

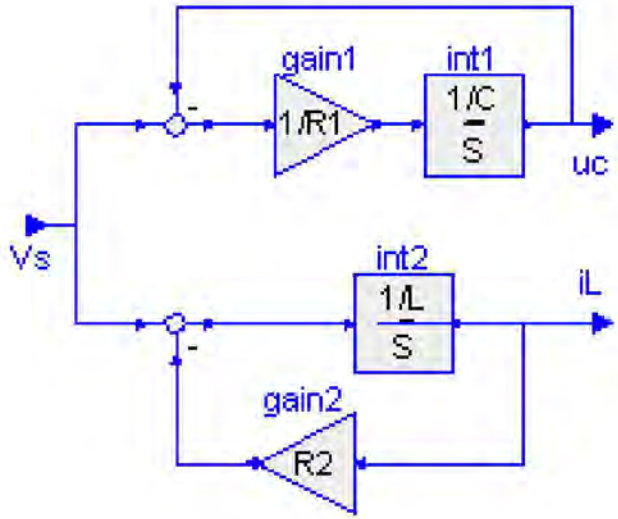
schlegel simulation



Quelle:
Kraftfahrt-Bundesamt 1/2005

Wie Modellieren: Objektorientiert (1)



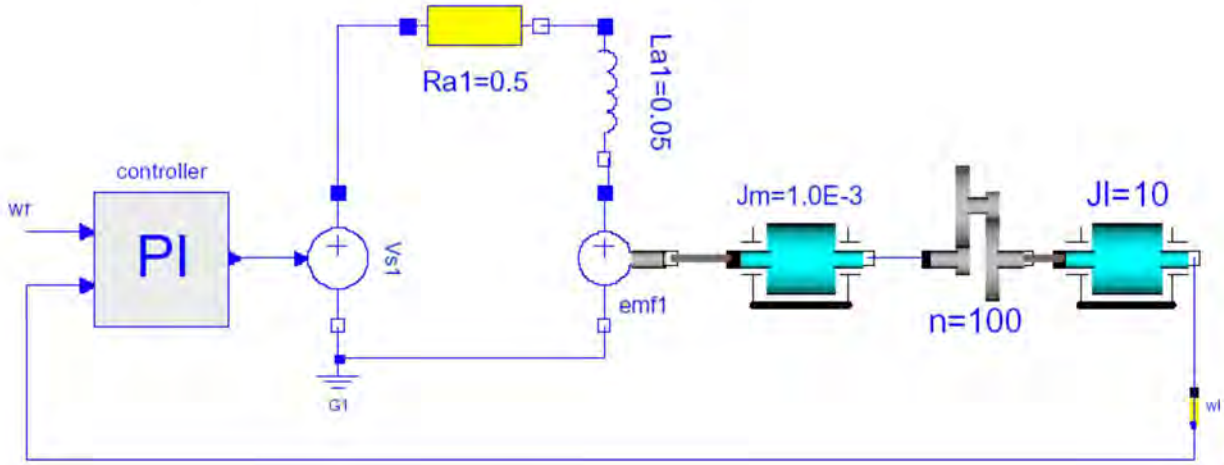


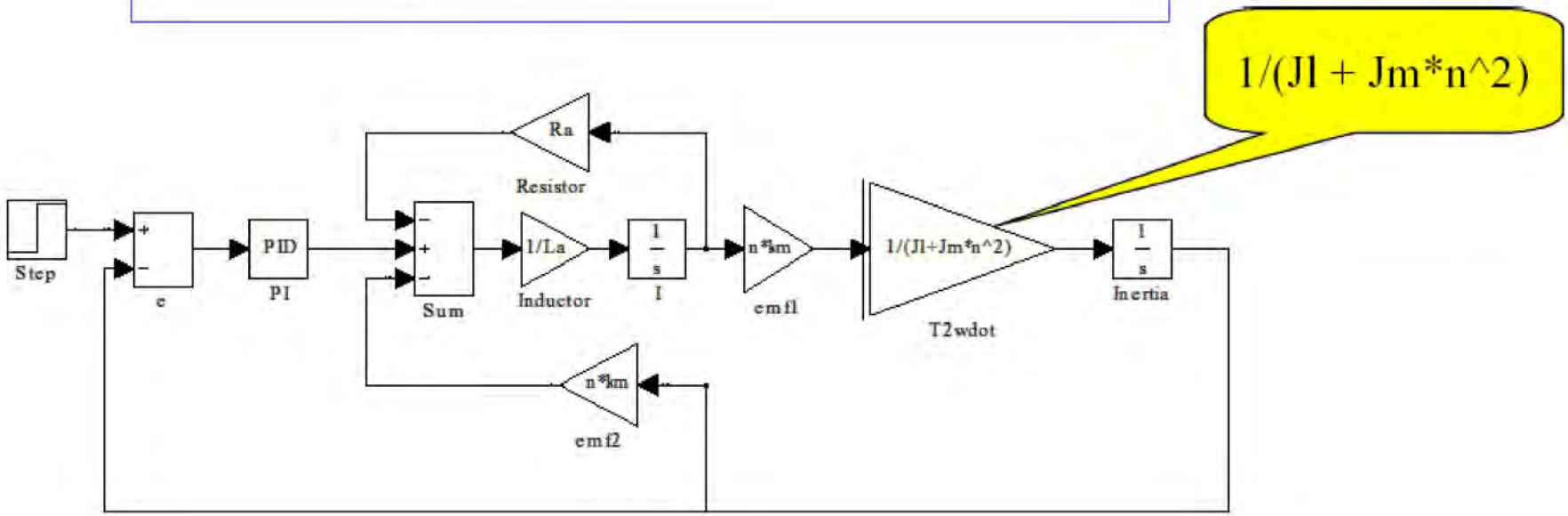
$$\frac{d}{dt} i_L = \frac{Vs - i_L R_2}{L}$$

$$\frac{d}{dt} u_c = \frac{Vs - u_c}{R_1 C}$$

Wie Modellieren: Objektorientiert (2)

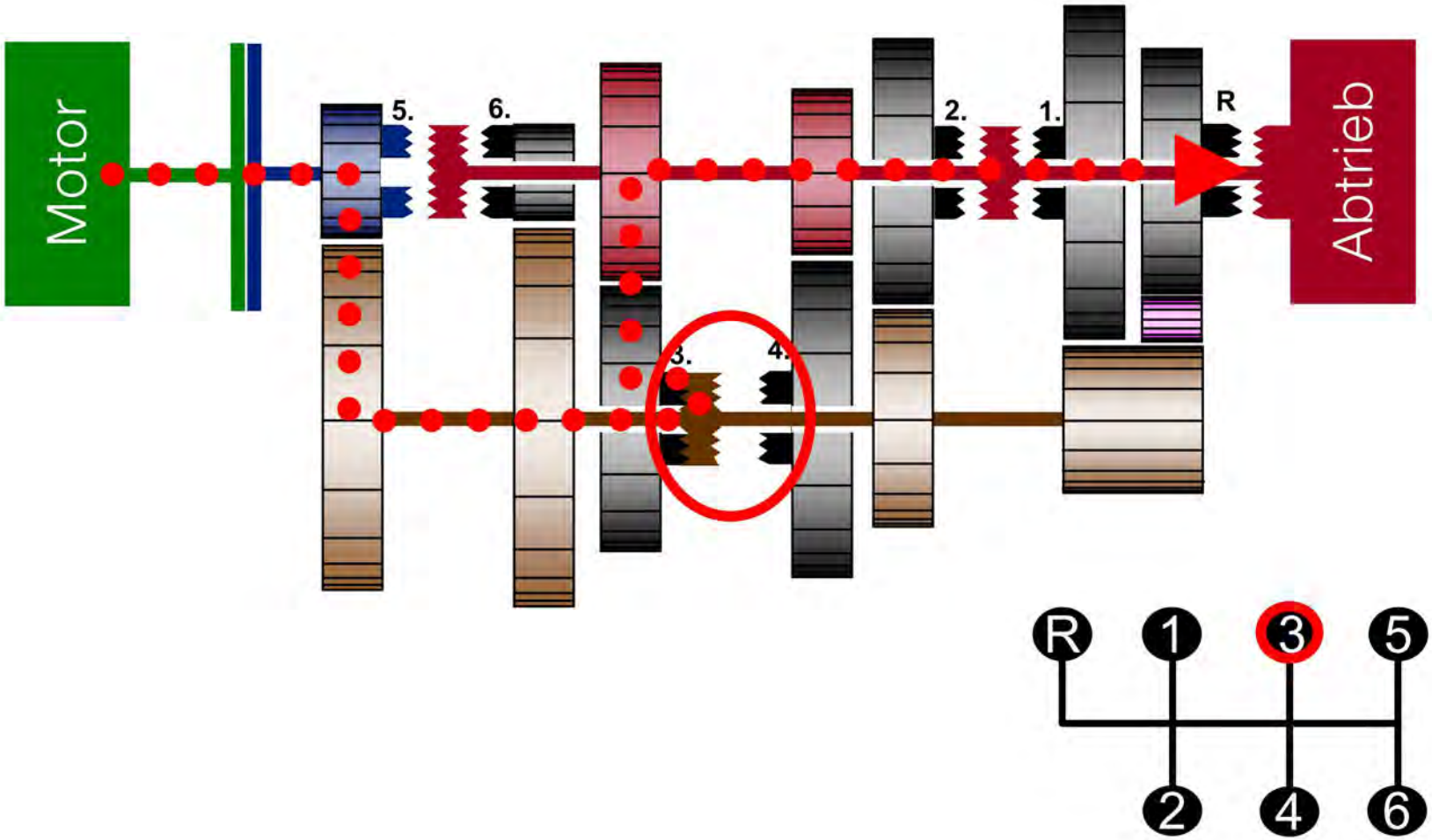
schlegel simulation





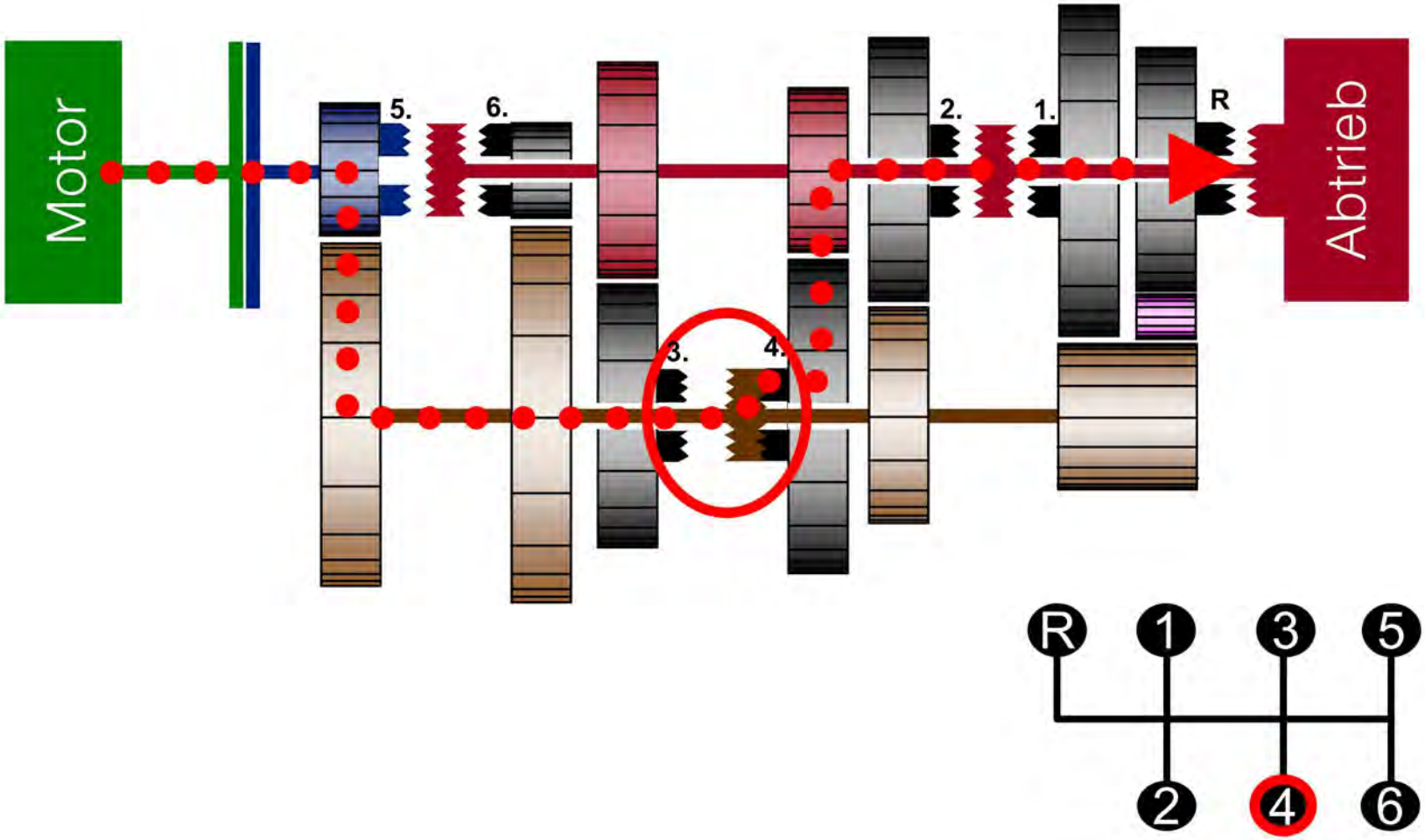
H-SMG: Getriebeschema 3. Gang

schlegel simulation



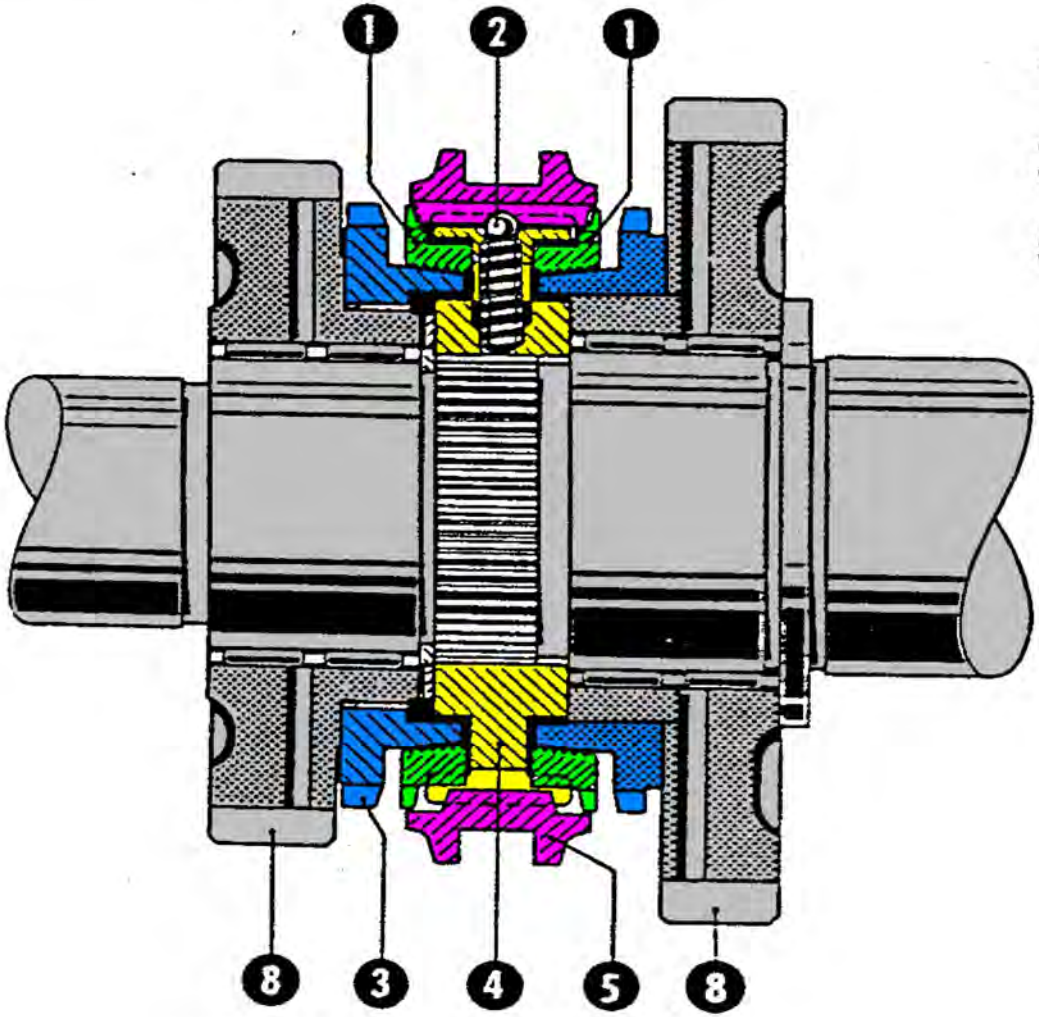
H-SMG: Getriebeschema 4. Gang

schlegel simulation

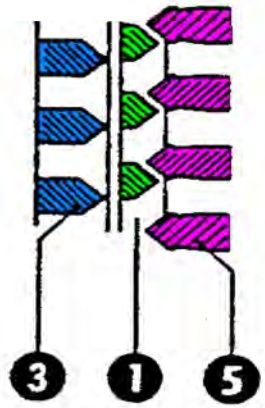


Schaltung: Elemente einer Synchronisation

schlegel simulation

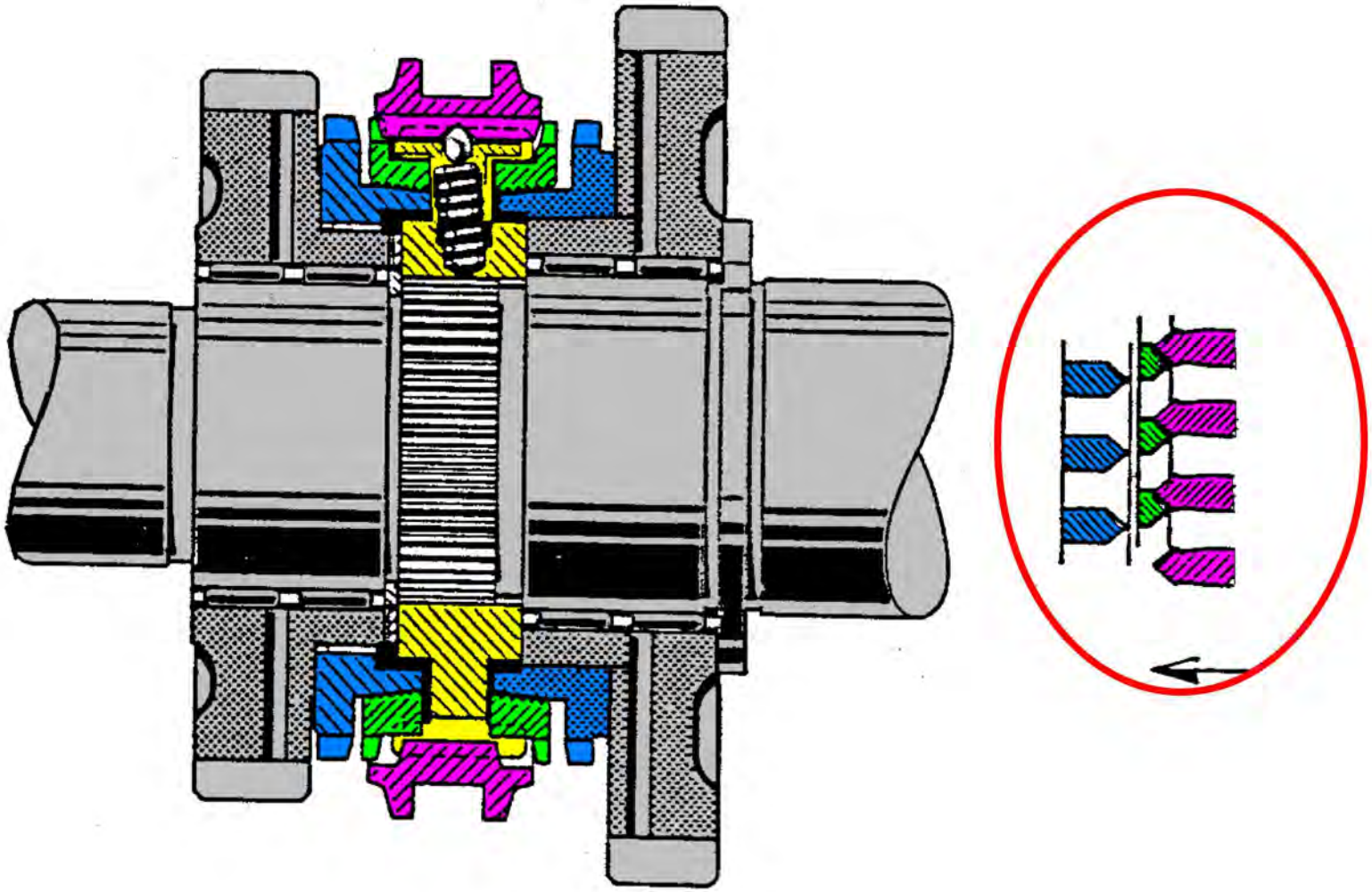


1 Synchronring
3 Zahnrad
5 Welle mit Muffe



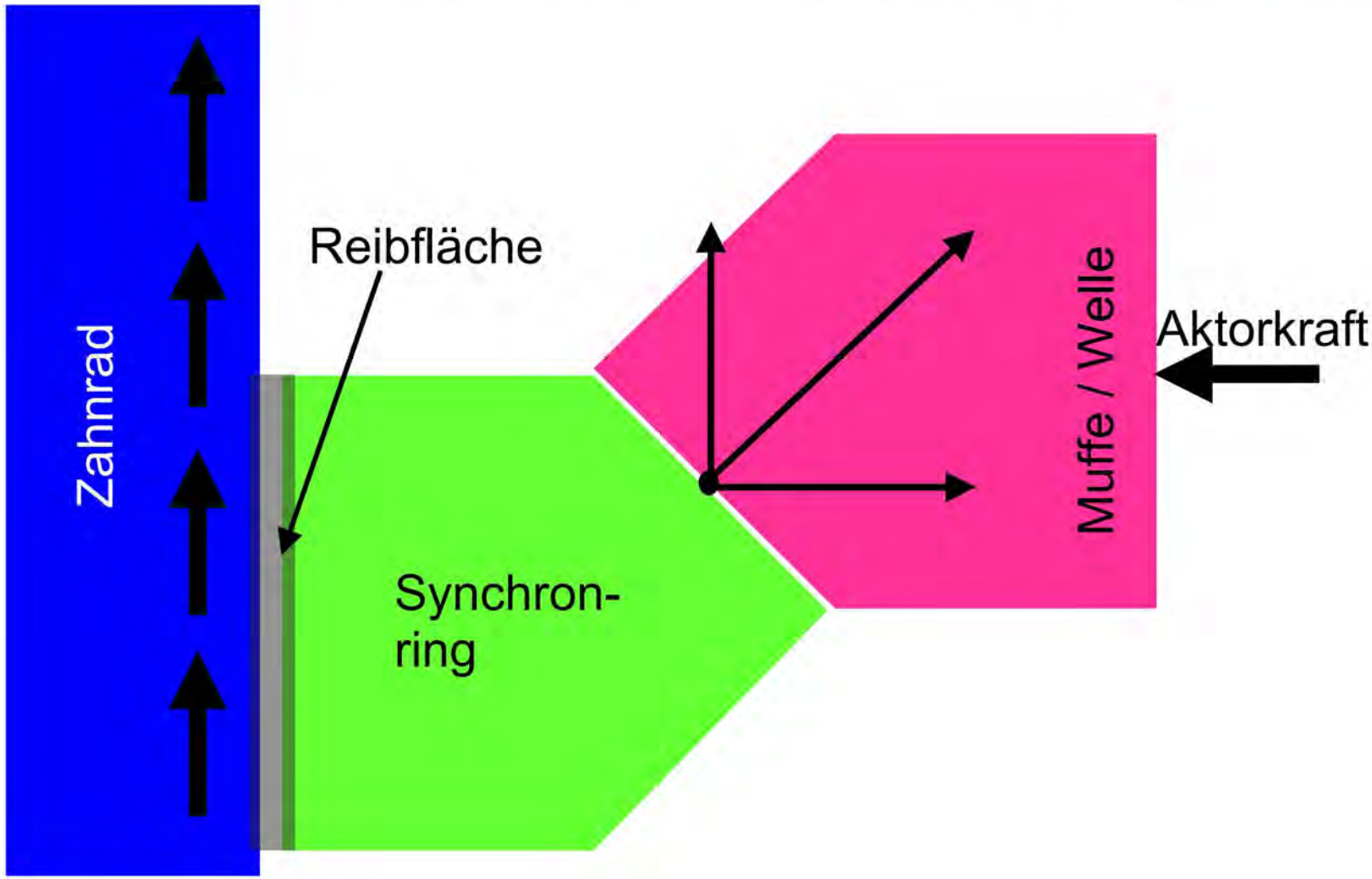
Schaltung: Synchronisationsphase (1)

schlegel simulation



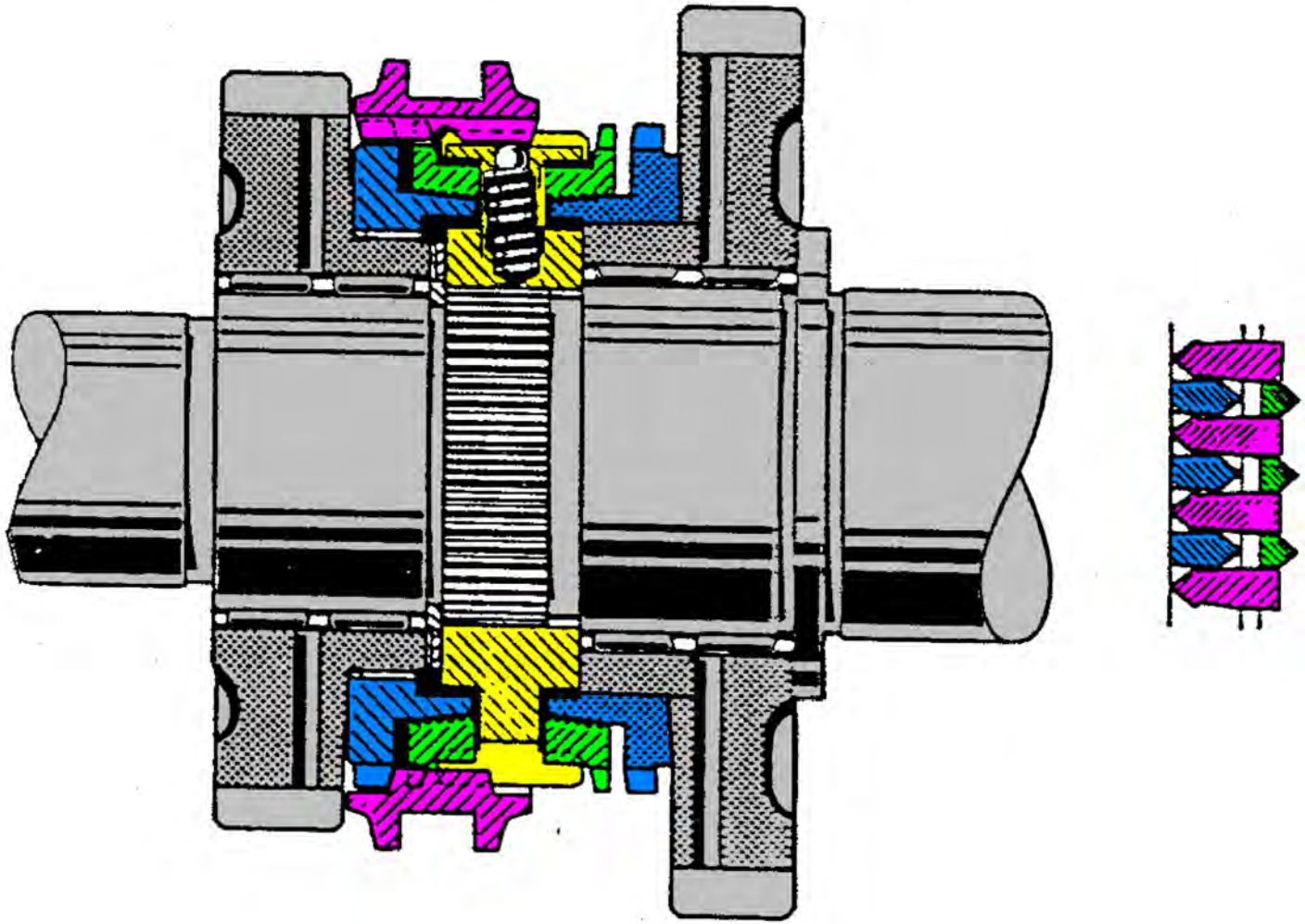
Schaltung: Synchronisationsphase (2)

schlegel simulation



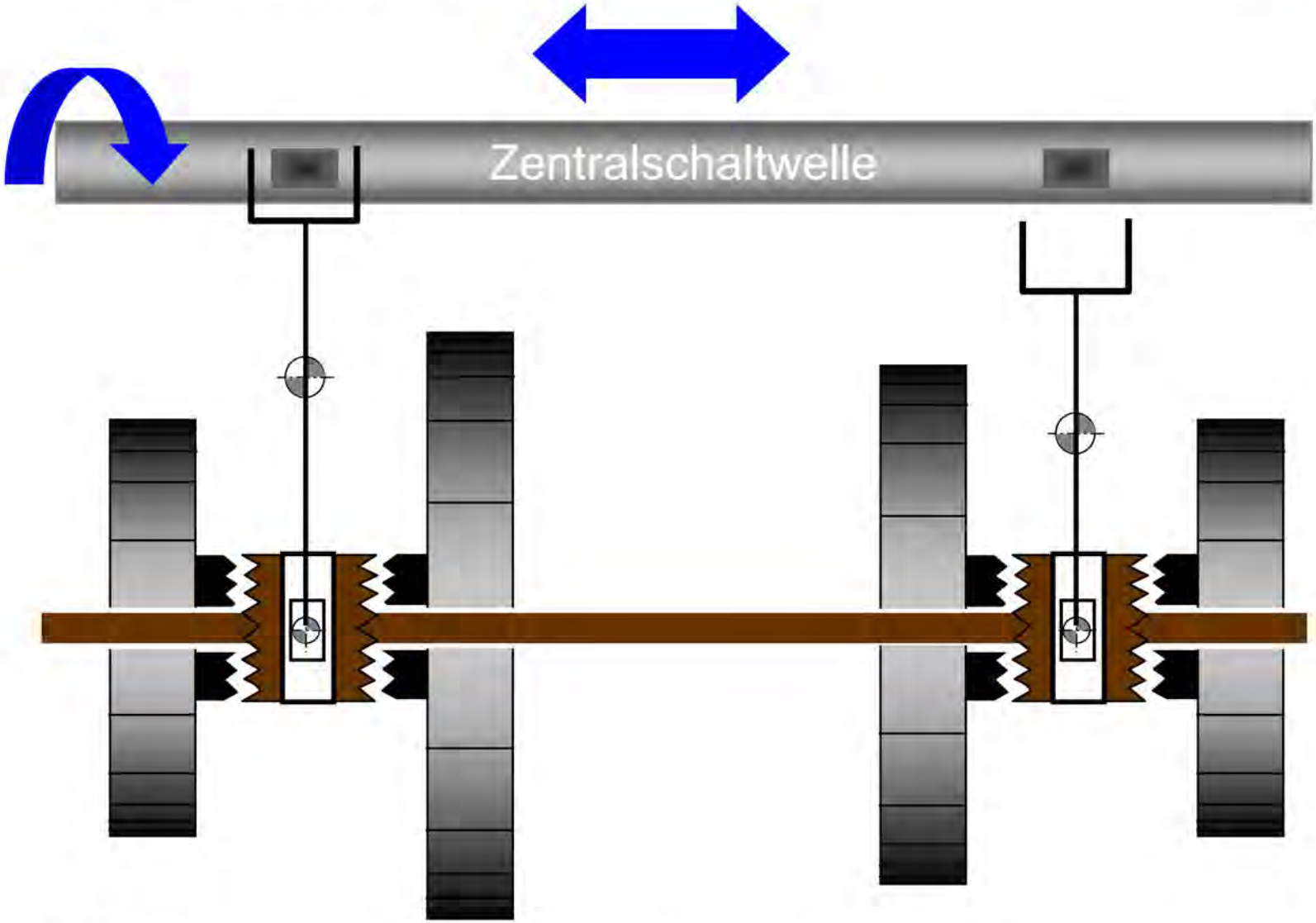
Schaltung: Einfädeln, Anschlag

schlegel simulation



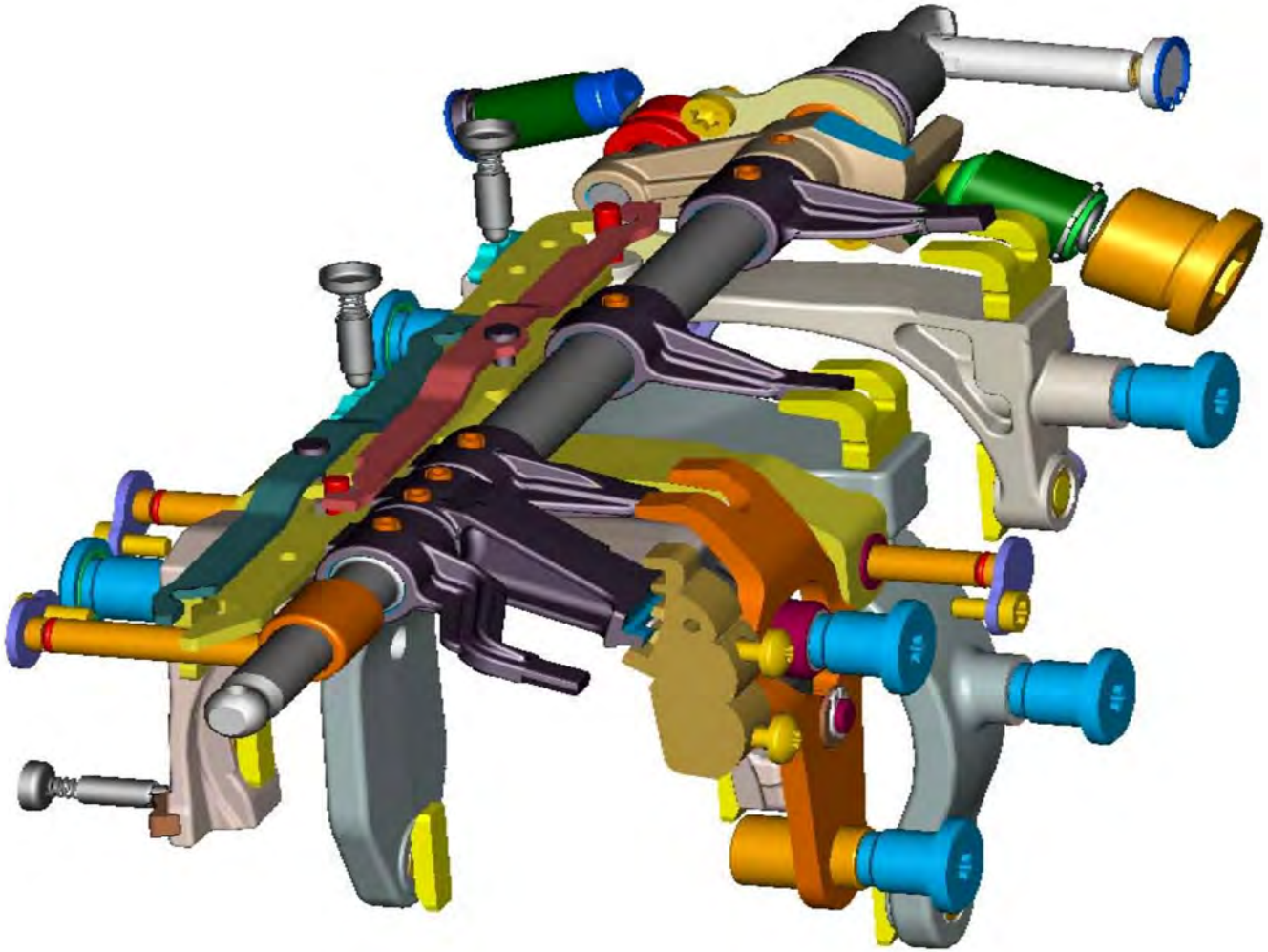
H-SMG: Innere Schaltung (1)

schlegel simulation



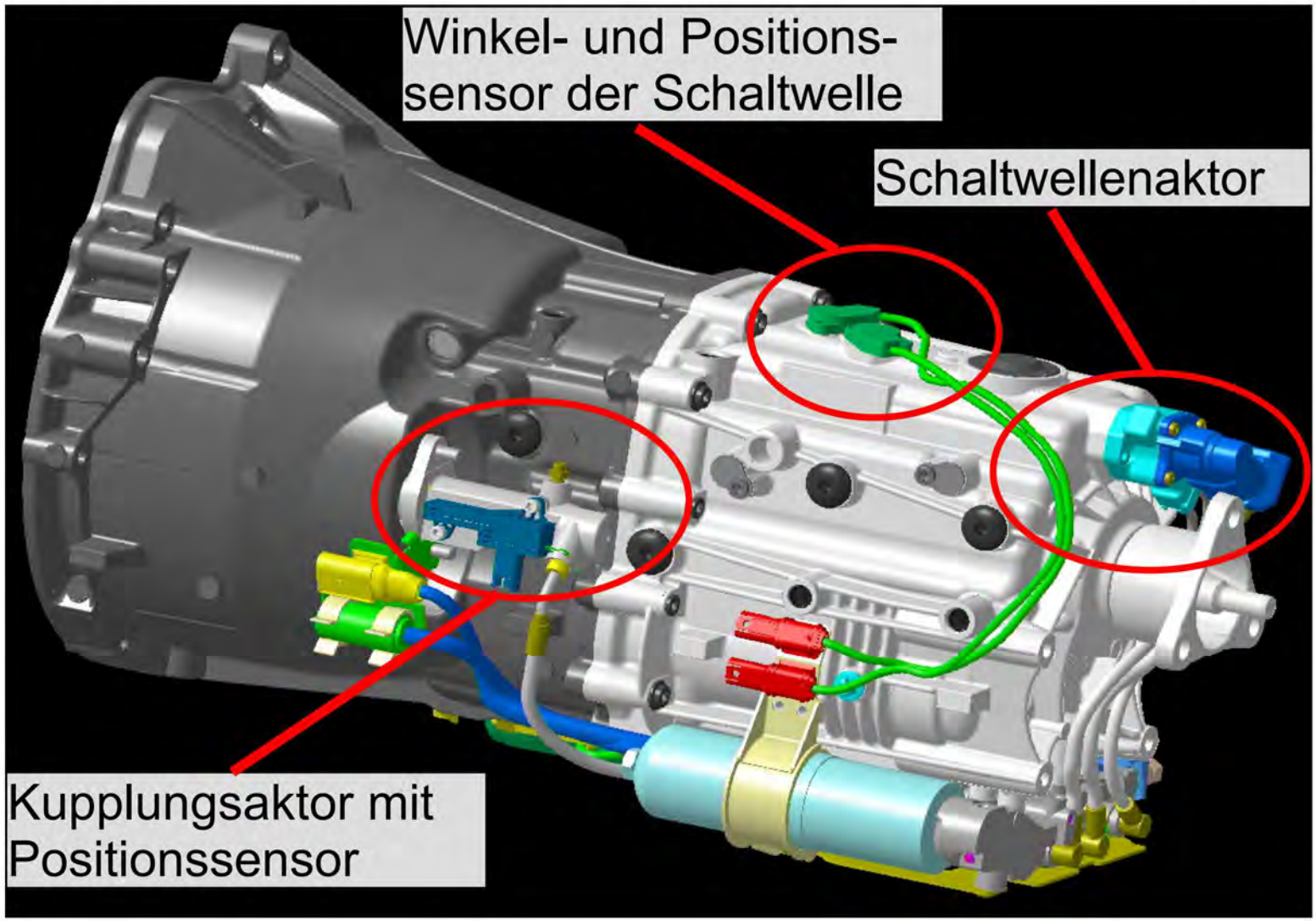
H-SMG: Innere Schaltung (2)

schlegel simulation



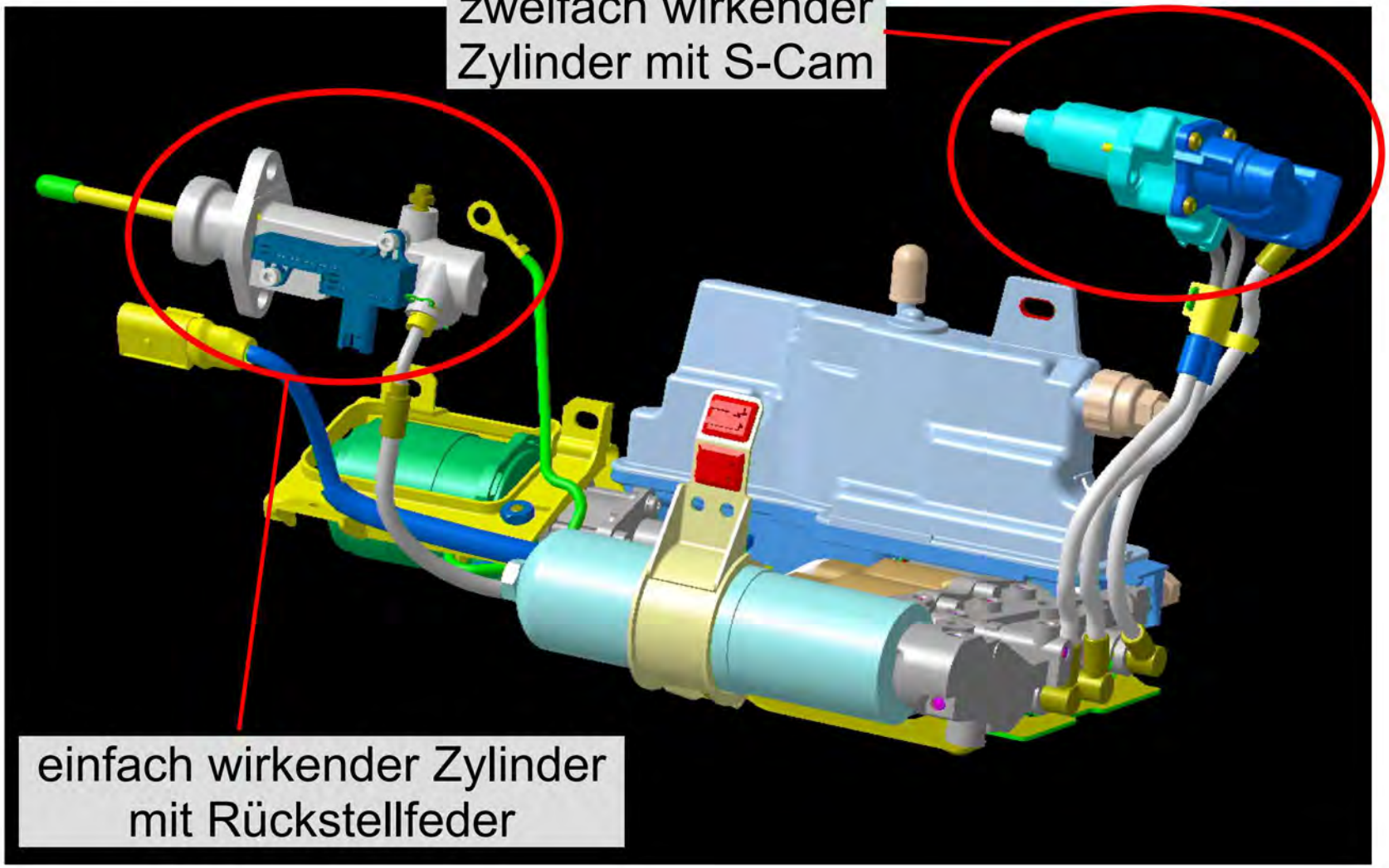
Aktuatorik / Sensorik: Übersicht

schlegel simulation



Aktuatorik / Sensorik: Übersicht

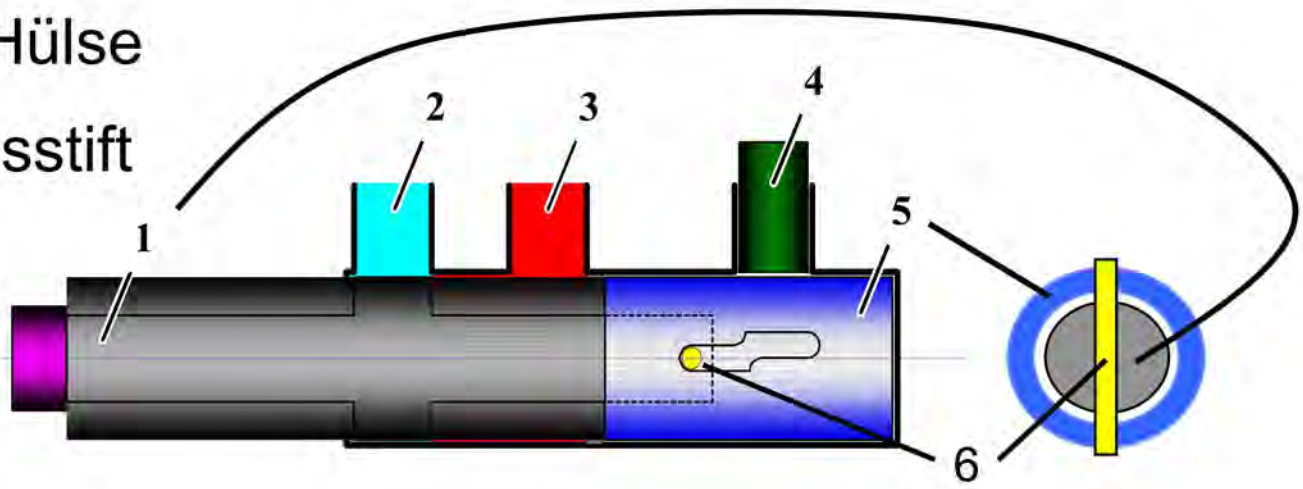
schlegel simulation



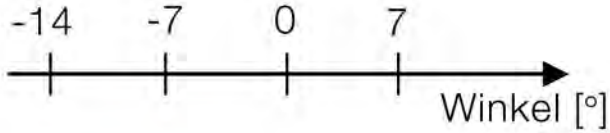
Aktuatorik / Sensorik: Funktion S-Cam

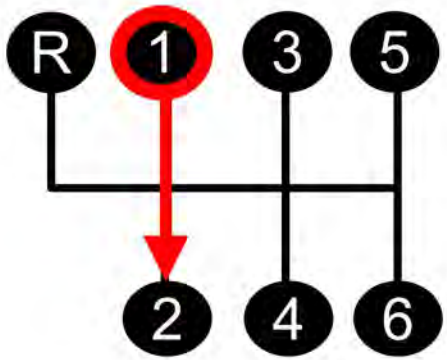
schlegel simulation

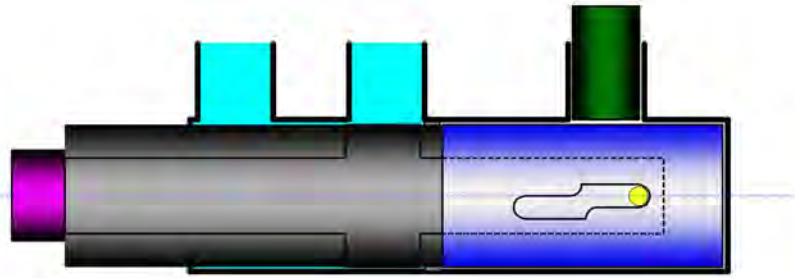
1. Anbindung an Zentralschaltwelle
2. Hydraulikanschluss Zylinder 1
3. Hydraulikanschluss Zylinder 2
4. S-Cam Bremse
5. S-Cam Hülse
6. Führungsstift

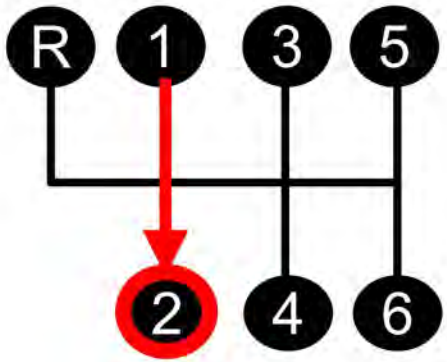


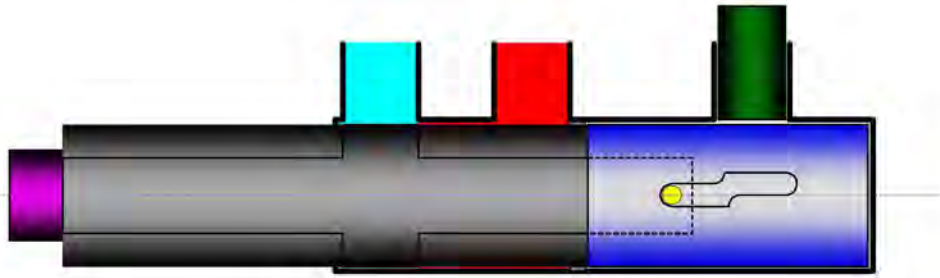
Aktuatorik / Sensorik: Schaltablauf S-Cam



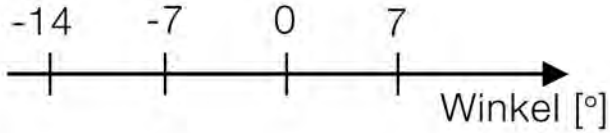


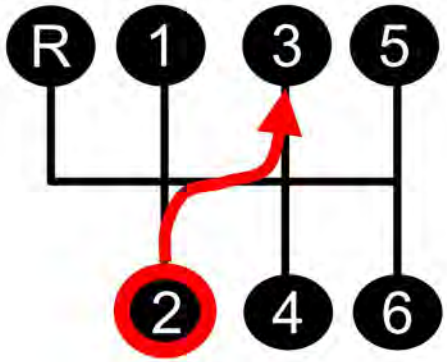


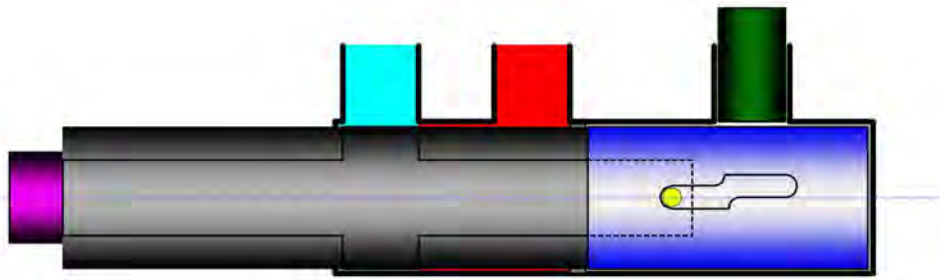


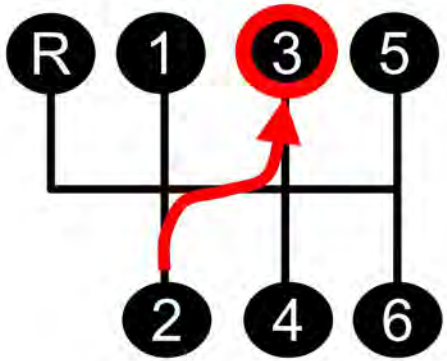


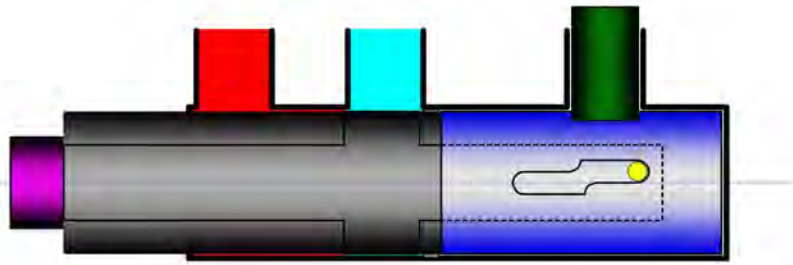
Aktuatorik / Sensorik: Schaltablauf S-Cam





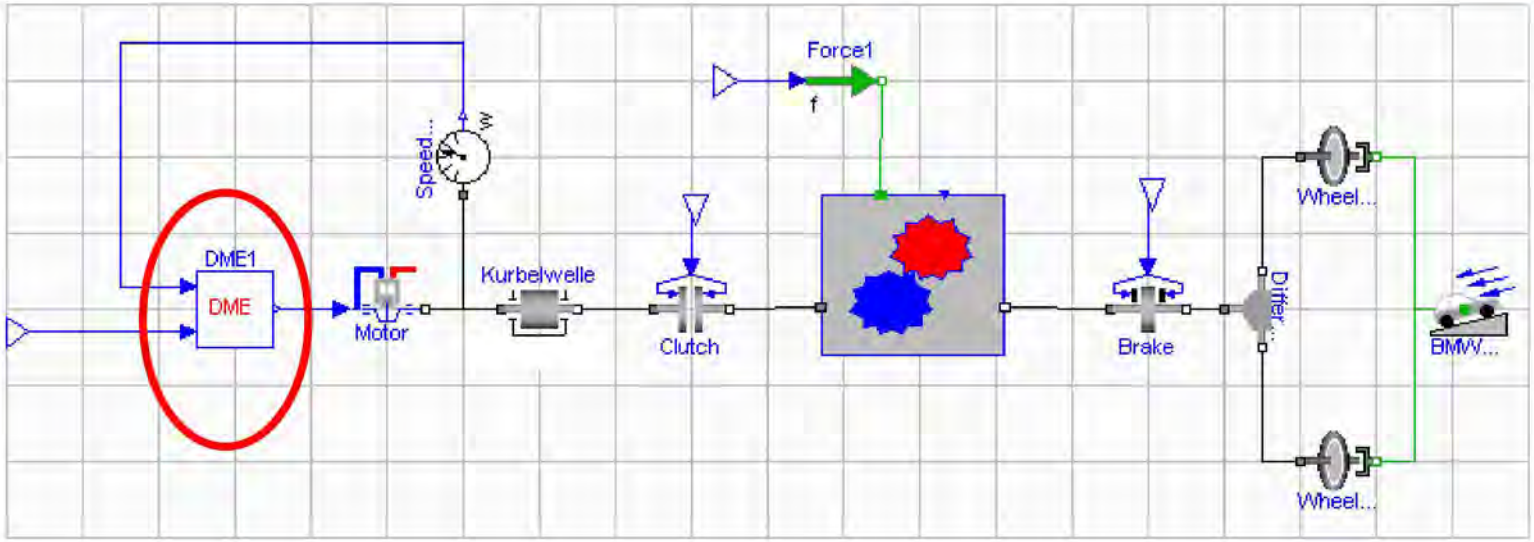






Modellierung: Antrieb

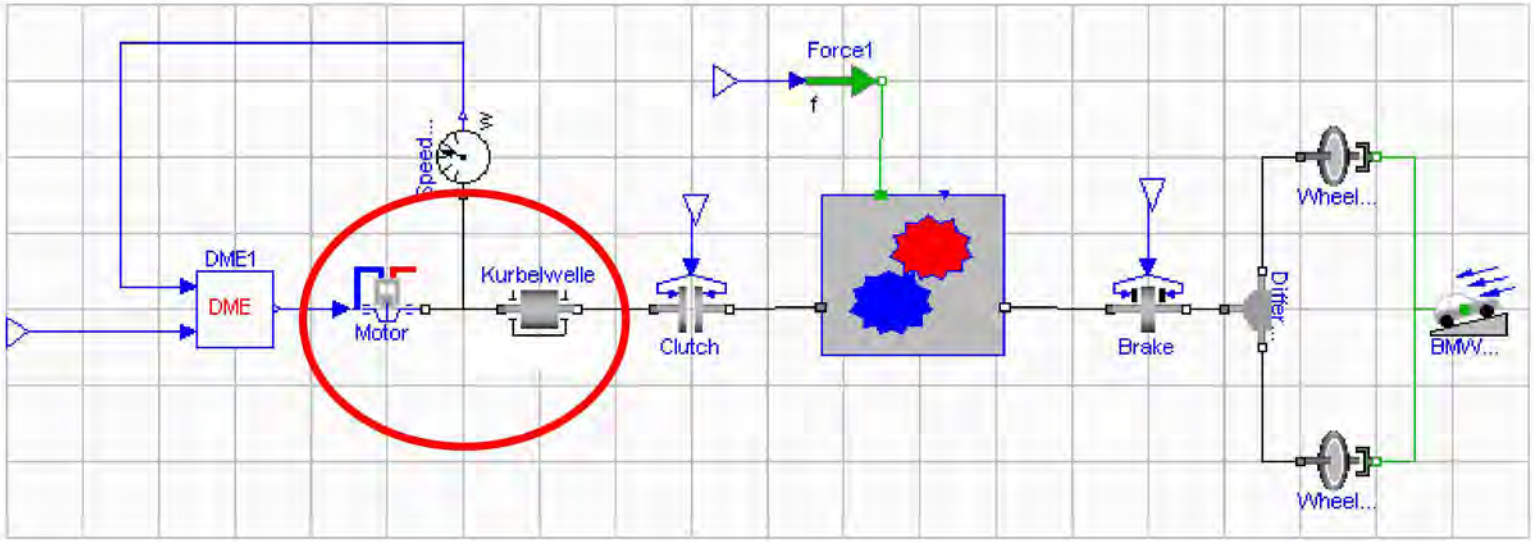
schlegel simulation



Teilmodell eines Motorsteuergeräts

- Leerlaufregler
- Interpretation Gaspedalstellung
- CAN-Kommunikation mit dem Getriebesteuergerät

Modellierung: Antrieb

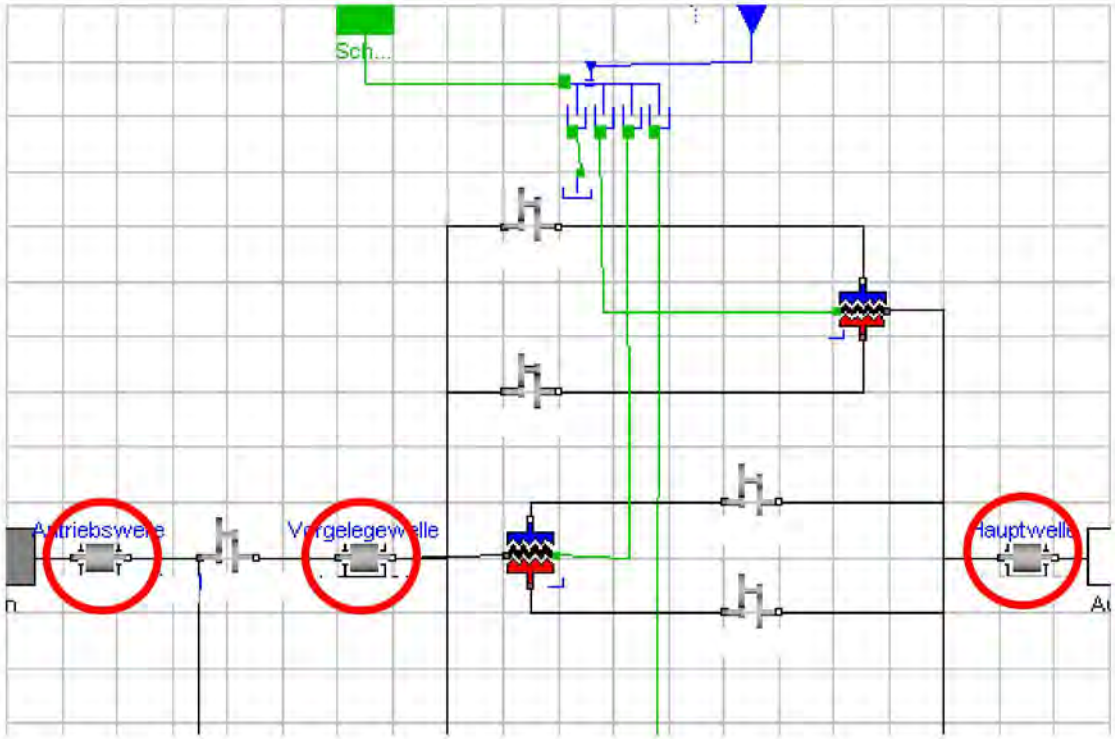


Motormodell

- Kennfeld Motormoment M=f(Nmot,Gaspedal)
- Luftmassendynamik als PT1

Modellierung: Getriebe

schlegel simulation

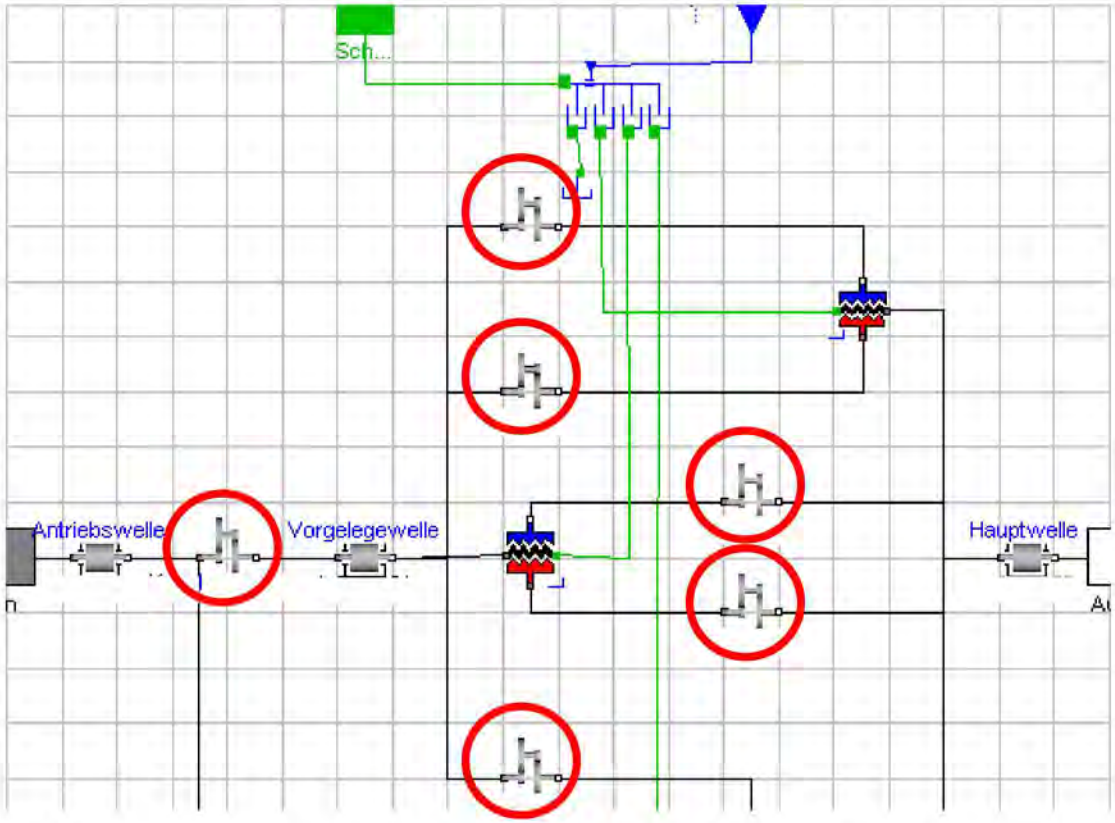


Trägheiten

- $d\phi/dt = \omega$
- $J * d\omega/dt = \sum M$

Modellierung: Getriebe

schlegel simulation

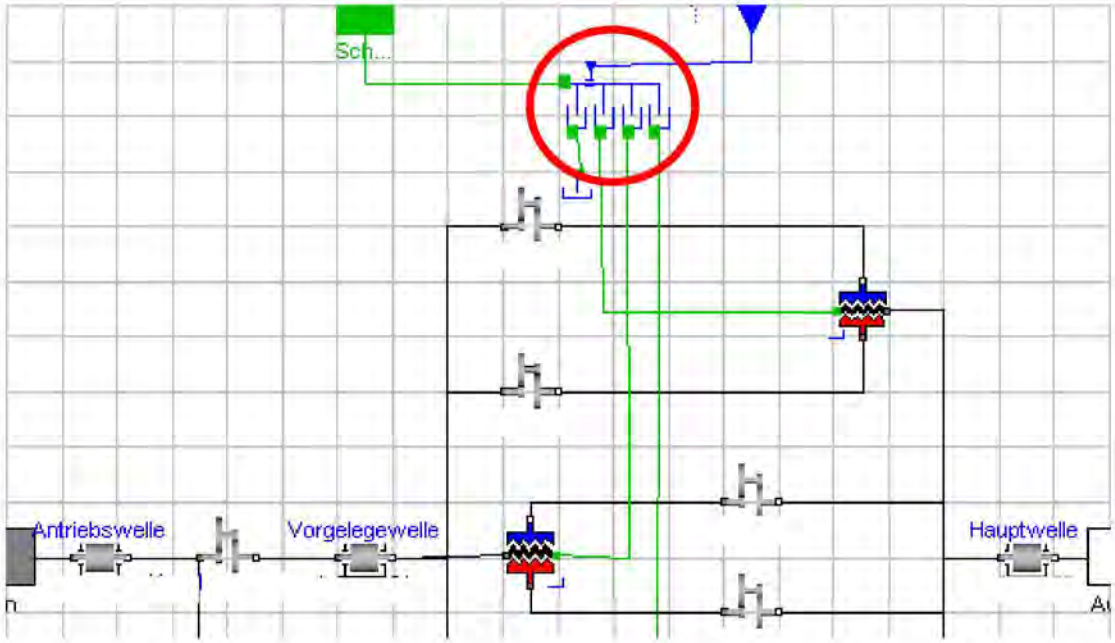


Übersetzungen mit Zähnezahlen z_1, z_2

- $\omega_1 / \omega_2 = z_1 / z_2$
- $M_1 / M_2 = z_2 / z_1$

Modellierung: Getriebe

schlegel simulation



Modell der inneren Schaltung

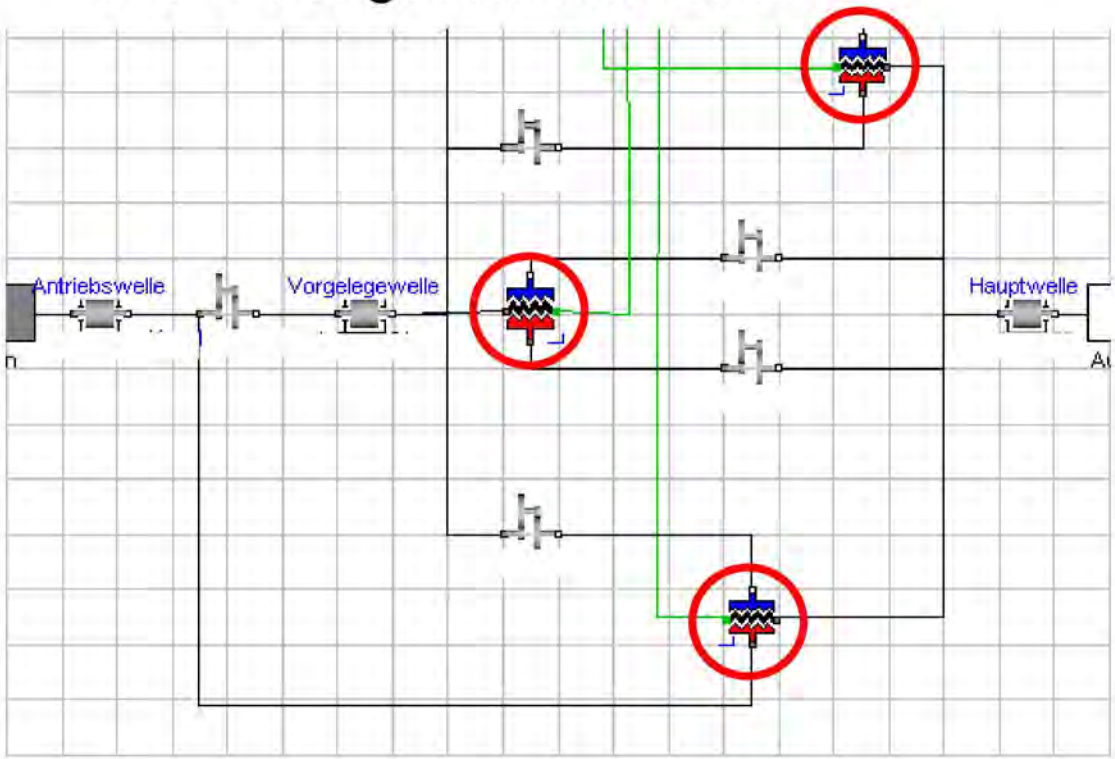
- Auswahl der gültigen Gasse (S-Cam-Funktion) und damit Anbindung der richtigen Muffe an die Zentralschaltwelle
- Bestimmung der Schaltwellen- und Muffenposition

Modellierung: Getriebe

schlegel simulation

Modell der Synchronisierung

- Verkopplung der jeweiligen Drehträgheiten über Reib- und Zwangsdrehmomente

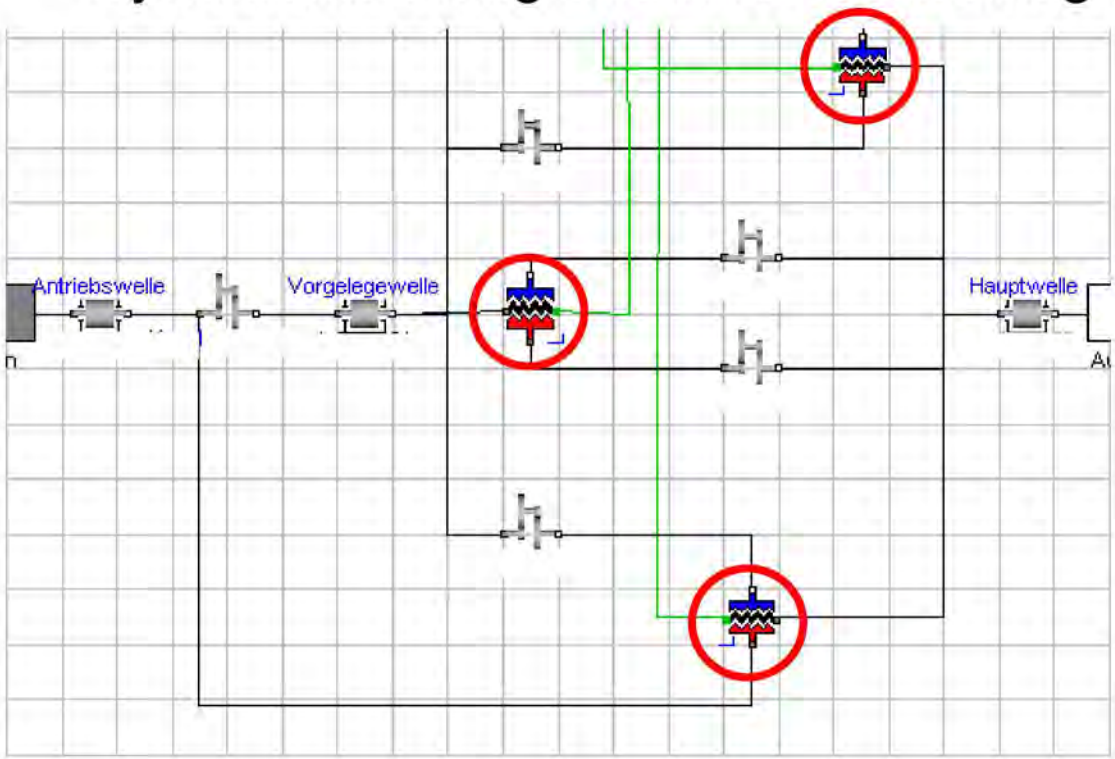


Modellierung: Getriebe

schlegel simulation

Modell der Synchronisierung

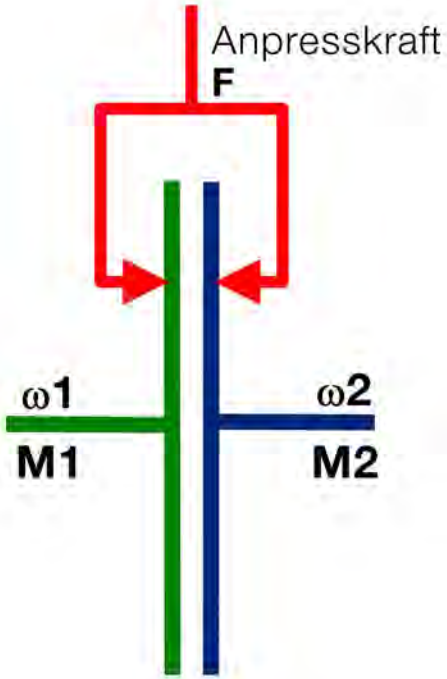
- Translatorische Zwangskräfte während der Synchronisierung und im Endanschlag der Muffe

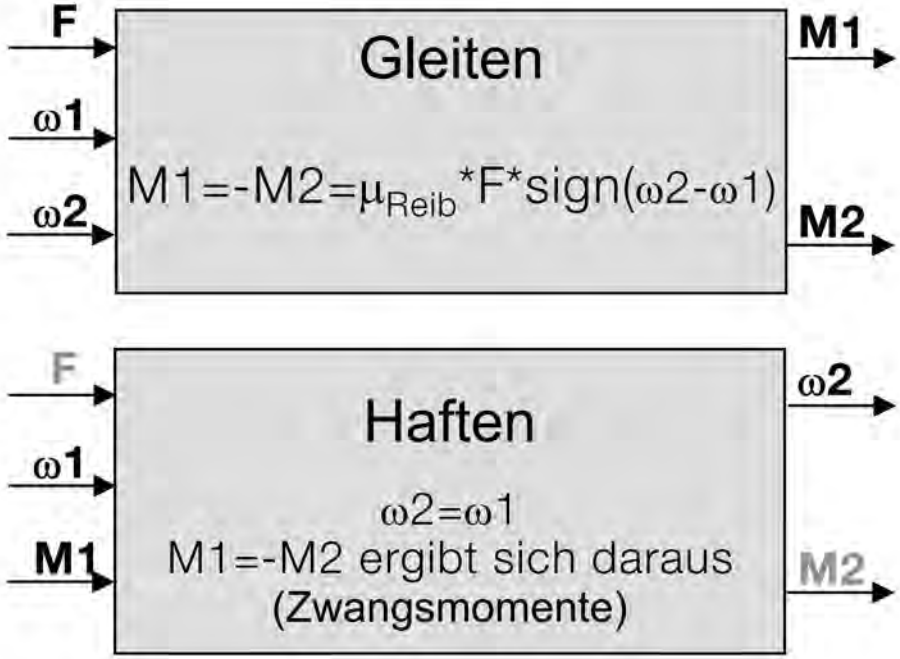


Modellierung von Reibung

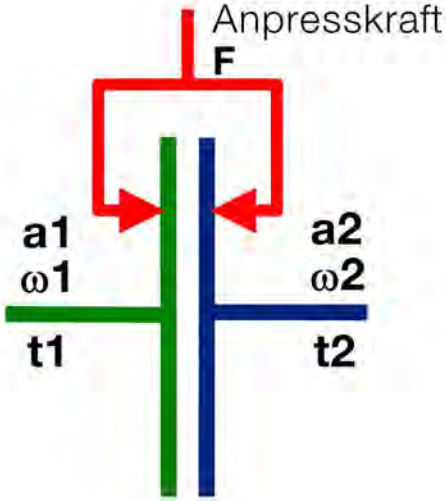
schlegel simulation

- Kausalitätswechsel: Änderung des Signalflusses
- Erläuterungen anhand einer Kupplung:



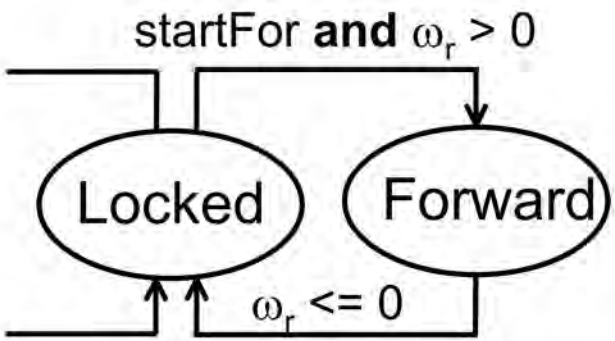


Akausales Reibungsmodell in Modelica



tz: in der Kupplung übertragenes Zwangsmoment

Locked: Kupplung haftet / gleitet (aus Zustandsautomat)



$\omega_r = \omega 2 - \omega 1$

0 = **if** *Locked* **then** *a2-a1* **else** *tz*

-t1 = **if** *Locked* **then** *tz* **else** $f(\omega_r, F)$

t1 = *t2*

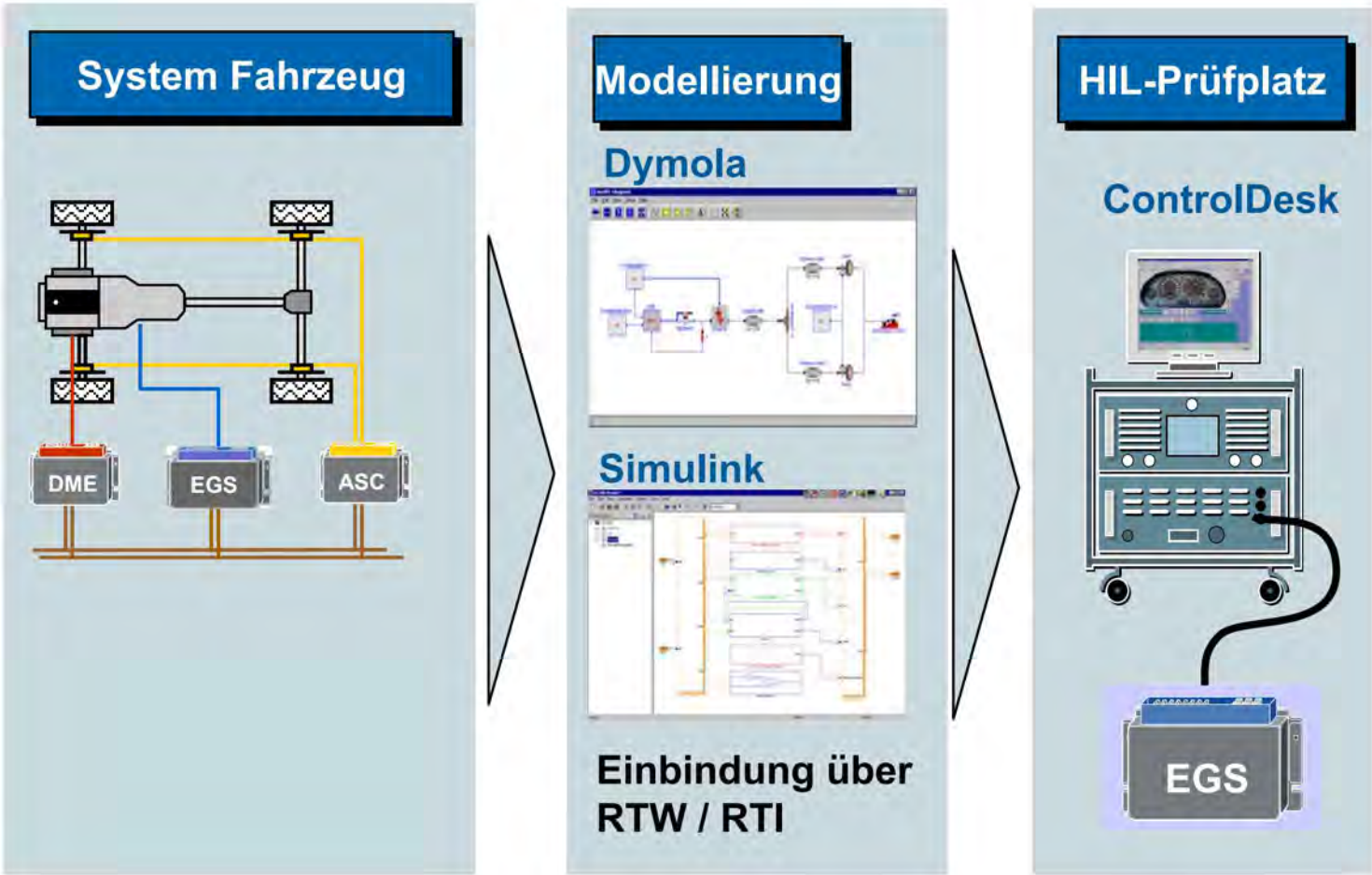
Reibungsmodellierung: Zusammenfassung

schlegel simulation

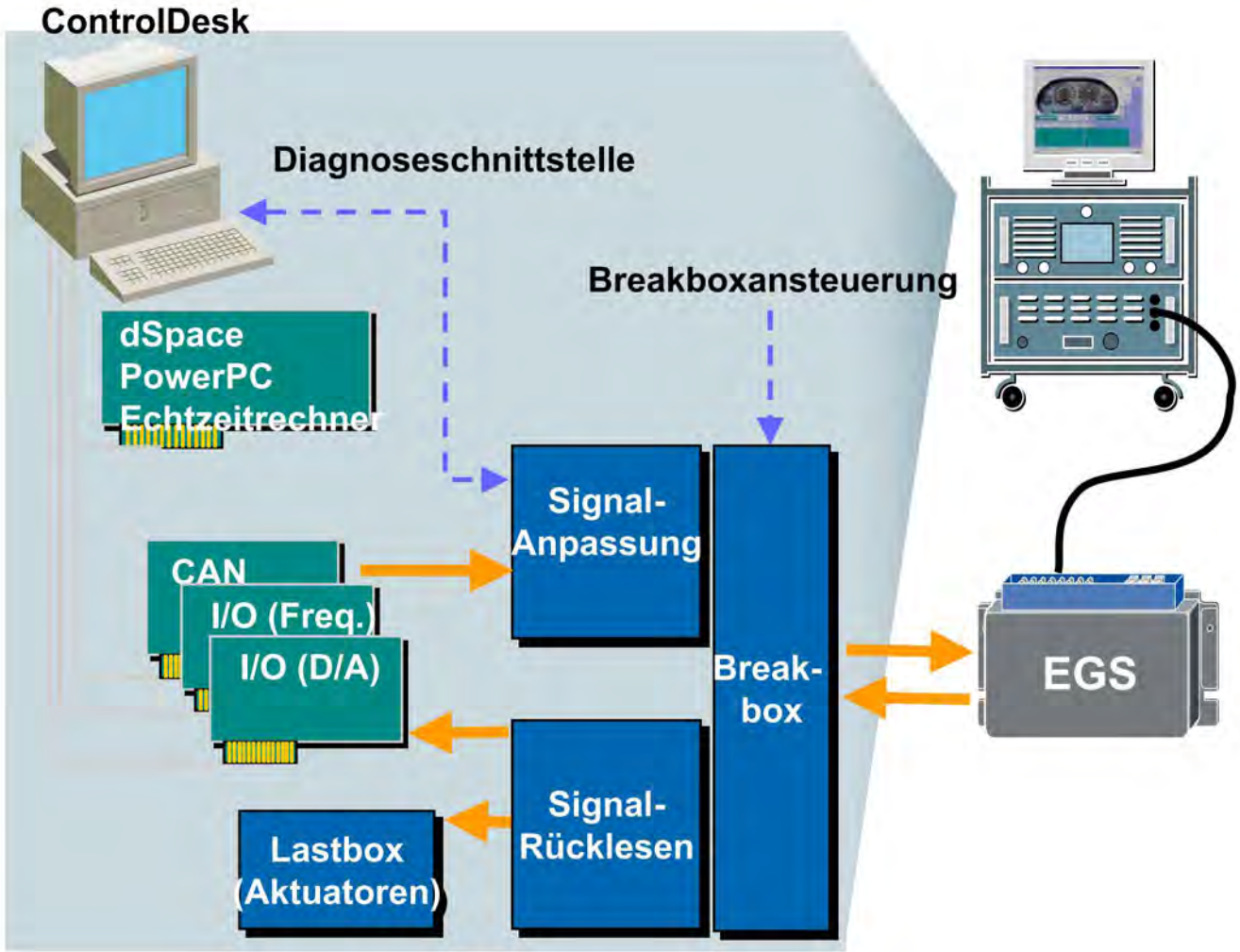
- Ein Zustandsautomat bestimmt den Zustand der Kupplung (Haften / Gleiten).
- Je nach Zustand wird die entsprechende Gleichung aktiv.
- Die dynamischen Kausalitätswechsel erlauben es, Komponenten wahlweise durch Kraft- oder Zwangsgesetze zu koppeln.
- Komponenten mit ähnlicher Problematik sind sehr häufig:
  - Masse mit Anschlag (z.B. Muffe / Schaltwelle mit Endanschlag, Synchronisierungsanschlag, etc.)
  - Freilauf

Modellierung & Simulation: Vorgehen

schlegel simulation

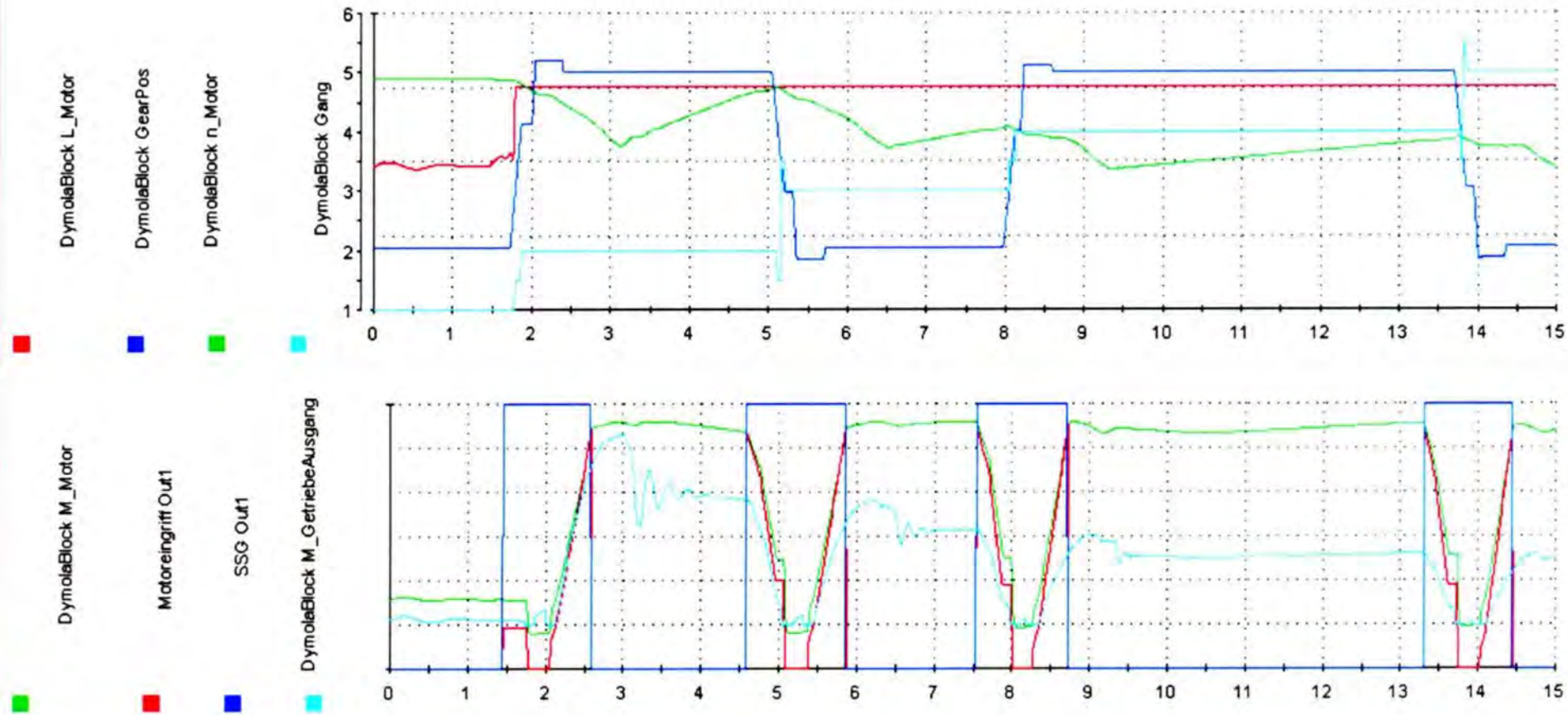


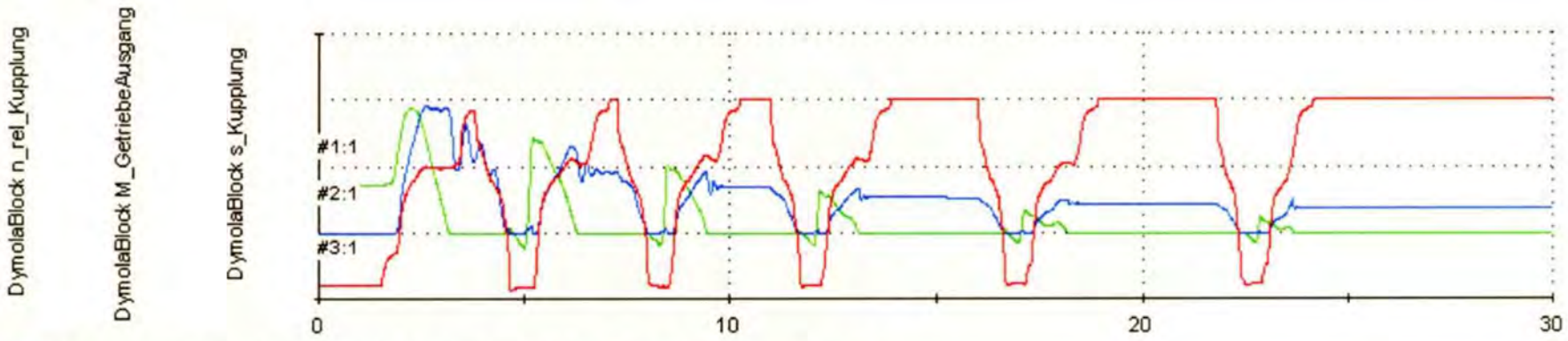
HIL Prüfstand



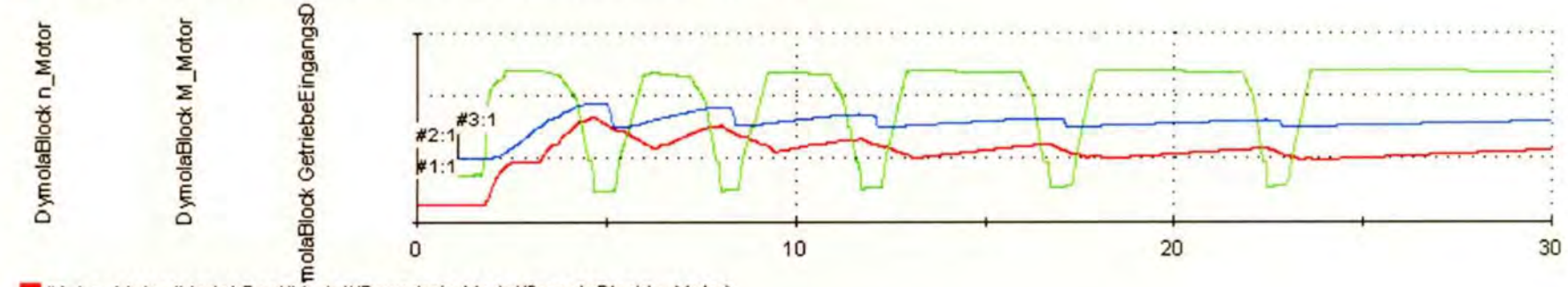
Simulationsergebnisse

schlegel simulation

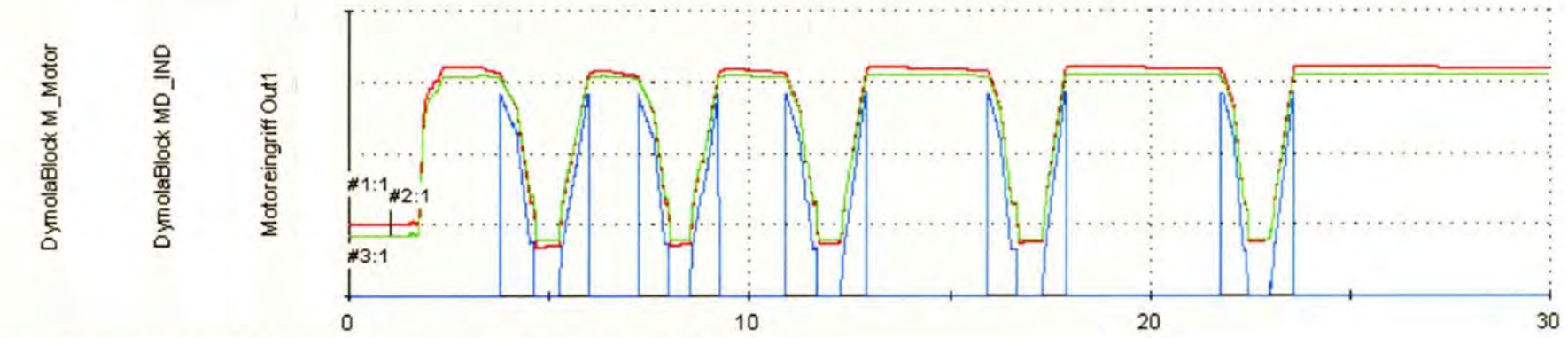




#1:1 n_rel_Kupplung (Model Root/Modell/Powertrain Model/DymolaBlock/n_rel_Kupplung)
#2:1 M_GetriebeAusgang (Model Root/Modell/Powertrain Model/DymolaBlock/M_GetriebeAusgang)
#3:1 s_Kupplung (Model Root/Modell/Powertrain Model/DymolaBlock/s_Kupplung)



#1:1 n_Motor (Model Root/Modell/Powertrain Model/DymolaBlock/n_Motor)
#2:1 M_Motor (Model Root/Modell/Powertrain Model/DymolaBlock/M_Motor)
#3:1 GetriebeEingangsDrehzahl (Model Root/Modell/Powertrain Model/DymolaBlock/GetriebeEingangsDrehzahl)



#1:1 M_Motor (Model Root/Modell/Powertrain Model/DymolaBlock/M_Motor)
#2:1 MD_IND (Model Root/Modell/Powertrain Model/DymolaBlock/MD_IND)
#3:1 Out1 (Model Root/Modell/Powertrain Model/DymolaBlock/Motoreingriff/Out1)

Zusammenfassung

schlegel simulation

- Die akausale Modellierung mit Dymola ermöglicht echtzeitfähige Simulationsmodelle, die
  - zu 100 % wiederverwendbar sind und
  - dynamische Kausalitätswechsel erlauben

- Einsatz des vorgestellten Modells zur Vorapplikation von Steuergeräten (u.a. „Einlernen“)

Kontakt

Schlegel Simulation GmbH
Meichelbeckstr. 8b
D-85356 Freising
Telefon 08161 / 4943960
Telefax 08161 / 4943962
info@schlegel-simulation.de

SMASH

single kernel

mixed signal multi-level multi language simulator

Aktuelle Version 5.6.2

download: http://www.disa.dolphin.fr/medal/smash/smash_download.html

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Betriebssysteme

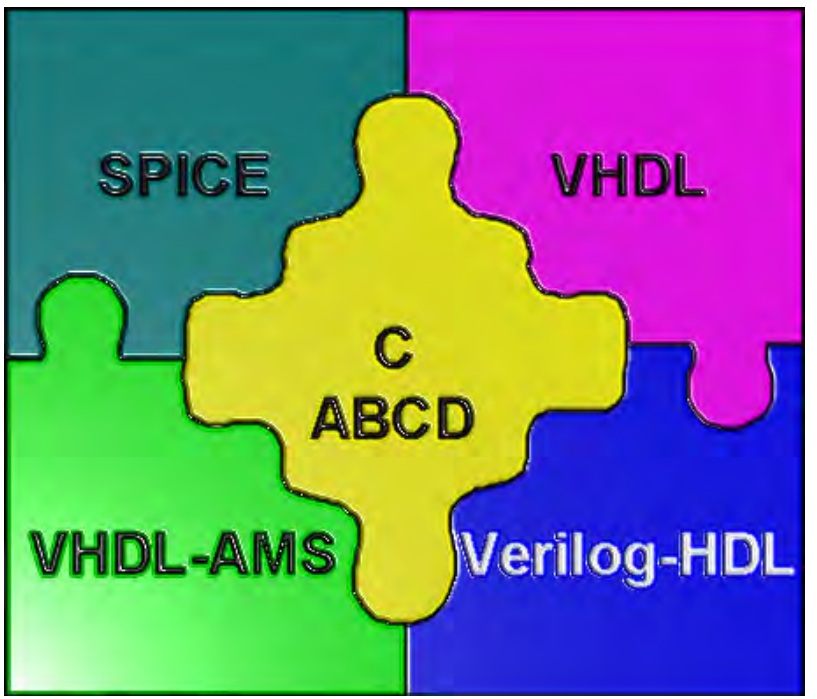
MS-WINDOWS 9x/NT/2000/XP
SUN-SOLARIS 7
LINUX
97 % des SMASH™ C/C++ Codes
ist auf allen Plattformen gleich

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Unterstützte Hardware-Beschreibungssprachen

- analog:
  - ABCD & C
  - SPICE
  - VHDL-AMS
  - Verilog-AMS (bald)
- logic:
  - SPICE (relaxation)
  - Verilog-HDL
  - VHDL
  - C

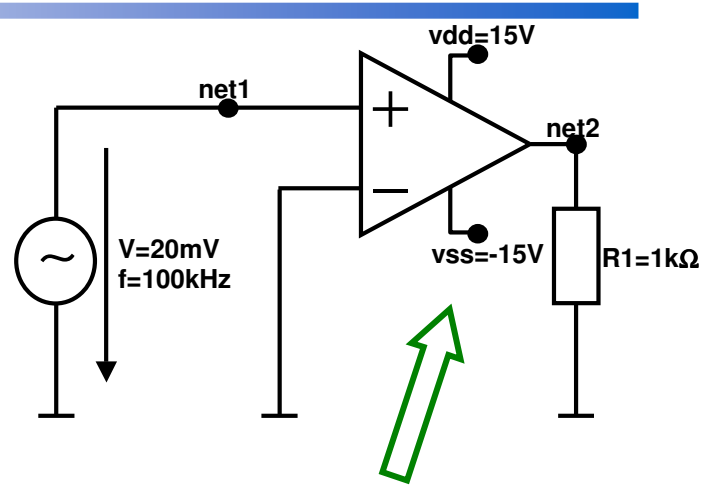


Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Anwendungsbeispiel Operationsverstärker

```
1.  ENTITY OpAmp IS
2.    GENERIC (vdd  : voltage :=  15.0;
3.             vss  : voltage := -15.0;
4.             gain : REAL);
5.    PORT (TERMINAL in_p, in_n: ELECTRICAL;
6.          TERMINAL output: ELECTRICAL);
7.  END ENTITY OpAmp;
8.
9.  ARCHITECTURE simple OF OpAmp IS
10.   QUANTITY vin ACROSS in_p TO in_n;
11.   QUANTITY vout ACROSS iout THROUGH output;
12. BEGIN
13.     IF vin'ABOVE(vdd/gain) USE
14.       vout == vdd;
15.     ELSIF NOT vin'ABOVE(vss/gain) USE
16.       vout == vss;
17.     ELSE
18.       vout == vin*gain TOLERANCE "MyTol";
19.     END USE;
20.
21.     BREAK ON vin'ABOVE(vdd/gain), vin'ABOVE(vss/gain);
22. END ARCHITECTURE simple;
```



Attribut ‘ABOVE in Verbindung mit BREAK Statement zeigt dem Simulator Unstetigkeiten!

VHDL-Syntax

VHDL-AMS-Syntax

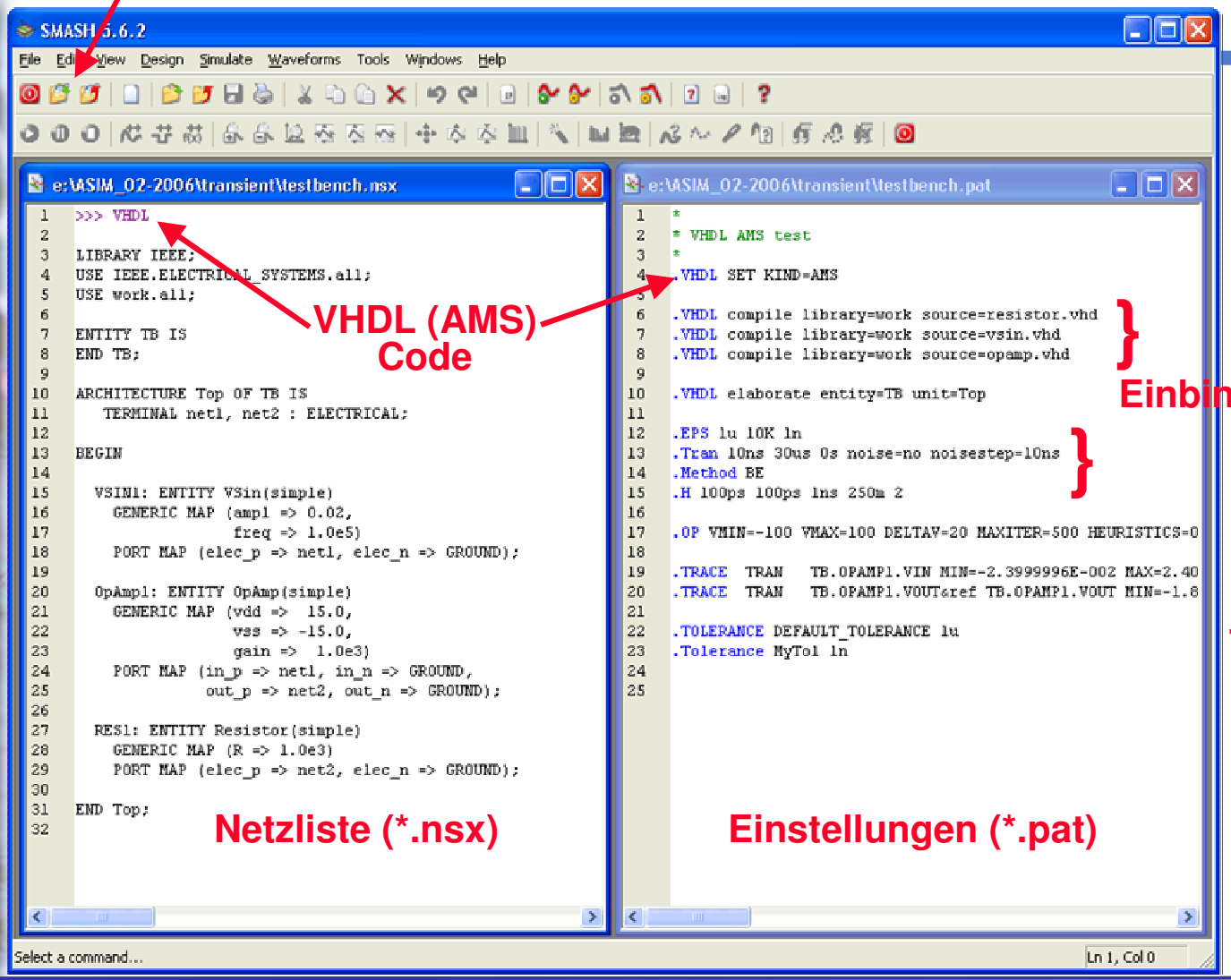
Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Vorbereitung

Open Circuit Ctrl-E

*.nsx + *.pat



VHDL (AMS) Code

Netzliste (*.nsx)

Einstellungen (*.pat)

VHDL-AMS Modelle Einbindung Testbench

Simulations-Einstellungen

AP-Einstellungen

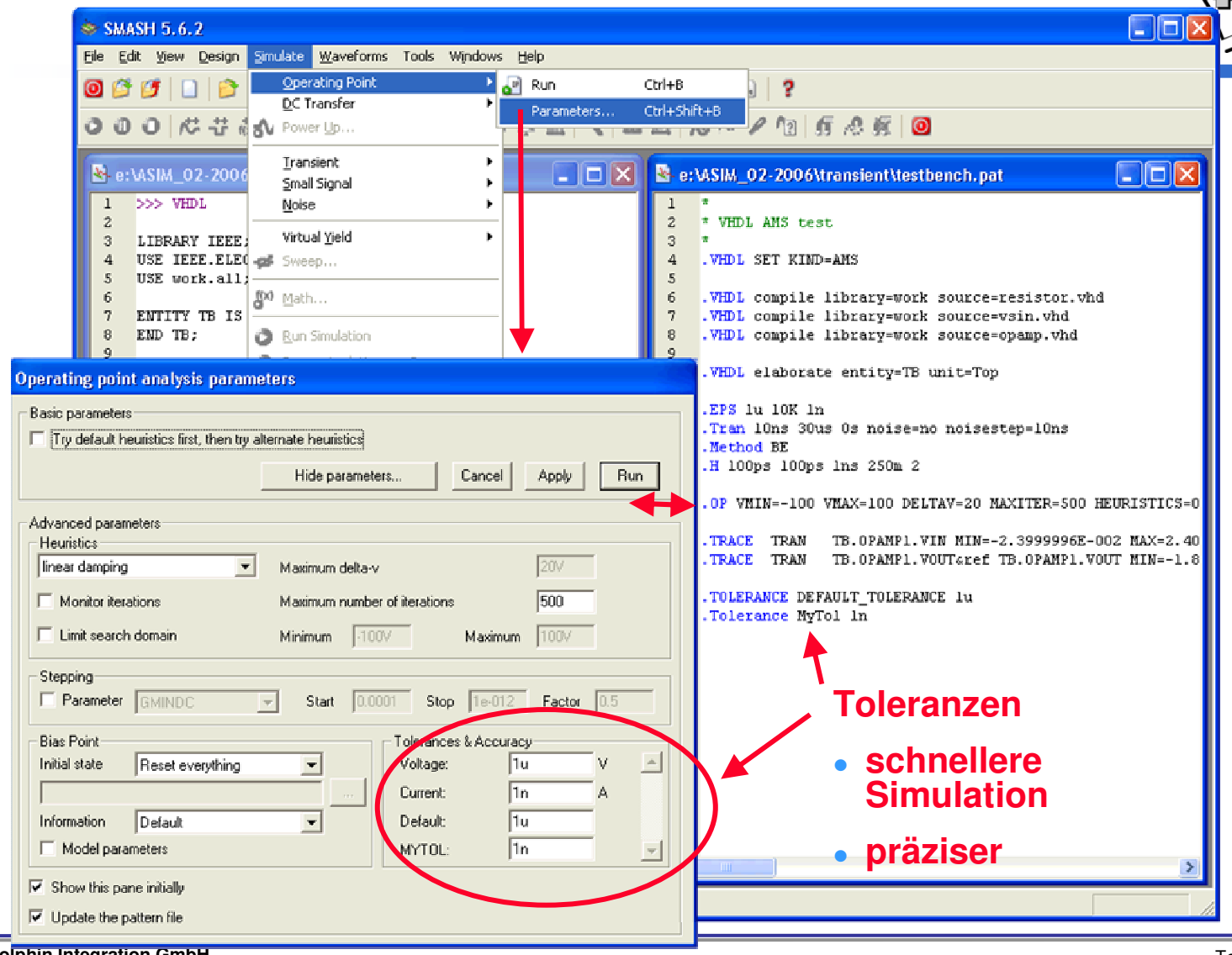
Signale

Toleranzangaben

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

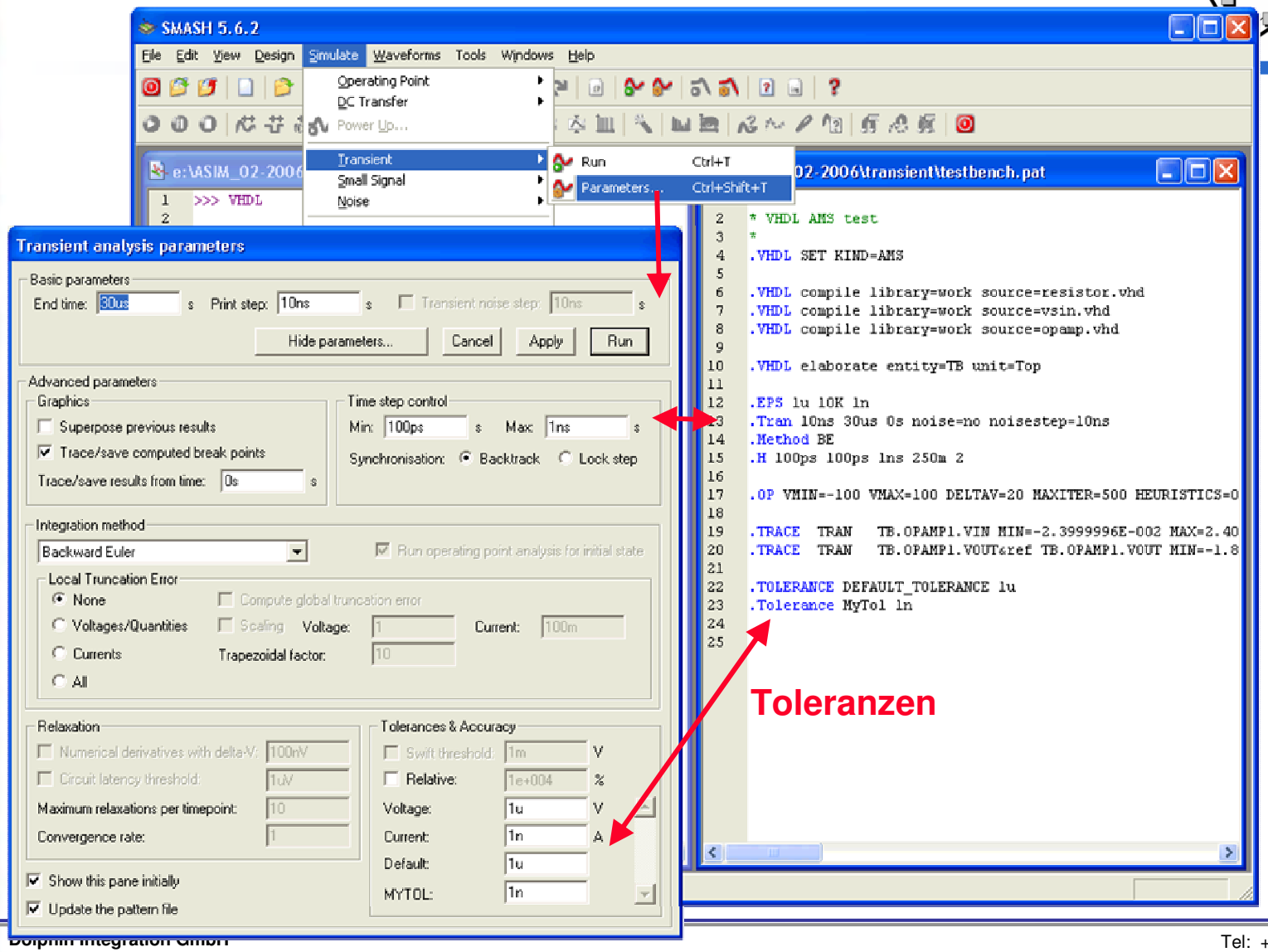
Arbeitspunktanalyse



Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

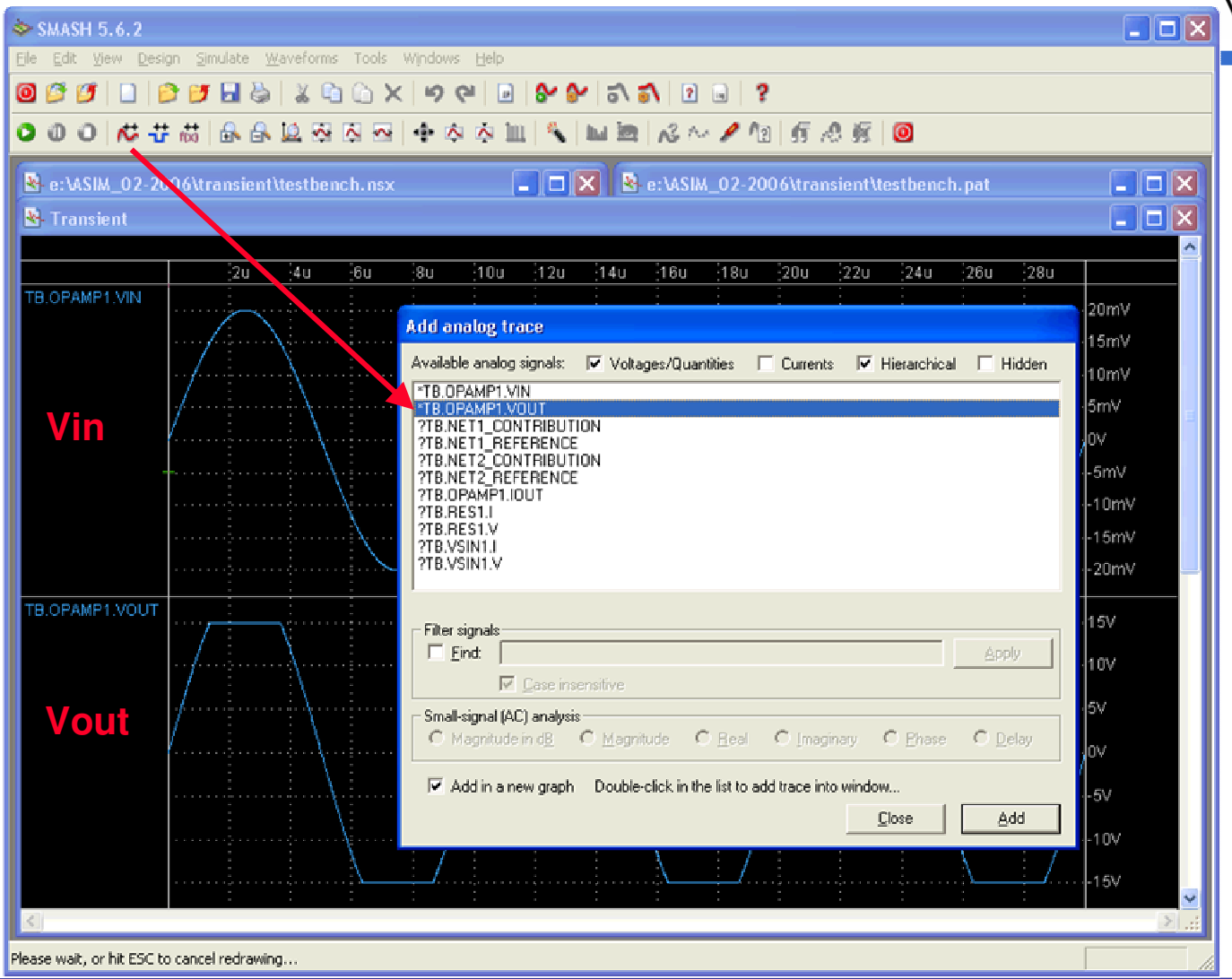
Transiente Analyse



Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Simulation

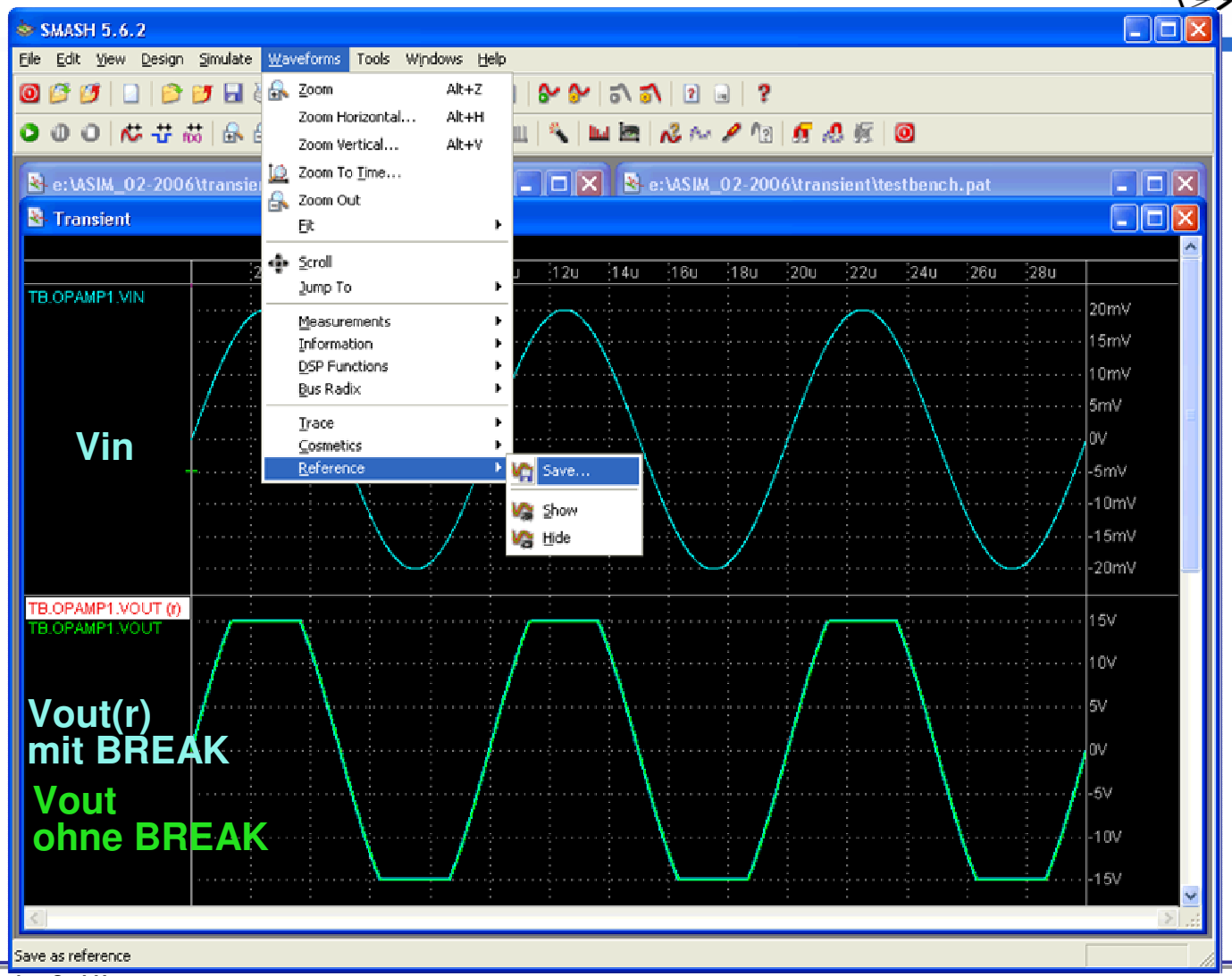


Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Simulation speichern/laden

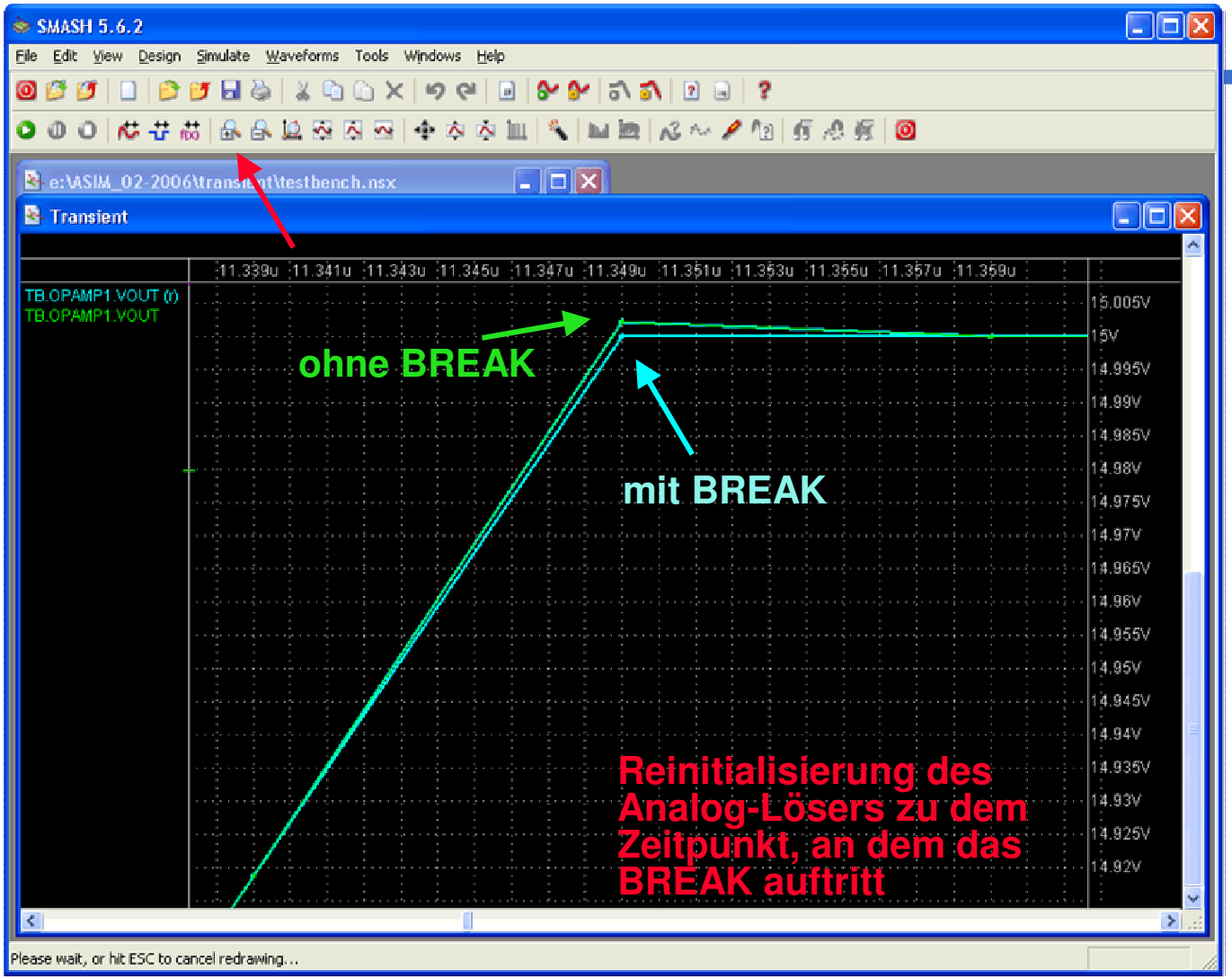
testbench.tmf & testbench.ref.tmf



Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Zoom



Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

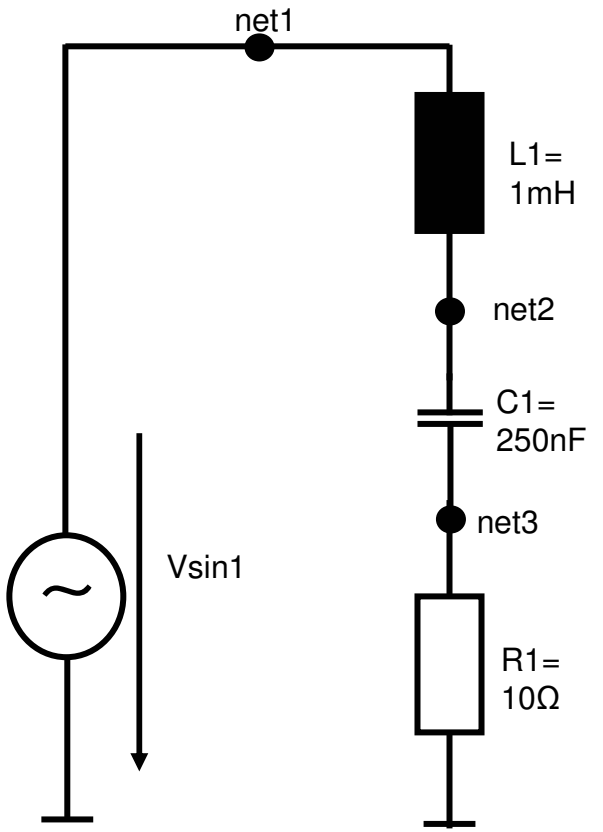
Modellierung für Small Signal Analyse

- Resonator
  - Resonance frequency f_r= 10kHz
  - Quality factor Q=6,3
  - Bandwidth B=1600

$$f_r = \frac{1}{2\pi\sqrt{LC}} \qquad B = \frac{R}{2\pi L} \qquad Q = \frac{f_r}{B} = \frac{1}{R}\sqrt{\frac{L}{C}}$$

```
ARCHITECTURE simple OF VSin IS
  QUANTITY v ACROSS i THROUGH elec_p TO elec_n;
  QUANTITY Vac: voltage SPECTRUM 1.0, 0.0;

BEGIN

  v == ampl * sin(MATH_2_PI * freq * NOW + phi0) + Vac;

END ARCHITECTURE simple;
```

Spectrum Quantity



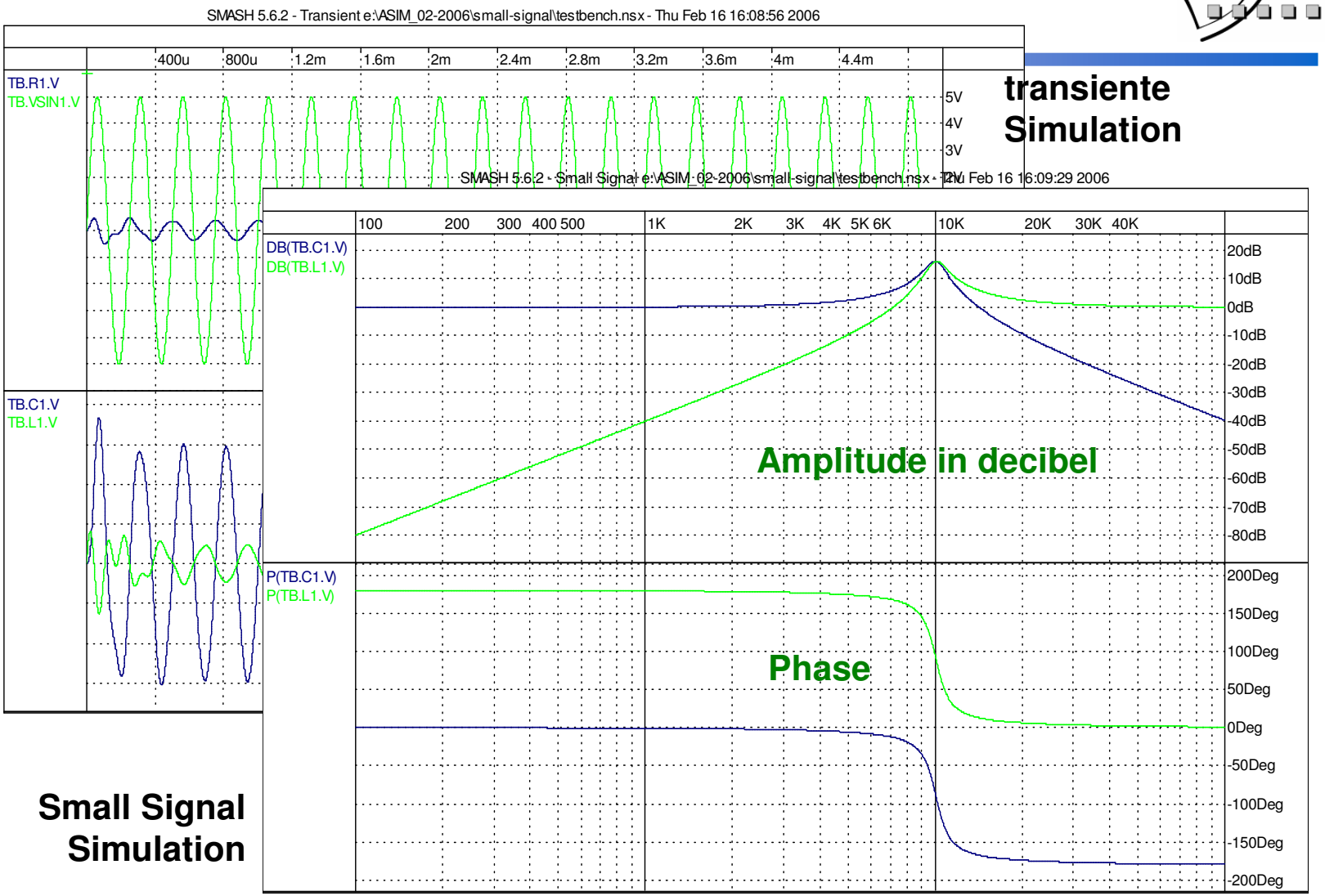
VHDL-Syntax

VHDL-AMS-Syntax

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Small Signal Simulation



Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Batch Mode Simulation

smash --batch --transient testbench.nsx

```
F:\Anwendungen\Dolphin\Smash562\bin\smash --batch --transient testbench.nsx
------------------------------------------------------------------------
- SMASH (TM) - Copyright (c) by Dolphin Integration, 1992-2006. -
------------------------------------------------------------------------
  Release 5.6.2 of Feb  3 2006

  Command line: F:\Anwendungen\Dolphin\Smash562\bin\smash --batch --transient te
stbench.nsx
------------------------------------------------------------------------

Cataloguing 'F:\Anwendungen\Dolphin\Smash562\lib\mingw'...
Cataloguing 'F:\Anwendungen\Dolphin\Smash562\lib\mingw' done.
Loading circuit e:\ASIM_02-2006\transient\testbench.nsx
Processing top-level
Library WORK parsing 1 / 4 (resistor.vhd)...
Library WORK parsing 2 / 4 (vsin.vhd)...
Library WORK parsing 3 / 4 (opamp.vhd)...
Library WORK parsing 4 / 4 (e:\ASIM_02-2006\transient\testbench-smash\testbench.
Library WORK generating 1 / 8 (OPAMP)...
Library WORK generating 2 / 8 (RESISTOR)...
Library WORK generating 3 / 8 (TB)...
Library WORK generating 4 / 8 (VSIN)...
Library WORK generating 5 / 8 (SIMPLE/OPAMP)...
Library WORK generating 6 / 8 (SIMPLE/RESISTOR)...
Library WORK generating 7 / 8 (SIMPLE/VSIN)...
Library WORK generating 8 / 8 (TOP/TB)...
Library WORK building 1 / 8 (work_opamp)...
Library WORK building 2 / 8 (work_resistor)...
Library WORK building 3 / 8 (work_tb)...
Library WORK building 4 / 8 (work_vsin)...
Library WORK building 5 / 8 (work_opamp_simple)...
Library WORK building 6 / 8 (work_resistor_simple)...
Library WORK building 7 / 8 (work_vsin_simple)...
Library WORK building 8 / 8 (work_tb_top)...
Cataloguing 'e:\ASIM_02-2006\transient\testbench-smash'...
Cataloguing 'e:\ASIM_02-2006\transient\testbench-smash' done.
Elaborating circuit...
Elaborating circuit (top-level)...
Elaborating circuit (creating devices)...
Elaborating circuit (elaborating devices)...
Elaborating circuit (VHDL)...
Elaborating library WORK entity TB unit TOP at top level...
Initializing...
Linking analog resolution matrix...
Analog elaboration of library WORK entity TB unit TOP at top level...
Load completed.
Operating-Point analysis (trying linear damping heuristics)...
Initializing library WORK entity TB unit TOP at top level...
Initializing library WORK entity TB unit TOP at top level...
Initializing library WORK entity TB unit TOP at top level...
Initializing library WORK entity TB unit TOP at top level...
Initializing library WORK entity TB unit TOP at top level...


Operating point analysis completed (converged).
Transient
Transient:  1 % done; Time to End: 0.0s; Tp:   151; It:   302; Sb:     0; Lai:
Transient:  2 % done; Time to End: 0.0s; Tp:   450; It:   900; Sb:     0; Lai:
 ...
Transient: 100 % done; Time to End: 0.0s; Tp: 29851; It: 59702; Sb:     0; Lai:
 2; Gai:  2

Transient analysis completed.
Unloading library WORK entity TB unit TOP at top level...
- Type any key to continue.
```

Simulation fertig

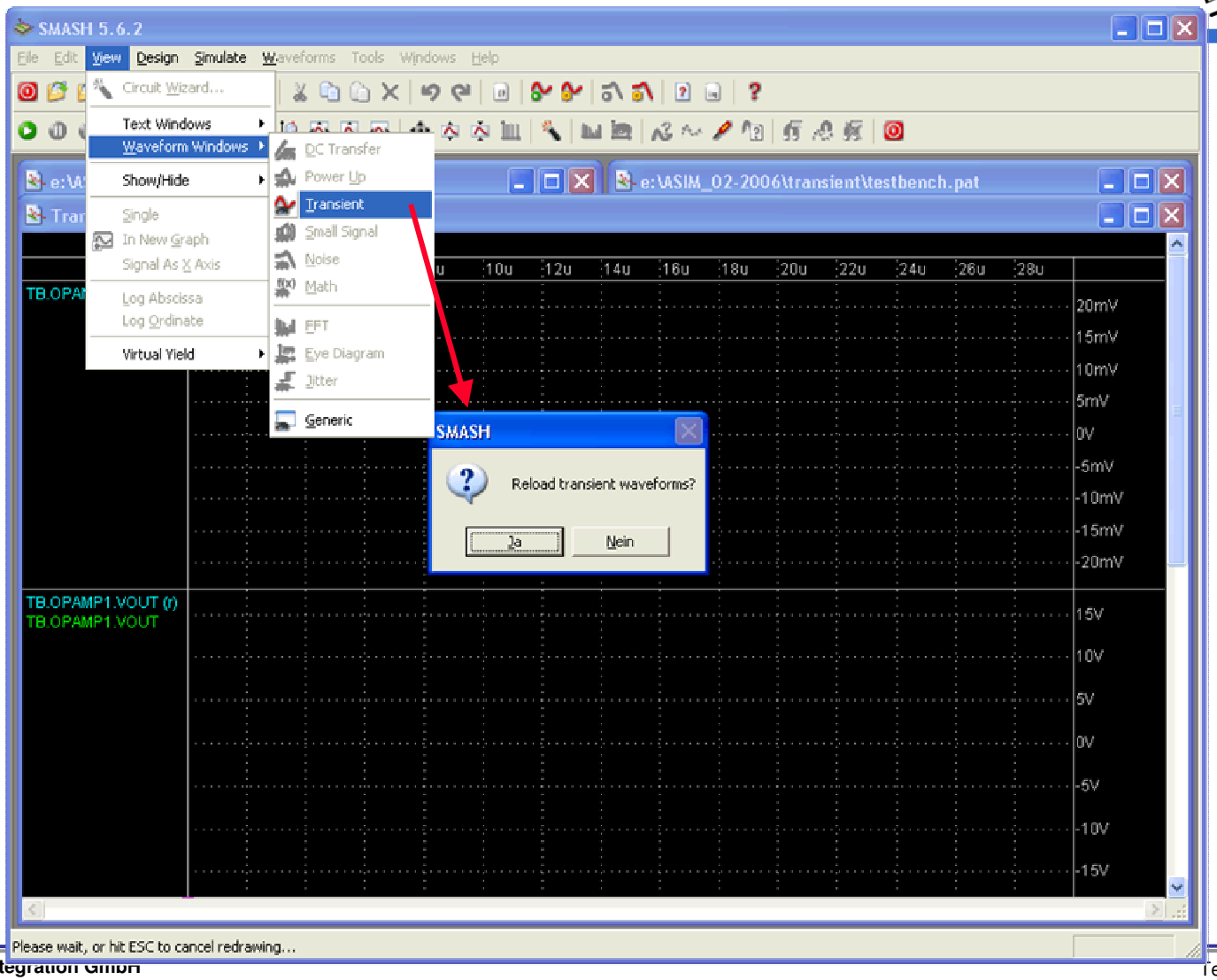
Ergebnisse in

testbench.tmf

optional

--no-wait

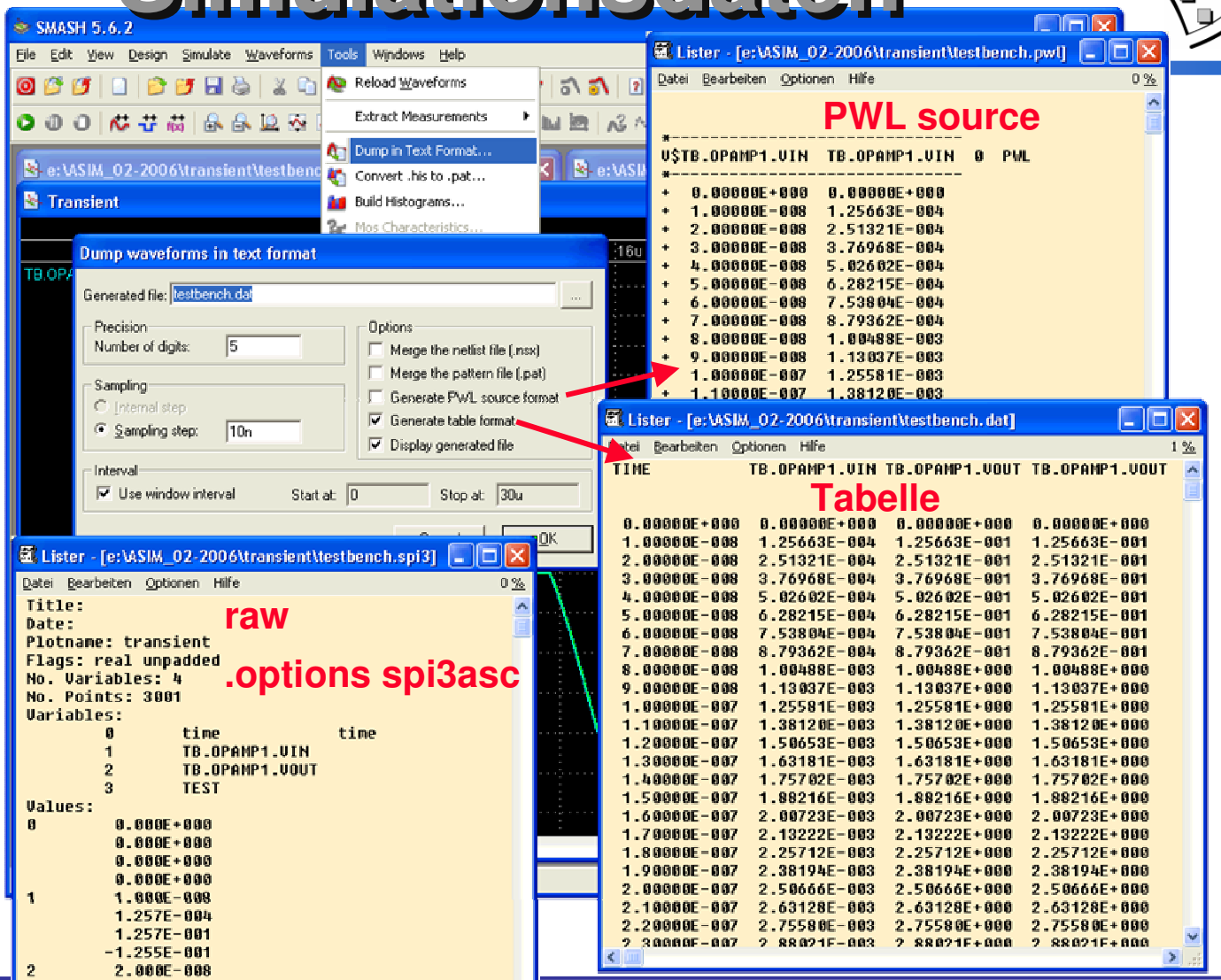
Batch Mode
Simulation ansehen



Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Export der Simulationsdaten



Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

LabVIEW

Simulation

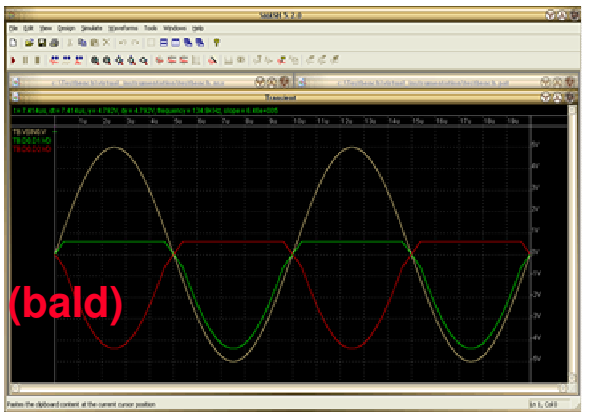
Prototyp

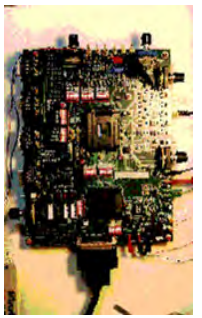
SPICE

ABCD

VHDL-AMS

Verilog-AMS (bald)



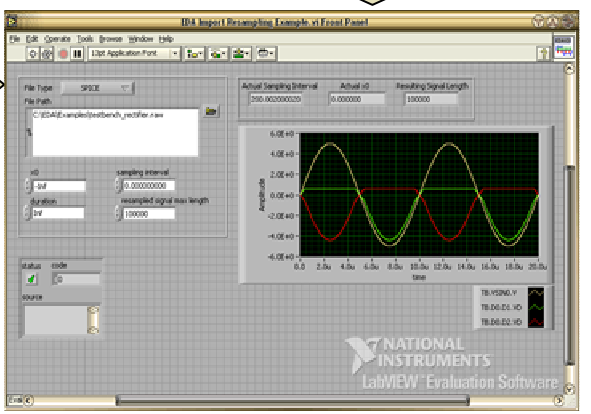


Datenerfassung

Transfer der Meßdaten

.raw

Transfer der Simulationsdaten über NIs SPICE VIs



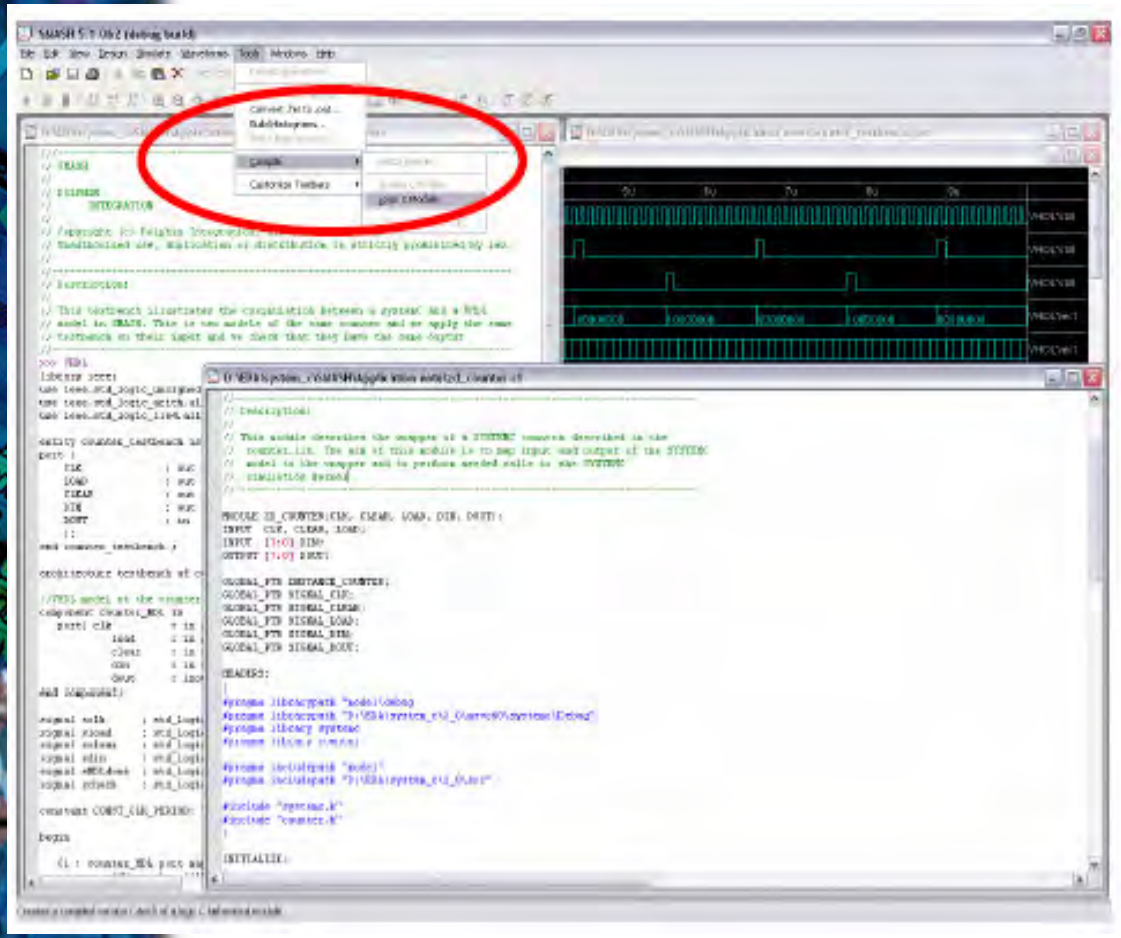
Verarbeitung der Simulations- und Meßdaten

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

SystemC

SystemC Tutorial: \SMASH\examples\Multi-Level\SystemC\

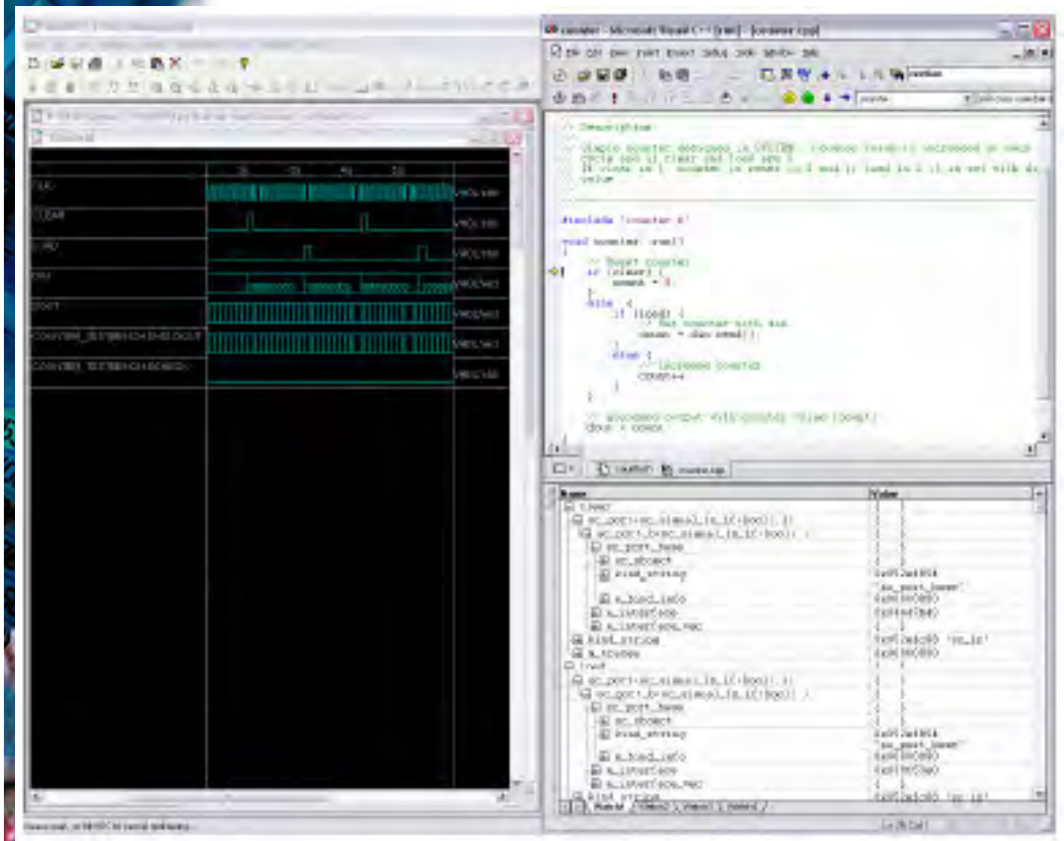


- **Einbindung logischer C Module in SMASH**
- **SystemC-AMS bald**
- **Unterstützt wird MS Visual C++**

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

SystemC

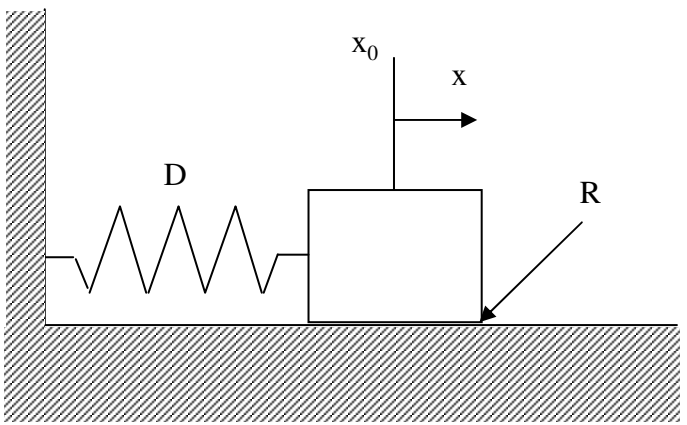


- Debuggen von SystemC Modellen während der Kosimulation mit SMASH
- Das SystemC Modell wird durch einen VHDL Prüfaufbau angeregt und in SMASH simuliert
- MS Visual C++ bringt die die Umgebung zum Debuggen des SystemC Modells

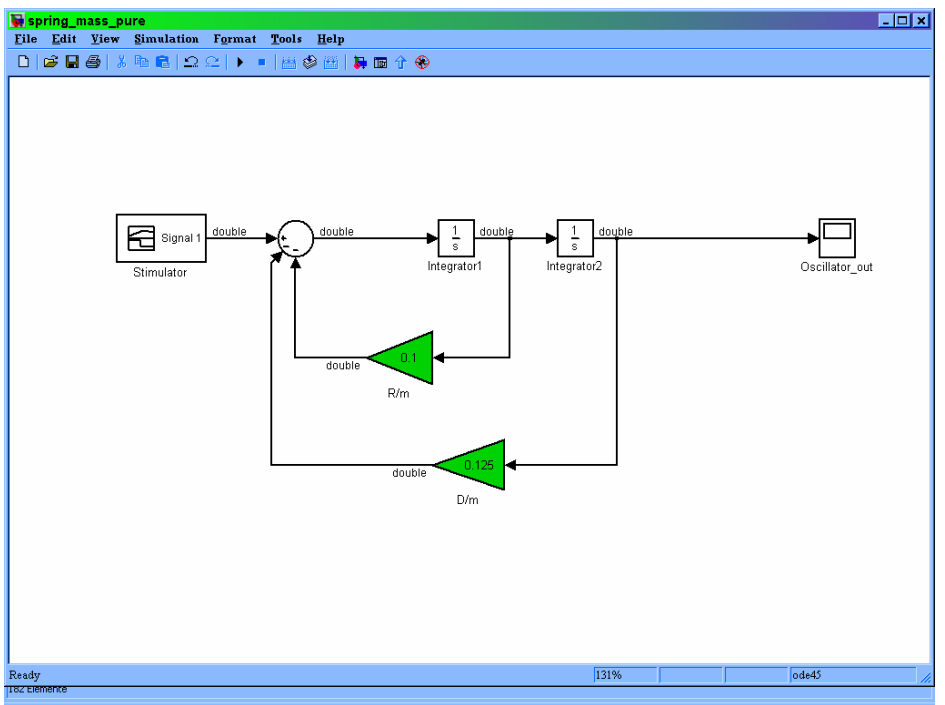
Simulink

Beispiel: \Smash\examples\Interfaces\Simulink\spring_mass

Beispiel: Feder-Masse-Dämpfer System mit Nulldurchgang-Zähler



- **einfache Modellierung der DGL in Simulink**



System DGL

$$\ddot{x} - \frac{R}{m} \cdot \dot{x} - \frac{D}{m} \cdot x = 0$$

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

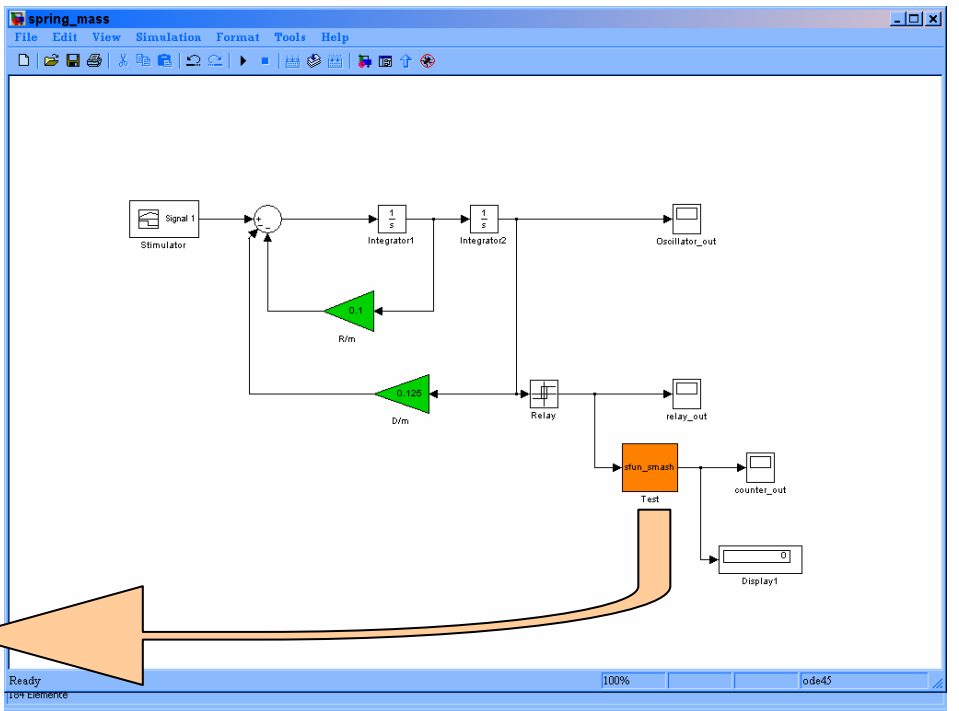
Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Simulink

Beispiel: \Smash\examples\Interfaces\Simulink\spring_mass

Das VHDL-Modell des Nulldurchgangszählers wird als s-Function im Simulink Modell instanziiert:

```
 1. entity counter is
 2.  port(toggle :  in real;
 3.        count : out real);
 4. end counter;

 5. architecture drive_counter of
counter is
 6.  begin
 7.   plus_one:process(toggle)
 8.     variable value : real := 0.0;
 9.   begin
10.    if toggle = 1.0 then
11.      value := value + 1.0;
12.    else value := value;
13.    end if;
14.    count <= value;
15.   end process plus_one;
16. end architecture drive_counter;
```

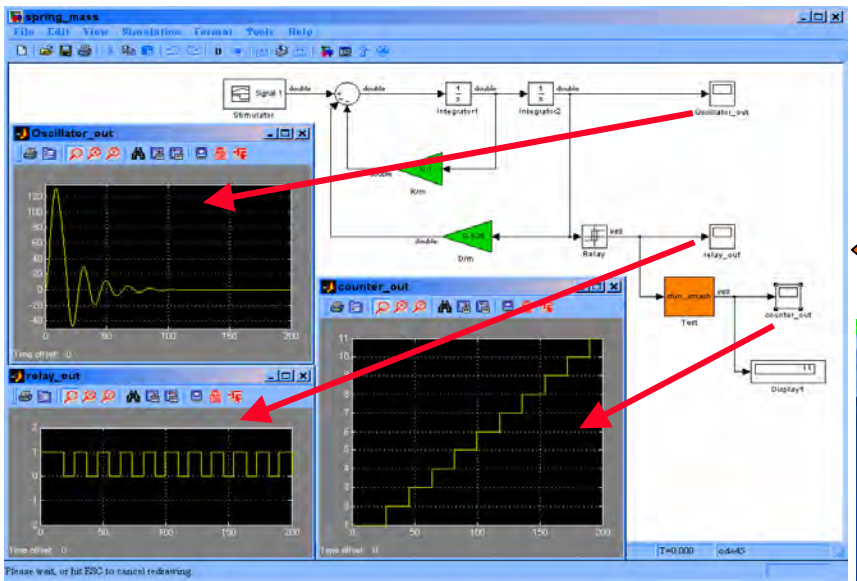


Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

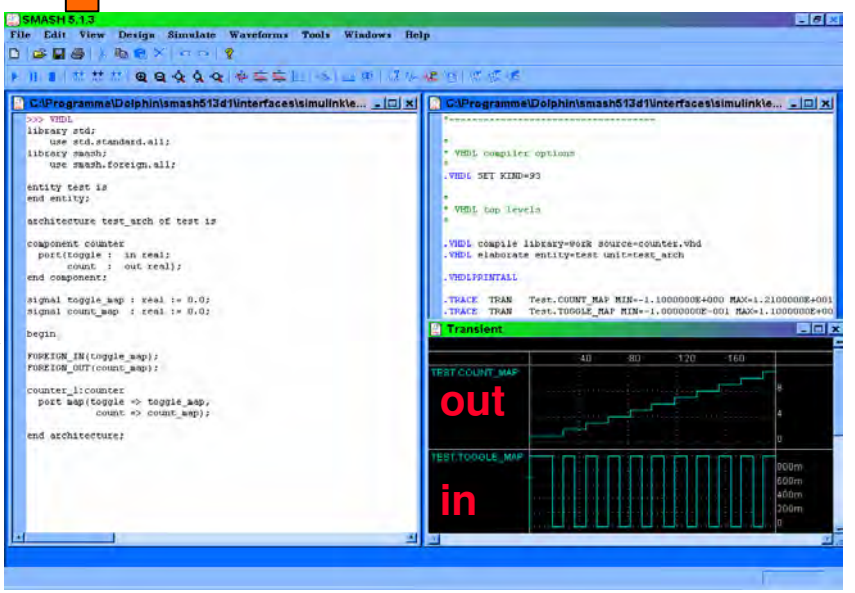
Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Simulink

Beispiel: \Smash\examples\Interfaces\Simulink\spring_mass



Simulink agiert als Master Simulator und SMASH als Slave



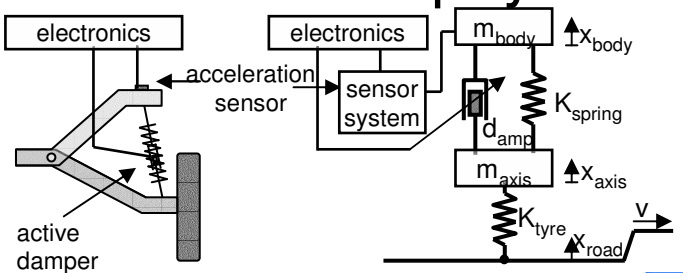
Beide Simulatoren haben Zugriff auf das VHDL-Zähler-Modell und dessen Daten

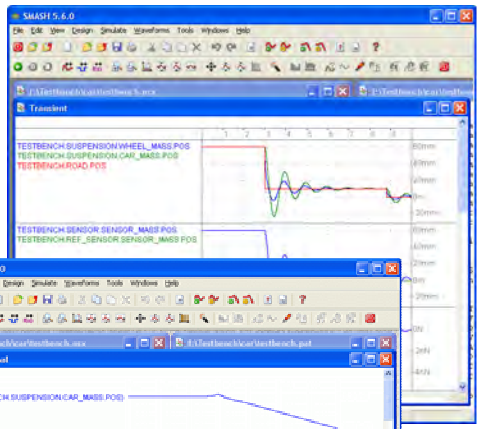
Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

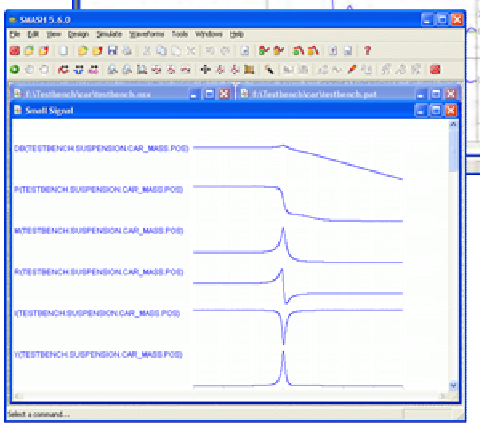
Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

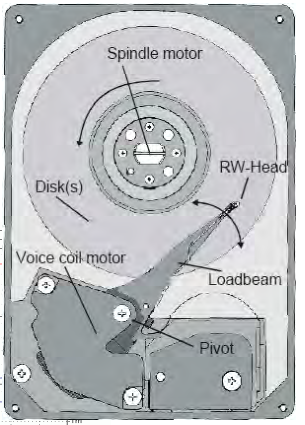
VHDL-AMS Modellbibliotheken

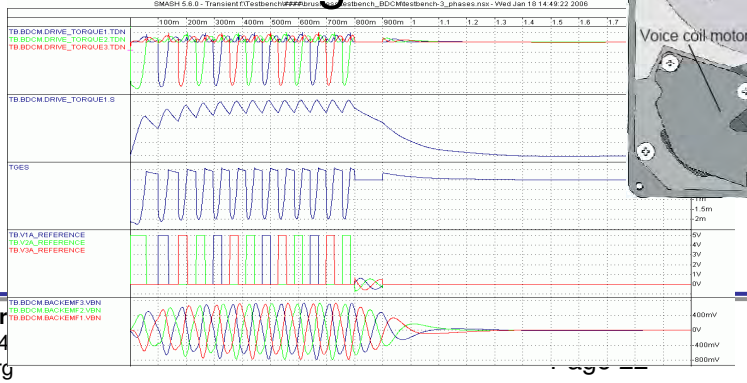
- EMBLEM – Modellbibliotheken physikalischer Effekte
  - Mecha
    - elektrisch
    - mechanisch
    - elektromechanisch
  - Drive
    - Elektromotor
    - Getriebe
    - Verbindungselemente











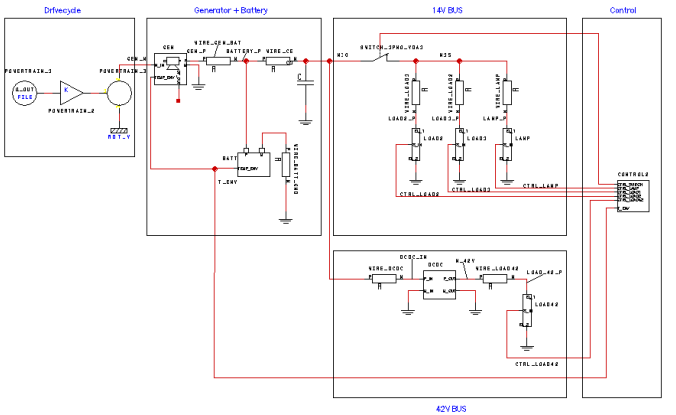
Dolphin Integr
Bismarckstr. 14
47057 Duisburg

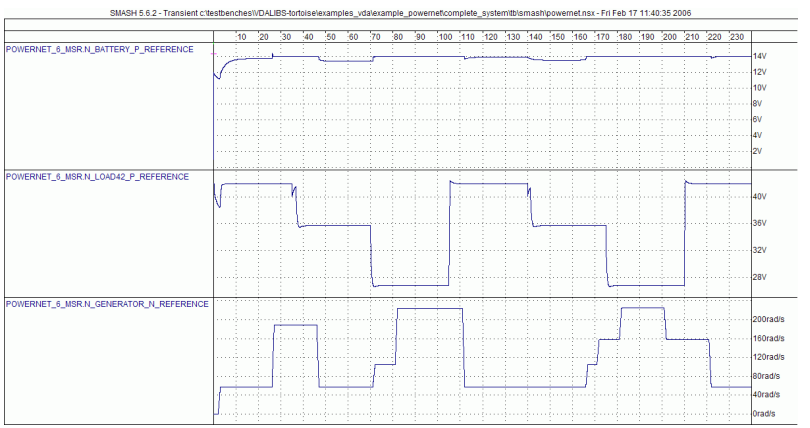
Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

VHDL-AMS Modellbibliotheken

- VDA FAT-AK30
  - Verband der Automobilindustrie
  - Forschungsvereinigung Automobiltechnik
  - Simulation gemischter Systeme mit VHDL-AMS
- open source library
- kompatibel zu SMASH

http://fat-ak30.eas.iis.fraunhofer.de/





Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Vielen Dank
für Ihre
Aufmerksamkeit

Aktuelle Version 5.6.2

download: http://www.disa.dolphin.fr/medal/smash/smash_download.html

Dolphin Integration GmbH
Bismarckstr. 142a
47057 Duisburg

Tel: +49 203 3062250
Fax: +49 203 3062269
Email: mems@dolphin-integration.com

Treffen der ASIM/GI-Fachgruppen
„Simulation technischer Systeme“ und
„Grundlagen und Methoden in Modellbildung und Simulation“
München, 20. und 21. Februar 2006

Anwendung mehrdimensionaler Netzwerke in VHDL-AMS

Joachim Haase

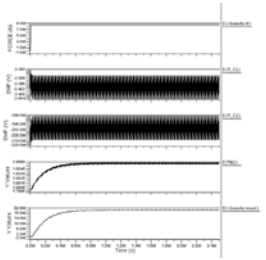
Fraunhofer-Institut für Integrierte Schaltungen
Außenstelle EAS Dresden

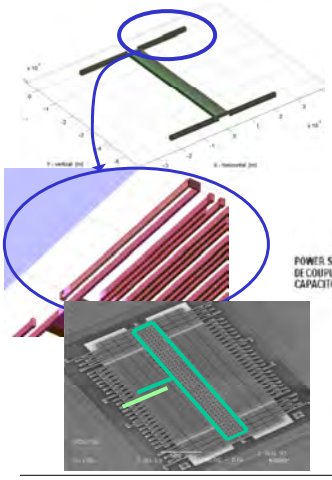
E-Mail: Joachim.Haase@eas.iis.fraunhofer.de

Inhalt

- Beschreibung mehrdimensionaler Netzwerke mit der Verhaltensbeschreibungssprache VHDL-AMS
- Frequenzgangberechnung für SC-Schaltungen mit mehrdimensionalen Netzwerken
- Simulation von RF-Schaltungen mit Basisbandmodellen
- Zusammenfassung

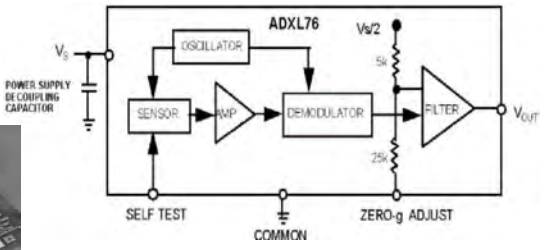
Beschleunigungsaufnehmer





Sensorgeometrie

- seismische Masse
- 2D-Modell für „Balkenfedern“



Sensor und Auswerteelektronik

Mehrdimensionale Netzwerke

- Deklaration von Knoten (Beispiel)
  terminal A, B : TRANSLATIONAL_VECTOR (1 **to** 2);

- Deklaration eines Zweiges

A ● ⟶ ● B

quantity V **across** I **through** A **to** B;

Kirchhoffsche Gesetze gelten komponentenweise

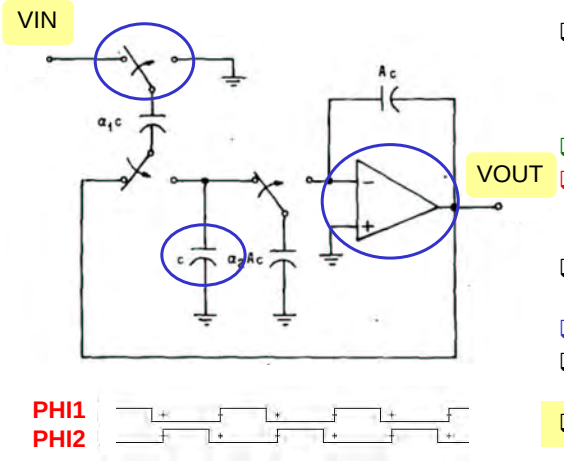
A(1) ● ⟶ ● B(1)

entspricht A(2) ● ⟶ ● B(2)

quantity V1 **across** I1 **through** A(1) **to** B(1);
quantity V2 **across** I2 **through** A(2) **to** B(2);

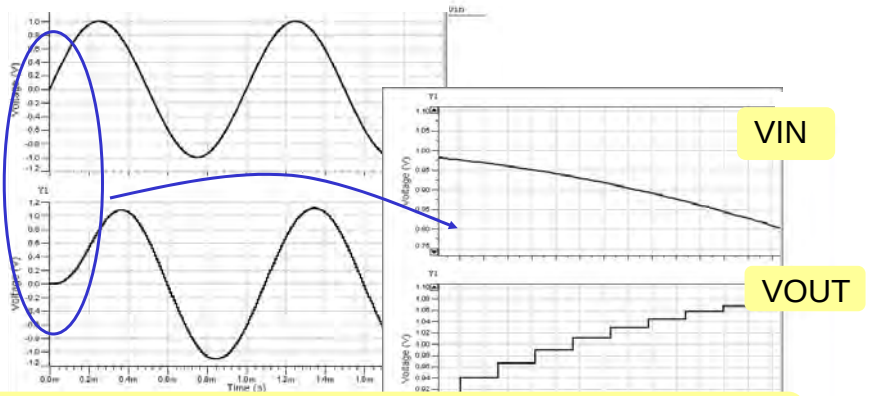
Frequenzgangberechnung für SC-Schaltungen mit mehrdimensionalen Netzwerken

Switched Capacitor (SC) Schaltungen



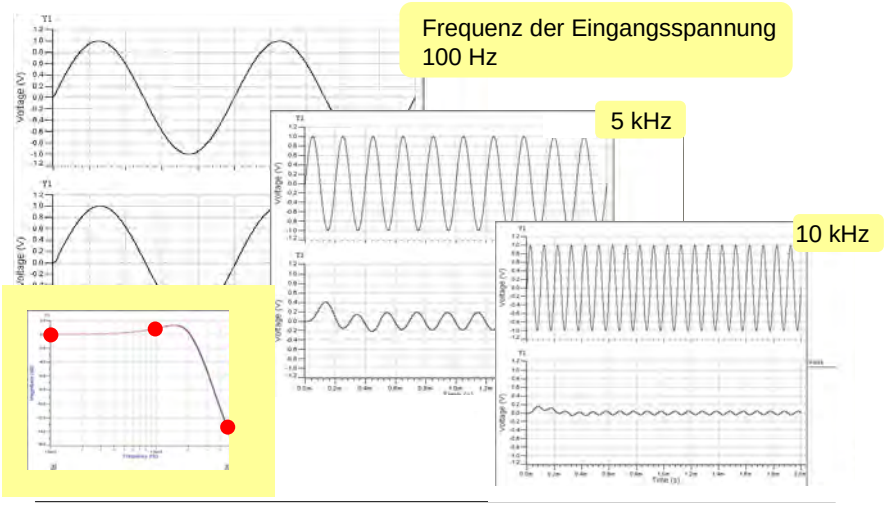
- Nachbilden des Verhaltens von Widerständen durch schnelles Umladen von Kapazitäten
- **Nichtüberlappende Takte**
- **Taktfrequenz sehr viel größer als Signalbandbreite**
- Lineare Spannungsübertragung
- **Integrierbarkeit**
- Einbeziehung in die Systemsimulation
- Frequenzgangberechnung

Simulation im Zeitbereich

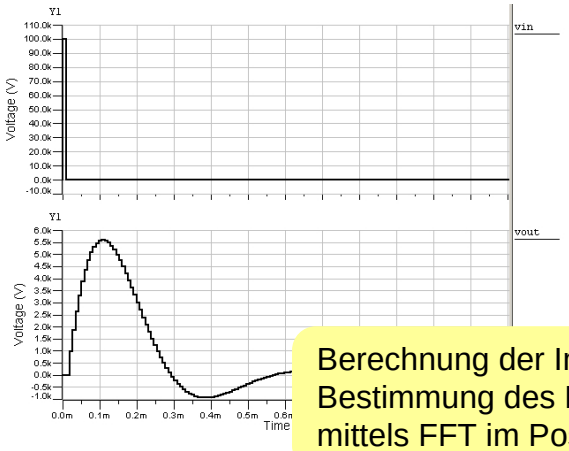


Spannungsänderungen bei Taktwechsel
keine direkte Anwendung der Kleinsignalanalyse möglich

Wiederholte Zeitbereichssimulationen



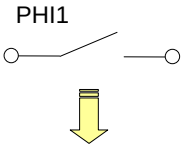
Berechnung der Impulsantwort

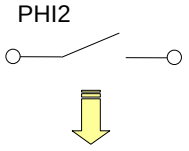


Berechnung der Impulsantwort
Bestimmung des Frequenzganges mittels FFT im Postprocessing

SC-Schaltermodelle für Frequenzgangberechnung

Schließen



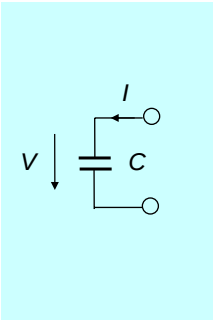


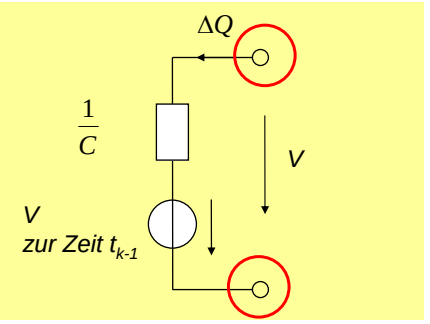
Phase PHI1

Phase PHI2

$$\begin{pmatrix} v_1 \\ \Delta q_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} \Delta q_1 \\ v_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

Ladungs-Spannungs-Modell im Zeitbereich



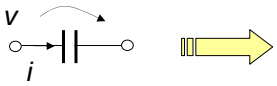


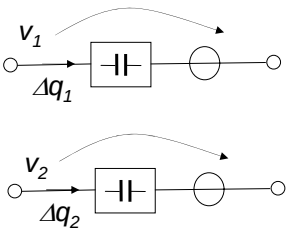
$$I = C \cdot \frac{dV}{dt}$$

$$\Delta Q = C \cdot \left(V - V_{t_{k-1}} \right)$$

$$\int_{t_{k-1}}^{t} I(\tau) d\tau = Q(t) - Q(t_{k-1})$$

SC-Kapazitätsmodell in VHDL-AMS (1)





$$i = C \cdot \frac{dv}{dt}$$

$$\begin{pmatrix} \Delta q_1 \\ \Delta q_2 \end{pmatrix} = \begin{pmatrix} C \cdot (v_1 - z^{-1} \cdot v_2) \\ C \cdot (v_2 - z^{-1} \cdot v_1) \end{pmatrix}$$

Verzögerung zwischen Ende der Taktphasen

SC-Kapazitätsmodell in VHDL-AMS (2)

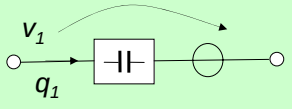
```
package SC_DECLARATIONS is

   nature SC_NATURE is
      REAL   across
      REAL   through
      SC_REF reference;

   nature SC_NATURE_VECTOR
            is array (NATURAL range <>) of SC_NATURE;

   -- clock period for PHI1 and PHI2
   constant  T : REAL;

end package SC_DECLARATIONS;
```


SC-Kapazitätsmodell in VHDL-AMS (3)

```
library SC_LIB;
use SC_LIB.SC_DECLARATIONS.all;
```

```
entity SC_CAP_2PHASE is
    generic (      CAP : REAL := 1.0);
    port (terminal P   : SC_NATURE_VECTOR (1 to 2);
          terminal N   : SC_NATURE_VECTOR (1 to 2));
end entity SC_CAP_2PHASE;
```

```
architecture A1 of SC_CAP_2PHASE is
    constant TSAMPL_2PHASE : REAL := SC_LIB.SC_DECLARATIONS.T/2.0;
    quantity V1 across Q1 through P(1) to N(1);
    quantity V2 across Q2 through P(2) to N(2);
    quantity V1_VAL,  V2_VAL  : REAL;
begin
    V1_VAL  == V1;
    V2_VAL  == V2;
    if DOMAIN = QUIESCENT_DOMAIN use
       Q1 == CAP*V1;
       Q2 == CAP*V2;
    else
       Q1 == CAP*(V1-V2_VAL'DELAYED(TSAMPL_2PHASE));
       Q2 == CAP*(V2-V1_VAL'DELAYED(TSAMPL_2PHASE));
    end use;
end architecture A1;
```



$$\begin{pmatrix}\Delta q_1\\ \Delta q_2\end{pmatrix} = \begin{pmatrix}C\cdot(v_1 - z^{-1}\cdot v_2)\\ C\cdot(v_2 - z^{-1}\cdot v_1)\end{pmatrix}$$

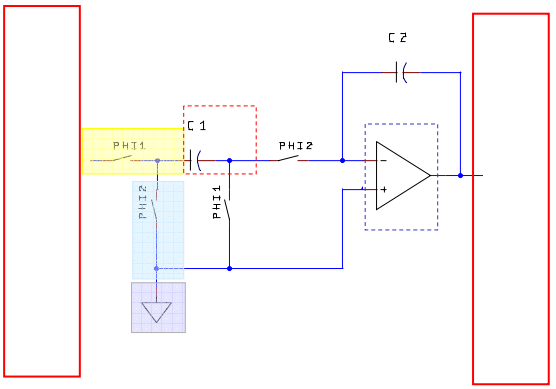
SC Modelle für Frequenzgangberechnung

Modell	Bemerkung
SC_CAP_2PHASE	Kapazitätsmodell
SC_SWITCH_2PHASE	Ideale Schalter (PHI1, PHI2 aktiv)
SC_SWITCH2_SC_REF_2PHASE	Ideale Schalter (verbunden mit SC_REF)
SC_TOGGLE_2PHASE	Idealer Wechselschalter
SC_OPAMP_2PHASE	Idealer Operationsverstärker
SC_VCONST_2PHASE	Konstantspannungsquelle
SC_VCONST2_SC_REF_2PHASE	Konstantspannungsquelle (an SC_REF)
SC_REF_2PHASE	Verbindung mit SC_REF
SC_INPUT_2PHASE	Konverter ELECTRICAL nach SC_NATURE
SC_OUTPUT_2PHASE	Konverter SC_NATURE nach ELECTRICAL

Beispiele

Nichtinvertierender SC Integrator (1)

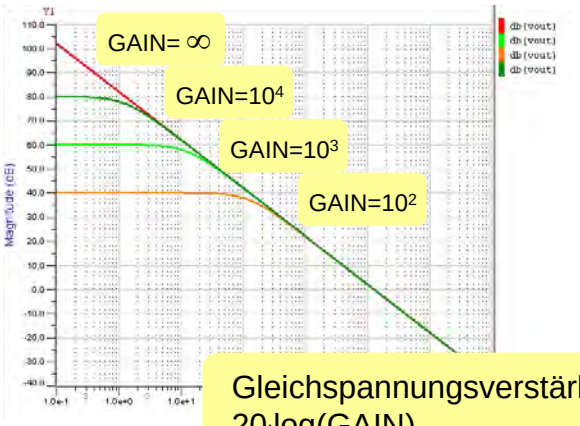
ELECTRICAL SC_NATURE ELECTRICAL



Nichtinvertierender SC Integrator (2)

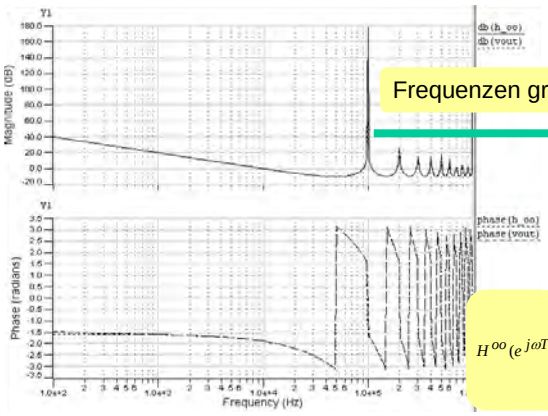
```
architecture arch_INTEGRATOR of INTEGRATOR is
   terminal N1, N2, N3, N4, NOUT : SC_NATURE_VECTOR(1 to 2);
   terminal SC_GROUND            : SC_NATURE_VECTOR(1 to 2);
   ...
begin
   S1: entity SC_LIB.SC_SWITCH_2PHASE(PHI1)
         port map (P => N1, N => N2);
   S2: entity SC_LIB.SC_SWITCH_2PHASE(PHI2)
         port map (P => N2, N => SC_GROUND );
   GR: entity SC_LIB.SC_REF_2PHASE(A1)
         port map ( P => SC_GROUND );
   C1: entity SC_LIB.SC_CAP_2PHASE(A1)
         generic map ( CAP => 0.6283)
         port map (P => N2, N => N3);
   ...
   OV1 : entity SC_LIB.SC_OPAMP_2PHASE
      generic map ( GAIN => REAL'HIGH )
      port map ( CP => SC_GROUND,  CN => N4, P => NOUT );
   ...
end architecture arch_INTEGRATOR;
```

Nichtinvertierender SC Integrator (3)



Gleichspannungsverstärkung in dB:
20·log(GAIN)

Vergleich mit analytischen Ergebnissen ohne Abtast- und Halteglied am Ausgang

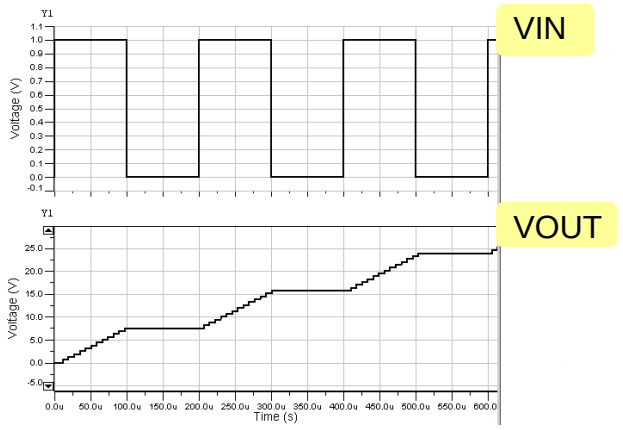


Frequenzen größer Taktfrequenz

$$H^{oo}(e^{j\omega T}) = \left(\frac{C_1}{j\omega T \cdot C_2}\right) \cdot \left(\frac{\omega \cdot \frac{T}{2}}{\sin\left(\omega \cdot \frac{T}{2}\right)}\right) \cdot e^{-j\omega \cdot \frac{T}{2}}$$

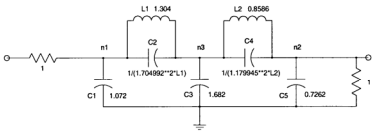
Allen, P. E.; Holberg, D. R..: CMOS Analog Circuit Design. Oxford University Press, 2002.

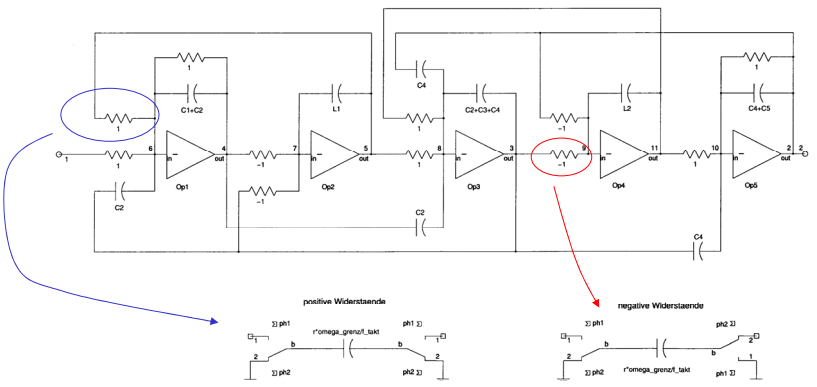
Zeitbereichssimulationsimulation



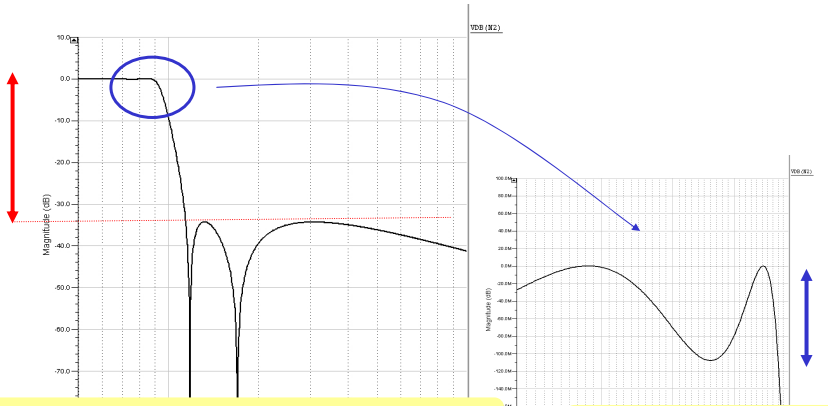
Cauer-Tiefpaß 5. Ordnung (1)

Cauer TP: C05/50





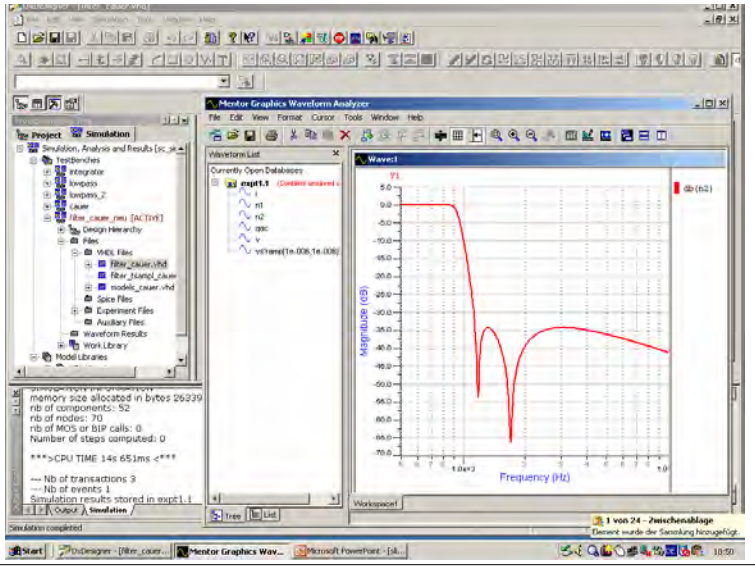
Cauer-Tiefpaß 5. Ordnung (2)



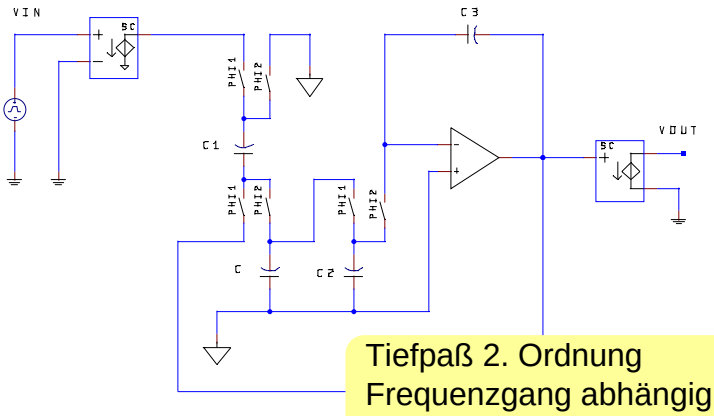
Minimale Sperrdämpfung 34 dB

Welligkeit < 0.1 dB

Simulation mit VHDL-AMS-Simulator

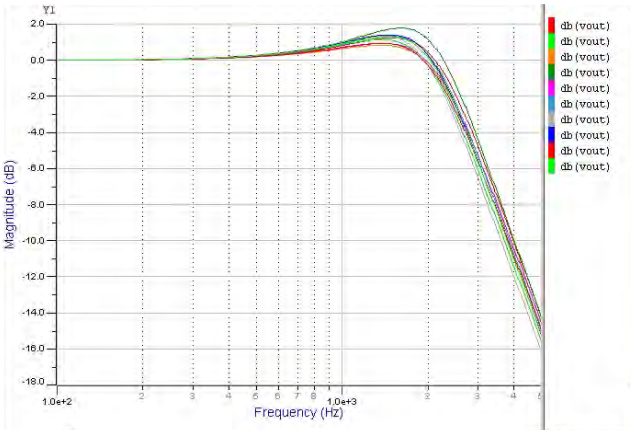


SC Filter (1)

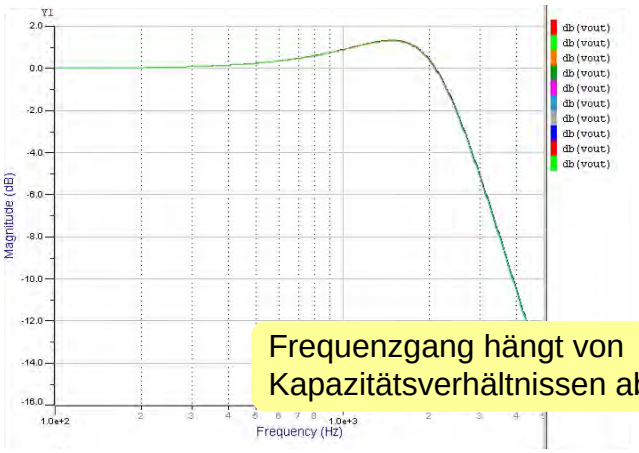


Tiefpaß 2. Ordnung
Frequenzgang abhängig von C, C1, C2 und C3

Frequenzgang
(Kapazitäten mit p=10 % Toleranz, unkorreliert)



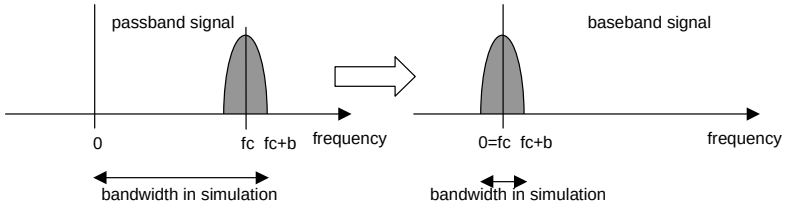
Frequenzgang
(Kapazitäten mit 10 % Toleranz, Korr. ≈ 1)



Frequenzgang hängt von Kapazitätsverhältnissen ab (C_a/C_b)

Basisbandmodelle

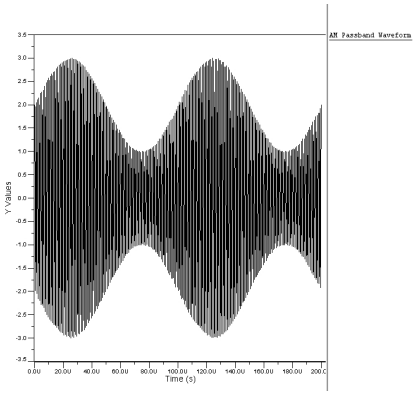
Übergang zu Basisbandmodellen



Passband-Beschreibung: $s(t) = I(t) \cdot \cos 2\pi \cdot f_c \cdot t - Q(t) \cdot \sin 2\pi \cdot f_c \cdot t$

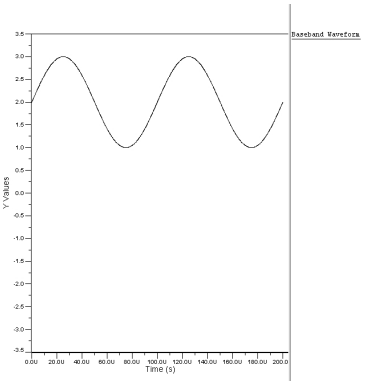
Basisbandbeschreibung: $B(t) = I(t) + j \cdot Q(t)$

Im Modell: Parallele Leitungen für I- und Q-Komponente



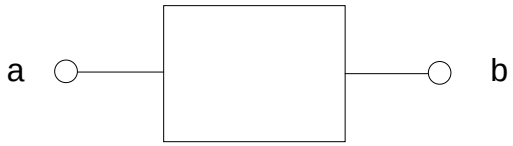
Passband Signal

Baseband Signal



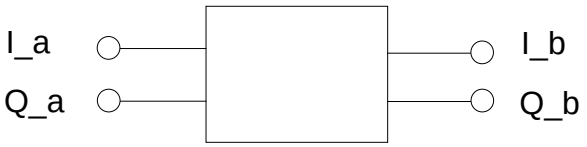
Übergang zu Basisbandmodellen

Passbandbeschreibung



Basisbandbeschreibung

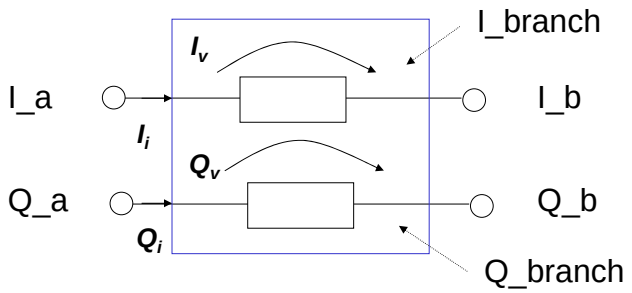
Überleitung der Modelle erforderlich



Basisbandmodell für Widerstand

Passbandmodell

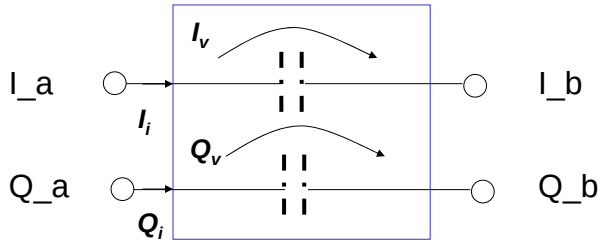
Basisbandmodell



Basisbandmodell für Kapazität (1)

Passbandmodell

Basisbandmodell



Basisbandmodell für Kapazität (2)

- Passbandbeschreibung

$$i(t) = C \cdot \frac{dv}{dt}$$

- Darstellung mit I- und Q-Komponenten

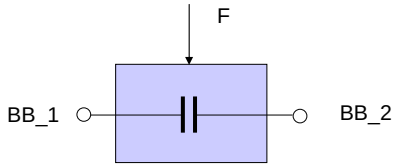
$$i(t) = I_i(t) \cdot \cos(2\pi \cdot f_c \cdot t) - Q_i(t) \cdot \sin(2\pi \cdot f_c \cdot t)$$

$$v(t) = I_v(t) \cdot \cos(2\pi \cdot f_c \cdot t) - Q_v(t) \cdot \sin(2\pi \cdot f_c \cdot t)$$

mit $$I_i(t) = c \cdot \left(\frac{dI_v}{dt} - 2\pi \cdot f_c \cdot Q_v(t) \right)$$

$$Q_i(t) = c \cdot \left(\frac{dQ_v}{dt} + 2\pi \cdot f_c \cdot I_v(t) \right)$$

Randbeschreibung in VHDL-AMS

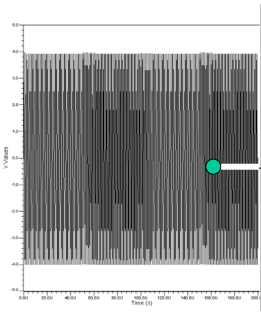



```
library BB_LIB;
use BB_LIB.BB_BASICS.all;

entity BB_CAPACITOR is
   generic (C    : REAL := 0.0
            -- Value of the capacitance [F]
           );
   port    (terminal BB_1 : BB_NATURE_VECTOR(1 to 2);
            -- First connection point

            terminal BB_2 : BB_NATURE_VECTOR(1 to 2);
            -- Second connection point

            signal   F    : REAL := 0.0
            -- Associated carrier frequency [Hz]
           );
end entity BB_CAPACITOR;
```


Modelle für Basisbandsimulation

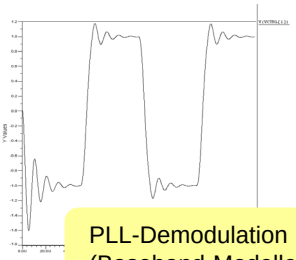
Modell	Bemerkung
BB_RESISTOR	Widerstand
BB_CAPACITOR	Kapazität
BB_INDUCTOR	Induktivität
BB_ADD	Addier-Block
BB_DELAY	Verzögerungsblock
BB_MULT	Multiplizierer
BB_AMPLIFIER	Nichtlinearer Verstärker
BB_FM_MODULATOR	FM Modulator
BB_AM_MODULATOR	AM Modulator
...	...

Beispiel

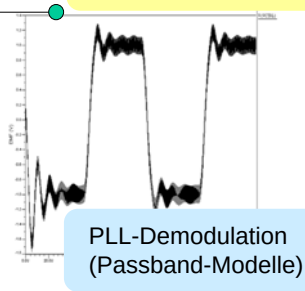


FM-Signal im Passband

PLL



PLL-Demodulation
(Baseband-Modelle)



PLL-Demodulation
(Passband-Modelle)

Zusammenfassung

- ❑ Frequenzgangberechnung für SC-Schaltungen mit VHDL-AMS-Simulationsprogrammen möglich
- ❑ Einschränkungen durch Ladungs-Spannungs-Modellansatz
  - Ideale Schalter und Operationsverstärker
  - Feste Verzögerungszeiten zwischen den einzelnen Taktphasen
- ❑ Rechenzeitreduzierung durch Basisbandmodelle für schmalbandige RF-Signale bei geringem Genauigkeitsverlust
- ❑ Nutzung der Möglichkeiten von VHDL-AMS
  - Mehrdimensionale Netzwerke
  - Erstellung eigener Grundelemente
  - ...

Literaturhinweise

- Kuo, C.F.; Moschytz, G.S.: Two-Port Analysis of Switched-Capacitor Networks Using Four-Port Equivalent Circuits in the z-Domain. IEEE Trans. Circuits Syst. 26(1976)3, pp. 166-179.
- Allen, P.E.; Holberf, D.R.: CMOS Analog Circuit Design. New York/Oxford: Oxford University Press, 2002.
- Haase, J.; Trappe, P.: Modellierung mit mehrdimensionalen Netzwerken. In Hülsmann, F.; Kowarschik, M.; Rüde, U.: Proceedings 18. Symposium Simulationstechnik ASIM 2005. Erlangen: SCS Publishing House e.V., 2005, S. 213-218.
- Chen, J.E.: Modeling RF Systems. Verfügbar: http://www.designers-guide.com/Modeling

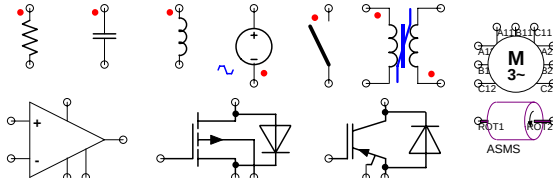
SIMPLORER

ASIM – 20.02.2006

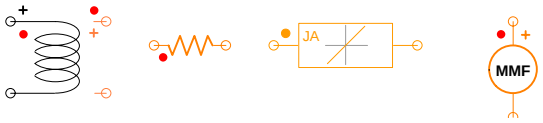
Olaf Hädrich, Application Engineer, Ansoft Munich

Multidomain – Physical and Cybernetic

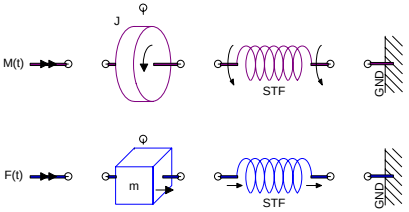
- Electrics



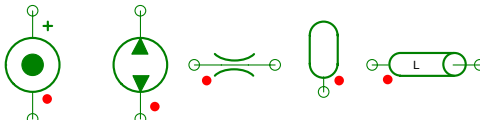
- Magnetics



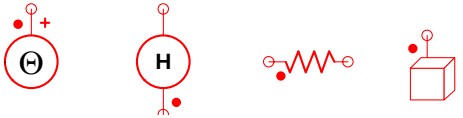
- Mechanic



- Hydraulics

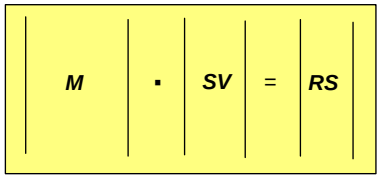


- Thermal

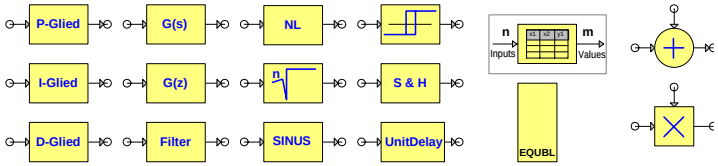


- ...

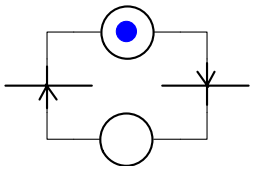
- Differential equations (n^{th}-Order)



- Block diagrams

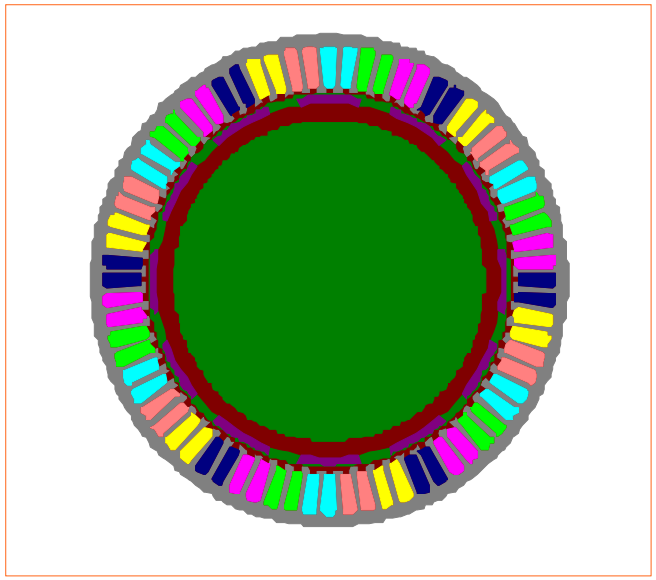


- State graphs

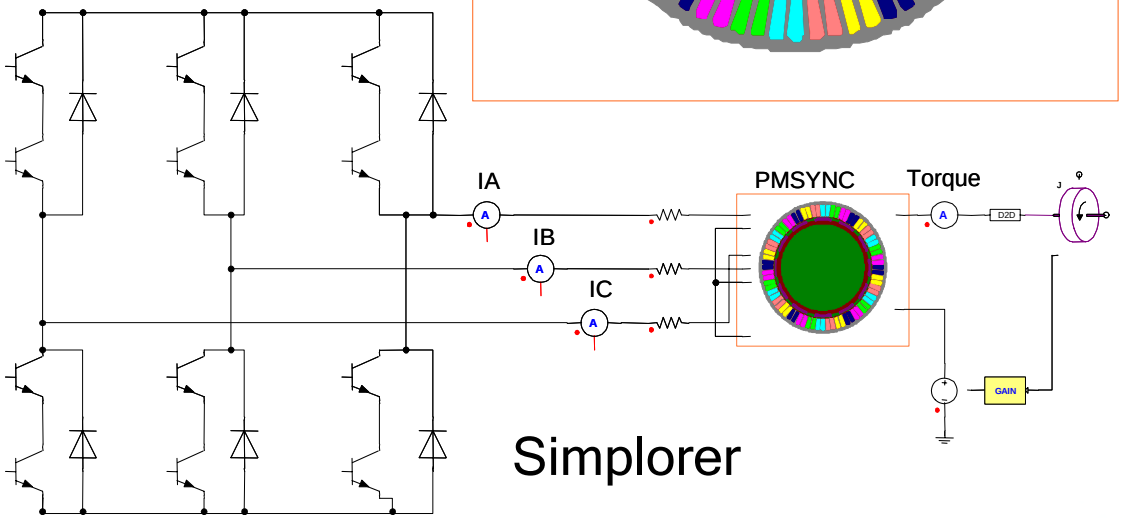


Multilevel – Low Frequency

- Component – detailed „in-field“ examination
- Parameter extraction, cosimulation
- System – interactions



Maxwell

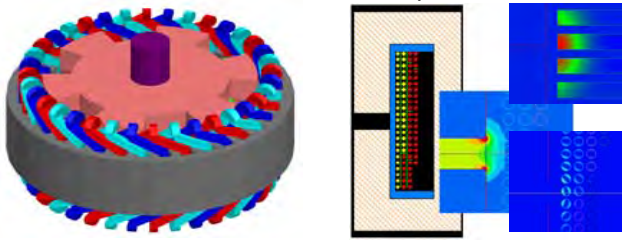


Simplorer

Integrated Multilevel Environment

RMxprt / PExprt
Analytical Design

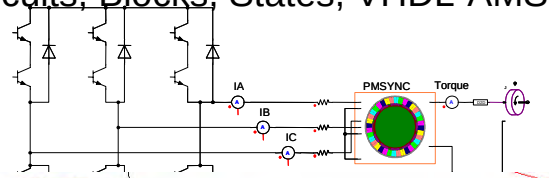
Machines, Transformers, Inductors



Model Generation

Simplorer
System Design

Circuits, Blocks, States, VHDL-AMS

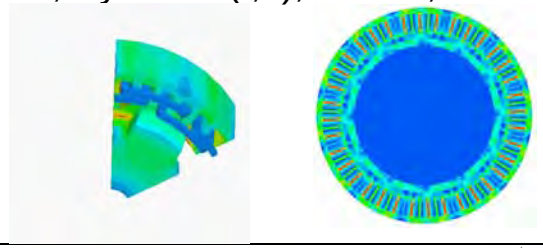


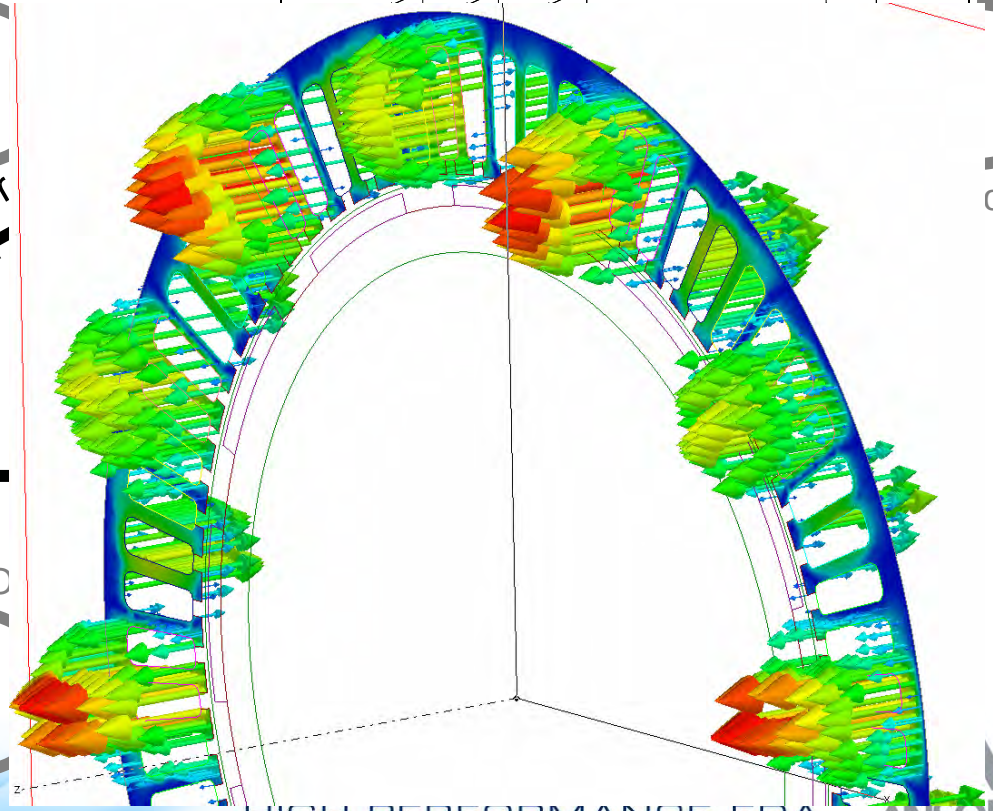
Model Generation

Optimization

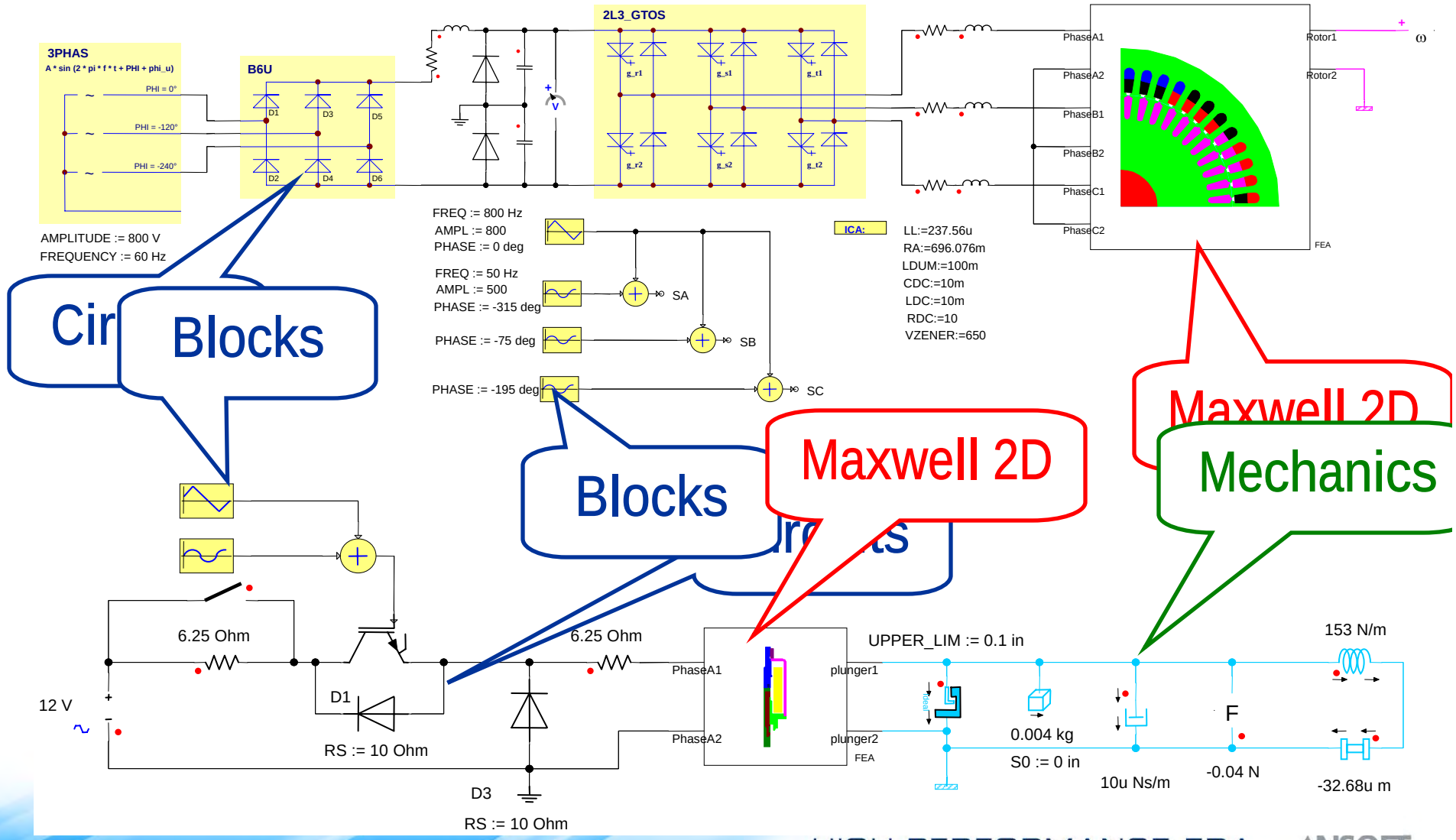
Maxwell 2D/3D
Component Design

Static, Dynamic (*t*, *f*), Motion, Circuits



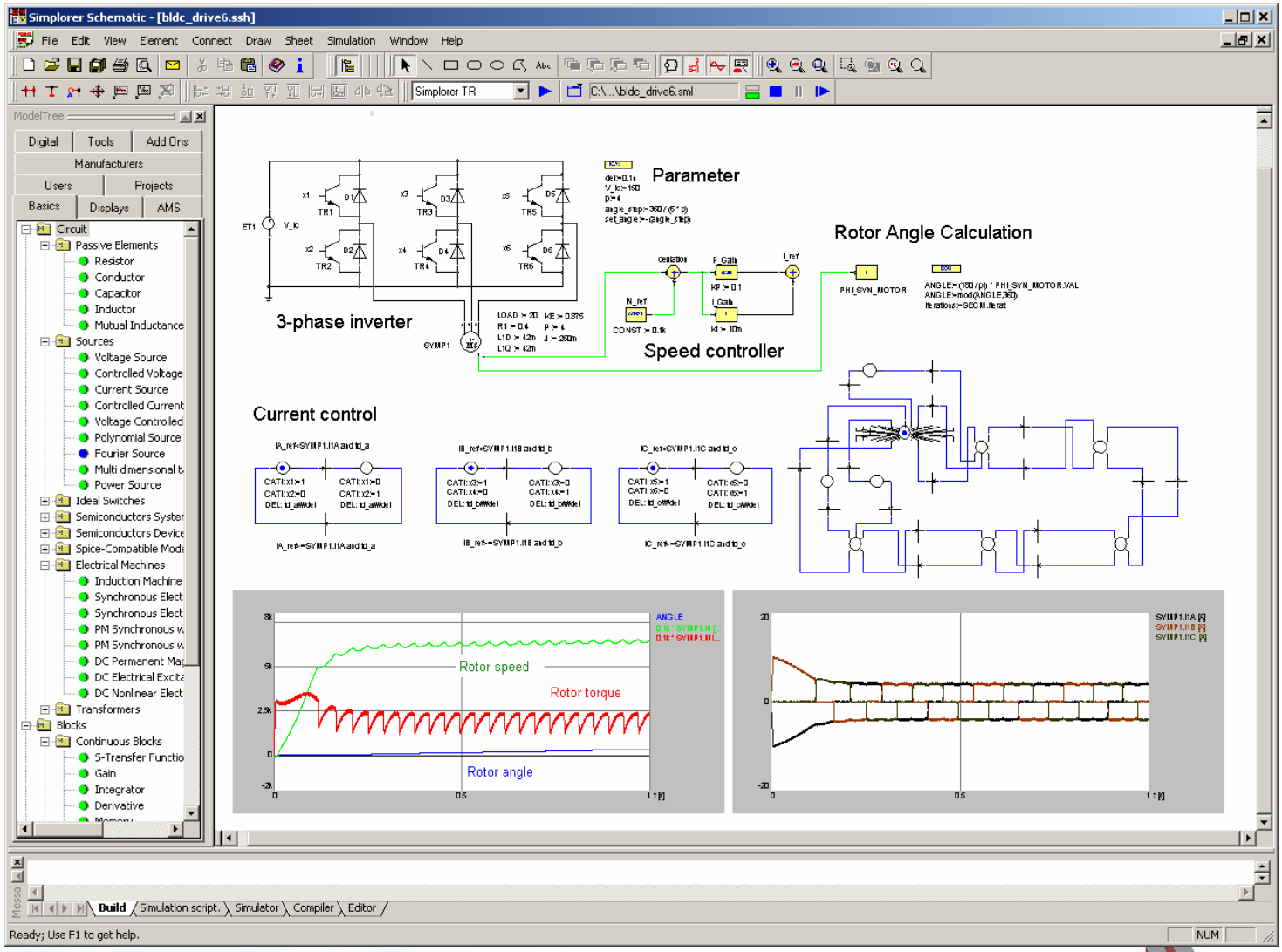


Simplorer – Maxwell Transient Link

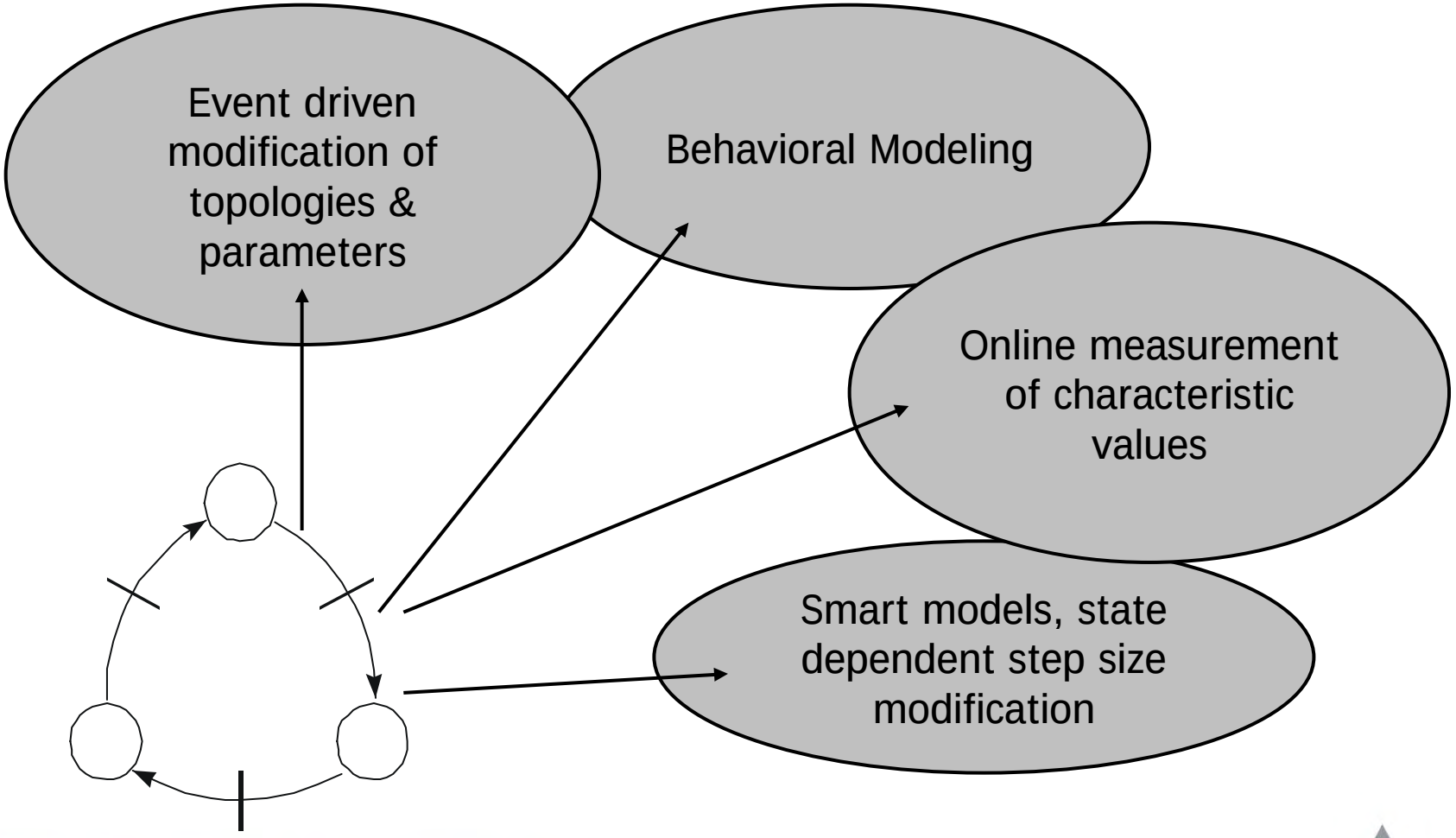


Simplorer Design Environment

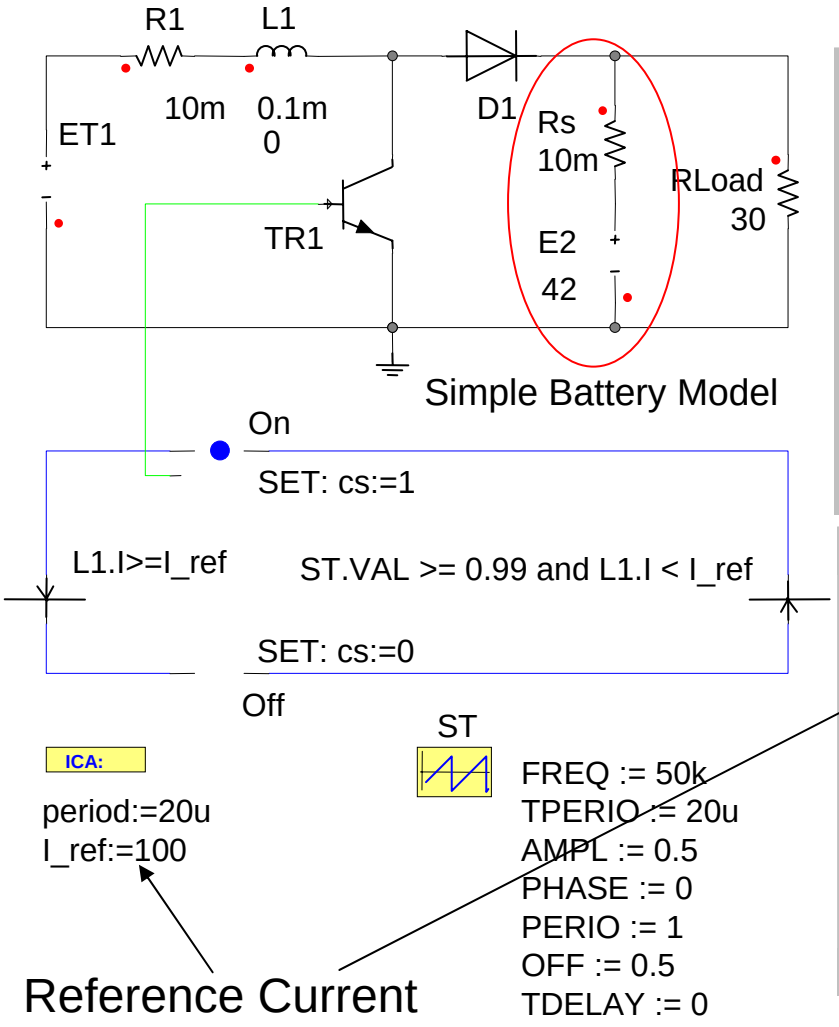
- Windows-like
- Modeling
  - Graphical
  - SML, VHDL-AMS, C/C++
- Hierarchical models
- All quantities accessable
- Postprocessing on sheet or in Postprocessor
- Scripting in VB, Tcl/Tk, ...
- Model libraries
- Wizards
- Physical units
- Multisimulation, Monte-Carlo, Optimization, ...

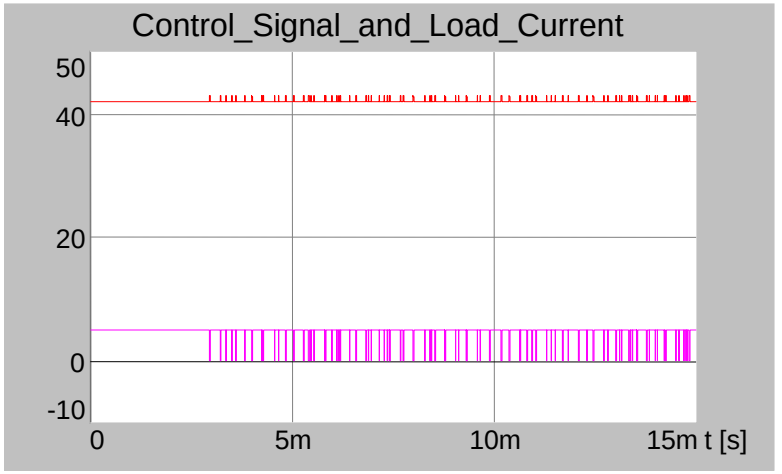


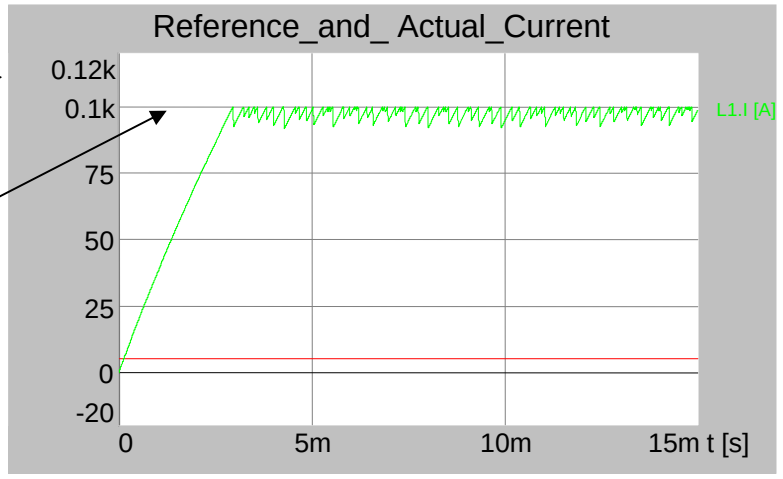
State Machines



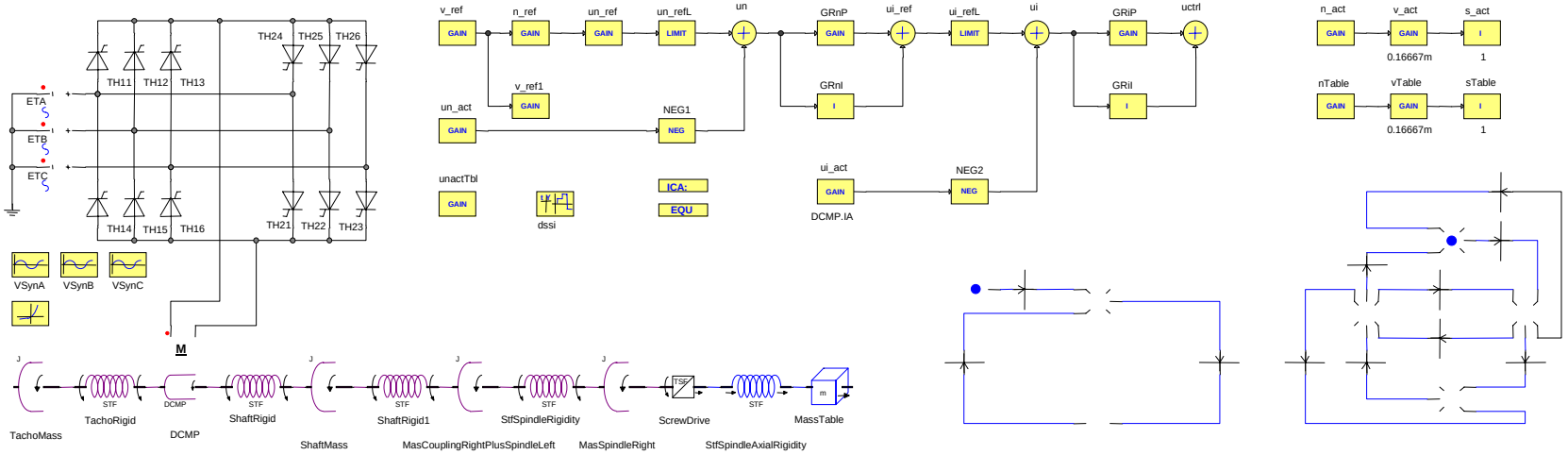
Simple Boost Converter Control Example

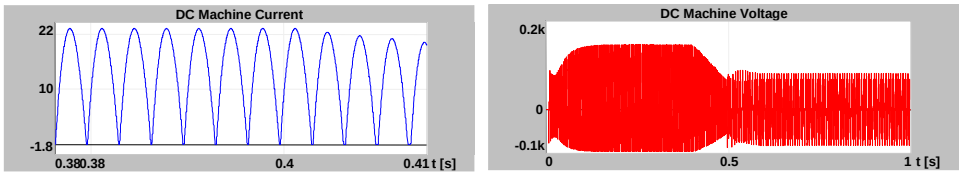


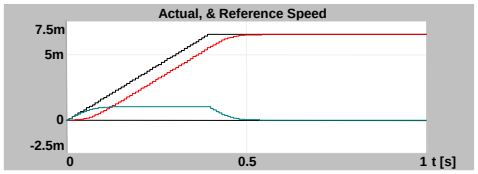


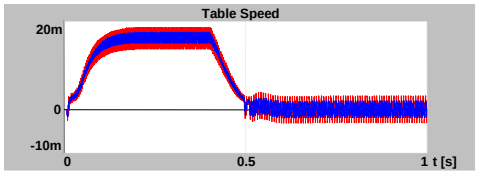


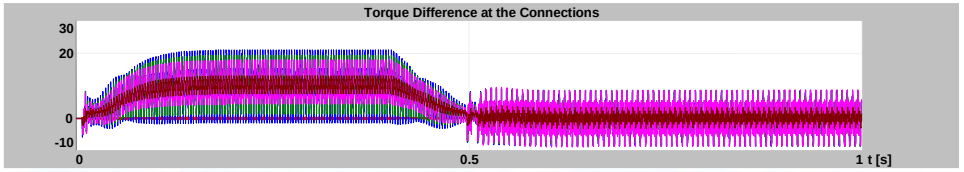
Position Controlled Drive Example

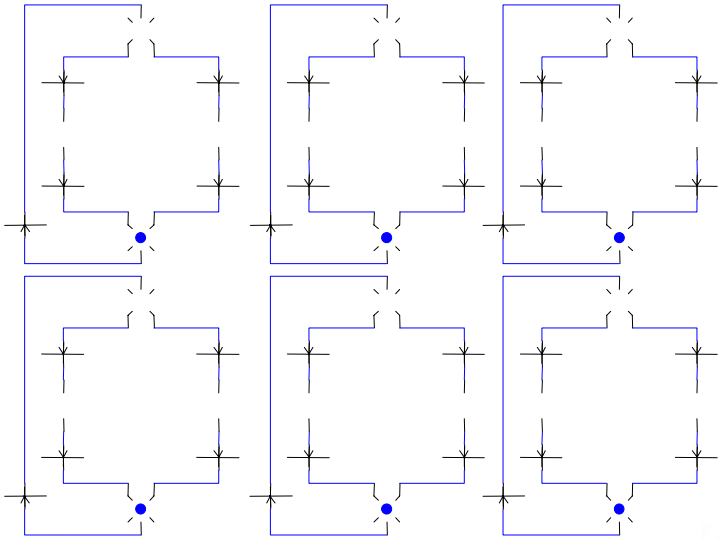












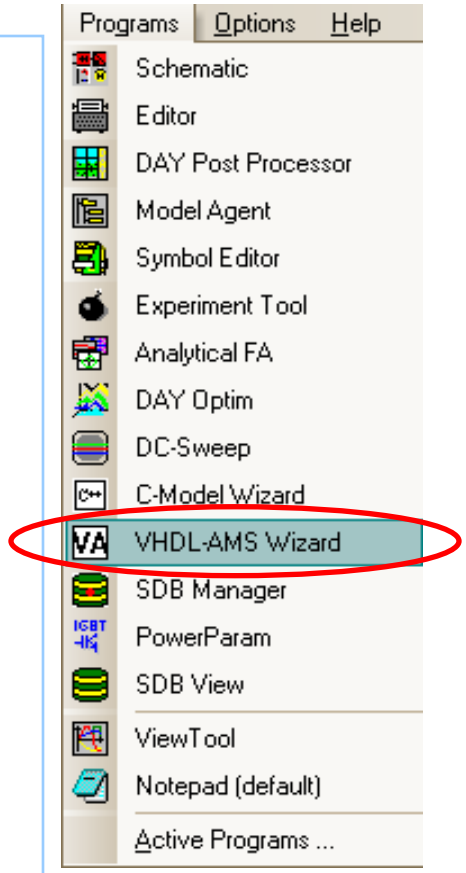
Simplorer Interfaces

- VHDL-AMS – standard IEEE approved system modeling language
- RMxprt, PExprt, Maxwell 2D/3D, Q3D, Fullwave

- SIM2SIM – integrated interface between
  - Simplorer and Simulink for online simulation
  - Simplorer/DAY and Matlab for pre-/postprocessing
- SPICE converter – integration of SPICE models
- External simulators, C/C++-interface, µ-controller code ...

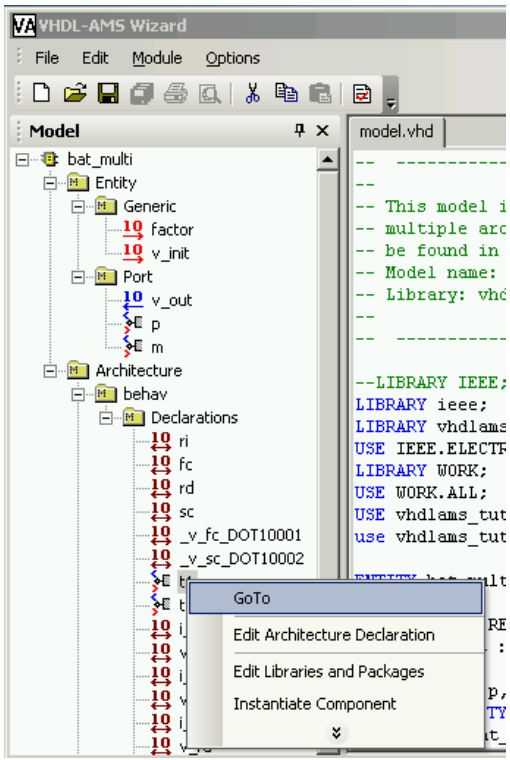
VHDL-AMS

- Modeling
  - VHDL-AMS-libraries (open)
  - Graphical and text modeling
  - Tools
    - Stimulus Generator
    - VHDL-AMS Wizard
- Model export and –import of
  - single component models
  - complete libraries
  - Sheets from Schematic
- *Foreign Function* and *Foreign Model*
  - Export of Schematics as text,
  - including SML- and C-models (no model exchange anymore)



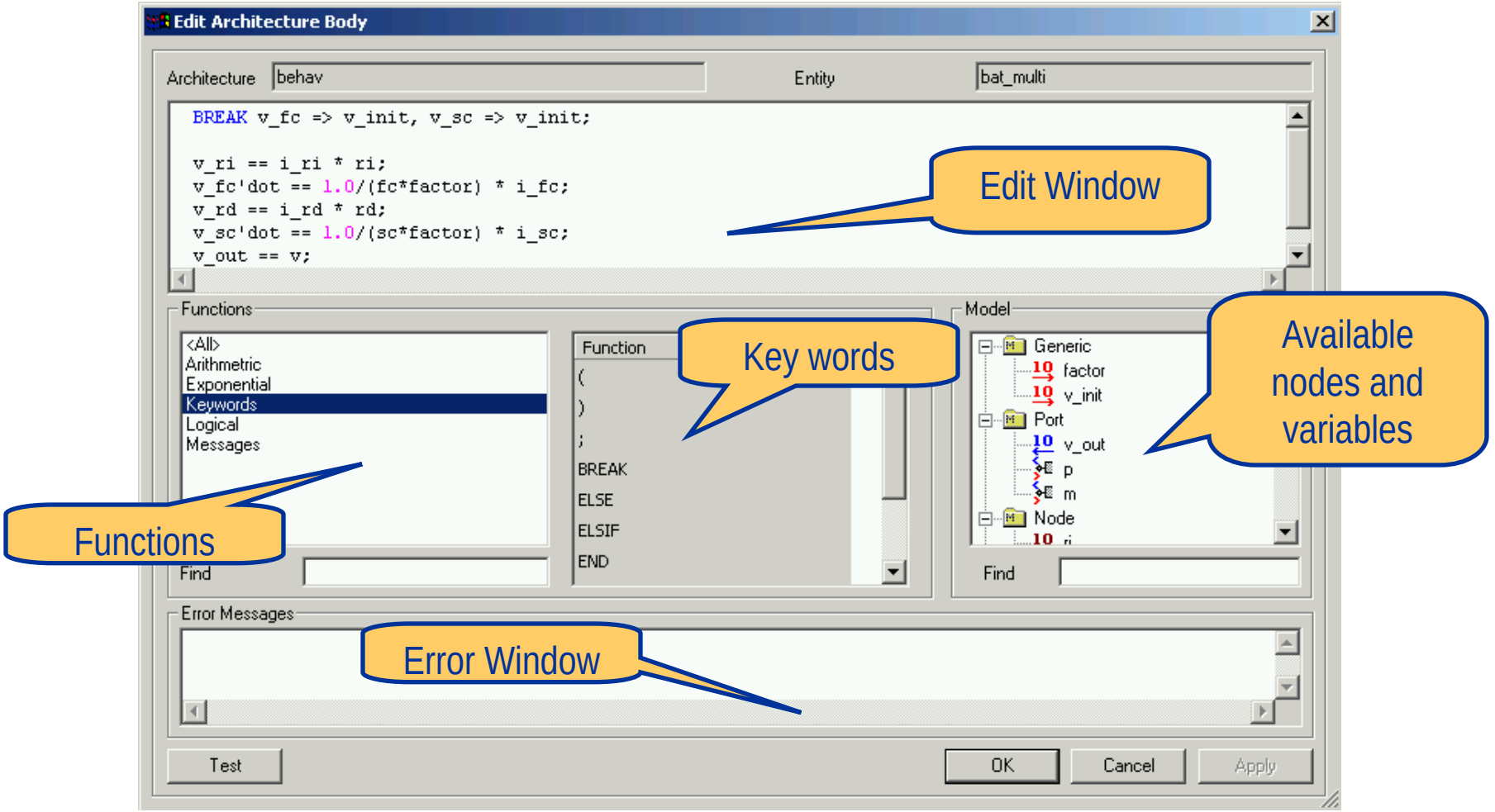
VHDL-AMS Wizard

- Easy model generation
  - User friendly dialogs
  - Specification of libraries and packages
  - *Entity* and *Architecture* description
  - Multiple architecture programming
- Syntax check and compilation
- *Go to* Functionality
- Allows ***Dual Mode*** **Editing**
  - Graphical editing in dialogs
  - Textual editing



Considerably accelerates learning VHDL-AMS

VHDL-AMS Wizard

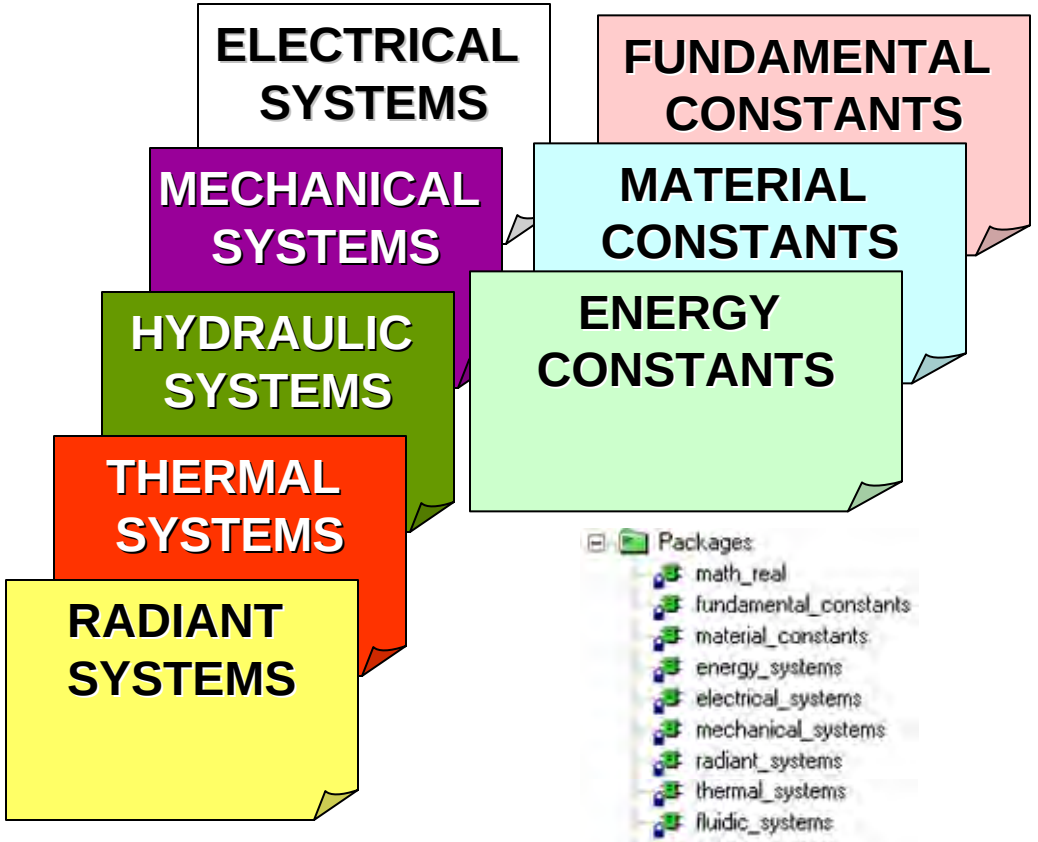


VHDL-AMS – AK30-Libraries

- Batch-mode / remote control
  - Yes, by minimum UI-usage
- Symbol format
  - Yes, „alpha version“ released in v7.0.5
- Encryption
  - Yes, new 64-bit encryption tool
  - Usable for Simplorer and Non-Simplorer users
  - Freely available on www.ansoft.com
- VDA-libraries in Simplorer
  - Yes – model.simplorer.com or OTS

IEEE VHDL-AMS 1076.1.1 Physical Domain Packages

- IEEE standardized the Physical Domain Packages in December 2004
- Packages now include definitions for several material and physical constants
- Newly standardized packages will replace the draft packages in Simplorer v7.0.3



SAE J2546 EDA Group
VHDL-AMS Statistical Packages

- SAE-STAT WG established in March ’05
- Objective is to standardize a package that provides support within VHDL-AMS for statistical modeling and simulation
- WG represented by Ansoft, Synopsys, Mentor and GM
- Timeline
  - Requirements Analysis completed in April ’05
  - Package standardization deadline was October ‘05

SAE International

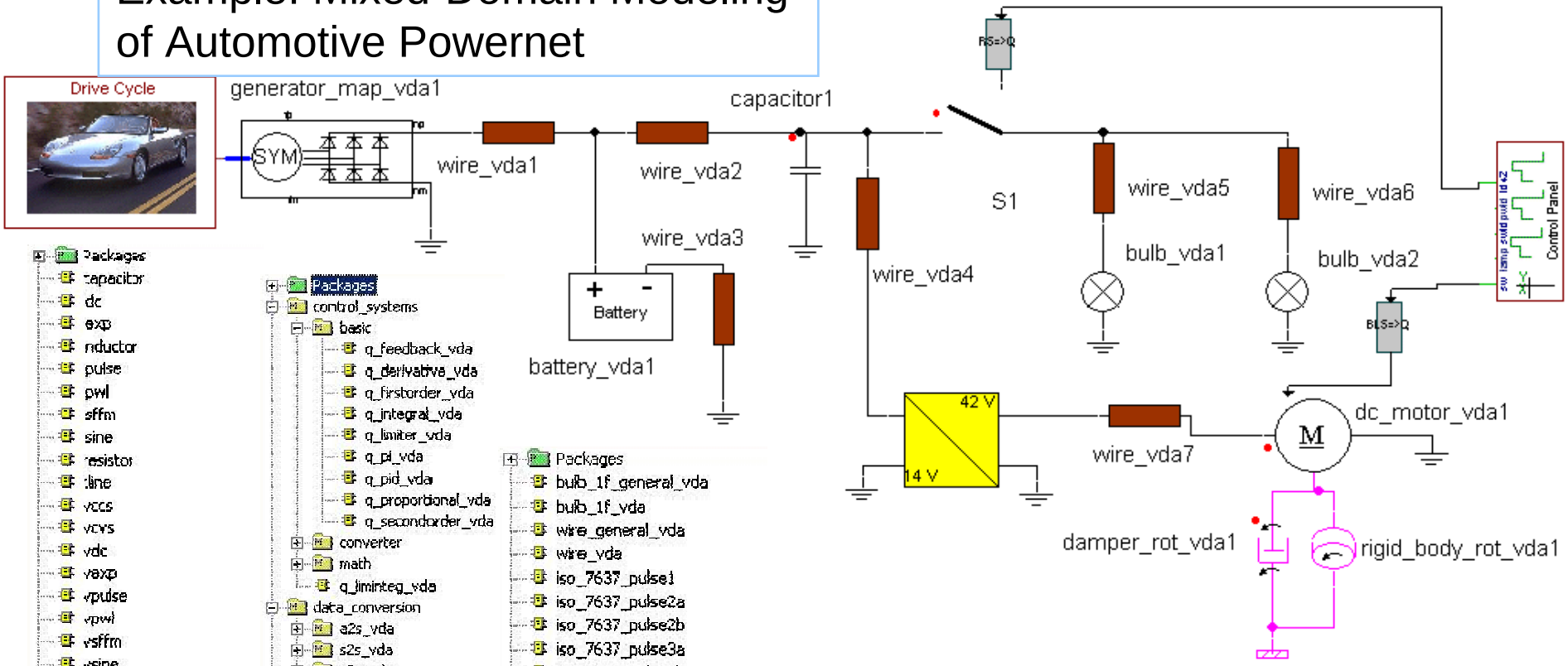
Requirements

1. It must be possible to assign a statistical distribution to each constant.
2. It must be possible to specify different statistical distributions for different constants.
3. Different instances of the same constant that have a statistical distribution must evaluate to different numeric values during a statistical analysis.
4. It must be possible to specify a correlation between constants.
5. It must be possible to use both continuous and discrete distributions.
6. It must be possible to define both symmetrical and asymmetrical distributions with respect to the nominal value.
7. It must be possible for the user to add their own statistical distributions without affecting any other distributions.
8. The use model for user-defined statistical distributions should be the same as for predefined statistical distributions.

GM ANSOFT SYNOPSYS Mentor Graphics

AK30 VHDL-AMS Libraries

Example: Mixed-Domain Modeling of Automotive Powernet



Simplorer v7.0.3

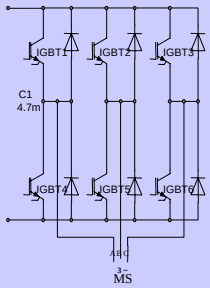
HIGH-PERFORMANCE EDA

ANSOFT

Integrated EM Design Flow

RMxprt

System Models

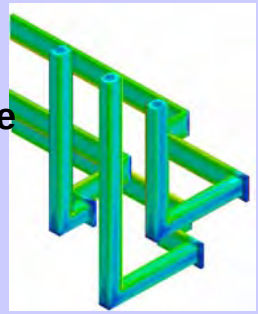


SIMPLORER

Current, Voltage

Inductance, Capacitance

Transient Link

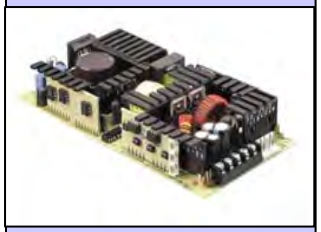


Maxwell

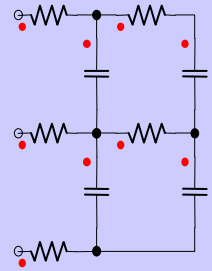
EM Force, Current

Link with other Ansoft desktops

System Models



PExprt



ECE

ePhysics

High Level Analysis

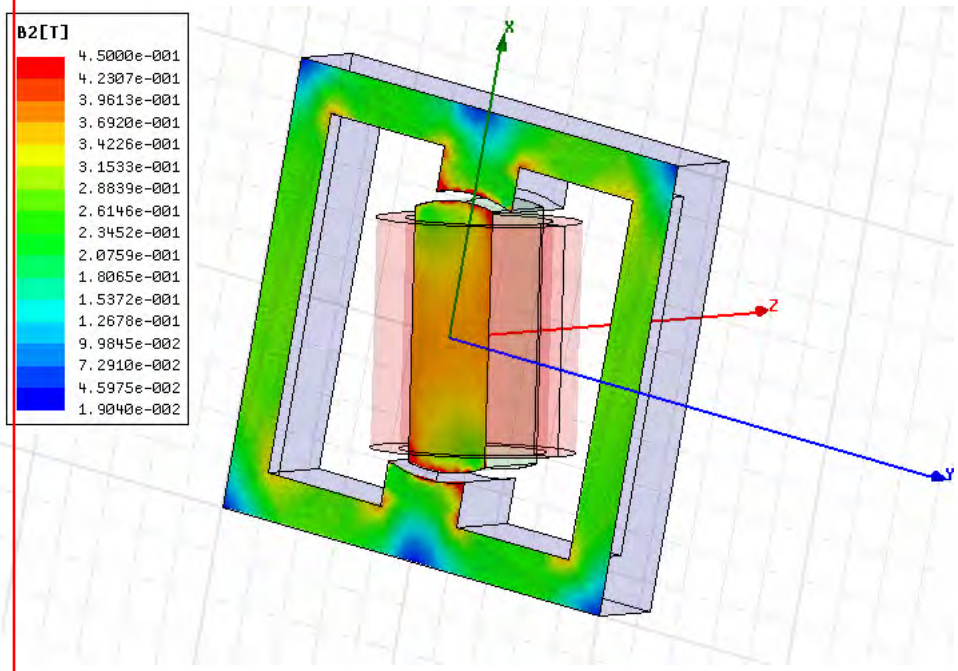
Parameterization
Optimization
Sensitivity Analysis
Statistical Analysis

Maxwell 2D/3D

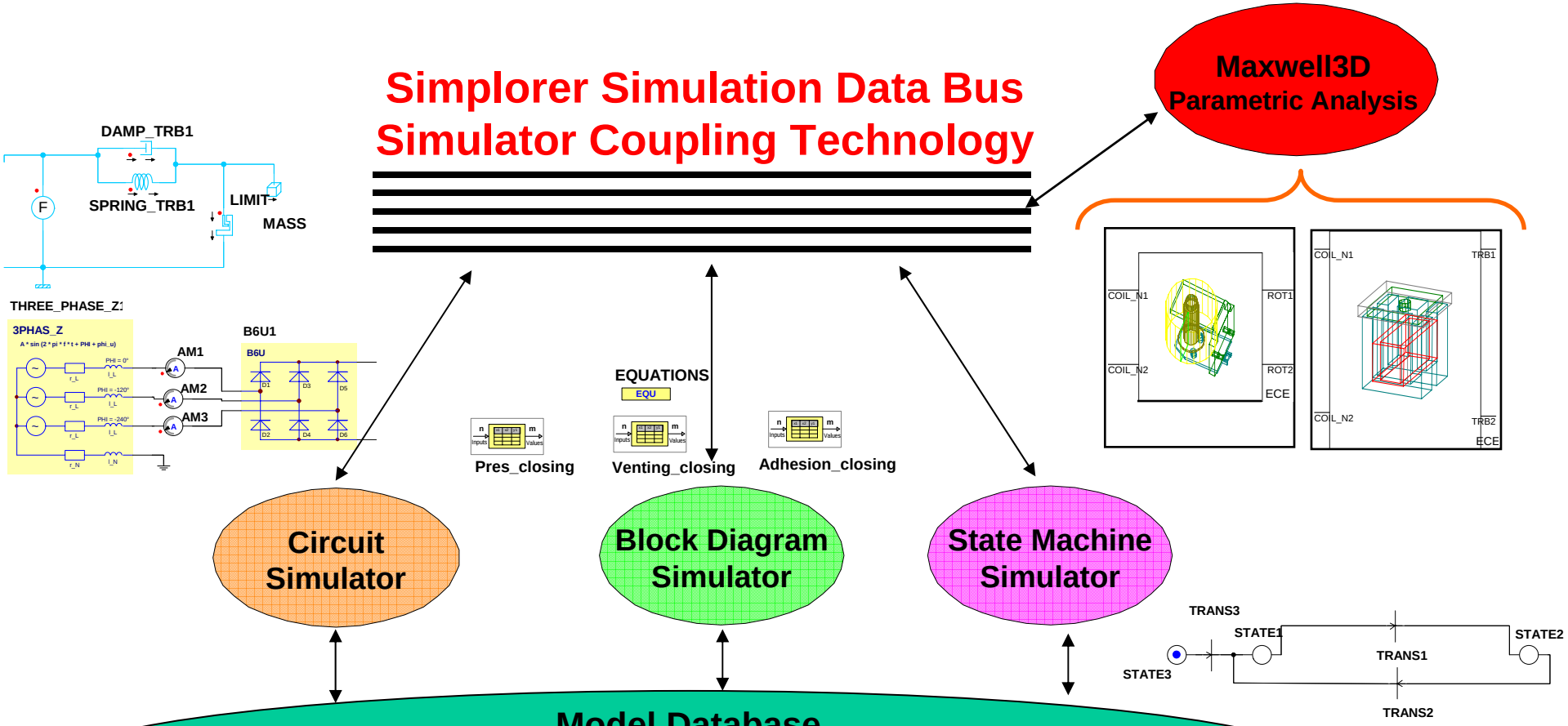
Transient, frequency-domain, static solution of electromagnetic fields

- Autoadaptive meshing
- Large motion
- dB/dt transients by motion
- Electric circuits (PE switching)

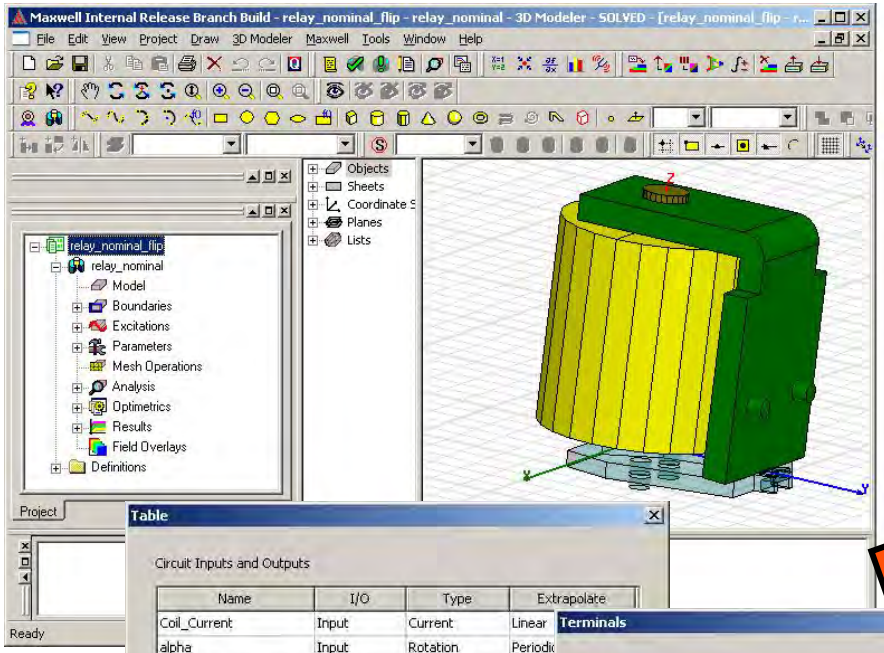
- Use whereever wave effects are subdominant

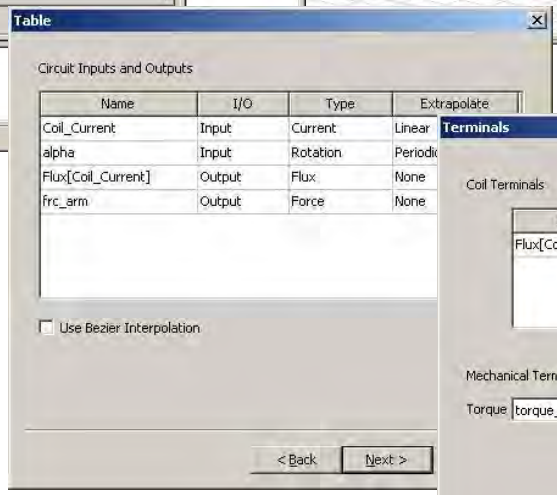


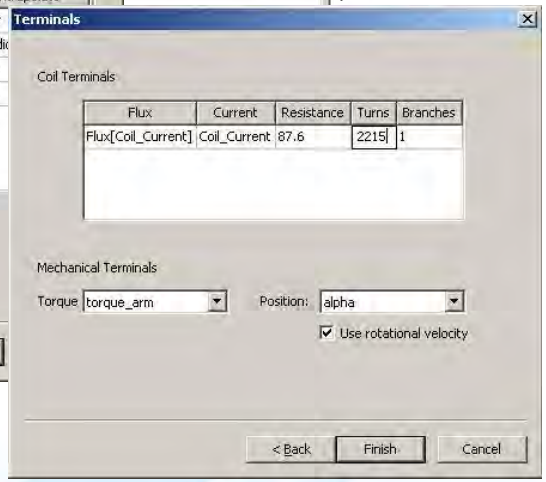
Integration with Maxwell for Complex Physics based Simulations

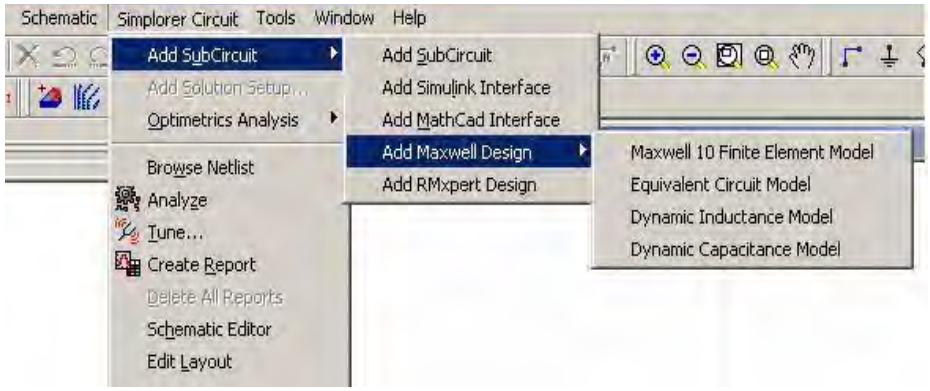


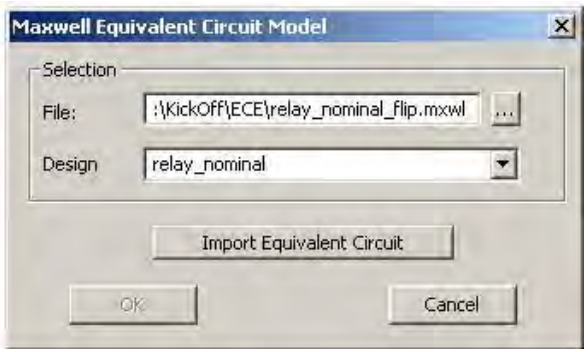
ECE – Equivalent Circuit Extraction
Maxwell – Simplorer



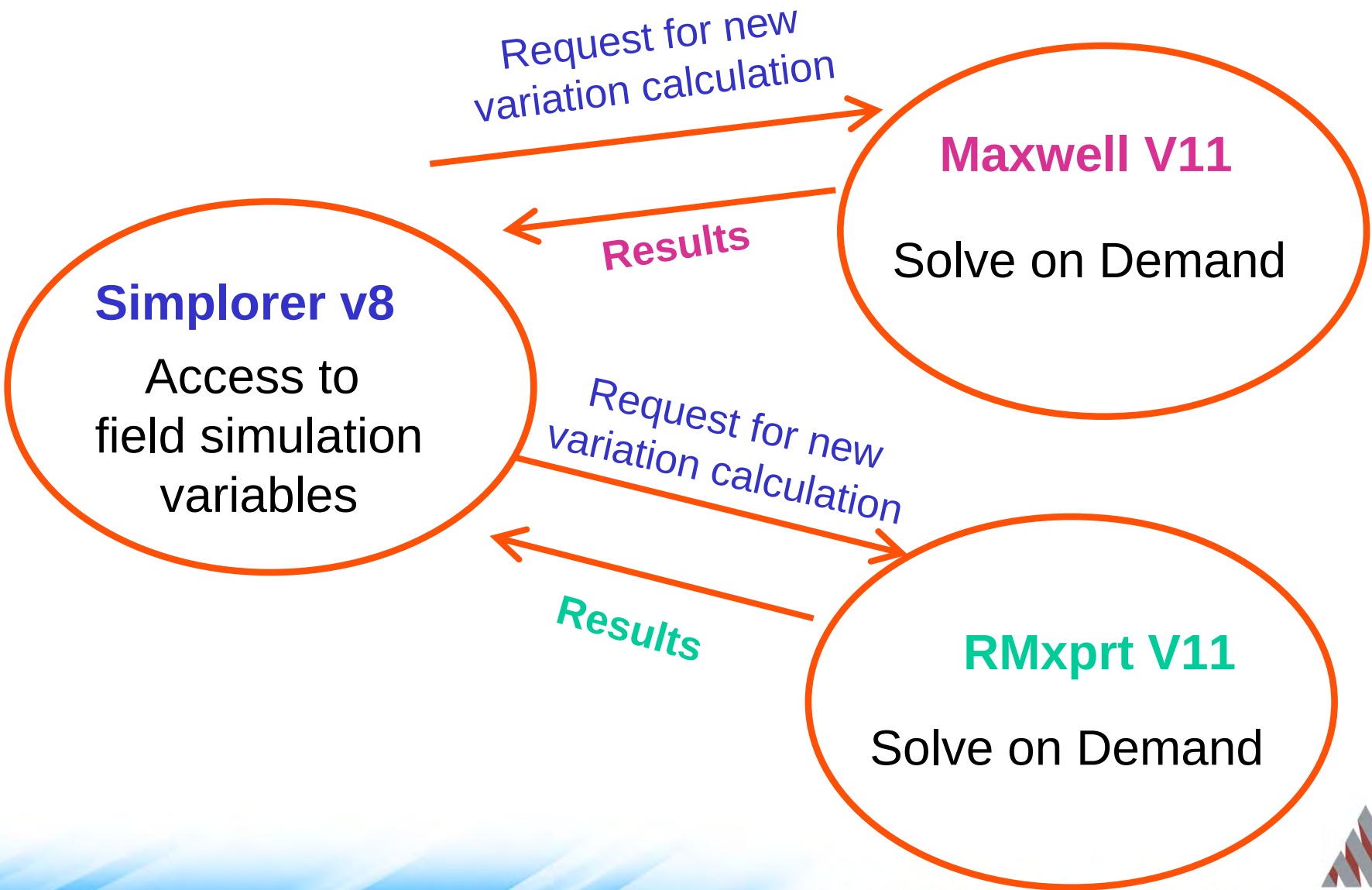






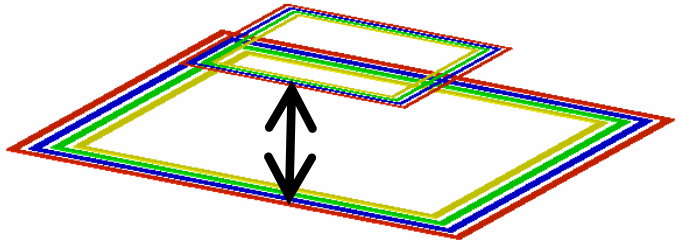


ECE – Solve on Demand



ECE – Solve on Demand

Maxwell 11 variables

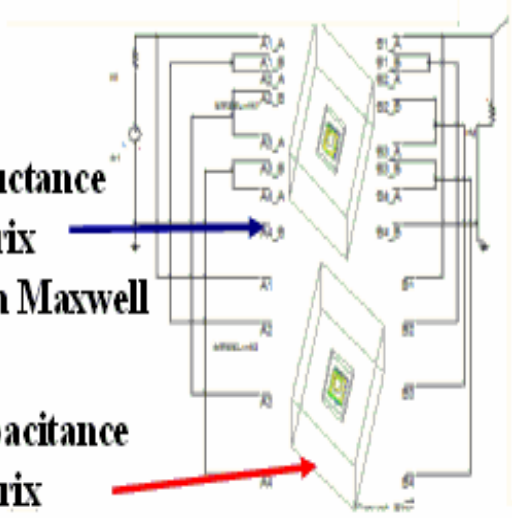


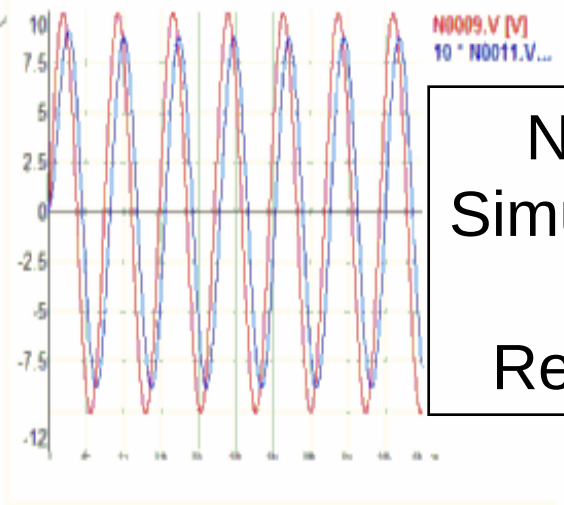
Changed in SIMPLORER

Automatic updates

Inductance matrix from Maxwell

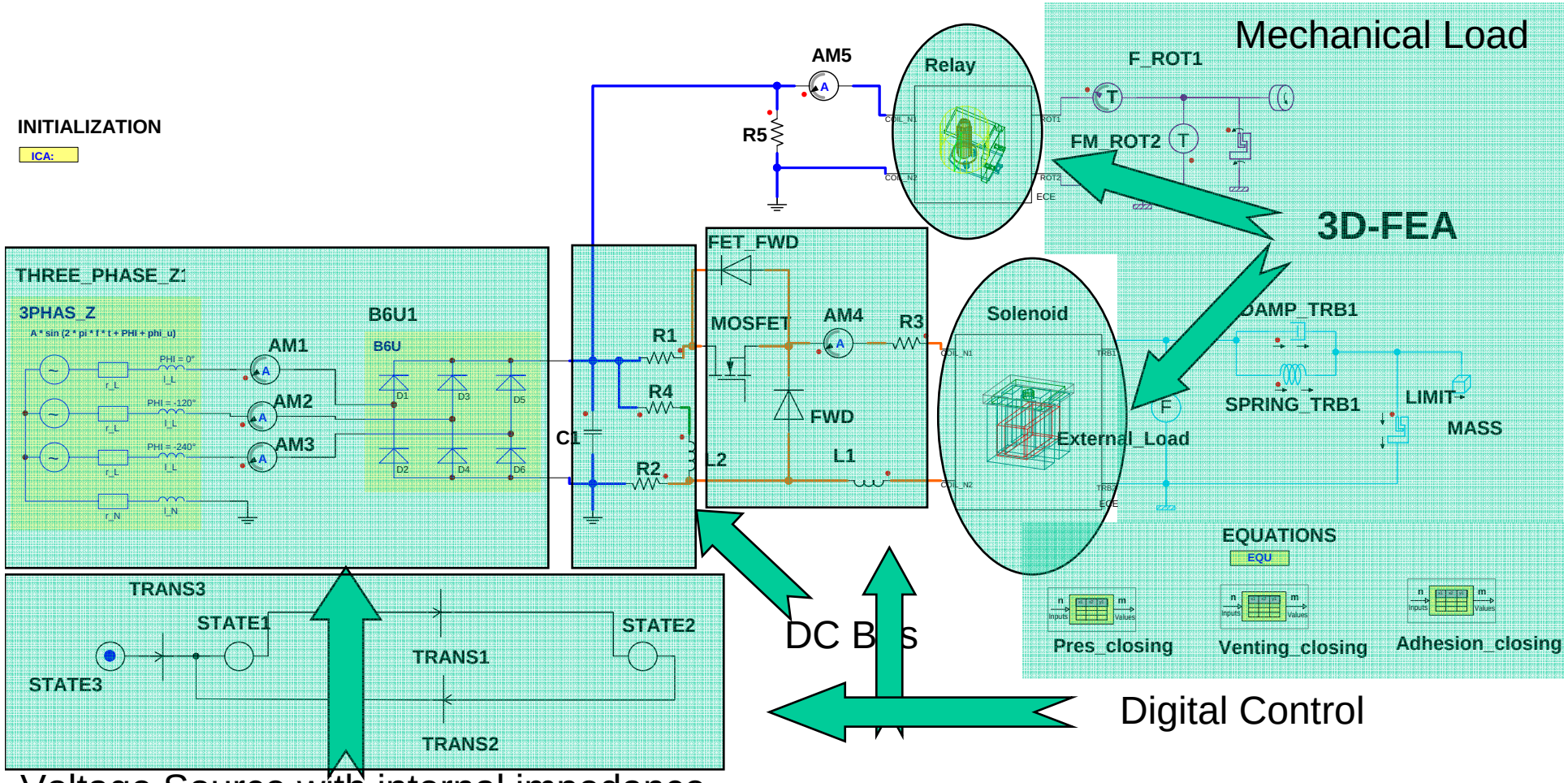
Capacitance matrix from Maxwell





New Simulation & Results

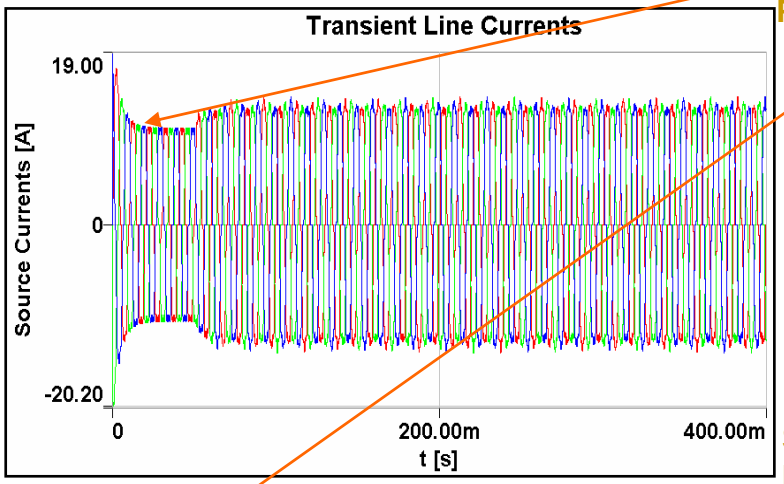
Case Study 1 – Integrated Sub-System

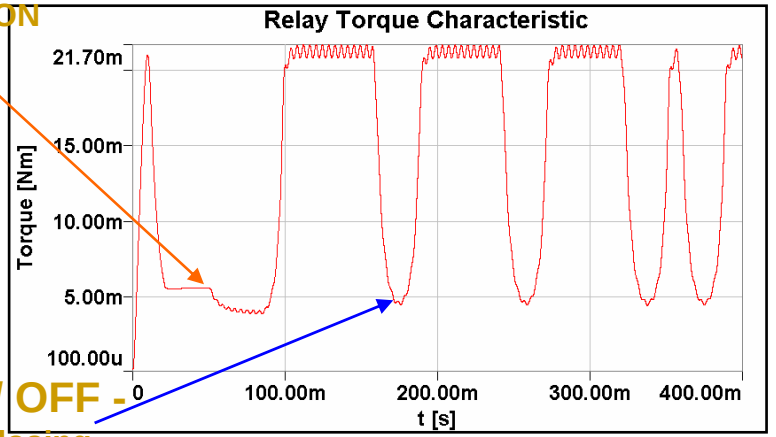


Case Study – Integrated Sub-System

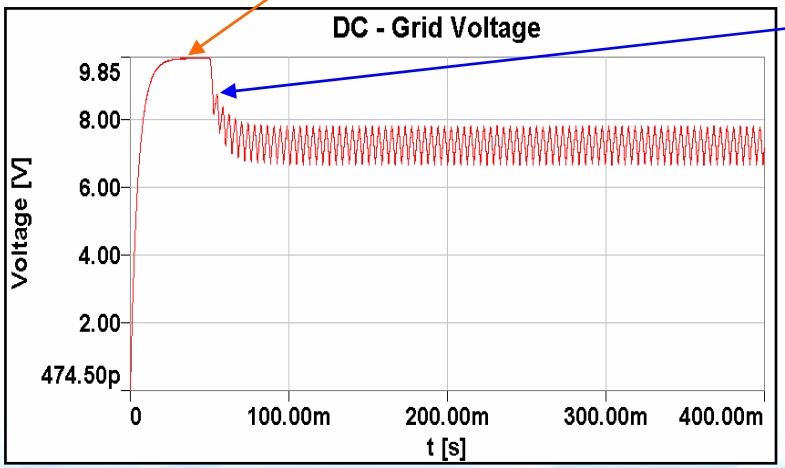
Sub-System Operation – Relay Chattering

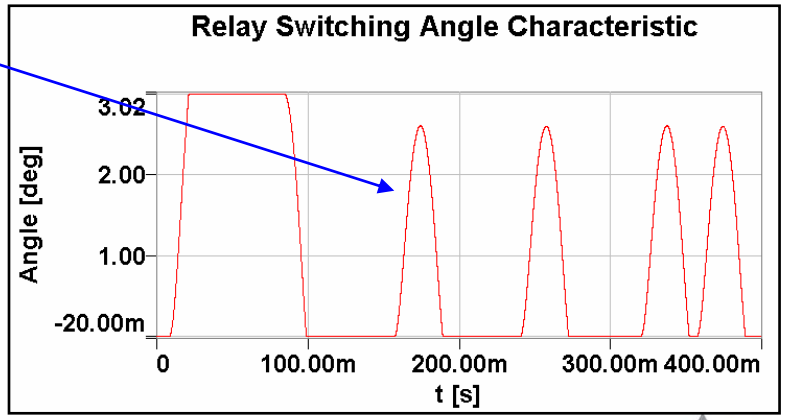
MOSFET - OFF -
Relay is switching ON





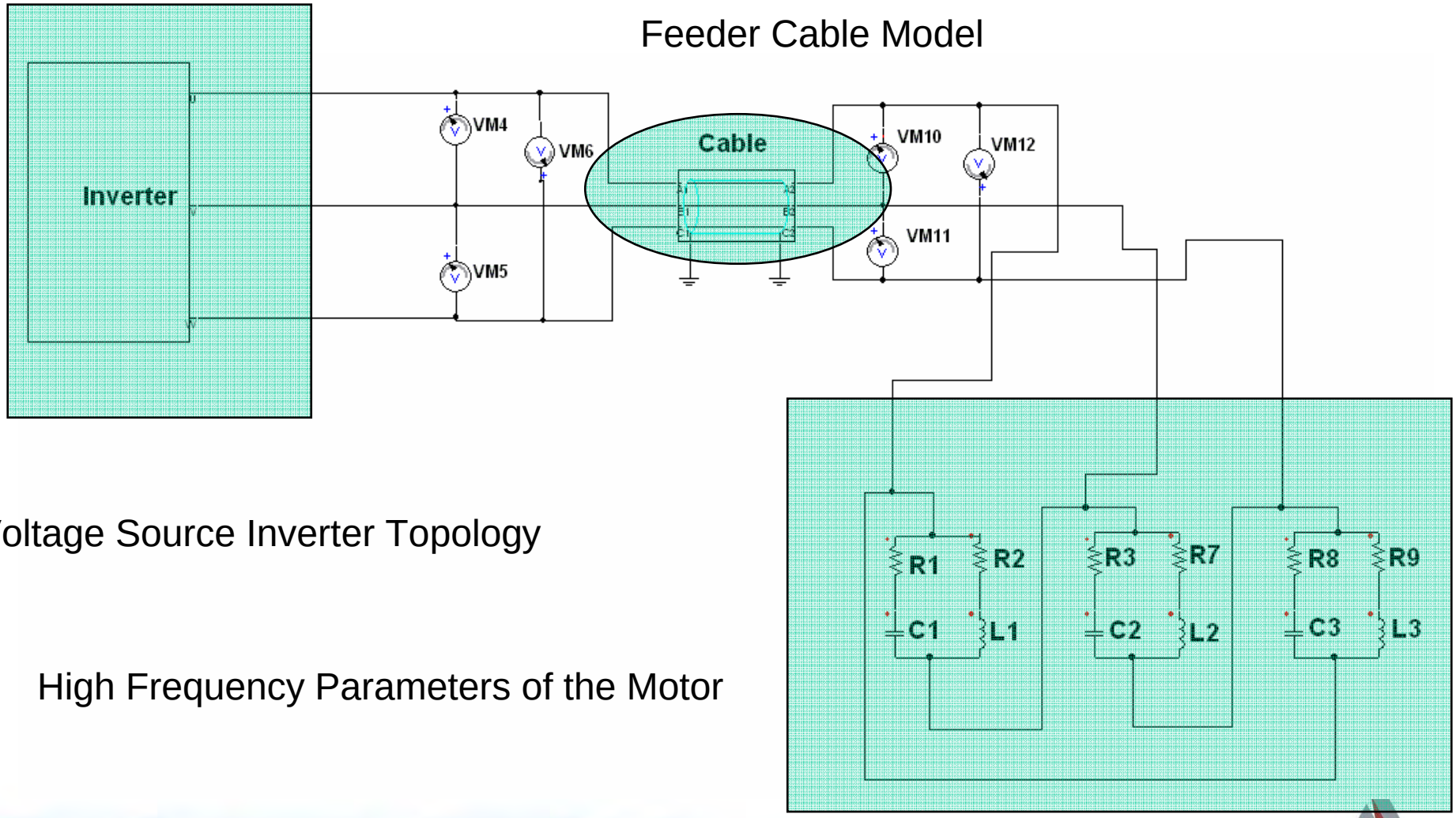
MOSFET - ON/ OFF -
Solenoid valve is closing
Relay is chattering





Case Study 2 – Inverter Fed Induction Motor Design

Feeder Cable Model



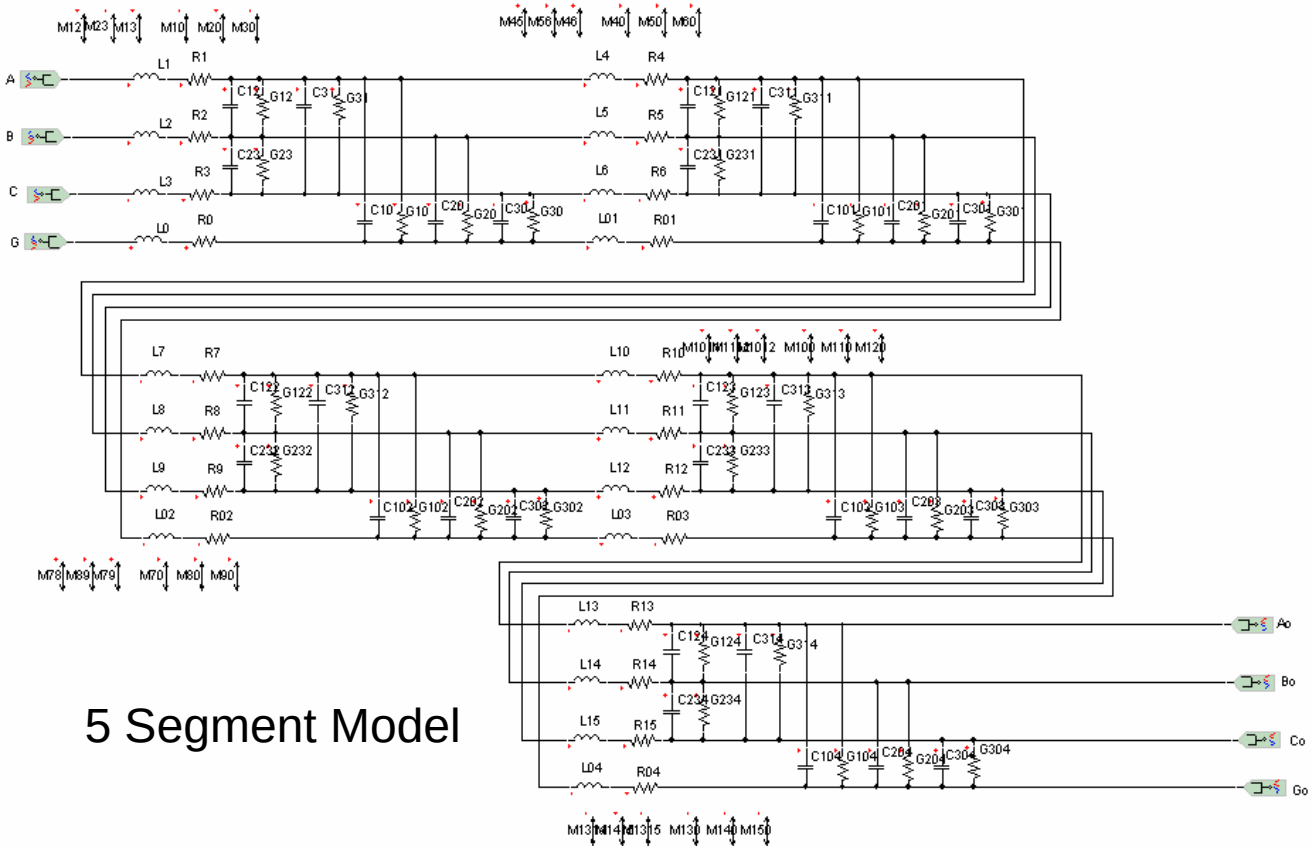
Voltage Source Inverter Topology

High Frequency Parameters of the Motor

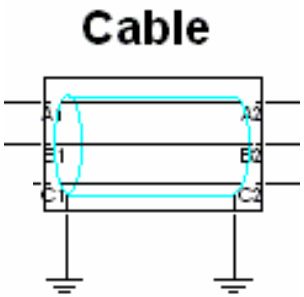
Case Study – Inverter Fed Induction Motor Design

Feeder Cable Model

Using hierarchical structure within SIMPLORER environment



5 Segment Model



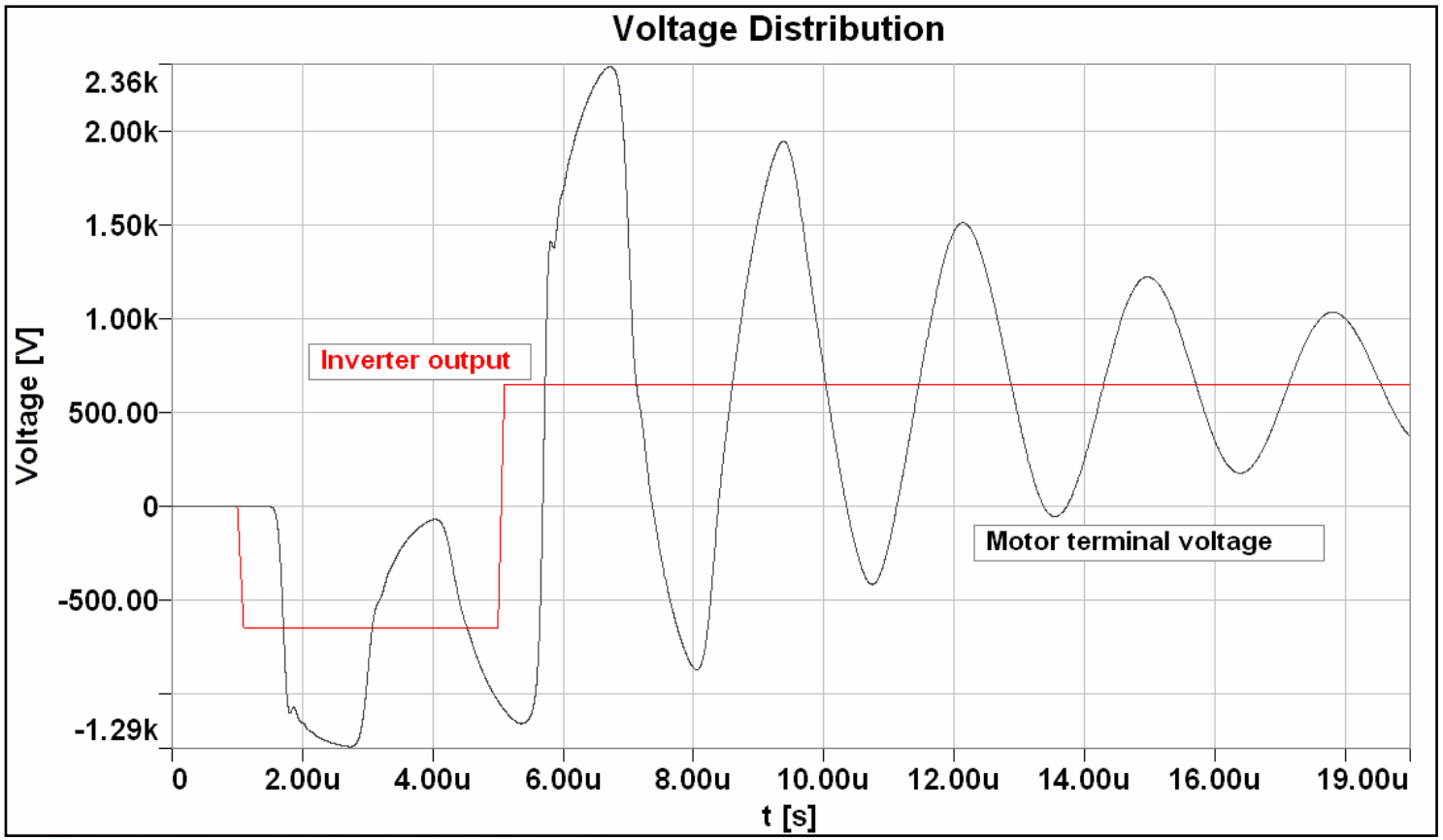
A2

B2

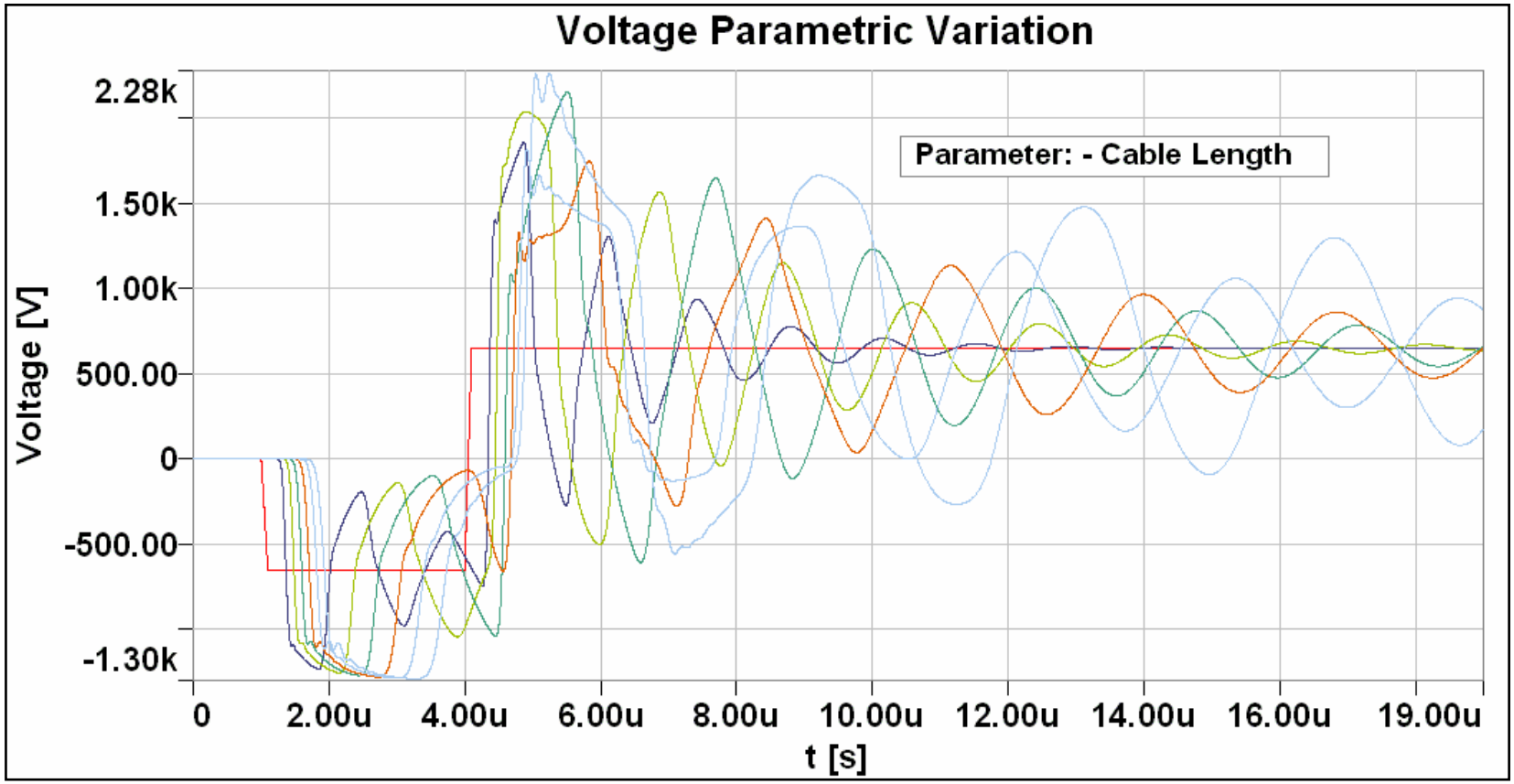
C2

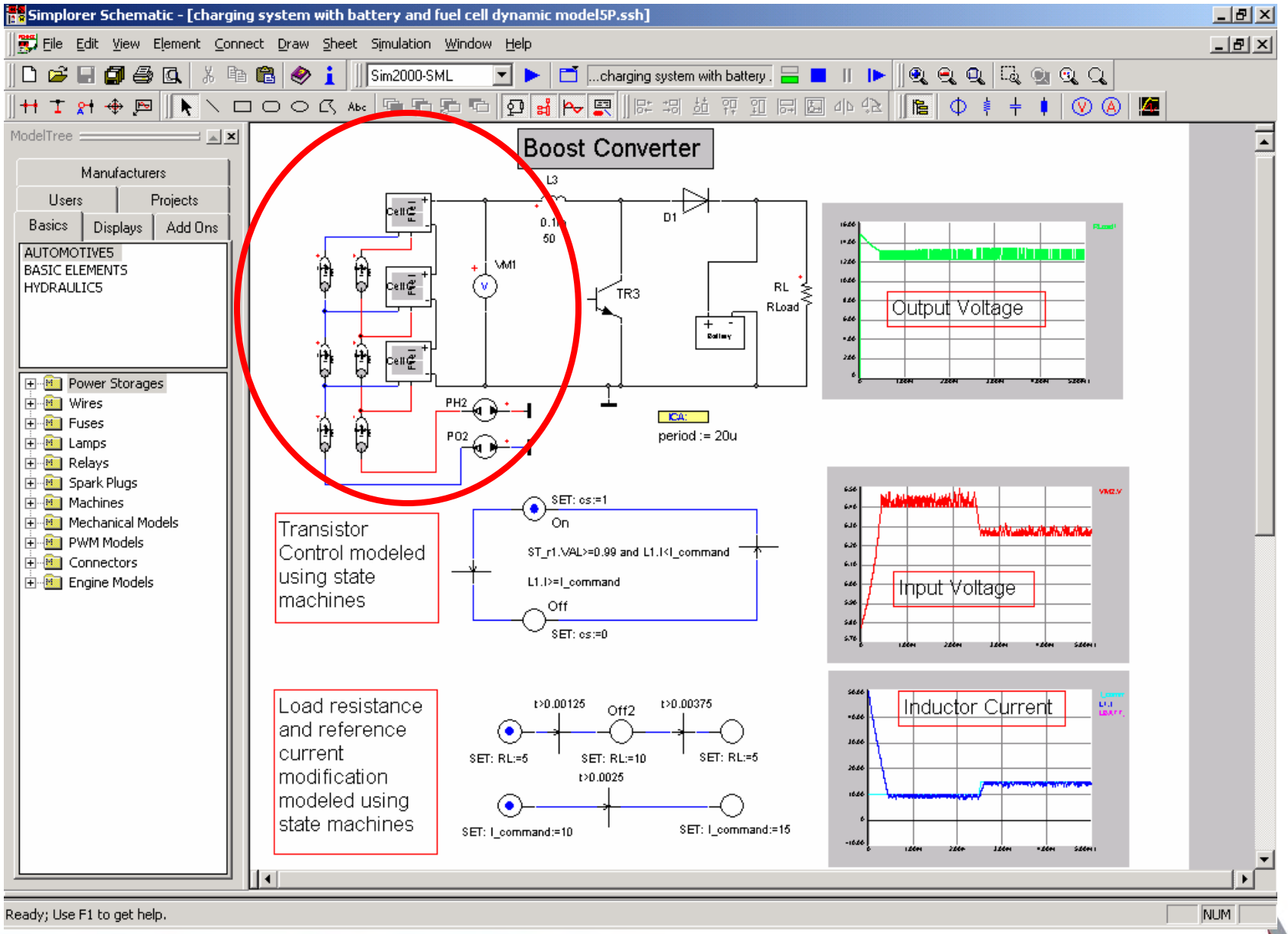
Case Study – Inverter Fed Induction Motor Design

Simulated inverter output voltage and motor terminal voltage with 35m long cable

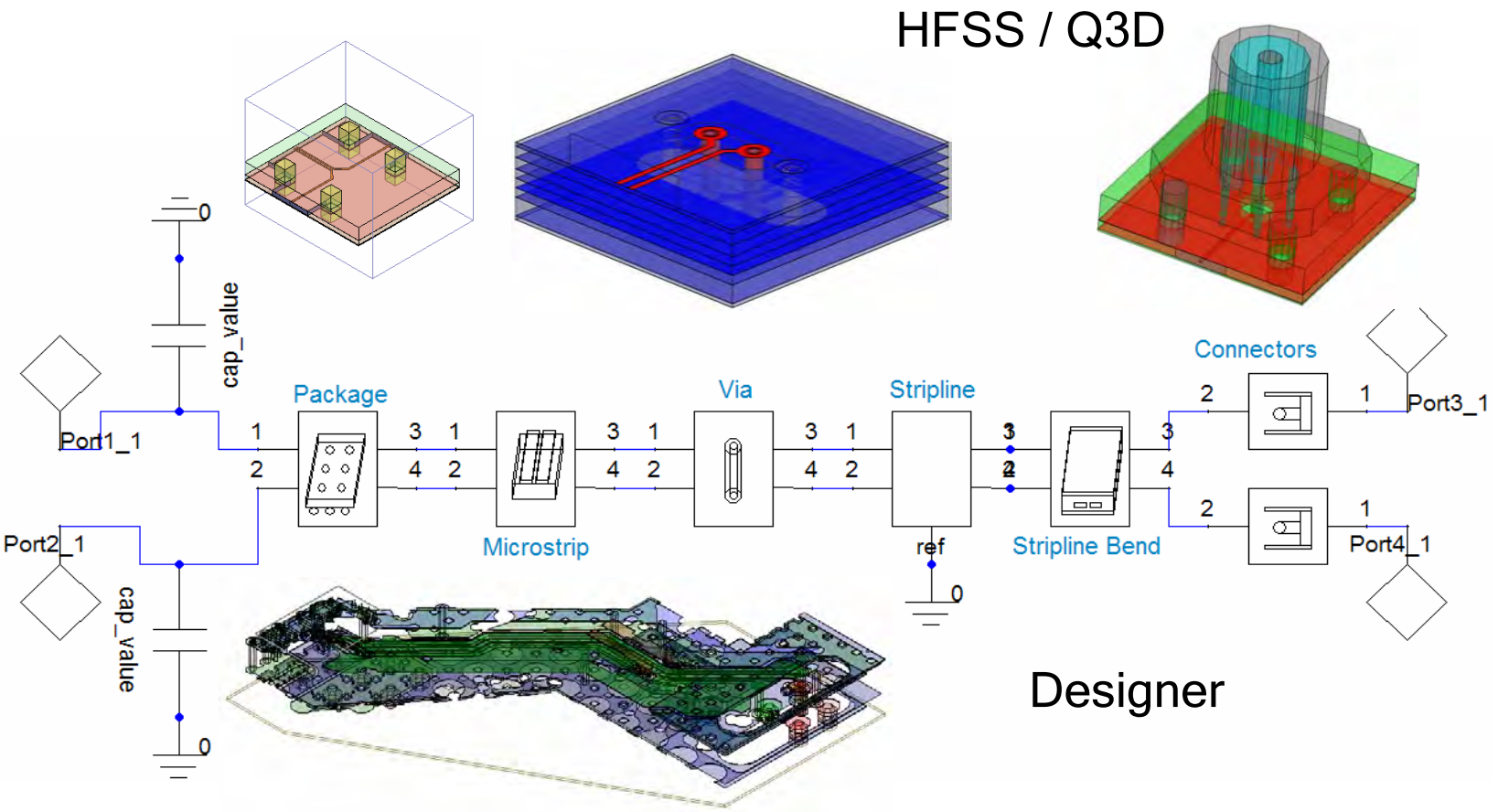


Case Study – Inverter Fed Induction Motor Design



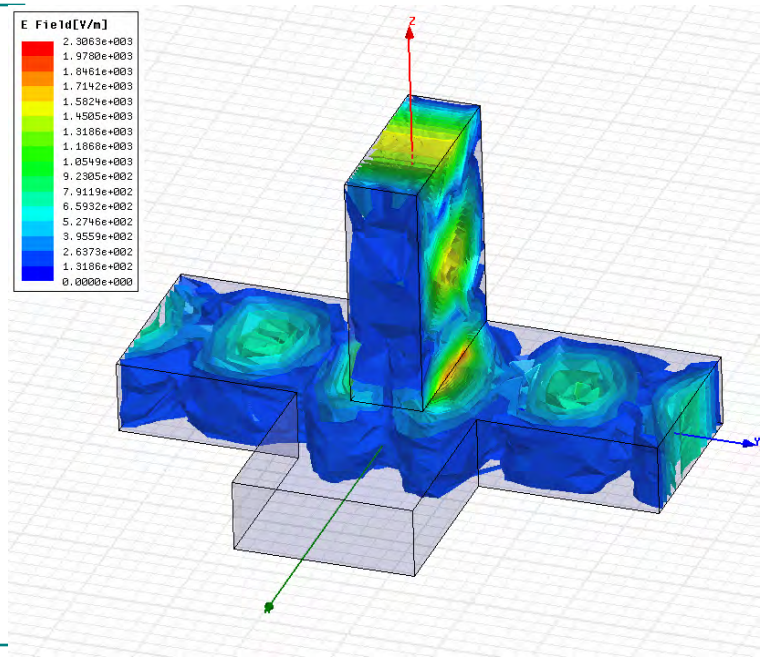

Simplorer Schematic - [charging system with battery and fuel cell dynamic model5P.ssh]
File Edit View Element Connect Draw Sheet Simulation Window Help
Sim2000-SML
...charging system with battery
ModelTree
Manufacturers
Users
Projects
Basics
Displays
Add Ons
AUTOMOTIVE5
BASIC ELEMENTS
HYDRAULIC5
Power Storages
Wires
Fuses
Lamps
Relays
Spark Plugs
Machines
Mechanical Models
PWM Models
Connectors
Engine Models
Boost Converter
L3
D1
TR3
RL
RLoad
Battery
PH2
PO2
period := 20u
Output Voltage
Input Voltage
Inductor Current
Transistor Control modeled using state machines
SET: cs:=1
On
ST_r1.VAL>=0.99 and L1.I<I_command
L1.I>=I_command
Off
SET: cs:=0
Load resistance and reference current modification modeled using state machines
t>0.00125
Off2
t>0.00375
SET: RL:=5
SET: RL:=10
SET: RL:=5
t>0.0025
SET: I_command:=10
SET: I_command:=15
Ready; Use F1 to get help.
NUM

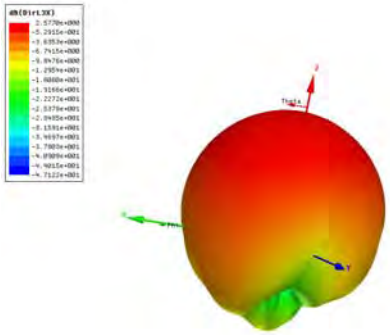
Multilevel – High Frequency

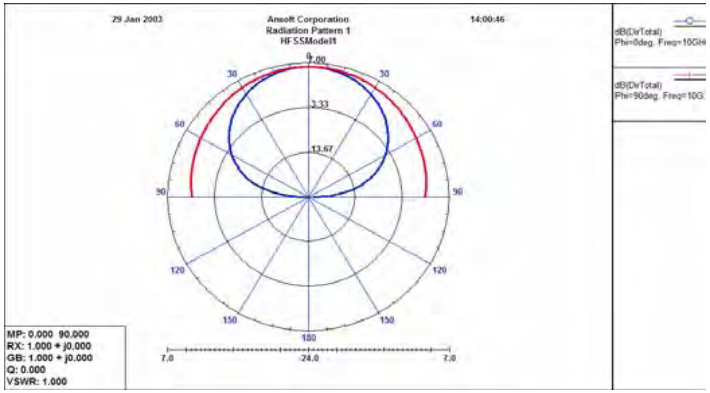


HFSS – High Frequency Structure Simulator

- Fullwave frequency-domain solution
  - Simulate arbitrary 3D-structure
  - Get radiation, S-parameters, electromagnetic fields, etc.
  - Use where wave effects are dominant

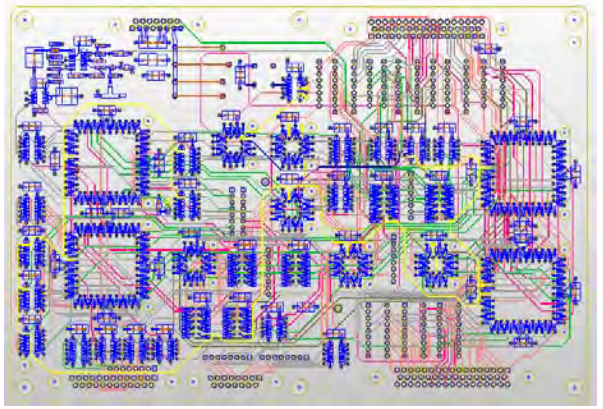


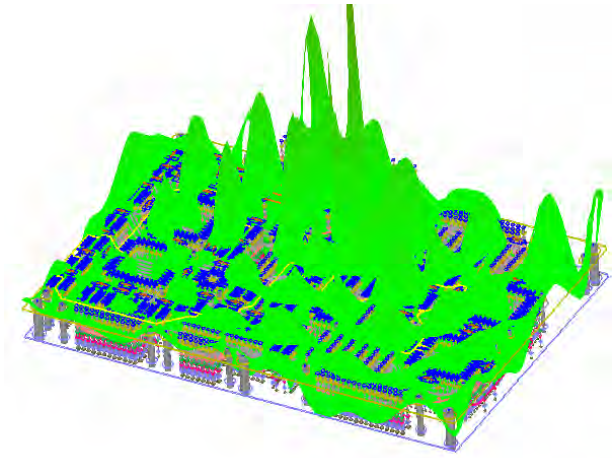




SIwave – Full Wave Signal Integrity Analysis

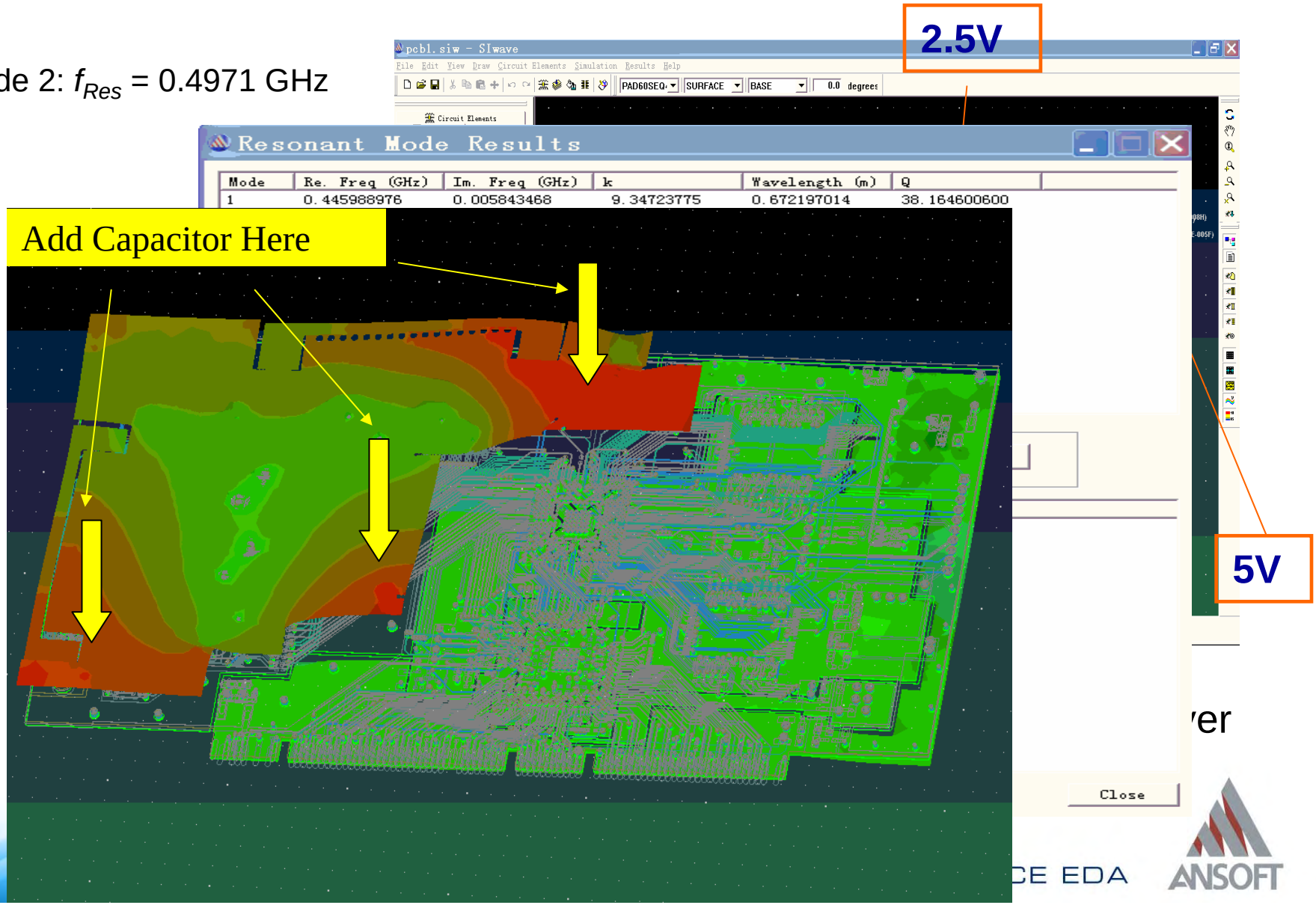
- Full wave PCB SI/PI simulation and design arbitrary 3D-structure
- Arbitrary power-ground resonances
- Signal trace full wave model extraction



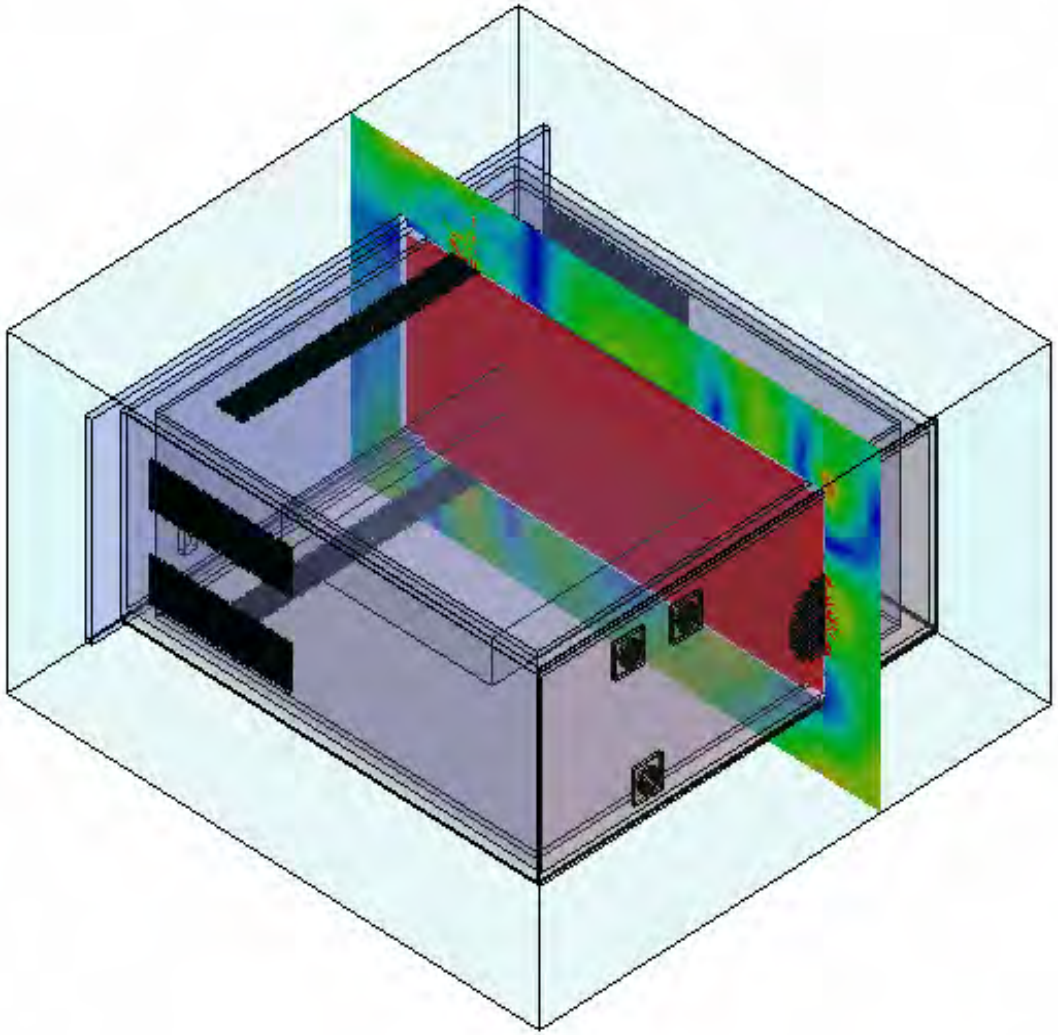


SIwave

Mode 2: f_{Res} = 0.4971 GHz

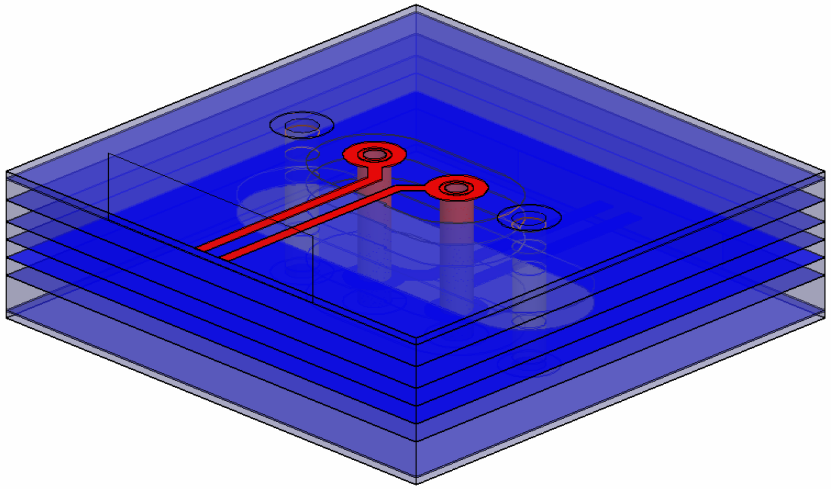


SIwave Model Embedded in HFSS

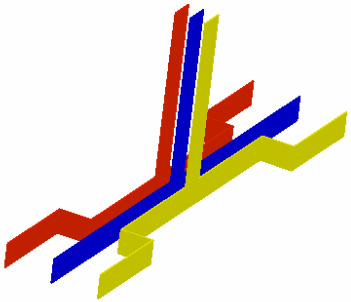


Q3D – 3D Parameter Extractor

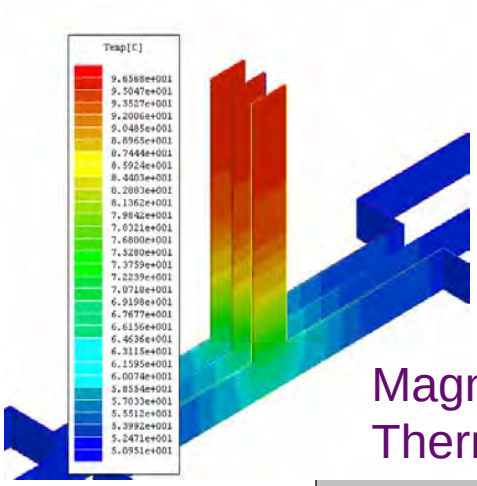
- Have a linear 3D-structure
  - Perform quasi-static electromagnetic field solutions
  - Get R-, L-, and C-matrix
  - Computes *XL* = f(*f*)
  - Creates equivalent models for Simplorer, Spice, and Designer



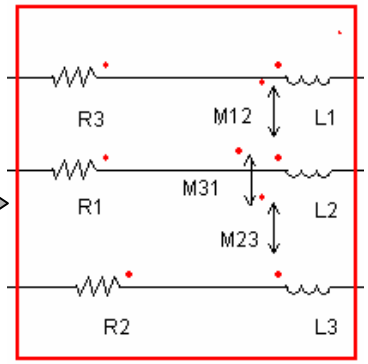
Parasitics Maxwell/Q3D – Simplorer



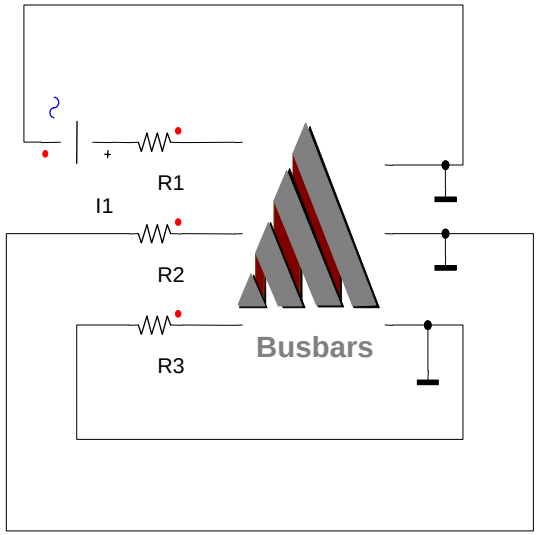
Geometry

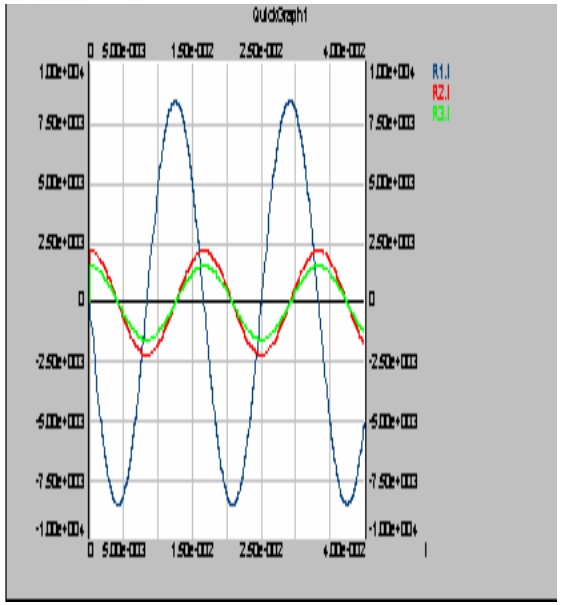


Magnetics, Thermal

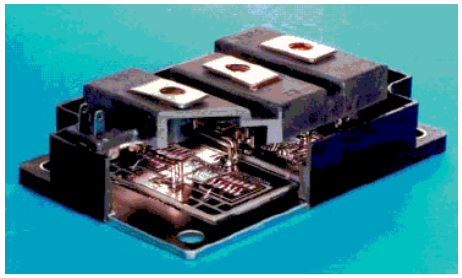


ECE

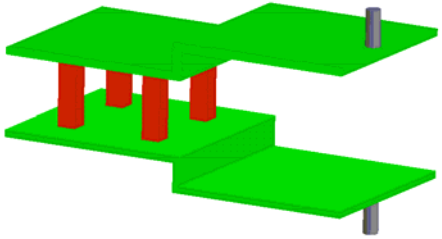




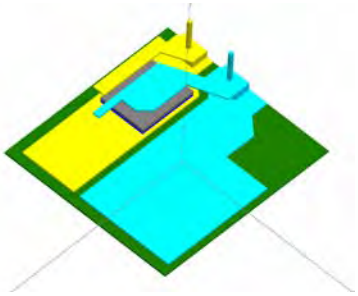
Switching Power Devices in Simplorer – An SI-Problem



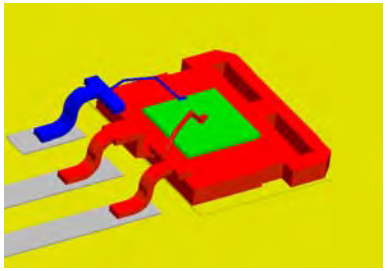
IGBT power devices



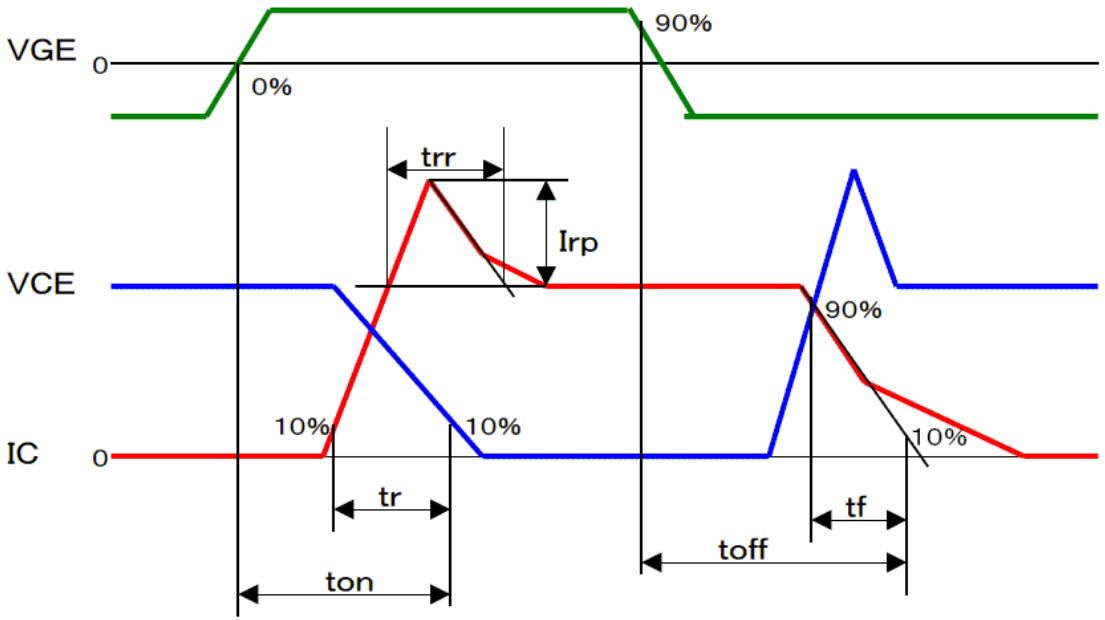
High power capacitors



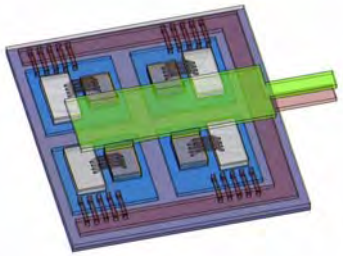
MOS transistors
(Direct bond copper substrate)



Power transistors

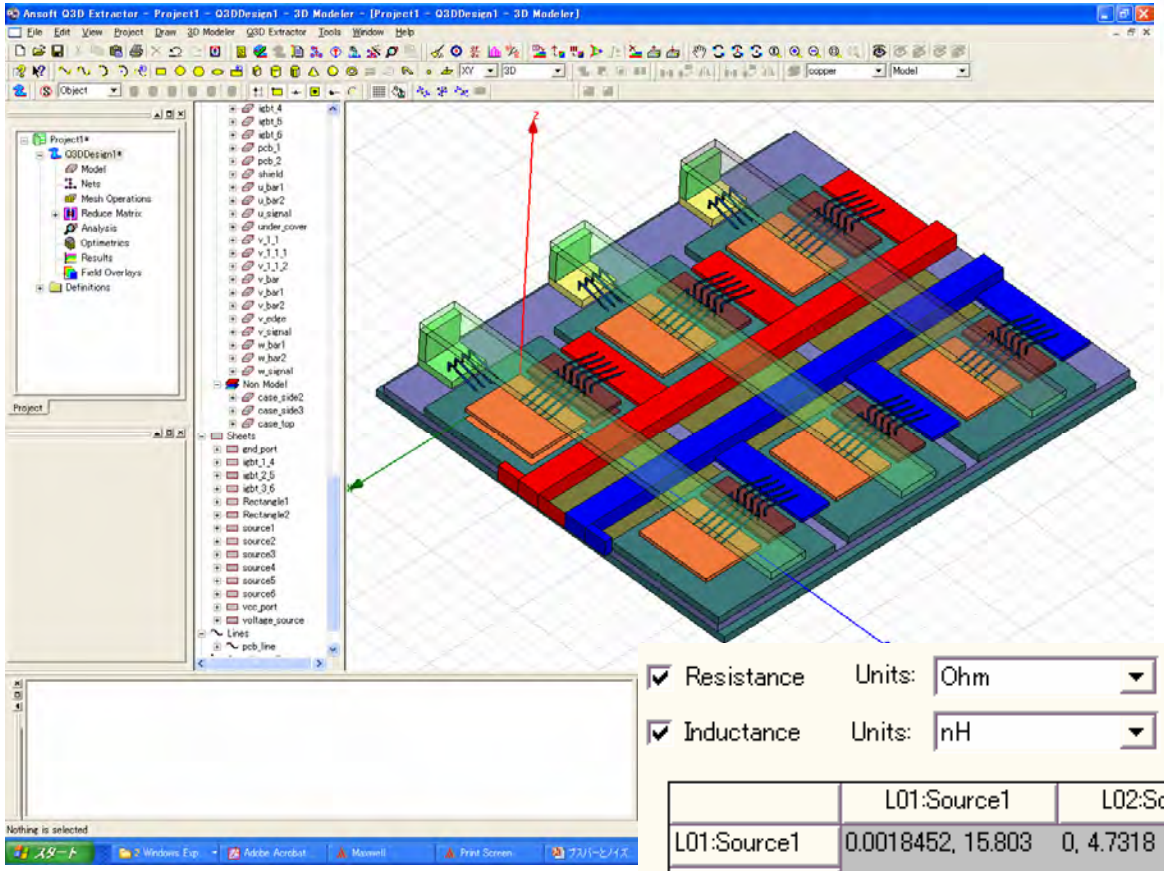


Package shape dependent electrical parasitics distort wave shape.



IGBT package

Parasitic Extraction by Q3D ...



IGBT package in Q3D

```
MODELDEF Q3DModel1
[

INTERN R R1 N1:=T21, N2:=T31 ( R:= );
INTERN R R2 N1:=T22, N2:=T32 ( R:= );
INTERN R R3 N1:=T23, N2:=T33 ( R:= );
INTERN R R4 N1:=T24, N2:=T34 ( R:= );
INTERN L L1 N1:=T31, N2:=T11 ( L:= );
INTERN L L2 N1:=T32, N2:=T12 ( L:= );
INTERN L L3 N1:=T33, N2:=T13 ( L:= );
INTERN L L4 N1:=T34, N2:=T14 ( L:= );
INTERN M K1_2 ( L1:=L1.L, L2:=L2.L, K:= );
INTERN M K1_3 ( L1:=L1.L, L2:=L3.L, K:= );
INTERN M K1_4 ( L1:=L1.L, L2:=L4.L, K:= );
INTERN M K1_5 ( L1:=L1.L, L2:=L5.L, K:= );

INTERN C C1_5 N1:=T1, N2:=T5 ( C:= );
INTERN C C1_6 N1:=T1, N2:=T6 ( C:= );
INTERN C C1_7 N1:=T1, N2:=T7 ( C:= );
INTERN C C1_8 N1:=T1, N2:=T8 ( C:= );
```

R

L

M

C

Simplorer model (R, L, M, C)

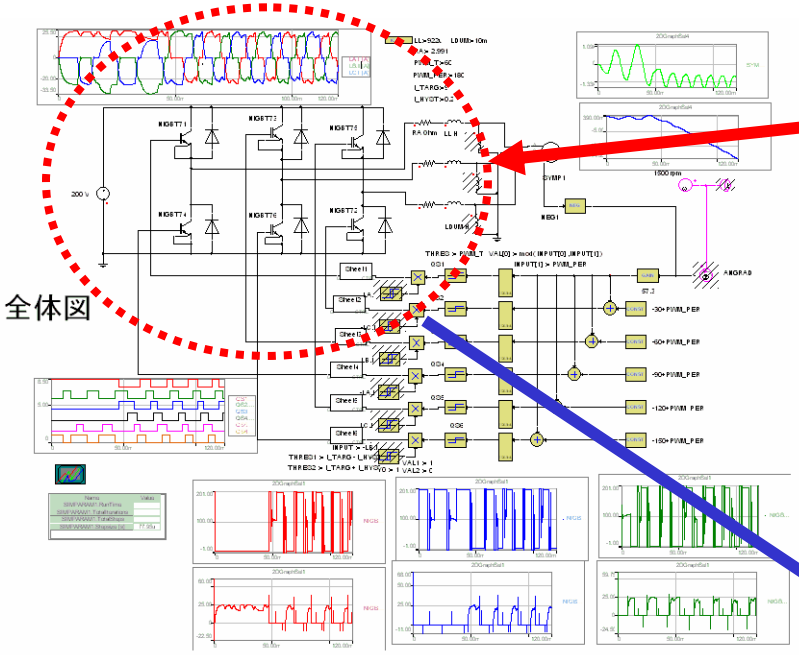
Extracted RL matrix

Resistance Units: Ohm Coupling Coefficient Export...

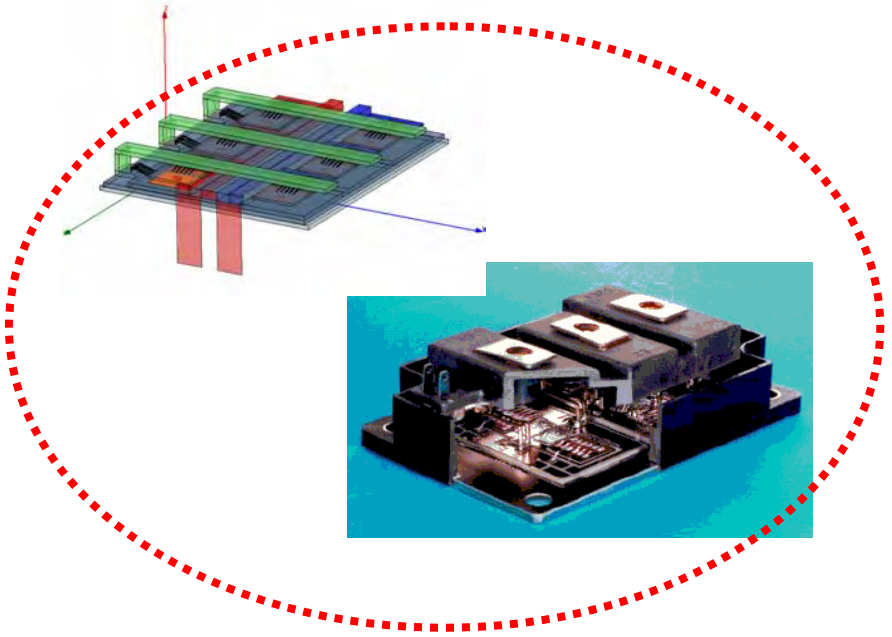
Inductance Units: nH Original

	L01:Source1	L02:Source2	L03:Source3	L04:Source4
L01:Source1	0.0018452, 15.803	0, 4.7318	0, 2.7524	0, 1.9312
L02:Source2	0, 4.7318	0.0018896, 15.745	0, 4.599	0, 2.748
L03:Source3	0, 2.7524	0, 4.599	0.0018967, 15.796	0, 4.7331
L04:Source4	0, 1.9312	0, 2.748	0, 4.7331	0.00185, 15.807

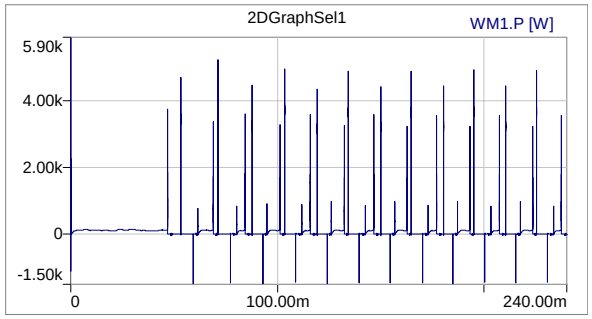
... and Automatic Parametrization of Simplorer Model



全体図

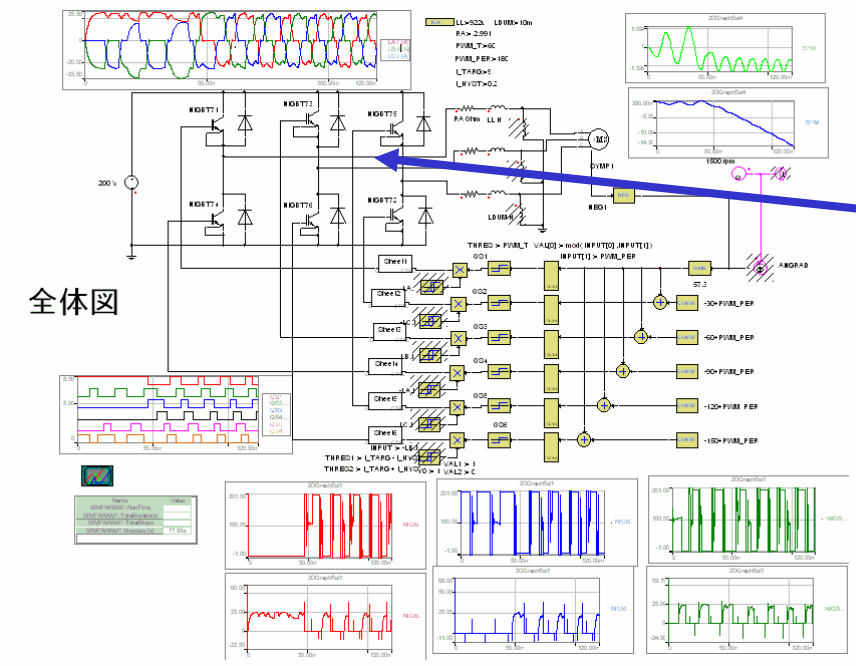


Realistic waveforms



Inverter Package – EMI Analysis Flow

Simplorer



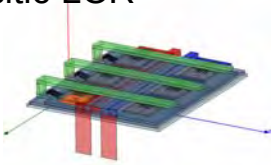
全体図

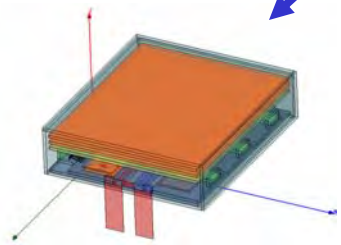
SI analysis:
- Parameter extraction
- Fullwave model

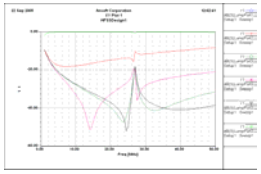
HF Electromagnetics
HFSS, (SIwave)

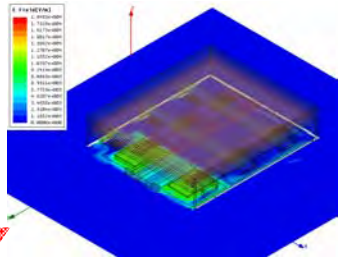
External
data import

Parasitic LCR



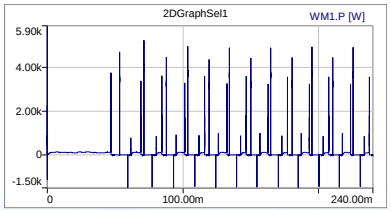




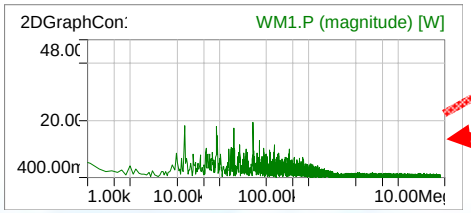


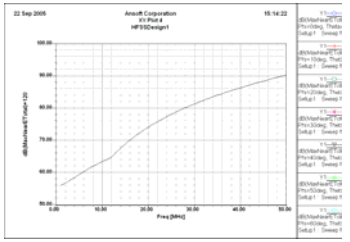
HF Electromagnetics
HFSS, (SIwave)

Currents, voltages



Spectra





MagE

Thank You.

The *SystemVision*™ *System Modeling Solution*

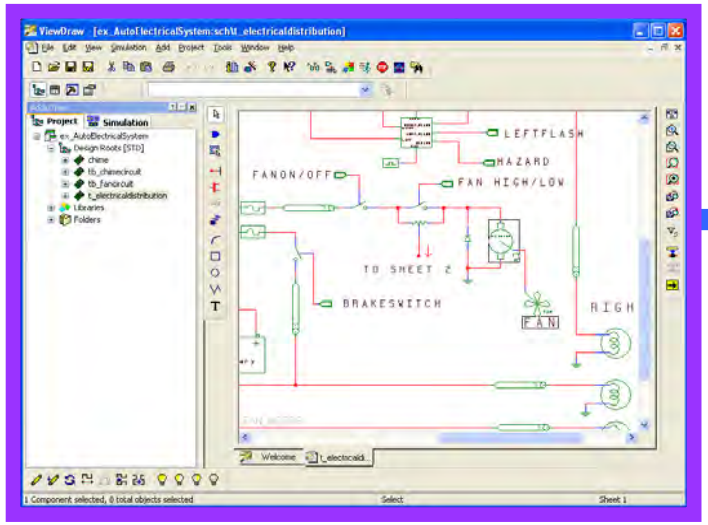
Thomas Heurung
European Product Specialist

Christian Lohner
Application Engineer

ASIM February 2006

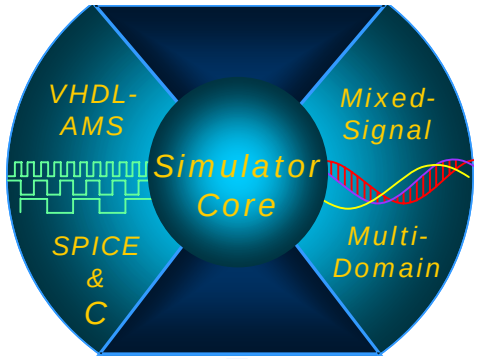
SystemVision Components

Graphical Design



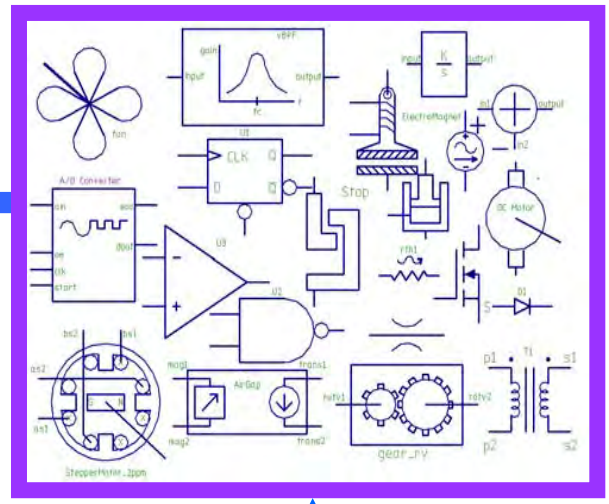
System Model

Component Models

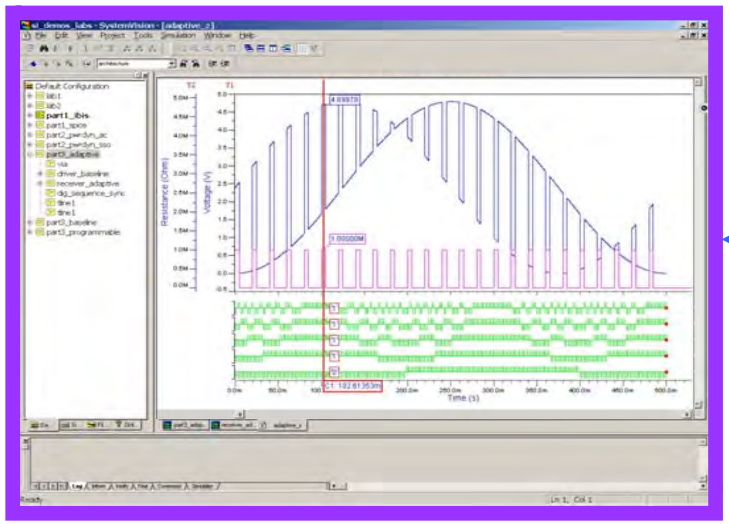


Results

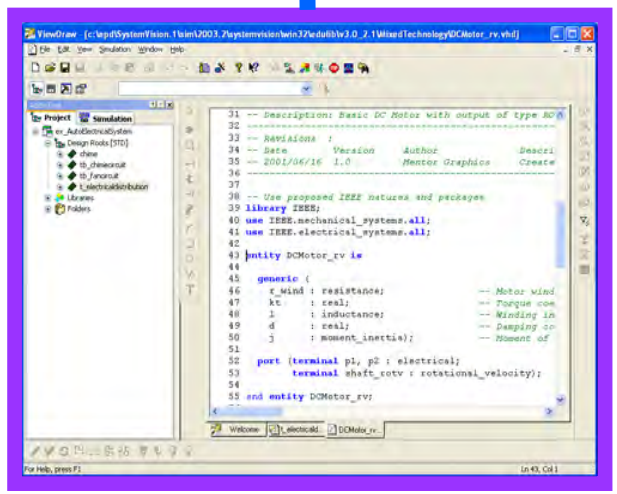
Model Libraries



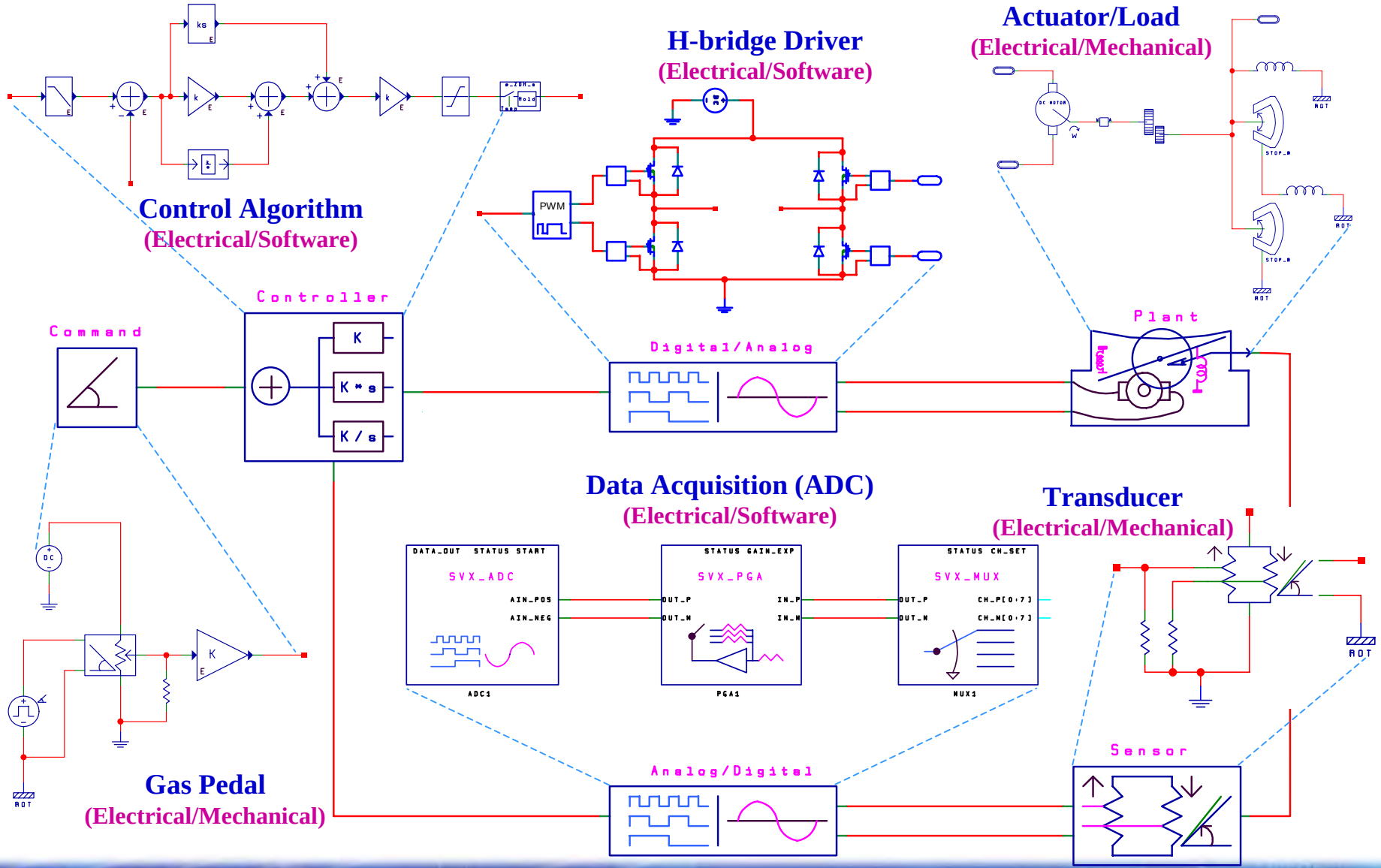
Data Analysis & Viewing



Model Creation



Electronic Throttle Control Embedded Control System
Implemented in SystemVision



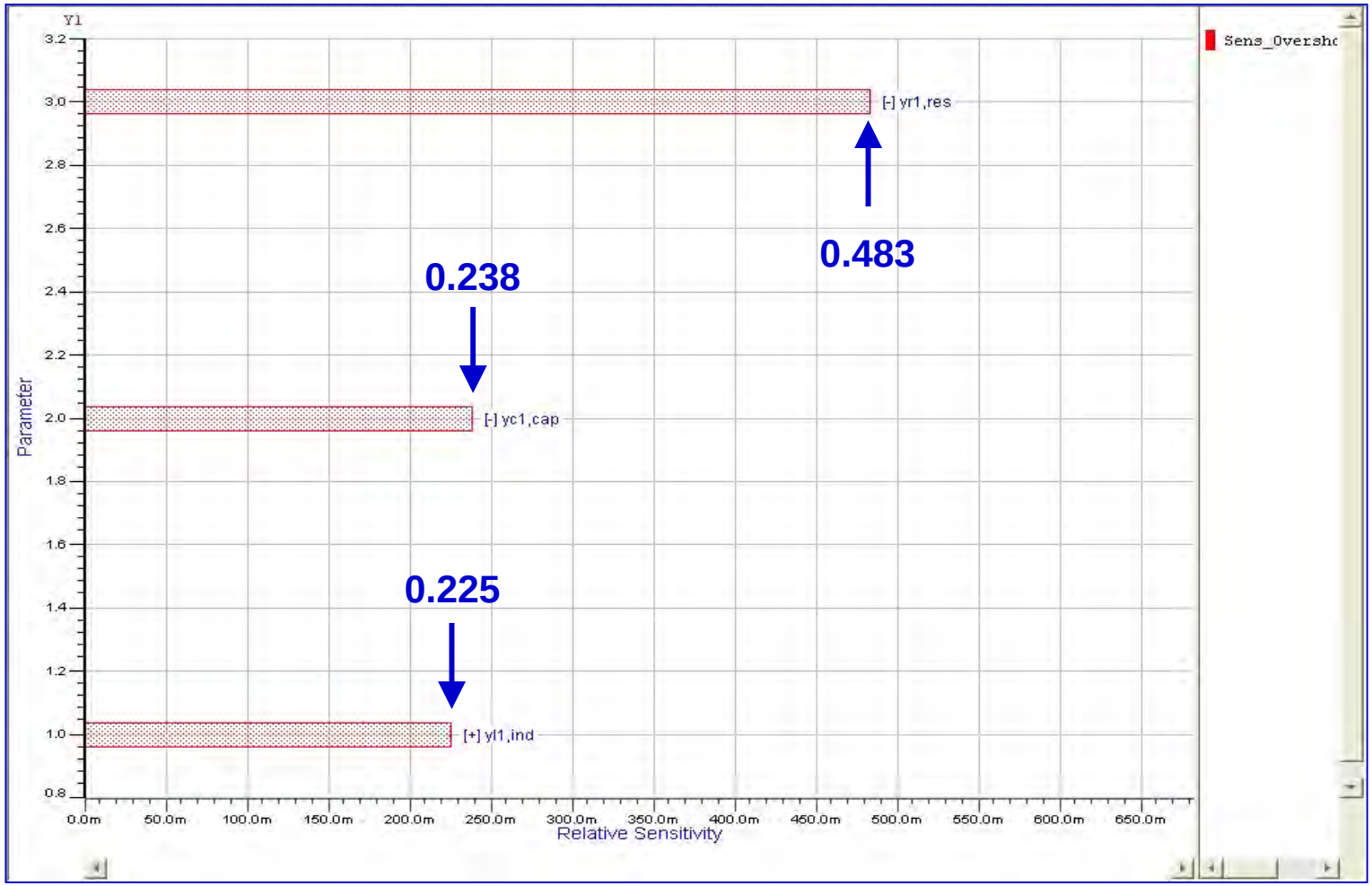
SystemVision Simulation Capabilities

- **Electrical (analog + behavioral)**
  - ***VHDL-AMS alone or mixed with SPICE***
- **Mixed-Signal**
  - ***Analog/Digital electronics***
- **Mixed-Technology**
  - ***Motors with drivers and mechanical loads***
  - ***Electro-magnetics (ex: solenoid)***
  - ***Electro-thermal (ex: self-heating devices)***
  - ***Electro-hydraulics (ex: actuator)***
- **Control Systems**
  - ***S and Z-domain modeling***
- **DC, Time Domain, Frequency Domain analyses**
  - ***Parametric variations, Monte Carlo Analysis***

SystemVision Parametric Analyses

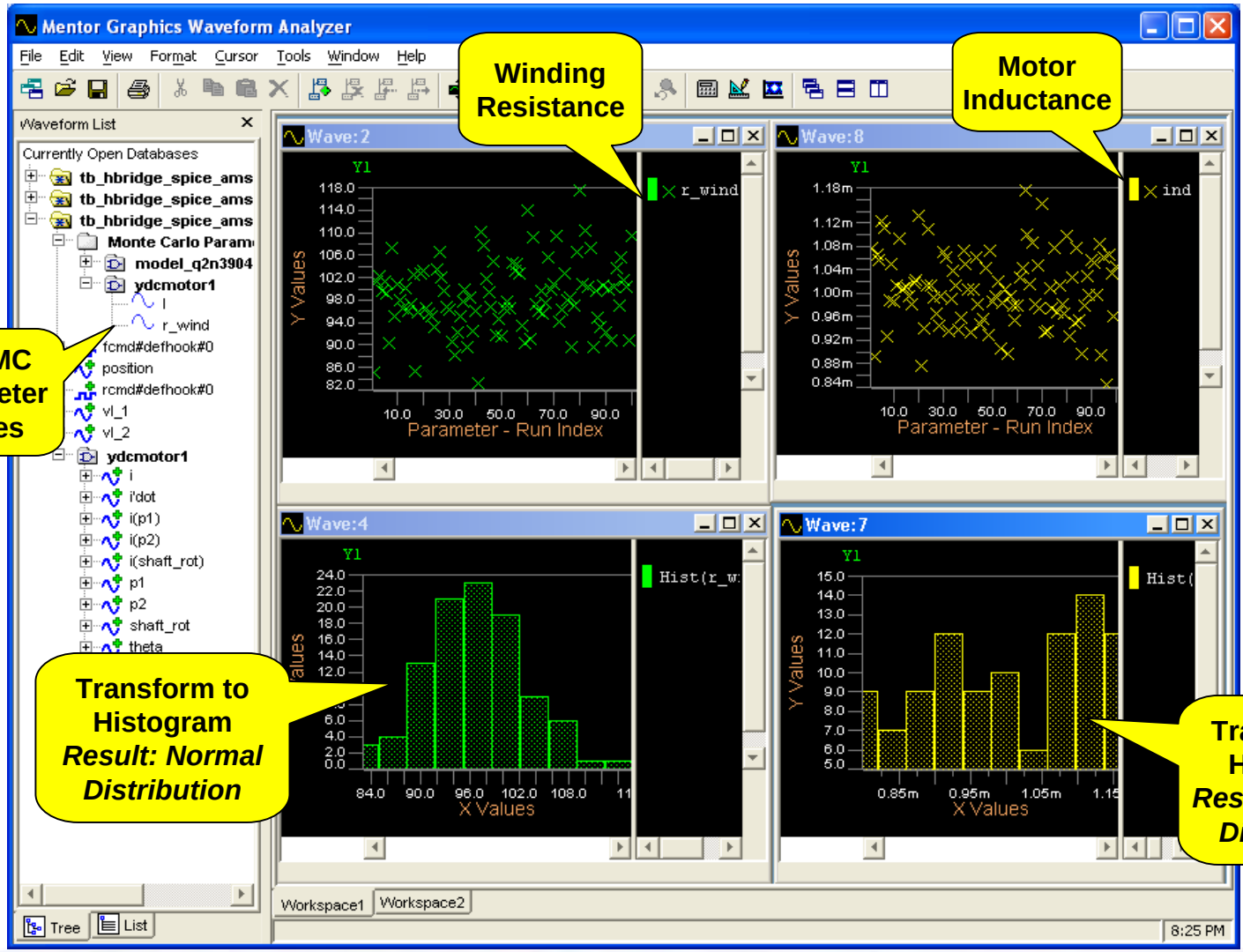
- **Parameter Sweeps**
  - ***DC Sweeps***
  - ***Time/Frequency-domain Sweeps***
- **Sensitivity Analysis**
  - ***Calculation of relative sensitivity of performance measures to component parameters***
- **Monte Carlo Analysis**
  - ***Nominal value, Tolerance and Distribution assigned for each parameter***
  - ***VHDL-AMS or SPICE parameters***
  - ***Powerful post-processing in Waveform Viewer***
    - Transforms data into information
- **Worst-Case Analysis (4.3)**

Sensitivity Bar Indicators



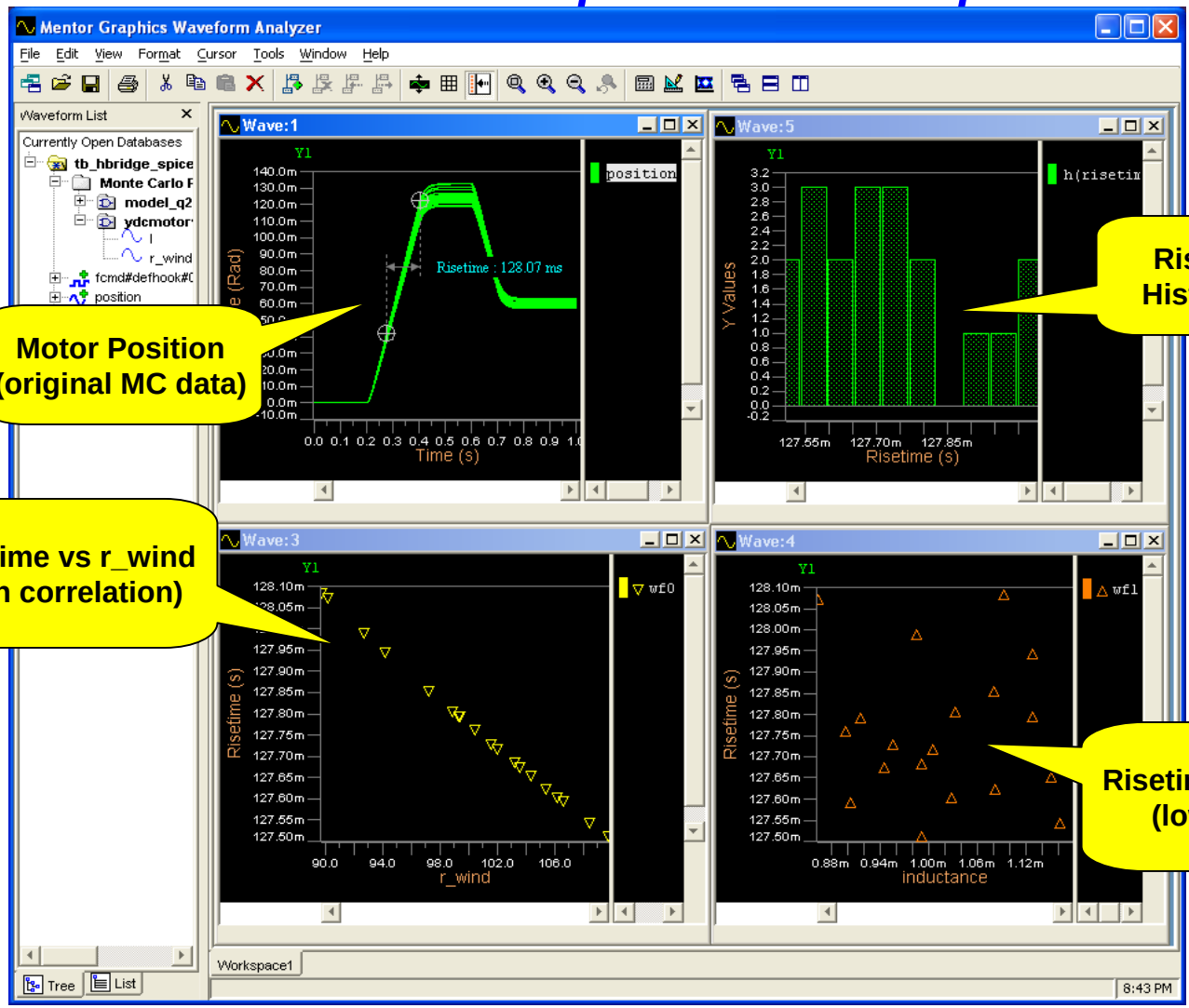
Post-Processing Monte Carlo Results

Calculator Example: Histograms



Post-Processing Monte Carlo Results

Calculator Example: Correlation plots



Motor Position (original MC data)

Risetime Histogram

Risetime vs r_wind (high correlation)

Risetime vs inductance (low correlation)

Stress Analysis

Three mechanisms for Stress Analysis:

- **Stress measures can be built directly into VHDL-AMS models. Models can output relevant data.**
- **Can use “stress monitor” models that are “attached” to various points in a circuit to measure stress based on pin data (can be used for SPICE or VHDL-AMS models).**
- **Can use SystemVision scripting capability to automatically populate a spreadsheet.**

SystemVision Scripting

- **Features**
  - ***Programmatic automation of netlisting, compilation, simulation, and waveform analysis, via windows COM objects***
  - ***Documented COM objects are programmable via VBScript, VB, C++, others.***
- **Benefits**
  - ***Automation of frequently used tasks***
  - ***Customization of SystemVision functionality***
  - ***Provides access via widely used languages***
  - ***Extends existing DxDesigner ActiveX controls used for automation of schematic functionality***

Waveform Analyzer

- **Powerful Post-processing Capabilities**
  - ***Transforms data into information***
- **Expanded Measurement set**
  - ***23 measurements***
  - ***Includes statistical measures***
- **Waveform Calculator**
  - ***Equation Stack***
  - ***Arithmetic, trigonometric, calculus, and complex functions***

VHDL-AMS / SPICE Models

Modelling Capabilities

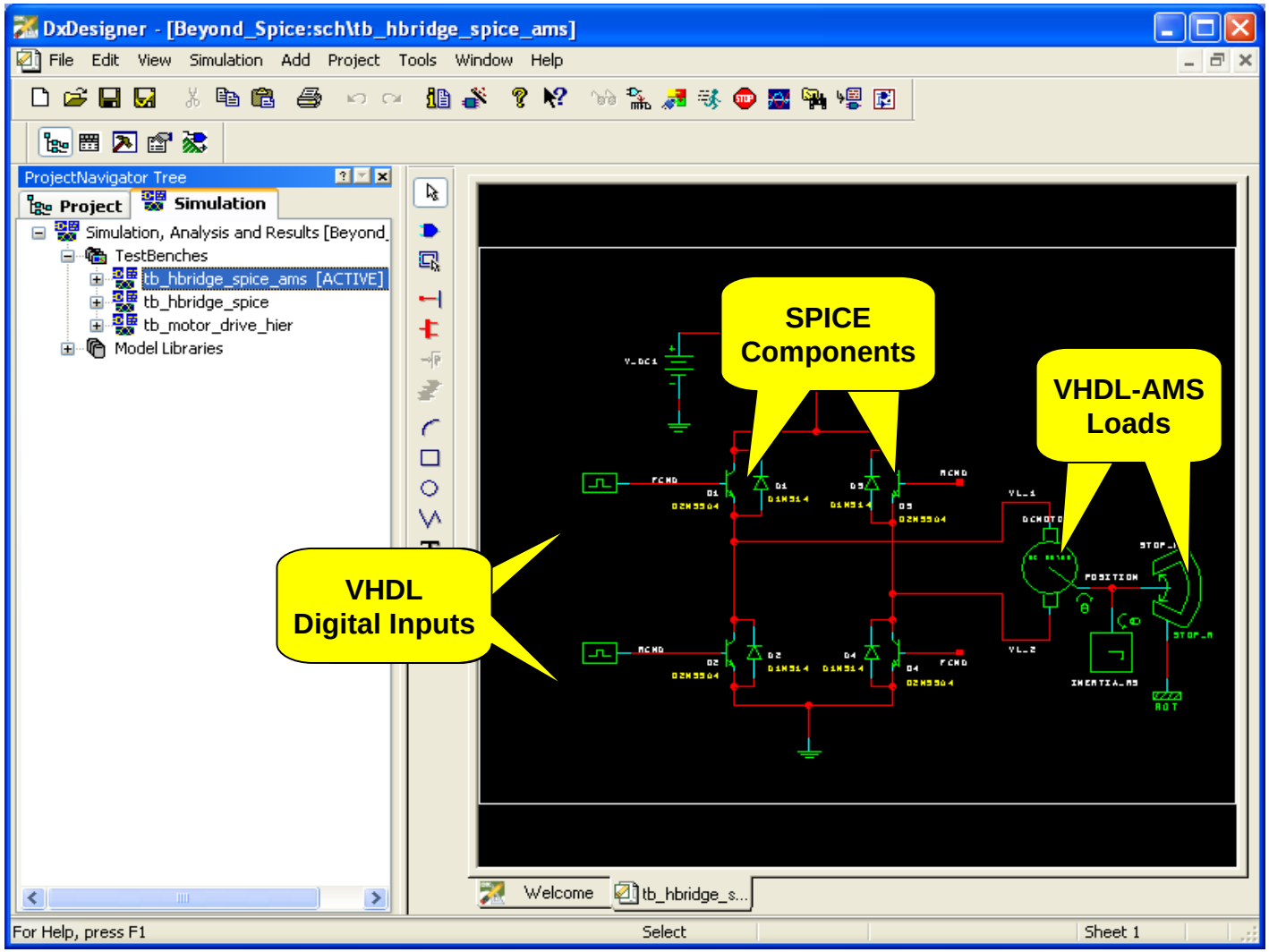
VHDLAMS and SPICE

- **Large SPICE component library**
  - ***>18,000 devices***
- **VHDL library for compatibility with PSPICE digital models**
  - ***63 VHDL models***
- **Libraries of models supplied by VDA–AK30**
  - **Cooperative model exchange between the automotive OEMs and Tier 1 suppliers**
- **VHDL-AMS Mixedsignal/Mixedtechnology Educational-Library**

Modelling Capabilities

- **PSPICE Library Conversion Utility**
  - ***Makes PSPICE models compatible with SystemVision SPICE format***
- **Existing Spicemodels can be easily attached and re-used**
- **VHDL-AMS Model Wizard**
  - ***Generates models and symbols automatically***
  - **Provides a structured GUI that eliminates the need to remember language syntax**
  - **Generates both the correct-by-construction code and a Dx symbol**
  - **Includes a powerful capability to import tables of data and automatically generate models**

Combining SPICE and VHDL-AMS Models



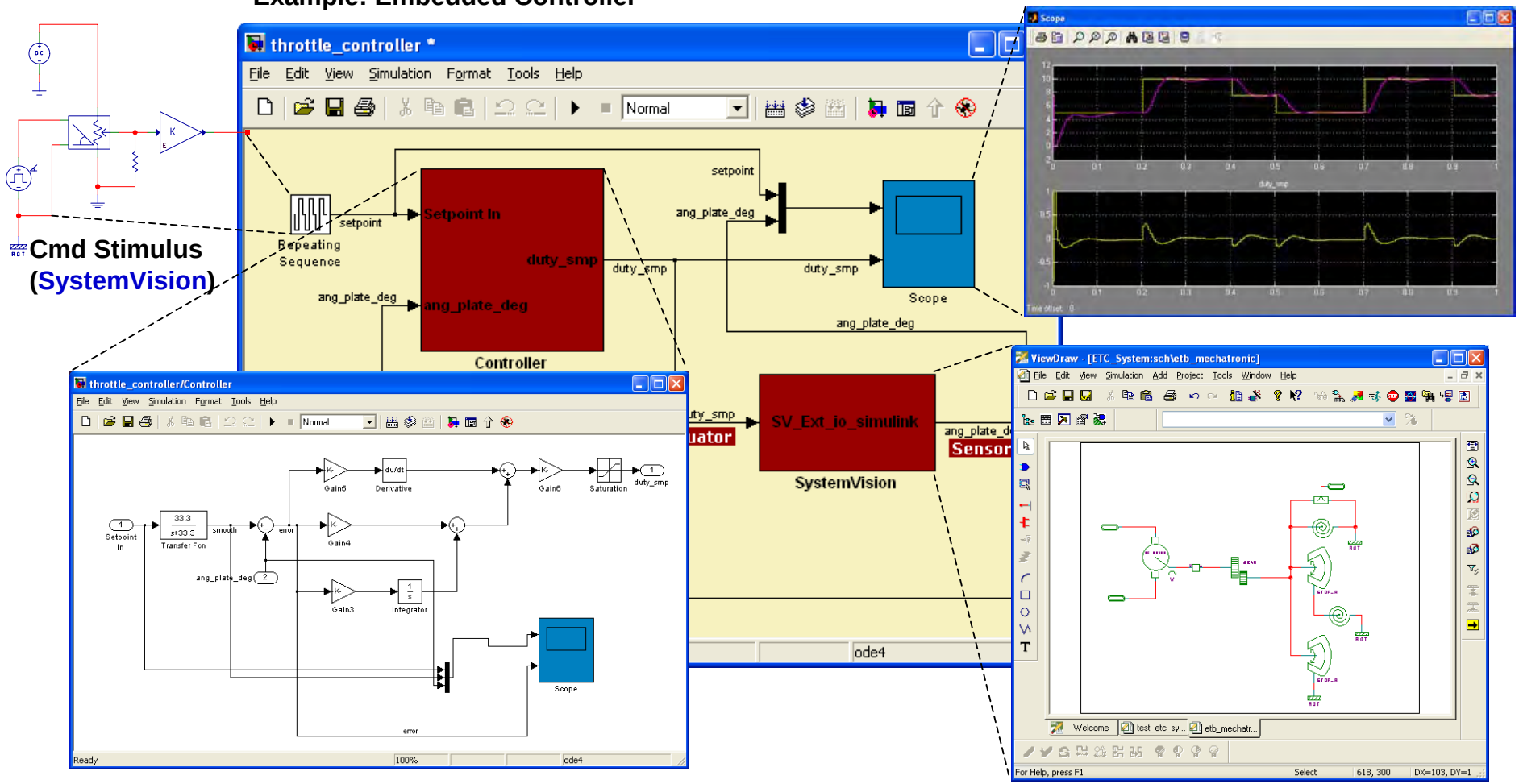
SystemVision and ModelSim

- **Modelsim SE&PE seamless integrated**
  - **Optional**
  - **Switching between the ModelSim and ADMS compiler as easy as checking/unchecking a box on a dialog**
  - **All other details are managed for the user**
- **Integration of HDL Designer**
  - **Already part of the DxDesigner environment**
  - **Also supported in SystemVision**

The Best of Both Worlds:
SystemVision and Simulink

Example: Embedded Controller

Simulation Results (SL or SV)



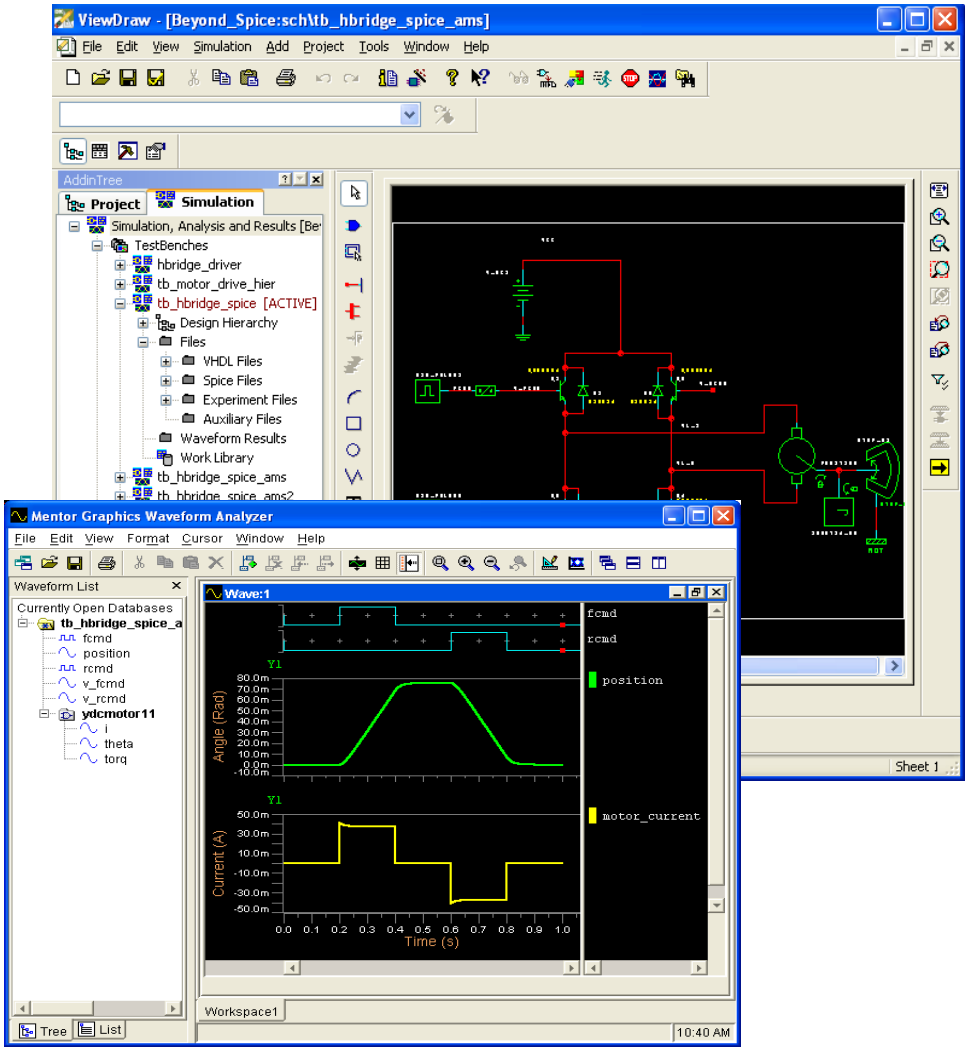
Cmd Stimulus (SystemVision)

Control Algorithm (Simulink)

Actuator/Load Mechanics (SystemVision)

SystemVision Summary

- Integration with DxDesigner
- Simulation Tab added for quick access to files
- Driven by Spice or VHDL-AMS netlist “on top”
- Easily add symbol and model libraries
- Drag and drop model files into schematic from Windows Explorer
- Intuitive waveform viewer with built-in measurements
- Modelsim
- Simulink



SystemVision Summary

- **Further information on :**
  - ***http://www.mentor.com/system***
    - With Access to the educational version
    - For requesting an evaluation license
    - For online demonstration of mixed technology system
    - For downloading examples


Thank you
Mentor Graphics®

Thermische Modellierung elektrischer Leitungen mit Computer-Algebra-Simulation und VHDL-AMS – Theoretische Grundlagen und praktische Anwendungen

Stefan Braun, SmartCAE München; Manfred Klinkenberg, Yazaki Europe Ltd. Köln

Inhalt

- Allgemeines zur Dimensionierung von Fahrzeugleitungen
- Entwicklung, Test und Verifikation von Fahrzeugleitungen durch Simulation
- Ein Beispiel
- Zusammenfassung

Allgemeines zur Dimensionierung von Fahrzeugleitungen

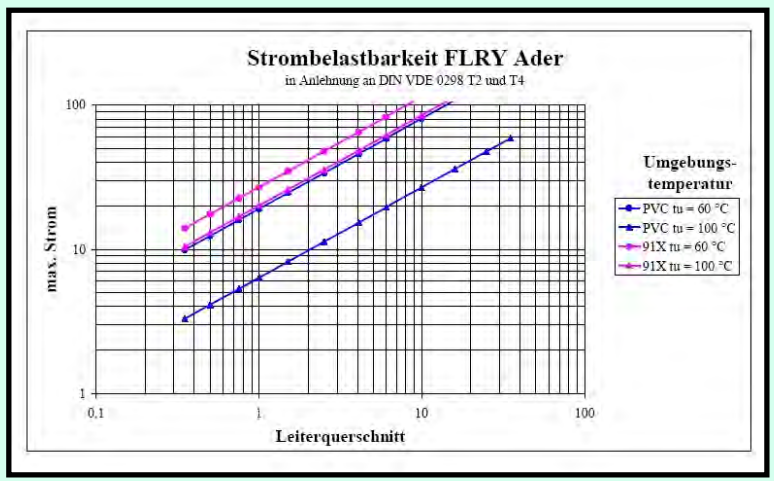
Allgemeines zur Dimensionierung von Fahrzeugleitungen

Festlegen des Ltg.-querschnittes

- **Ermittelung des Verbraucherstromes I [A]:**
  I = U/R [A] oder I = P/U [A]
- **Berechnen des Leiterquerschnittes A [mm²]:**
  A = I*Rho*l/Uvl [mm²]

A_nenn-Leiter [mm²]	Drahtan-zahl	R_max/ Meter [mohm/m]	d_max-Leiter [mm]	Isolations-dicke [mm]	d_außen_ max Leitung [mm]
0.35	12	52	0.9	0.25	1.4
0.5	16	37.1	1	0.3	1.6
...					
95	475	0.196	14.8	1.6	18.0
120	608	0.153	16.5	1.6	19.7

- **Auswahl des Leiterquerschnittes A [mm²]:**
  Siehe Tabelle.
- **Tatsächlicher Spannungsabfall Uvl ermitteln [V]:**
  Uvl = I*Rho*l/A [V]
- **Überprüfen der Stromdichte S [A/mm²]:**
  S = I/A [A/mm²]; Siehe Diagram.



Allgemeines zur Dimensionierung von Fahrzeugleitungen

Ltg. nach ISO 6722:2002(E)

- **FLRY – Leitungen:**
  - Einadrige, unverzinnte, PVC-isolierte Leitungen mit reduzierter Wandstärke
  - FL: Niederspannungsleitung
  - R: Wanddicke der Isolierung reduziert
  - Y: PVC – Polyvinylchlorid
  - Temperaturbereich: -40°C bis 105°C

A_nenn-Leiter [mm²]	Drahtan-zahl	R_max/ Meter [mohm/m]	d_max-Leiter [mm]	Isolations-dicke [mm]	d_außen_ max Leitung [mm]
0.22	12	84.4	0.7	0.25	1.2
0.35	12	52	0.9	0.25	1.4
...					
16	126	1.16	6.3	1.0	8.3
25	196	0.743	7.8	1.3	10.4

- **FLY – Leitungen:**
  - Einadrige, unverzinnte, PVC-isolierte Leitungen mit normaler Wandstärke
  - FL: Niederspannungsleitung
  - Y: PVC – Polyvinylchlorid
  - Temperaturbereich: -25°C bis 90°C

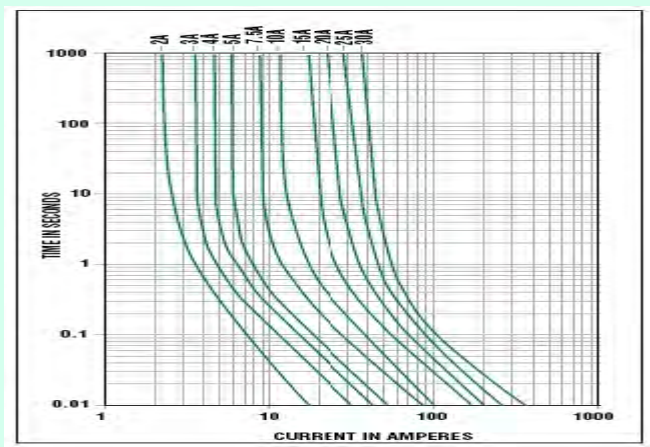
A_nenn-Leiter [mm²]	Drahtan-zahl	R_max/ Meter [mohm/m]	d_max-Leiter [mm]	Isolations-dicke [mm]	d_außen_ max Leitung [mm]
0.5	16	37.1	1.1	0.6	2.3
0.75	24	24.7	1.3	0.6	2.5
...					
95	475	0.196	14.8	1.6	18.0
120	608	0.153	16.5	1.6	19.7

Allgemeines zur Dimensionierung von Fahrzeugleitungen

Absicherung

- In Fahrzeugen werden, mit nur wenigen Ausnahmen, alle Fahrzeugleitungen abgesichert.
- Eine solche Ausnahme können beispielsweise Starterleitungen sein.
- Absicherung erfolgt durch Schmelz- oder elektronische Sicherungen.
- Der maximaler Dauerstrom am Verbraucher darf nicht größer sein als 80% des Nennstromes der Sicherung.

Leitungs-querschnitt [mm²]	Sicherungs-nennwert [A]	Maximaler Dauerstrom [A]
0.35	5	4
0.5	7.5	6
0.75	10	8
1	15	12
1.5	20	16
2.5	30	24



Entwicklung, Test und Verifikation von Fahrzeugleitungen durch Simulation

Entwicklung, Test und Verifikation von Fahrzeugleitungen durch Simulation

VDA Verband der Automobilindustrie

FAT AK 30

Allgemein:

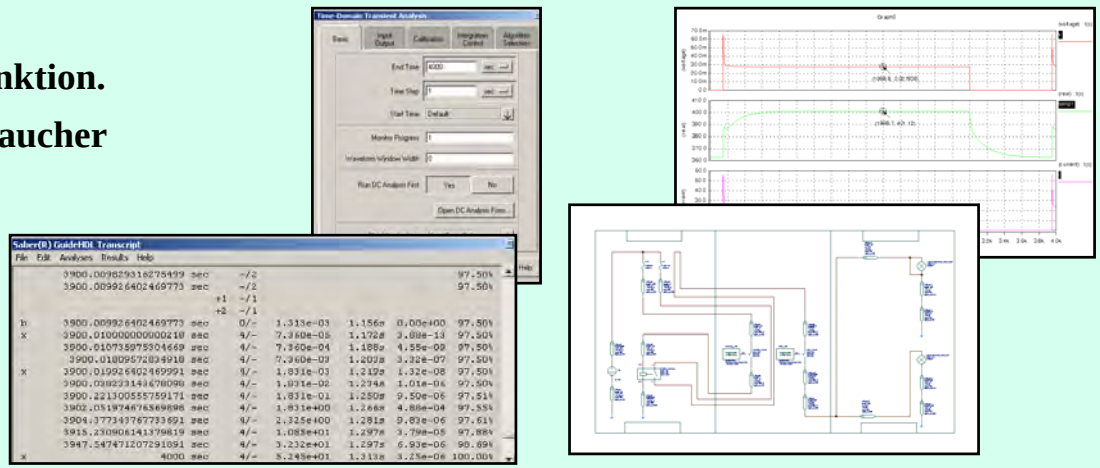
- **Entwicklung, Test und Verifikation von Fahrzeugleitungen durch Simulation ist die Möglichkeit, die Dimensionierung, Absicherung und Funktion eines elektrischen Stromkreises im Fahrzeug in Abhängigkeit seiner Last, Umgebungstemperatur und Lage im Fahrzeug unter Berücksichtigung bestimmter Betriebszustände, mit Hilfe von Simulation zu betrachten.**

Betriebszustände:

- **Normalbetrieb: Ein-/Ausschaltfunktion.**
- **Fehlerfall: Kurzschluß am Verbraucher**

Art der Überpüfung:

- **Stromfluß**
- **Spannungsabfall**
- **Temperaturentwicklung**



Anwendungsbereiche:

- **Leitungsbezogene Entwicklungsanteile von neuen Fahrzeugprogrammen.**
- **Leitungsbezogene Design-Verifikatonen bei Änderungen in laufenden Fahrzeugprogrammen.**
- **Leitungsbezogene, allgemeine Betrachtungen und Entwicklungen.**

Ein Beispiel

Ein Beispiel (1)

Aufgabe:

- **Überprüfen der Funktion/Leitungsauslegung eines Begrenzungslichtstromkreises im Normalbetrieb und Fehlerfall**

Verwendete Software-Tools und Bibliotheken:

- **Systemzeichen-Software: SaberSketch, Version 2005.09**
- **Simulation, Analyse und Anzeige: SaberHDL, Version 2005.09**
- **Modellbibliotheken: Automotive_vda, Fundamentals_vda und spice2vhd des VDA/FAT-AK30**
- **Symbolbibliothek: SaberSketch, Version 2005.09**

Verwendete VHDL-AMS Modelle:

- **Wire_vda.vhd – Einadriges Fahrzeugleitungsmodell**
- **Fuse_vda.vhd – Sicherungsmodell**
- **Switch_1pno_vda.vhd - Schaltermodell**
- **Swdrv_prog_vda.vhd – Positiongeber-Modell für Schalter**
- **Relay_1pno_vda.vhd – Relaismodell**
- **Bulb_1f_vda.vhd – 1-Fadenlampenmodell**
- **Vdc.vhd – Modell Gleichspannungsquelle (Batterie)**

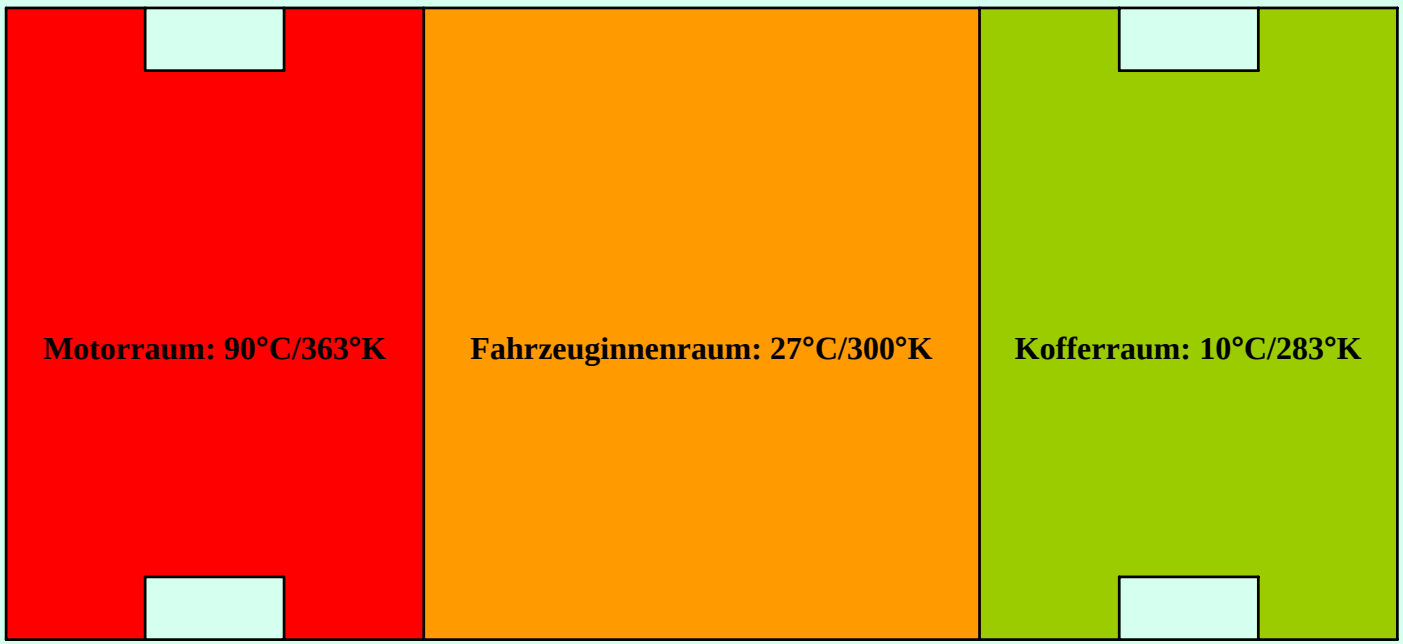
Ein Beispiel (2)

Technische Details:

- **Leitungsaufbau und -dimensionierung richten sich nach dem internationalem Standard ISO 6722:2002(E): Road vehicles - 60V and 600Vsingle-core wire - Dimensions, test methods and requirements**
- **FLRY – B Leitungen (-40°C bis 105°C) Motorraum**
- **FLY – Leitungen (-25°C bis 90°C) Fahrzeuginnen- und Kofferraum**
- **Verwendeter Leitungsquerschnitt: 0.5, 0.75, 1.0, 1.5, 6.0, 16.0 mm^2**
- **Verwendeter Sicherungstyp: 7.5A + 20A MINI-Sicherung und 50A MAXI-Sicherung**
- **Verwendeter Lampentyp: W5W: 12V, 5W**
- **Batteriespannung: 13.8V**
- **Einschaltdauer: 60 min Fahrzeit, 47.5 min Lampen**

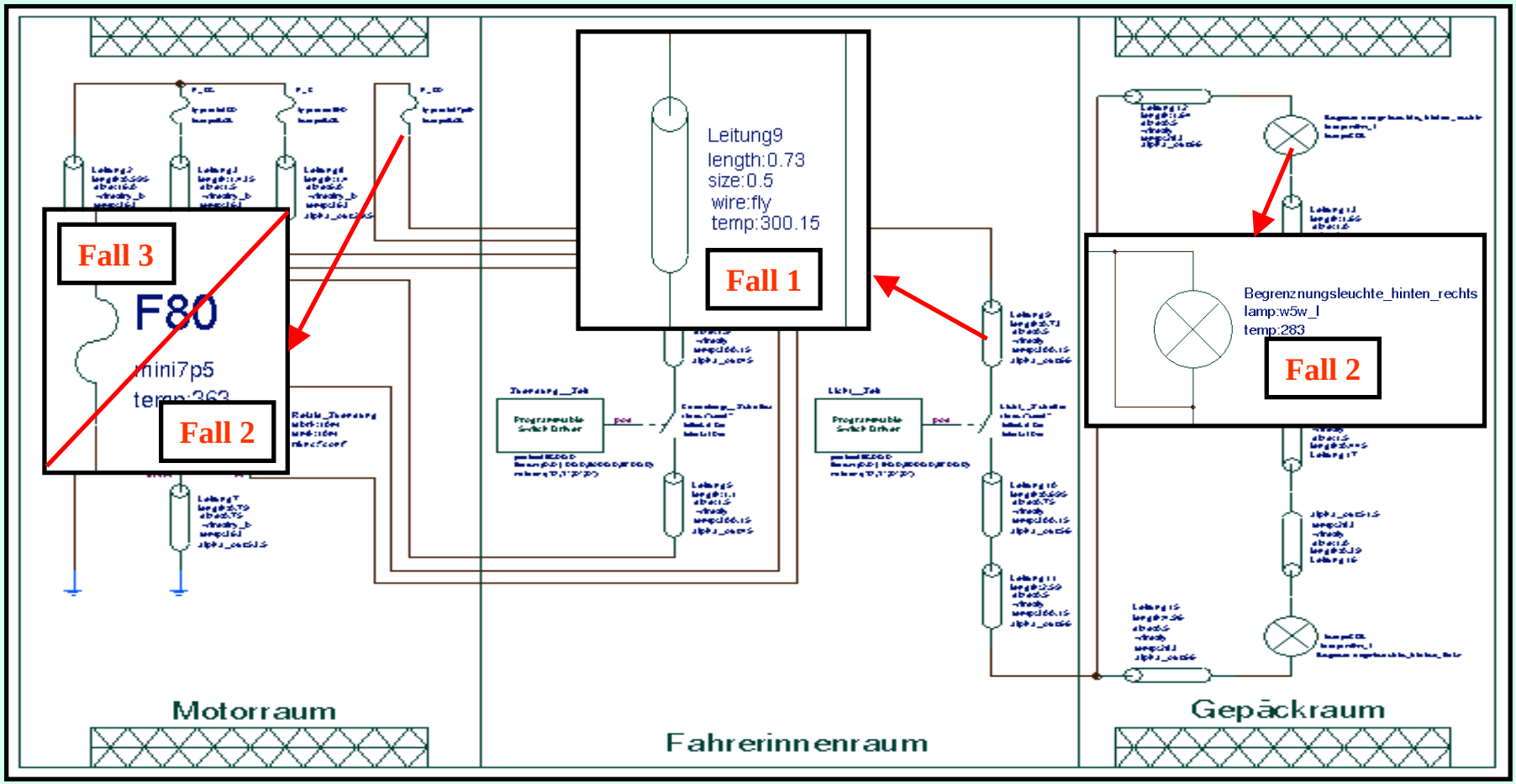
Ein Beispiel (3)

- **Umgebungstemperaturaufteilung im Fahrzeug**



Ein Beispiel (4)

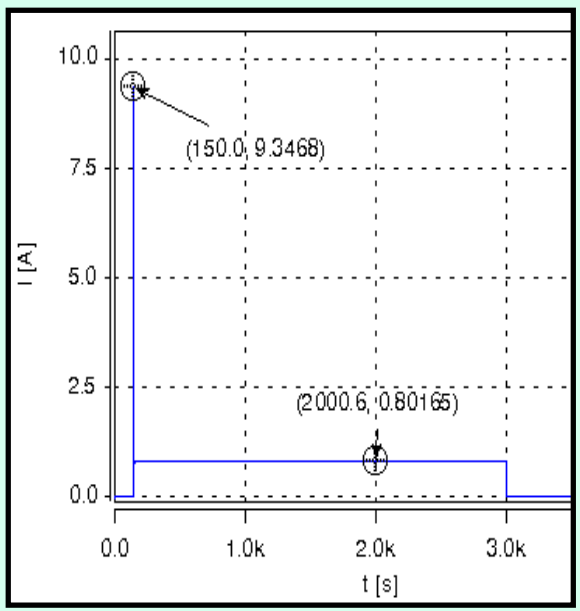
- **Layout**



Ein Beispiel (5)

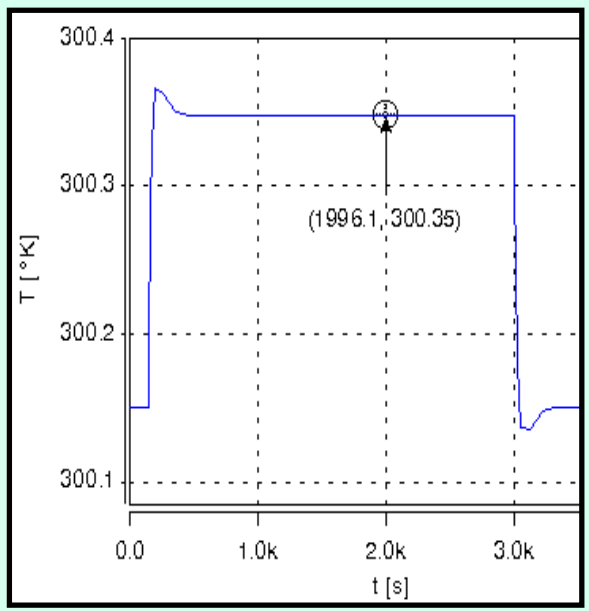
- **Fall 1: Normalbetrieb**

Stromfluß I [A]



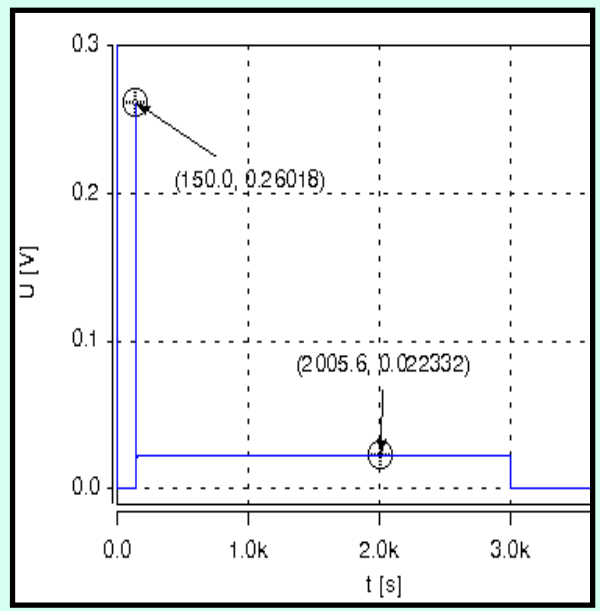
I = 0.8A

Temperatur T [°K]



T = 300.35°K

Spannung U [V]

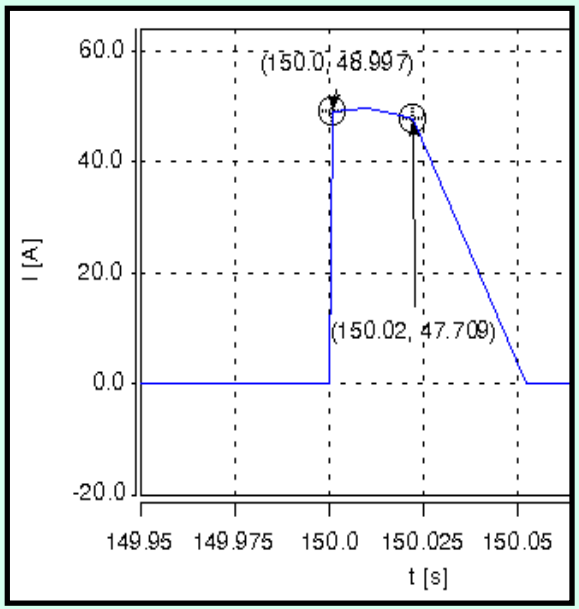


U = 0.02V

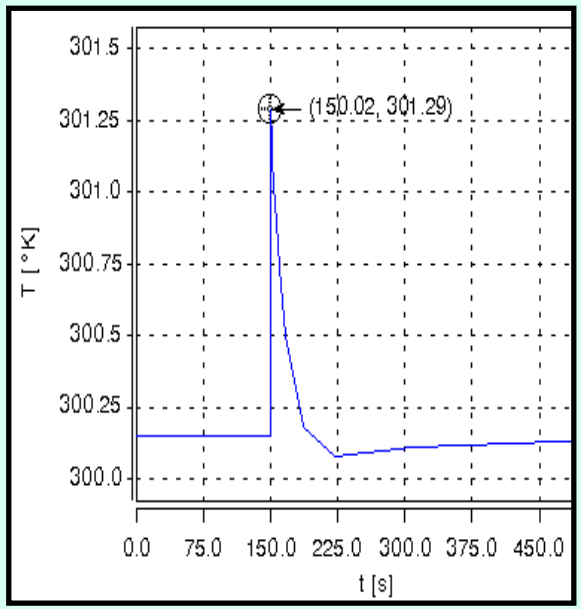
Ein Beispiel (6)

- **Fall 2: Fehlerfall**

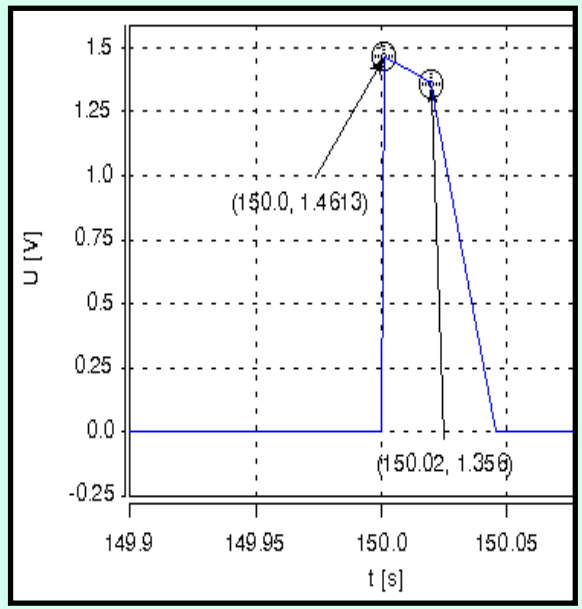
Stromfluß I [A]



Temperatur T [°K]



Spannung U [V]



I_Max = 49A

T_Max = 301.29°K

U_Max = 1.46V

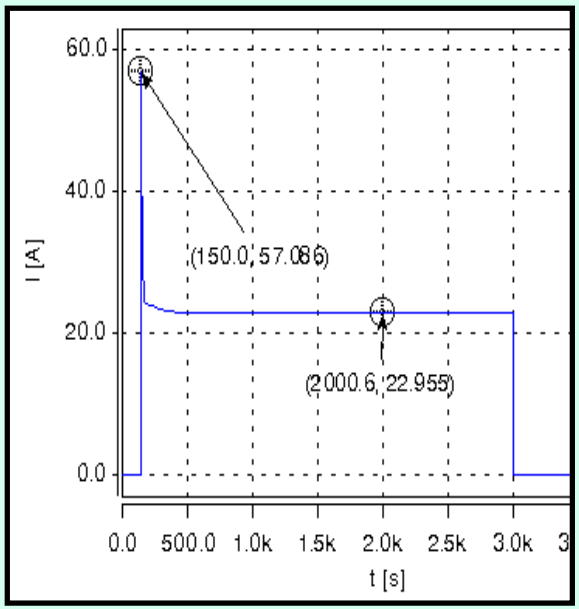
```
REPORT (WARNING) [p1:extended]: WARNING: Fuse melts.
        150.022343245374467 sec   -/2
```

t_Reaktion Sicherung F80 = 22.3ms

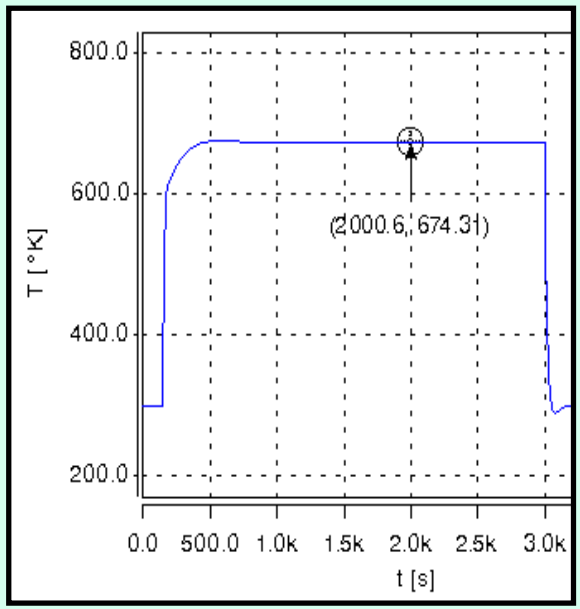
Ein Beispiel (7)

- **Fall 3: Fehlerfall ohne F80**

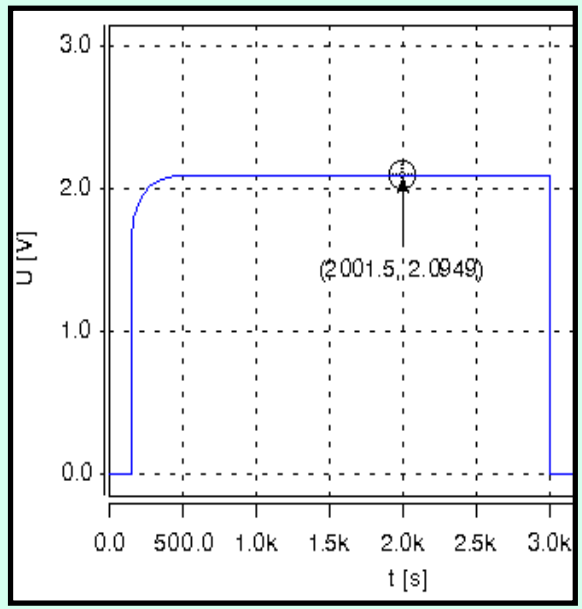
Stromfluß I [A]



Temperatur T [°K]



Spannung U [V]



I_Max = 57A | **I = 23A** | **I = 17A** | **T = 674.31°K** | **T_cu = 1358°K** | **U = 2.1V**

```
ASSERTION (ERROR) [extended]: ERROR: Violation of temperature class
(Temperature > TEMP_MAX)
```

T_Max = 363°K

Zusammenfassung

Zusammenfassung

Systemzeichensoftware in Verbindung mit geeigneten Simulationswerkzeugen eignen sich gut für die

- **Untersuchung leitungsbezogener Entwicklungsanteile neuer Fahrzeugprogramme,**
- **leitungsbezogene Design-Verifikaton bei Änderungen in laufenden Fahrzeugprogrammen,**
- **allgemeinen Betrachtungen und Entwicklungen leitungsbezogener Themen.**

Unterstützt durch

- **einen entsprechend starken Sprachenstandard wie VHDL-AMS,**
- **umfangreiche Bibliotheken wie die fundamentals_vda, automotive_vda, spice2vhd, spice2vhd_devices und megma,**
- **die sich bietenden Analyse- und Betrachtungsmöglichkeiten der VHDL-AMS - Modelle, wie beispielsweise des wire_vda.vhd – Modelles,**
- **Referenz-Modelle,**
- **die Verwendbarkeit eines Designs mit VHDL-AMS Modellen auf den verschiedenen Simulationstools und der somit erheblichen Verbesserung des Datenaustausches,**

ergeben sich gute und brauchbare Alternativen zu reinen Testverfahren, die kosten- und zeitintensiv sind.

Weitere Informationen unter http://fat-ak30.eas.iis.fraunhofer.de/

Danke für Ihre Aufmerksamkeit!

VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Dr. Thomas Lang, BMW Group
Ewald Hessel, Hella KgaA, Hueck&Co

BMW Group

VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Gliederung.

- Einleitung
- Heutige Vorgehensweise/Ausgangsbasis
- Zukünftige Prozessverbesserung
- Praxisbeispiel: Modellaustausch DC/DC-Wandler
  - Prozessverlauf und Aufgabenverteilung
  - Integration des Wandlermodells in ein Fzg.-Bordnetzmodell
  - Simulationsergebnisse
- Zusammenfassung

VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Heutige Vorgehensweise/Ausgangsbasis.

- In der Systemsimulation herrscht bisher eine Vielzahl unterschiedlicher Modelliersprachen und Plattformen.
- Die gemeinsame Modellentwicklung zw. Zulieferen und OEMs ist dadurch stark erschwert.
- Aus Sicht der OEMs erfordert die Integration von heterogenen Modellen z.B. in ein elektrisches Gesamtsystem des Fahrzeugs einen hohen Aufwand verbunden mit gesteigertem Projektrisiko und erhöhter Entwicklungszeit.
- In der Praxis werden die Modelle meist umgeschrieben oder toolspezifisch auf die jeweilige Plattform angepasst. Diese Integration „von Hand“ ist wenig prozesssicher.
- Die alternative Co-Simualtion erweist sich als sehr wartungsintensiv.

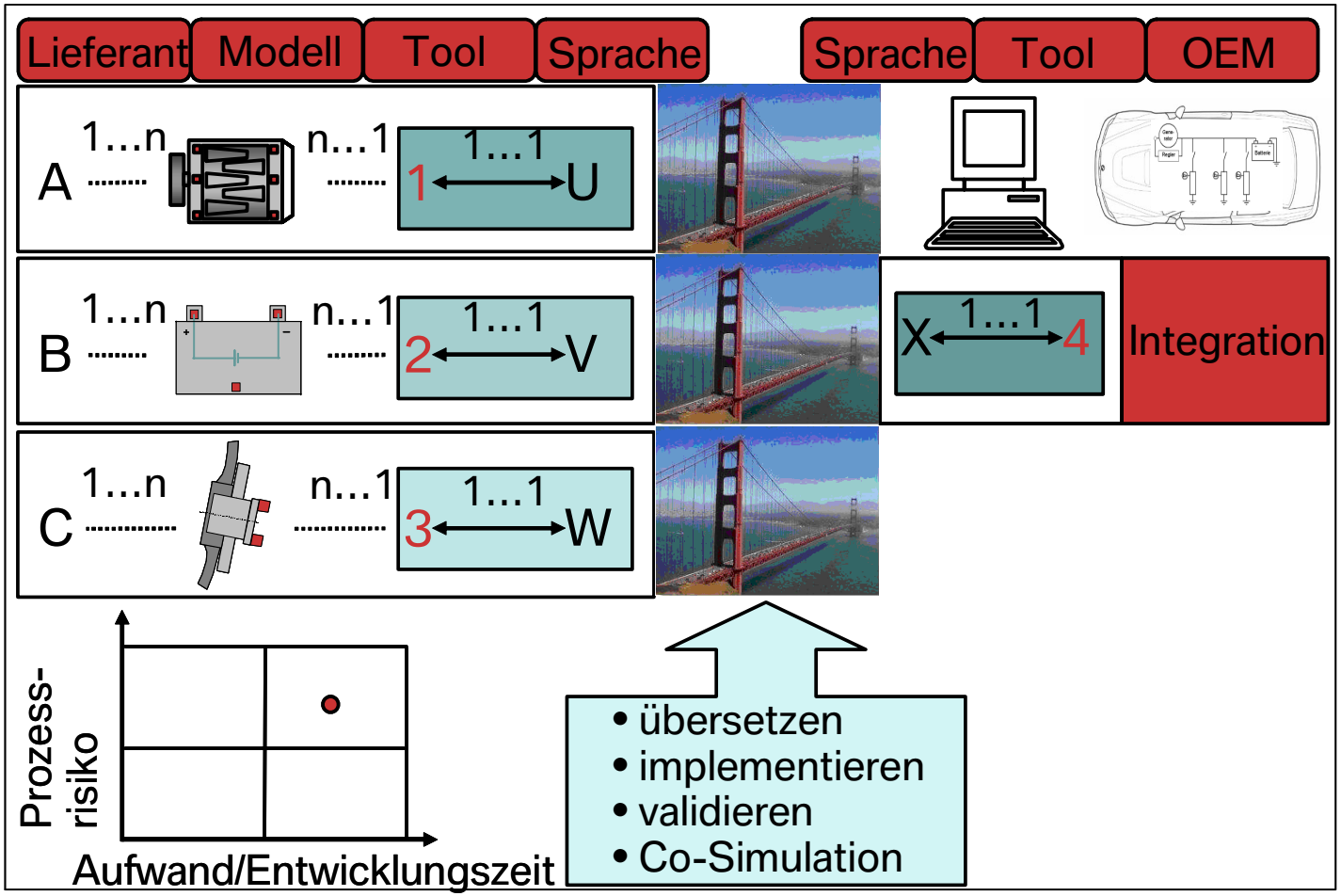
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Heutige Situation.



Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Zukünftige Prozessverbesserung.

- Grundlage für den zukünftigen Prozess bildet die standardisierte Modelliersprache VHDL-AMS und nicht ein spezifisches Simulationswerkzeug.

- Die Toolunabhängigkeit wird dadurch erreicht, dass voneinander unabhängige und renomierte Toolanbieter Simulatoren und Software für VHDL-AMS anbieten.

- Die zukünftige Vorgehensweise enthält VHDL-AMS als zentrales Brückenelement. Die Prozesskette wird enorm vereinfacht (ca. dreifach). Die Modelle müssen nur noch eingebunden werden. Prozessschritte wie übersetzen, validieren und Co-Simulation entfallen ganz oder werden stark minimiert.

- Der Entwickler kann sich künftig stärker den wesentlichen Aufgaben der Komponenten- und Systemauslegung widmen, ohne sich mit toolspezifischen Details beschäftigen zu müssen.

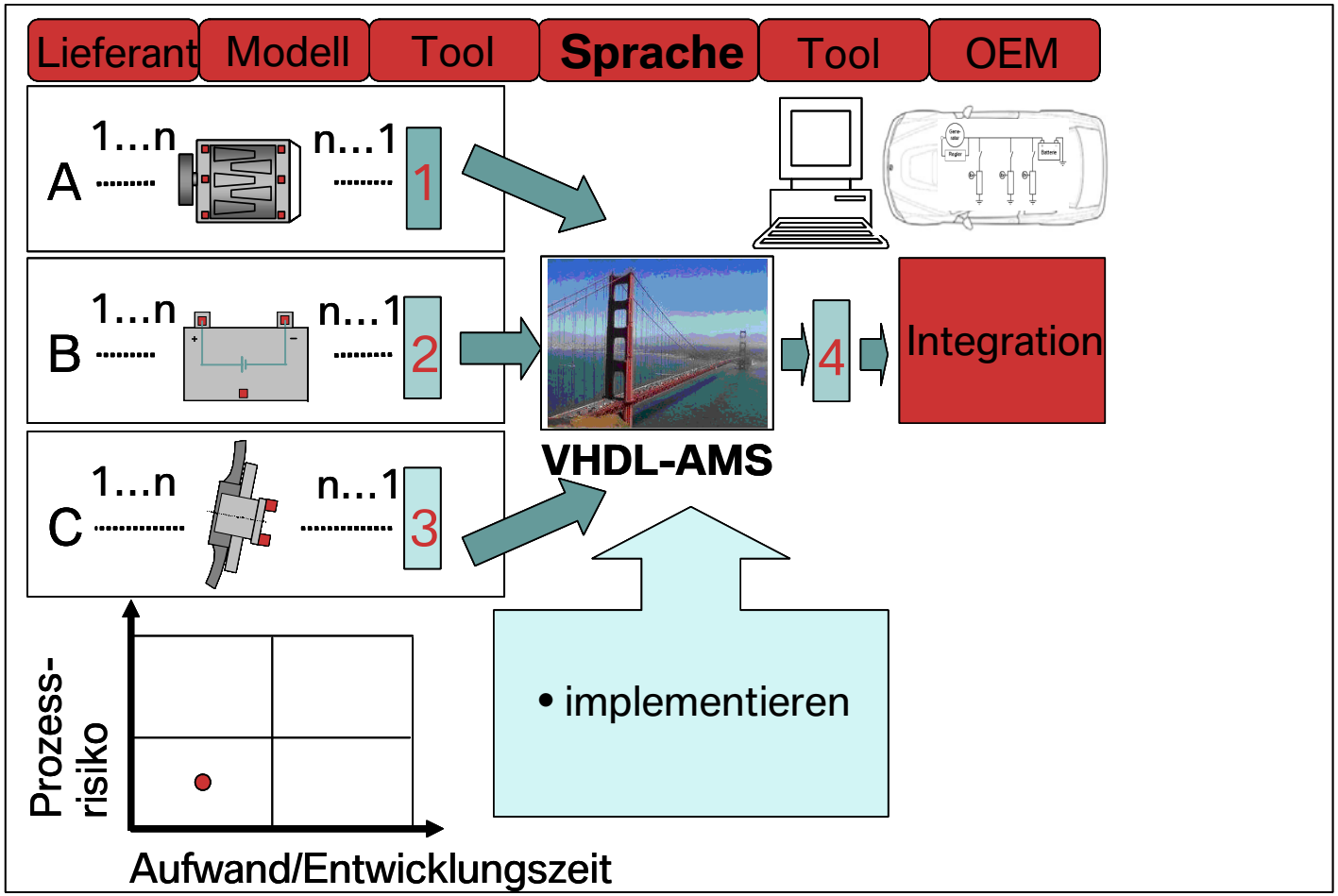
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Zukünftige Situation.



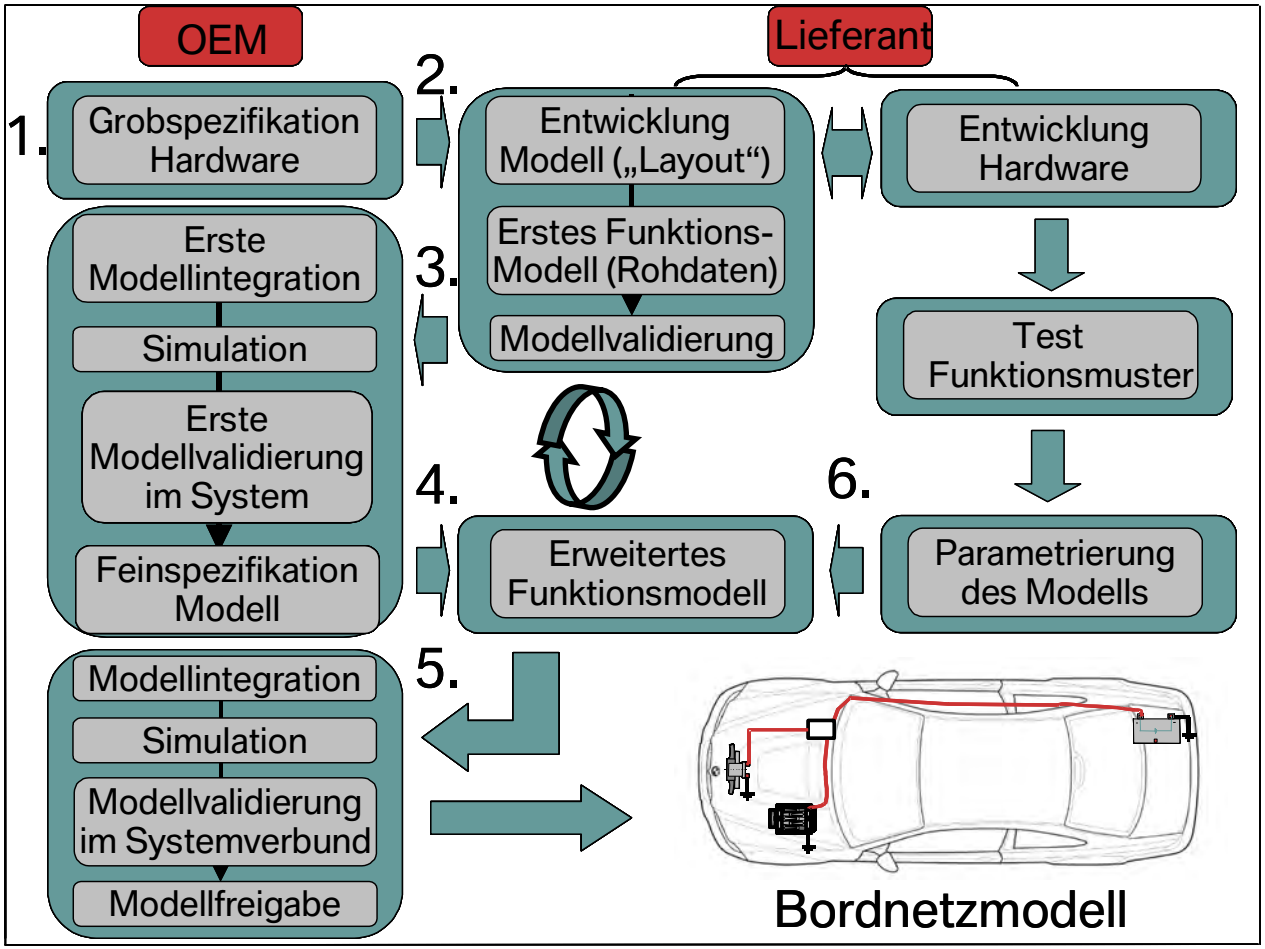
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Prozessverlauf und Aufgabenteilung (2).



Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Prozessverlauf und Aufgabenteilung (1).

- Der Prozess wurde zwischen Zulieferer und OEM wie folgt abgestimmt.

1. Grobspezifikation der Hardware u. Funktionen seitens des OEM.
2. Erster Modellentwurf seitens des Lieferanten. Beginn der Hardwareentwicklung.
3. Erste Modellintegration beim OEM. Simulation und Validierung. Anforderungen überprüfen, erweitern. Erstellung einer Feinspezifikation.
4. Erweiterung des Funktionsmodells seitens des Lieferanten aufgrund der Feinspezifikation
5. Modellintegration, Modellvalidierung und Modellfreigabe auf Seiten des OEMs.
6. Befüllung der Modellparameter mit „Echtdaten“ seitens des Zulieferers

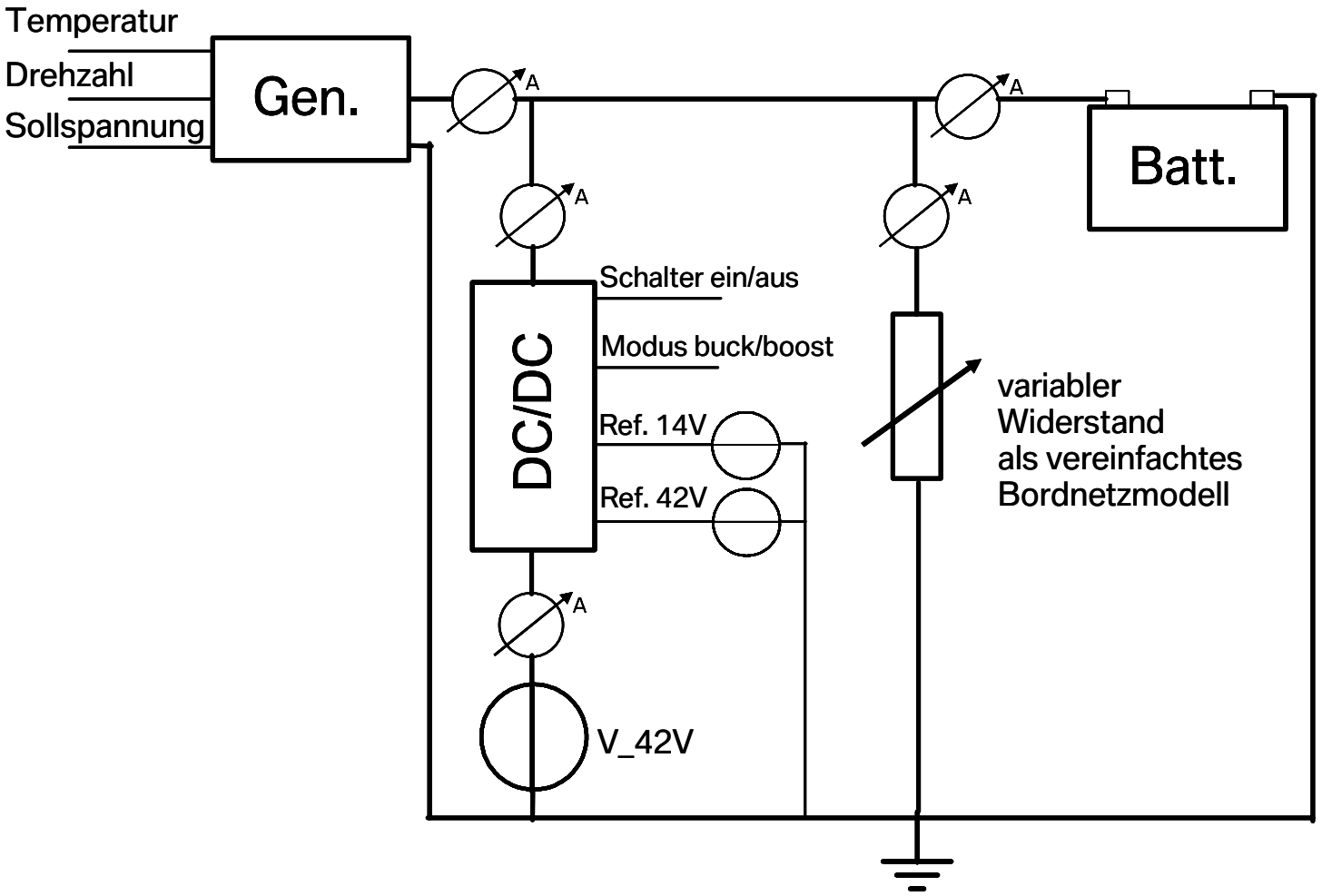
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Integration des DC/DC-Wandlermodells (1)



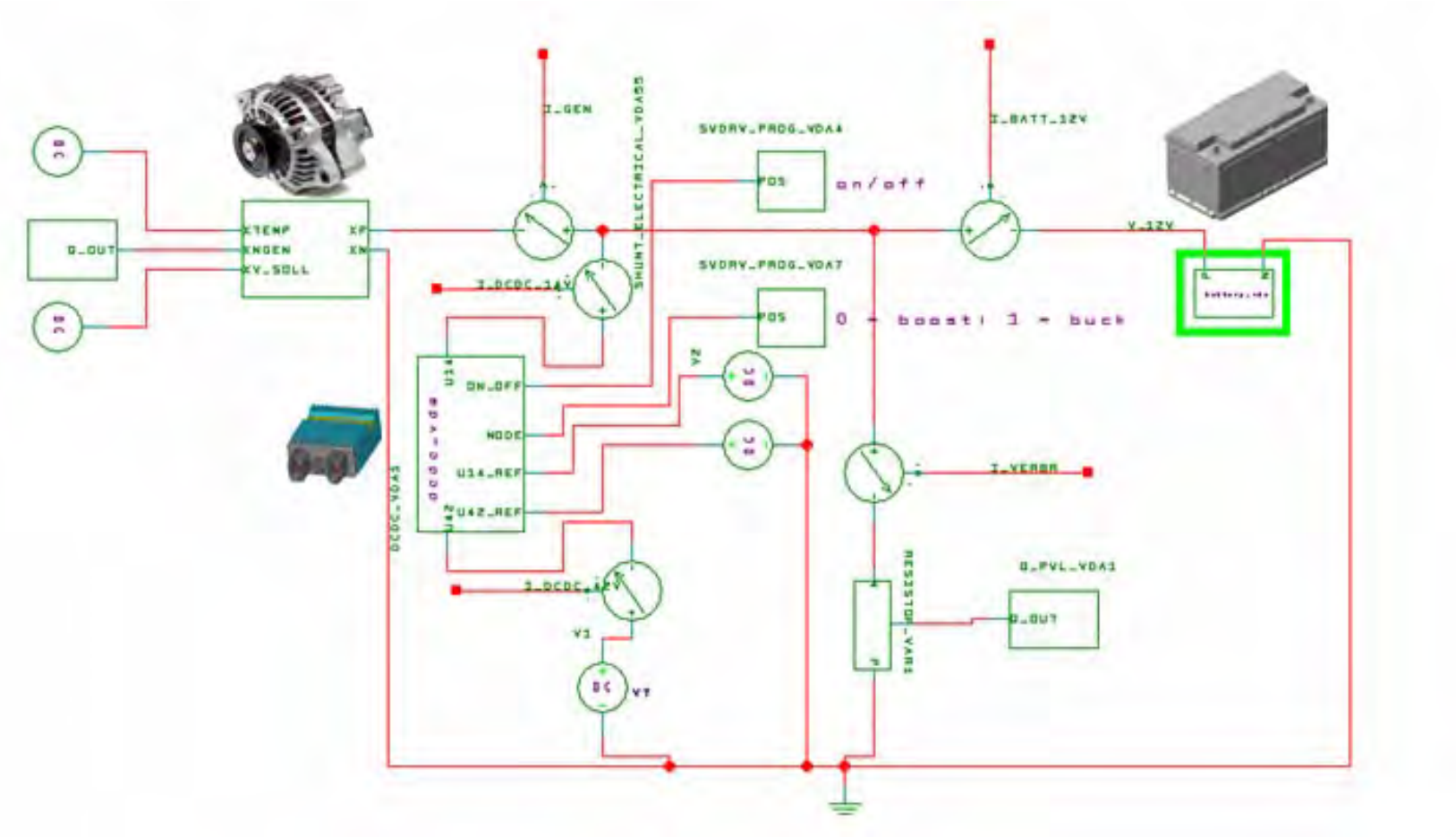
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2(

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Integration des DC/DC-Wandlermodells (2)



VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Bild des DC/DC-Wandlers.

Quelle: Hella KGaA

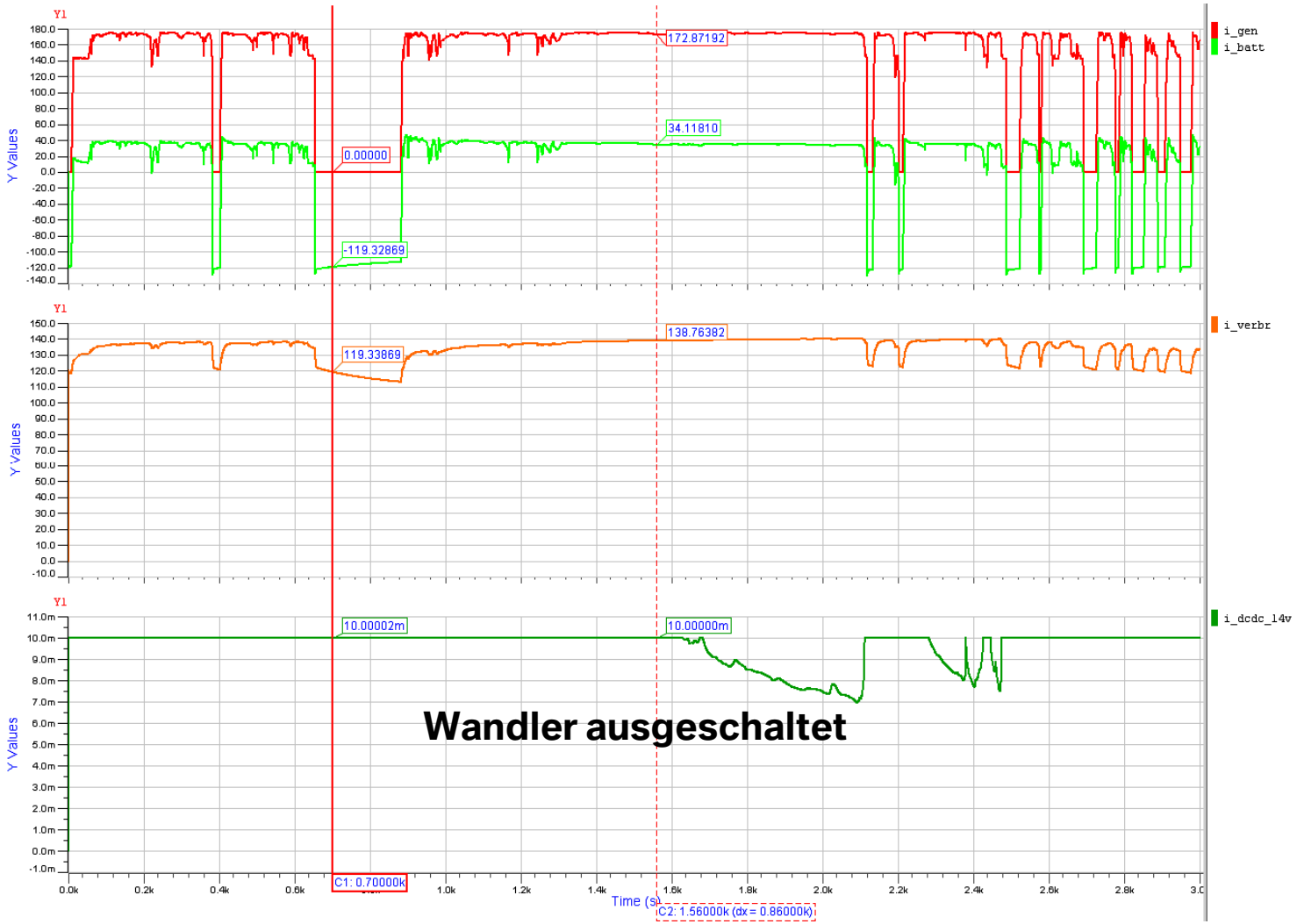
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Simulationsergebnisse – buck mode (1).



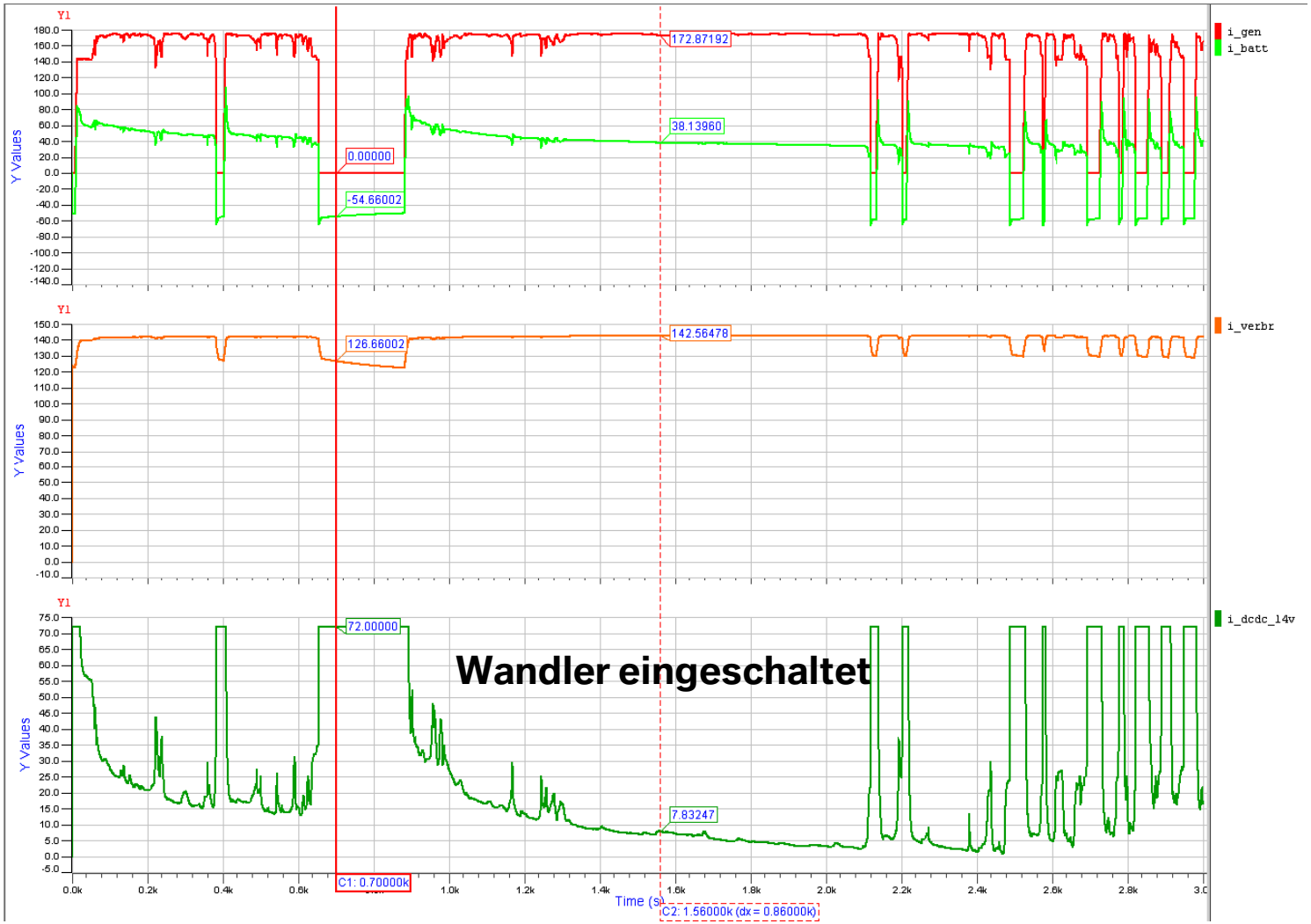
VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Simulationsergebnisse – buck mode (2).



Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Zusammenfassung

- VHDL-AMS eignet sich sehr gut als Beschreibungssprache für die elektrische Bordnetzsimulation.

- Der Modellaustausch zwischen Zulieferer und OEM lässt sich prozessicher gestalten. Ein möglicher Prozessablauf wurde zwischen der Hella KgaA und der BMW Group beispielhaft umgesetzt.

- Die standardisierte Modelliersprache gewährleistet eine wenig aufwendige Implementierung der Modelle in ein bestehendes System, da ein Übersetzen der Modelle nicht mehr notwendig ist.

- weitere Untersuchungen und weitere gemeinsame Modellierungen werden werden folgen.

VHDL-AMS
Praxisbeispiel
Modellaustausch
DC/DC-Wandler

Dr. Thomas Lang
BMW Group
Ewald Hessel
Hella KgaA,
Hueck&Co

ASIM-Tagung
München
20.-21.Februar.2006

Systemsimulation mit VHDL-AMS - Modellaustausch DC/DC-Wandler.

Vielen Dank für Ihre Aufmerksamkeit !

Einführung in VHDL-AMS

Dr. Klaus Panreck, Hella KGaA, Lippstadt

Einführung in VHDL-AMS

Inhalt

- Motivation
- VHDL-AMS, IEEE 1076.1
- Arbeitskreis 30 des VDA/FAT

Einführung in VHDL-AMS

Motivation:

Fachdisziplin-spezifische Simulation weitgehend ausgereift

- Analoge elektronische Schaltkreise
- Digitale elektronische Schaltkreise
- Mechanische Teilsysteme
- Pneumatische Teilsysteme
- Thermische Teilsysteme
- Steuer- und Regelkreise
- Software (uC, SystemC, C++)
- Verhaltensmodelle

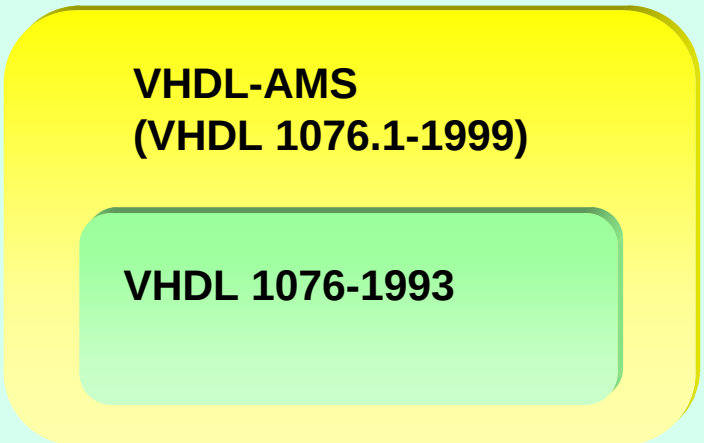
Herausforderung:

Zusammenführung der Fachdisziplinen in der Simulation!

Simulation gemischter Systeme von immer größerer Bedeutung !

Einführung in VHDL-AMS

V Very High Speed Integrated Circuits
HDL Hardware Description Language
AMS Analog and Mixed Signal



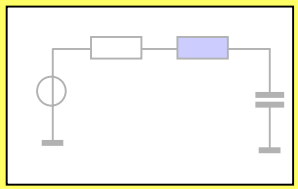
- **VHDL 1076-1993**
  - **Zeitdiskrete Systeme**
  - **Ereignis-getriebene Simulationsalgorithmen**
- **VHDL-AMS**
  - **Erweiterung für zeitkontinuierliche (analoge) Systeme**
  - **Analoge DAE-Solver**
  - **IEEE-Standard 1076.1-1999**
  - **http://www.vhdl.org/analog**

VHDL-AMS ist eine Obermenge von VHDL.
Konzepte u. Sprachkonstrukte für digitales VHDL bleiben gültig

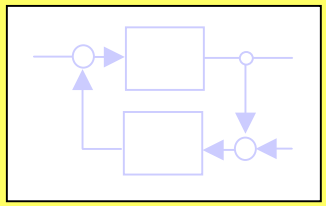
Einführung in VHDL-AMS

VHDL-AMS: Unterstützte Modellierungskonzepte

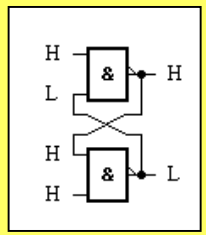
Mathematische Beschreibung



konservative Netzwerke
(Kichhoff'sche Netzwerke)



nichtkonservative Systeme
(Signal-Fluß-Diagramme)



digitale Systeme
(zeitdiskret)

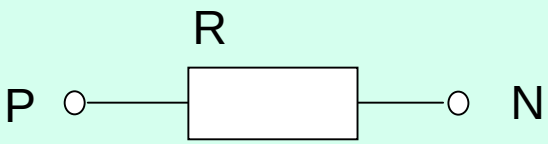
Mögliche Mischbeschreibungen sind:
- analog – digital
- elektrisch – nicht-elektrisch
- konservativ – nicht-konservativ
- ...

Implementation der Modelle

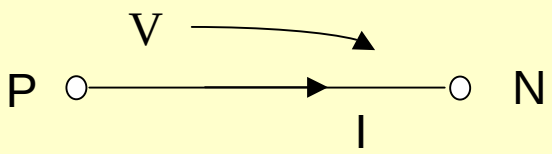
VHDL-AMS Modelle

Einführung in VHDL-AMS

VHDL-AMS: Beispiel



branch quantity declaration



simultaneous statements

$V = R \cdot I$

context clause

```
library IEEE;
use IEEE.ELECTRICAL_SYSTEMS.all;
```

```
entity RESISTOR is
  generic (R : REAL);
  port (terminal P, N : ELECTRICAL);
end entity RESISTOR;
```

```
architecture A0 of RESISTOR is
  quantity V across I through P to N;
begin
V == R * I;
end architecture A0;
```

Einführung in VHDL-AMS

VHDL-AMS: Simulatoren

Tool	EDA Vendor	URL
AMS DESIGNER	*Cadence*	*www.cadence.com*
ADVance MS	*Mentor Graphics*	*www.mentor.com*
SystemVision	*Mentor Graphics*	*www.mentor.com*
SaberHDL	*Synopsys*	*www.synopsys.com*
SIMPLORER	*Ansoft*	*www.ansoft.com*
SMASH	*Dolphin*	*www.dolphin.fr*

Einführung in VHDL-AMS

VHDL-AMS: Merkmale

- Tool-unabhängige Modellierungssprache (IEEE 1076.1)
- Unterstützt durch zahlreiche Tools
- Unterstützung digitaler Systeme (IEEE 1076-1993)
- Unterstützung analoger Systeme verschiedener Disziplinen
- Unterstützung von Mixed-Signal-Systemen

VHDL-AMS ist sehr gut geeignet für die Modellierung und Simulation von Automotive-Systemen.

Einführung in VHDL-AMS

Arbeitskreis 30 des VDA/FAT

Thema:

- Simulation gemischter Systeme mit VHDL-AMS

Ziele:

- Förderung der Beziehungen zwischen Automobilherstellern und ihren Zulieferern im Hinblick auf die Simulation gemischter Systeme und den Modellaustausch
- Förderung von VHDL-AMS (IEEE 1076.1) als allgemeine Modellierungssprache

Einführung in VHDL-AMS

Gegründet:

- 28 Juli 2003

Arbeitsweise:

- Kostenlose Mitgliedschaft (FAT- oder Gast-Mitglied)
- Definition von Arbeitspaketen und –aufgaben und Bearbeitung von Teilproblemen in Unterarbeitsgruppen
- Bearbeitung und Beauftragung größerer Projekte
- Regelmäßige Treffen (etwa alle 6 Wochen)
- Einige Modellbibliotheken im Internet frei verfügbar!
- Homepage: **http://fat-ak30.eas.iis.fraunhofer.de**

Einführung in VHDL-AMS

Mitglieder:

Automobilhersteller:

- Audi, BMW, DC, Ford, Porsche, VW

Zulieferer:

- Bosch, Brose, Conti, DELPHI, Hella, Siemens/VDO, Yazaki

Bauteillieferanten:

- Infineon, Atmel

Einführung in VHDL-AMS

Mitglieder:

Institute :

- FHG Dresden, FHG Paderborn, FKA Aachen, IKA Aachen

Dienstleister:

- SmartCAE, Bausch-Gall, DAnalyse, Adapted Solutions

Tool-Hersteller:

- Ansoft, Mentor Graphics, Synopsys, Cadence, Dolphin

Einführung in VHDL-AMS

Aktivitäten des AK 30

- Modellierungsrichtlinien
  - Definition eines einheitlichen Beschreibungsstandards
  - Erzeugung der Modelldokumentation aus dem Source-Code
  - Definition eines VHDL-AMS-Subsets
  - Regeln für die verteilte Modellentwicklung
  - . . .

- Modell-Bibliotheken
  - Grundmodelle in FUNDAMENTAL_VDA
  - Spice-ähnliche VHDL-AMS-Modelle in SPICE2VHD
  - Spezielle Automotive-Bausteine in AUTOMOTIVE_VDA (gefördert durch die FAT)
  - . . .

Einführung in VHDL-AMS

Weitere Aufgaben für die Zukunft:

- Schnittstellen zu bewährten Tools und Sprachen
  (Matlab/Simulink, Modelica, ...)
- Kontakt zur IEEE working group (Erweiterung des Standards)
- Kontakt zu verwandten Arbeitsgruppen (SAE etc.)
- Definition eines RT-Subsets für Echtzeit-Anwendungen
- Definition weiterer Bibliotheken für Spezialbereiche
- Test und Verbesserung des Prozesses zum Modellaustausch
- Definition von Anforderungen zur Verbesserung der Tools
- **Förderung des Einsatzes der Methode !!!**

Einführung in VHDL-AMS

Vielen Dank für Ihre Aufmerksamkeit !

Weitere Informationen:

http://fat-ak30.eas.iis.fraunhofer.de

System Architect Designer – SyAD®

A design and simulation framework for system design of heterogeneous microelectronic systems

contact.TM@CISC.at

ISSUE: February 2006

Outline

- Introduction into SyAD®
- System Overview
  - GUI Applications
  - Server Side Applications
  - System Management
- Design Steps covered by SyAD®
  - System Design
  - Automatic Generation of a Verification Platform
  - Model & Symbol Creation
- Summary

SyAD®: System Architect Designer

- System Design Tool
  - Heterogeneous multi-language design
  - Supports (IP) model-based design
  - Modular and evolutional design
- System Verification Tool
  - Homogeneous and heterogeneous simulation using industry's standard simulators
  - Distributed simulation
  - Simulation control
- Design Management Tool
  - User manager
  - Project manager
  - Library manager

Applications with SyAD®

- System integration for automotive and telecommunication applications
- Virtual prototyping and test engineering
- System verification using behavioral models
- HW/SW co-simulation

SyAD‘s Key Features

- System Design
  - Supports multi-language design
  - Provides a schematic editor for graphical design
  - Facilitates language and tool independent design
  - Modular block-based design
  - Promotes concurrent system development

SyAD's Key Features

- System Verification
  - Able to incorporate many state-of-the-art simulators
  - Automatic generation and set-up of a verification platform for
    - Homogeneous simulation
    - Heterogeneous cosimulation
  - SyAD's generic cosimulation interface features
    - Efficient communication between simulators
    - Integration of multiple simulators
    - Time accurate cosimulation
    - Distributed simulation
    - Tunable simulation parameters to accelerate the simulation execution
  - GUI-based simulation control

SyAD‘s Key Features

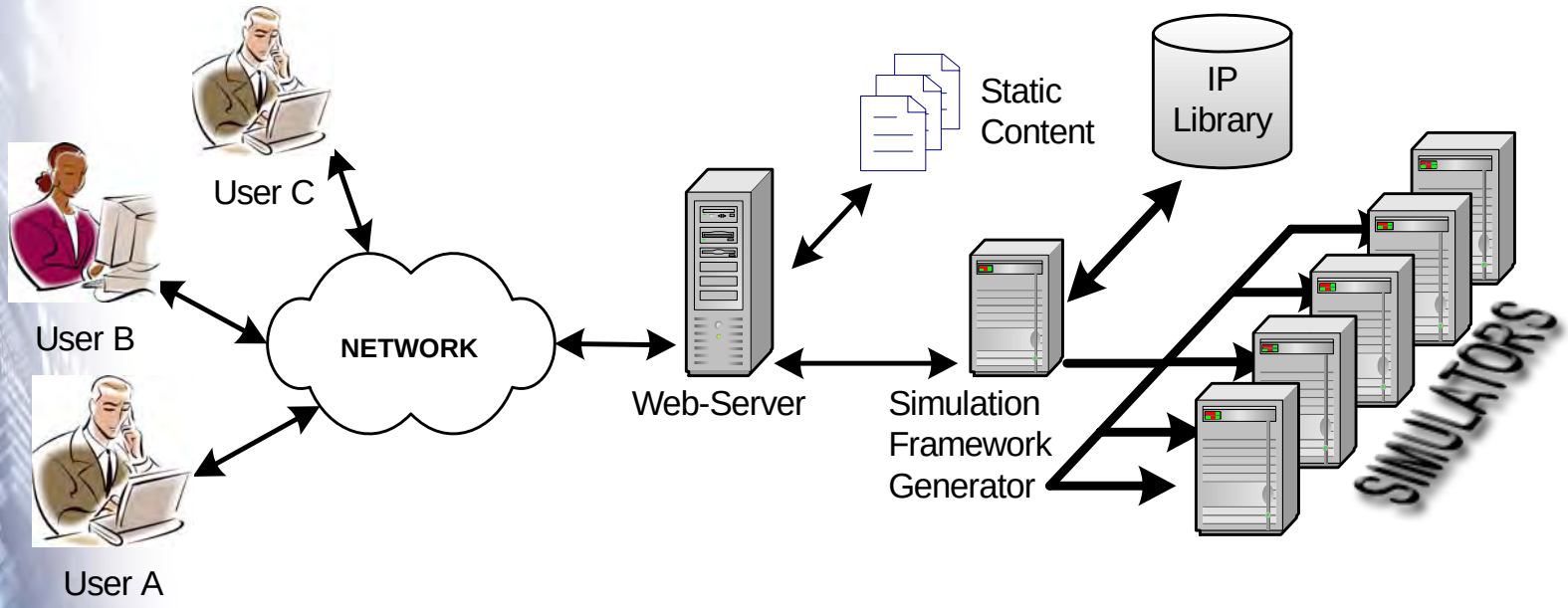
- Software Architecture
  - Client (distributed) Server application
  - Client(s)-Server communication based on HTTP protocol
  - User Entry: Stand-alone and applet application (Windows, Linux Sun Solaris)
  - Designed to promote Computer-Supported Concurrent Work (CSCW)
  - Web-based User-, Project-, and Library-Management Units

Design language features

- Supported languages/simulators
  - MATLAB/Simulink/Stateflow
  - SystemC(-AMS)
  - VHDL/Verilog: ModelSim,NC-Sim
  - VHDL-AMS/Verilog-AMS: AMS Designer, AdvanceMS
  - MAST: Saber (only homogeneous simulation)
  - SPICE: Eldo, Spectre
  - MODELICA: OpenModelica (1)

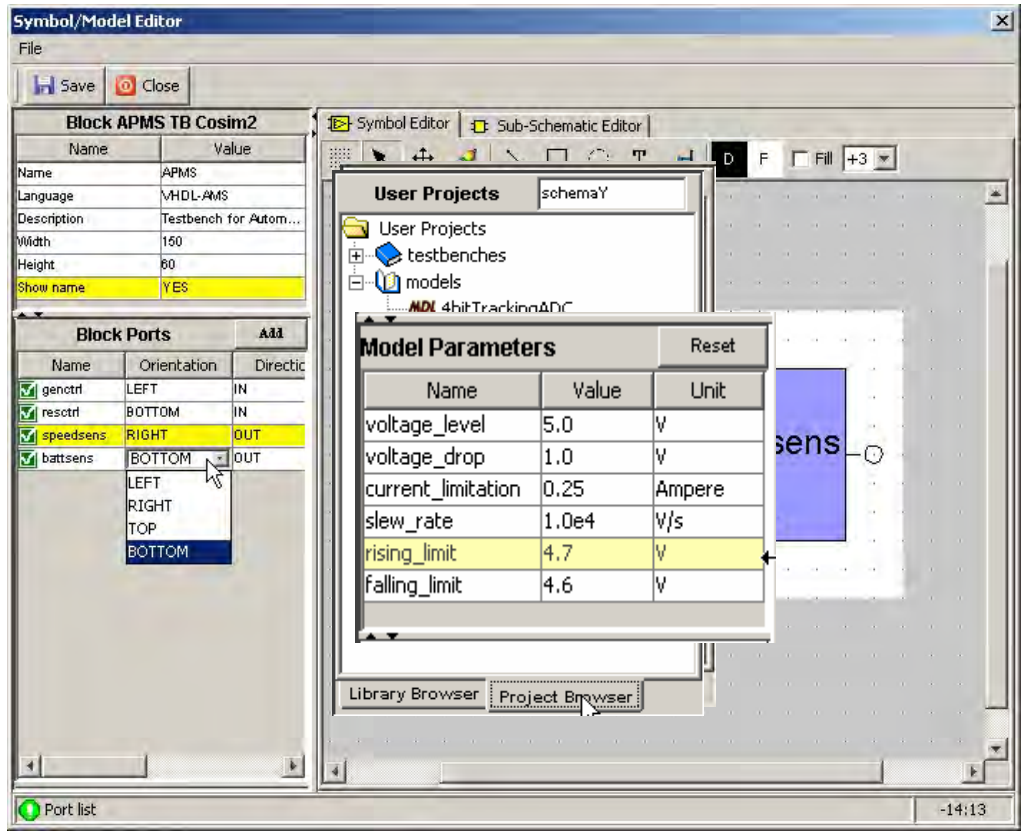
(1) Under development

System Overview



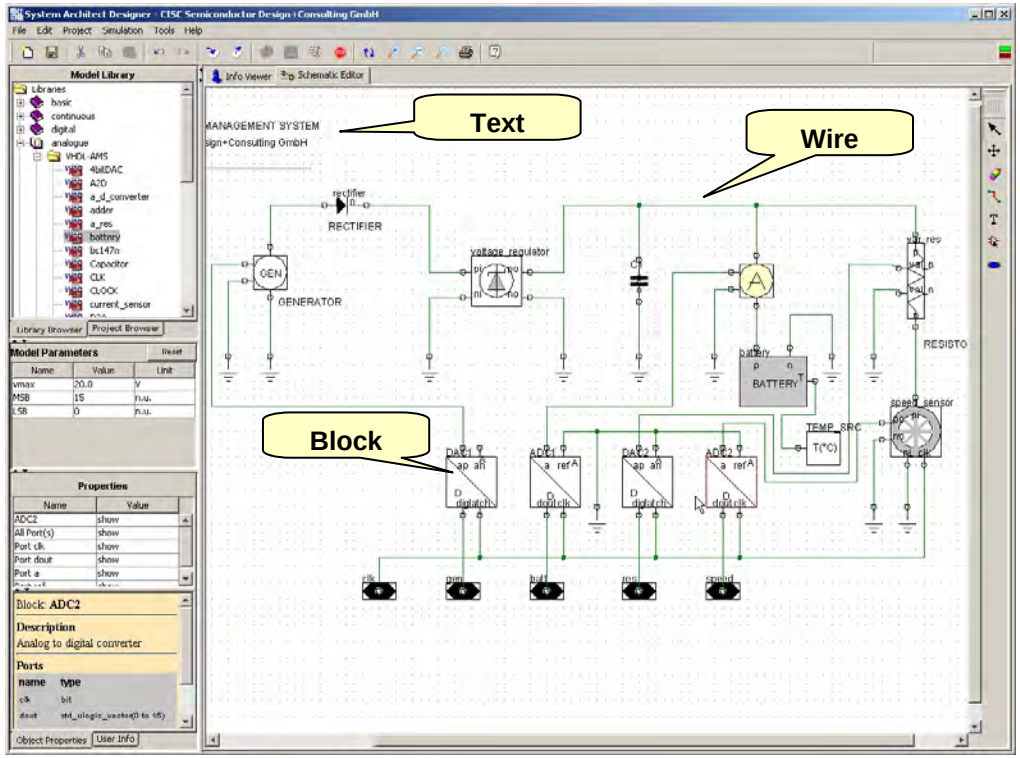
Client Side Application

- Schematic Editor
- Model Source Editor
- Library Browser
- Project Browser
- Block Parameter Editor
- Block Properties Editor
- Block Information Viewer
- Symbol Editor



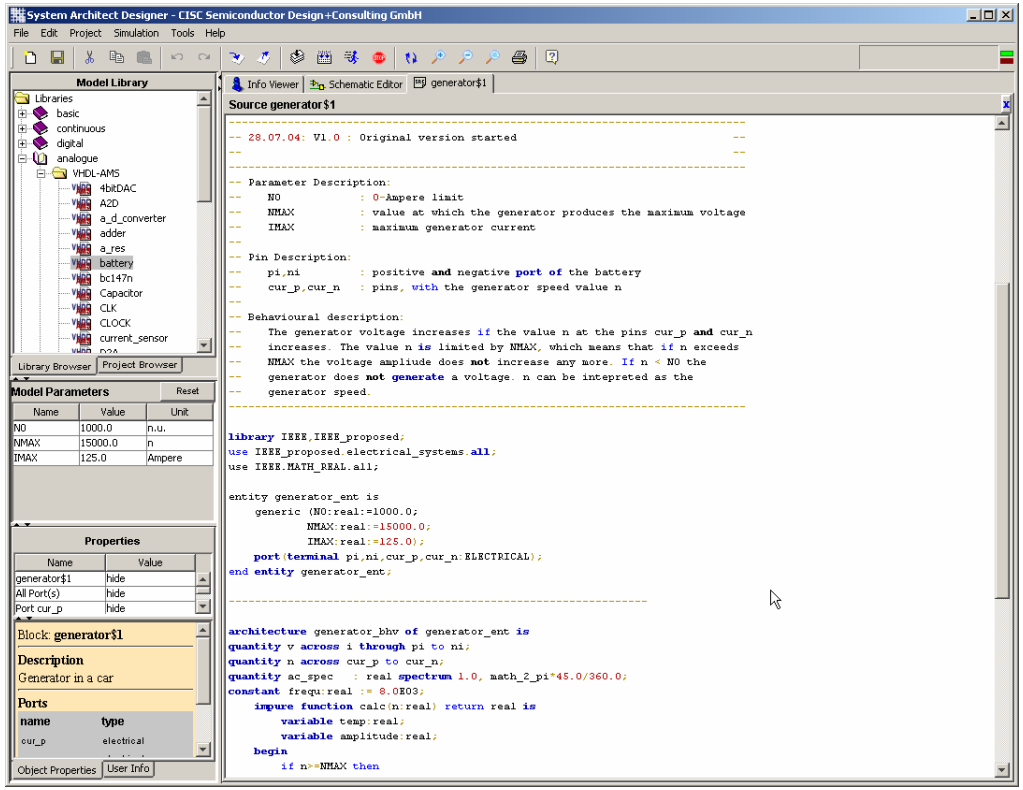
Schematic Editor

- Standard schematic functions
  - Blocks
  - Texts
  - Wires
- Block
  - open, rename
  - move, delete,
  - disconnect, rotate
- Text
  - change, move
  - delete
- Wire
  - rename, delete



Model-Source Editor

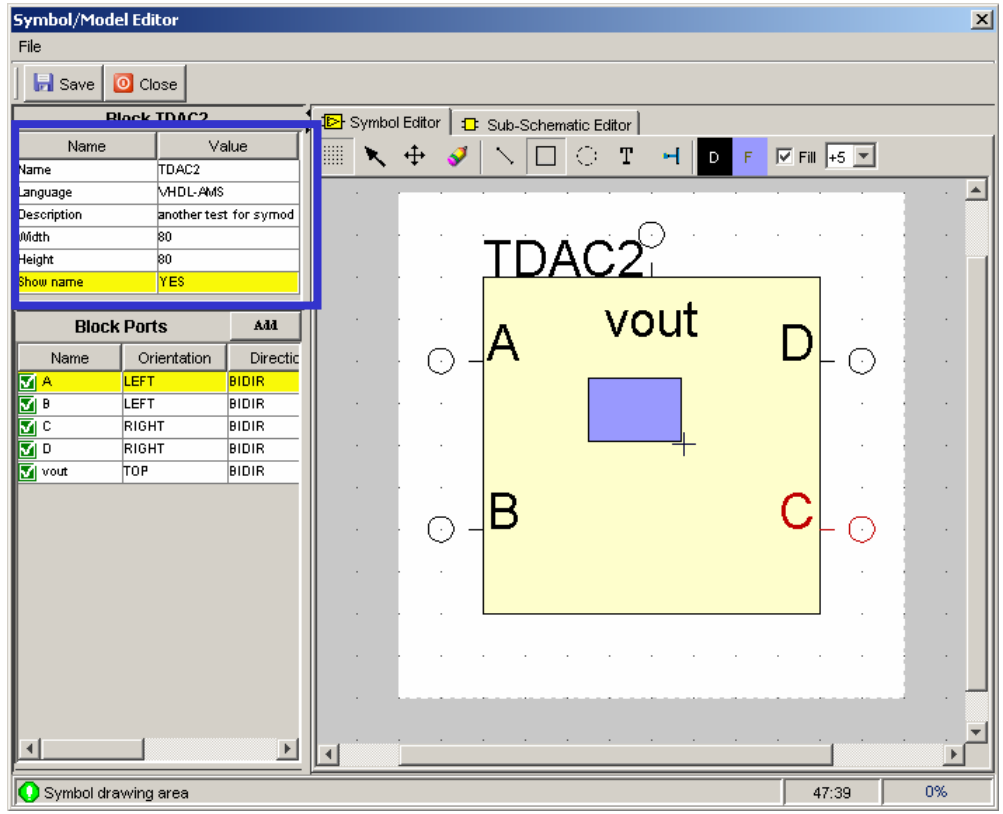
- Universal Viewer for multiple languages
  - Syntax highlighting for supported HDLs
  - Read-only mode for IP Lib models
  - User access control
  - Version control (1)



(1) Under development

Symbol Editor

- Graphical Symbol Creation
- Model properties setting
- Port parameter setting



- XML based symbol description

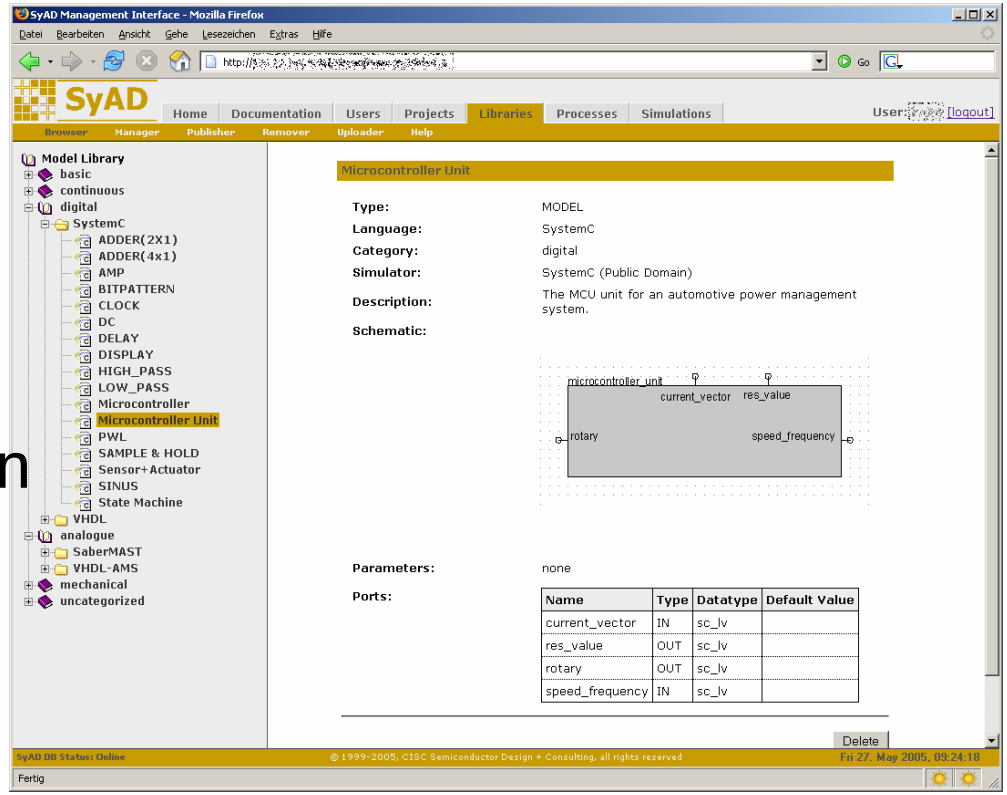
CISC semiconductor

Server Side Applications

- Simulation Framework Generator (SFG)
- Cosimulation Interface Generator (CSIG)
- IP-Library
- Web-Server
  - SyAD standard web content
  - Web-based System Management
  - Simulation (SyAD applet)

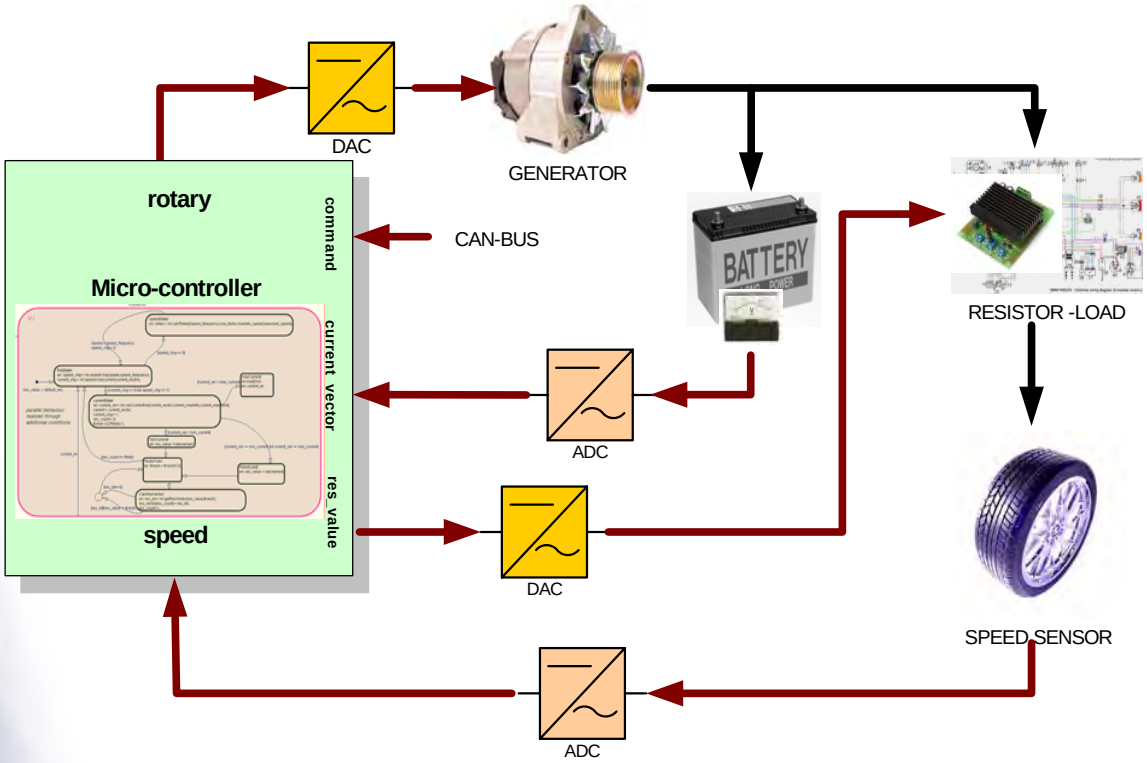
Web-Interface

- User Management
- Project Management
- Library Management
- Process Control
- Simulation
- Dynamically created model documentation



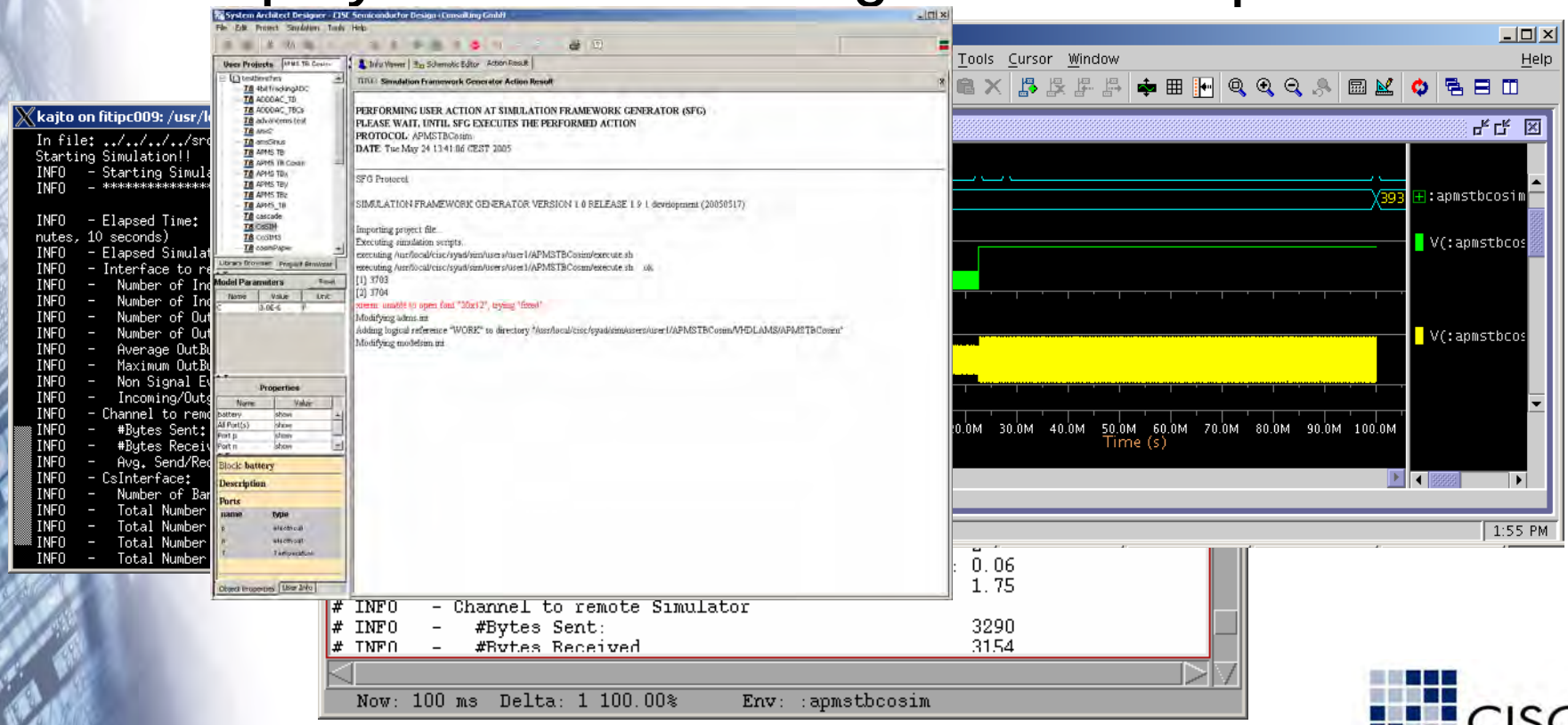
System Design

- An Example Application
  - Automotive Power Management System



Verification

- Automatic invocation of simulators
- Display redirection using external X-protocol

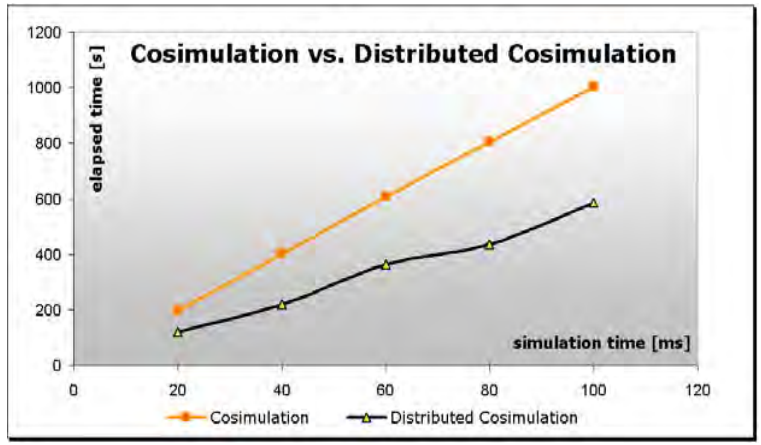


Co-simulation Results

- Distributed Cosimulation vs. Single-PC Co-simulation Speed-Up 1.75
- Simulation performance depends strongly on the structure of the system
- Optimization via CSIG parameters possible

		Cosimulation	Distributed Cosim.
Num.	Simulation Time [ms]	Elapsed Time [s]	Elapsed Time [s]
1	20	197	119
2	40	401	220
3	60	606	364
4	80	804	434
5	100	1003	586
	Dist. Speed-Up	1	1,75
Cosimulation: ADVanceMS and SystemC			

a)



b)

Summary

- Block-based schematic system design entry
- Fast & reliable automatic code generation
- Single and parallel co-/simulations
- Multi HDL system design
- System verification of concurrent developed subsystems coded in different HDLs.
- Based on latest web technologies
- OS independent implementation

Contact for SyAD Info

- CISC Semiconductor Design+Consulting GmbH
  - Lakeside B07
  - 9020 Klagenfurt - Austria
  - Tel +43 (463) 508 808
  - Fax +43 (463) 508 808 18
- Web: www.cisc.at/syad
- E-Mail: contact.**TM**@cisc.at

fachhochschule münchen

munich university of applied sciences

In Zusammenarbeit mit

T-Systems

Systemsimulation im Entwicklungsprozess aus der Sicht von Systems Engineering

ASIM
20. / 21. 2. 2006

Prof. Dr.-Ing. Werner Kohl

Fachbereich Elektrotechnik und Informationstechnik
Systems Engineering

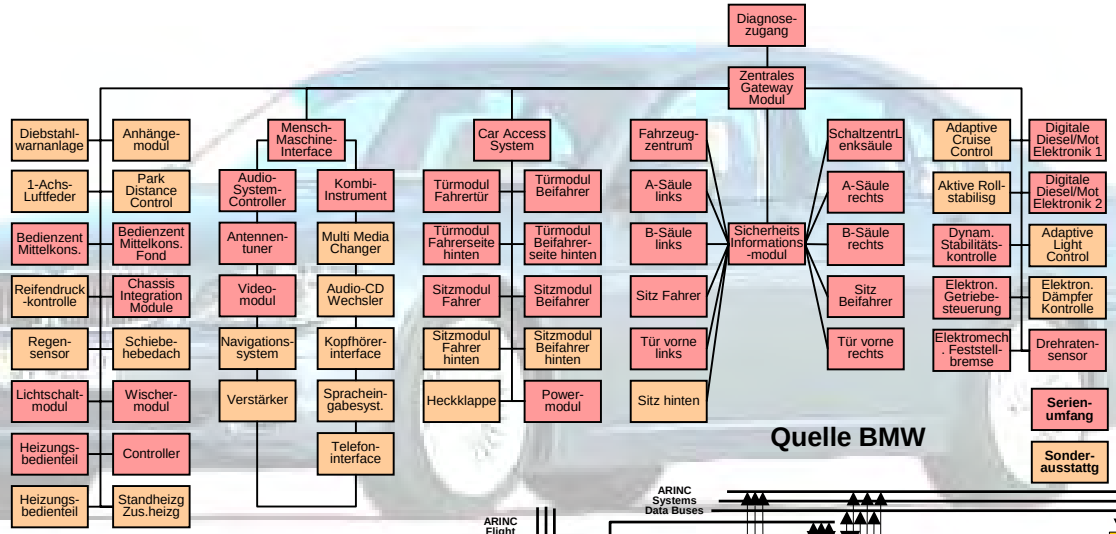
Inhalt

Charakteristik komplexer Systeme

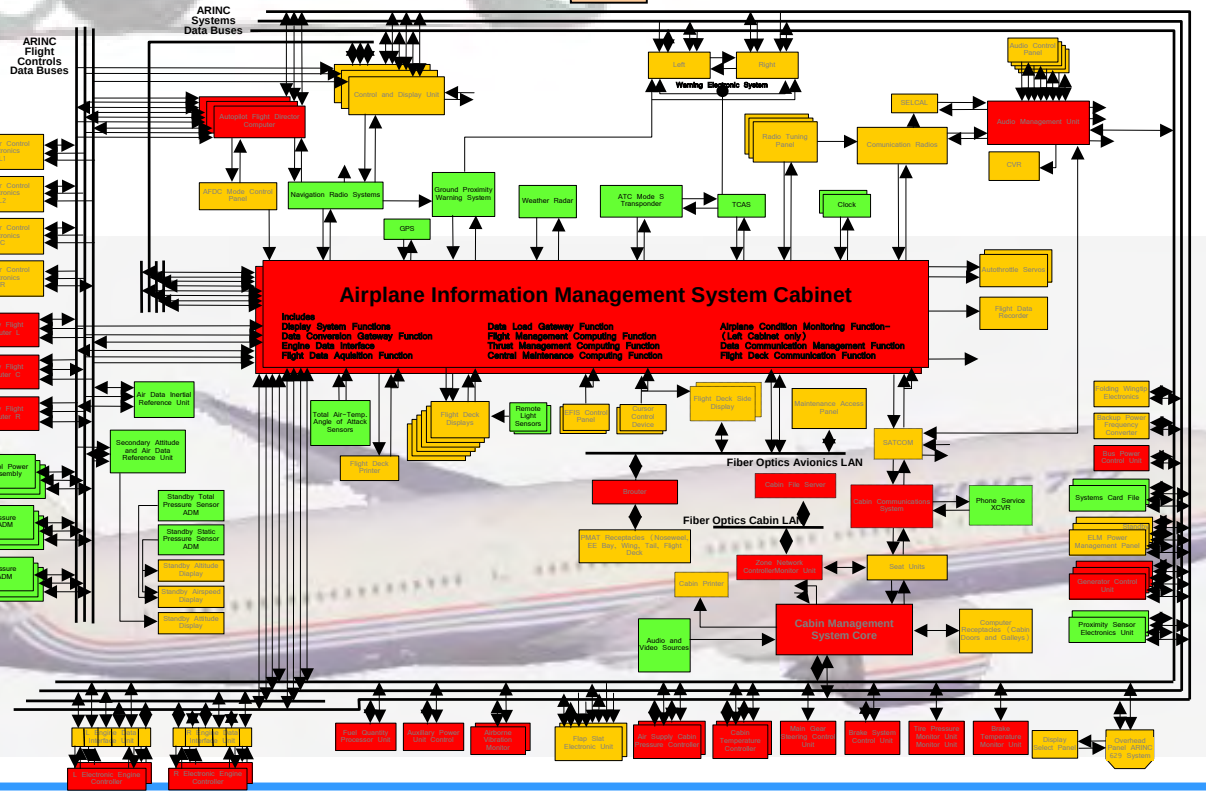
Risiken bei der Entwicklung

Lösungsansatz

komplexe Systeme



Quelle BMW



- **Prozessoren**
- **Sensoren**
- **Aktoren**

Systemcharakteristiken

- **Heutige Elektronik-Systeme bestehen aus hochgradig vernetzten embedded Komponenten**
  - Hoher Kostenanteil des Elektroniksystems (30 - 40% der Herstellkosten des Gesamtsystems)
  - 90% aller Innovation sind getrieben durch Elektronik/Software

- **Multi-Prozessor Environment**
  - Software bestimmt wesentlich die Funktion der Produkte
  - Softwarequalität bestimmt Produktqualität
- **Komponenten des Systems weitgehend von Zulieferern entwickelt**
  - Spezialisierung, Know How
  - Arbeitsteiligkeit, reduzierte Fertigungstiefe
  - Kostengesichtspunkte

- **Entwicklungsergebnis erst spät verifizierbar mit hohem Testaufwand zum Nachweis der Systemintegrität**

- **Embedded Real Time Systems haben ein hohes technisches und Projektmanagement Risiko**

Risiko Systemkomplexität

Zu große Komplexität bezüglich Funktionalität und Systemarchitektur

- **Mit zunehmender Komplexität der Systeme nimmt die Möglichkeit einer Gesamtsicht für den einzelnen Ingenieur ab.**
- **Die Integration von Subsystemen zum Gesamtsystem ist bei ungenügender Gesamtsicht ein potentieller Problembereich.**
- **Es entstehen unbeabsichtigte und unnötige Wechselwirkungen zwischen Systemfunktionen und Komponenten.**
- **Lokale Designentscheidungen und ihre Auswirkungen auf das Gesamtsystem werden erst spät im Projektverlauf sichtbar.**
- **Zu starke Kopplungen im System führen zu starker Projekt-abhängigkeit von Einzelkomponenten bezüglich Zeit und Kosten.**

Risiko Anforderungsspezifikation

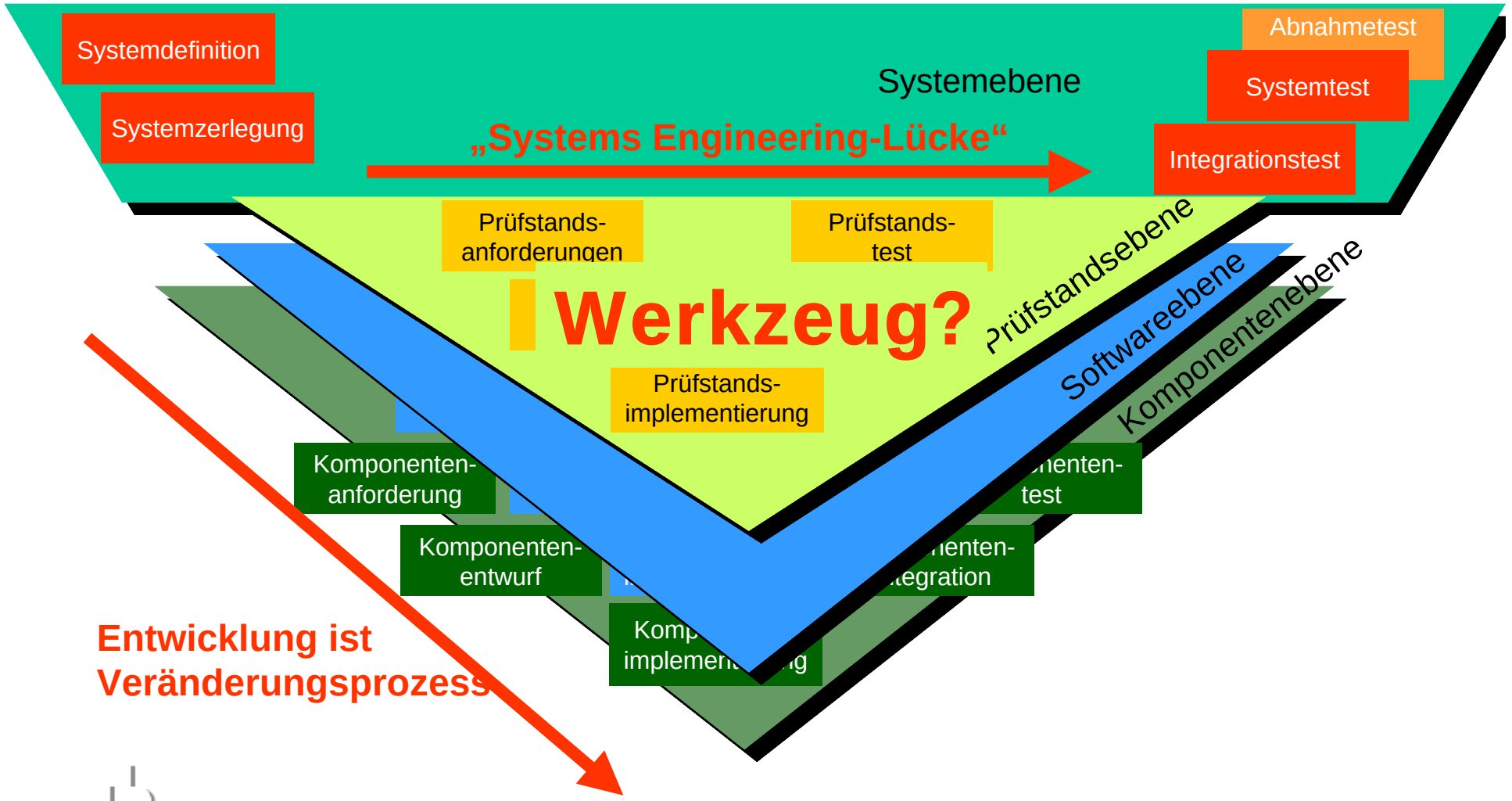
Vollständigkeit und Richtigkeit der Funktionalität, Architektur und Schnittstellendefinitionen

- **Mehr als 80% der im Lebenszyklus eines Embedded Real Time Systems festgestellten Fehler sind Spezifikationsfehler.**
- **Spezifikationsfehler sind problematisch, da sie in der Regel erst spät im Entwicklungsprozess erkannt werden.**
- **Entdeckung von Spezifikationsfehlern:**
  - **ca. 50% beim SW-Entwurf**
  - **ca. 30% beim Testen**
  - **ca. 20% bleiben zunächst unentdeckt.**
- **Testproblematik: Ableitung der Prüfspezifikationen aus der Anforderungsspezifikation. Was falsch spezifiziert wurde, wird falsch geprüft.**

Vertragliche Schnittstellen zu Komponentenzulieferern

- Zu technischen Problemen zusätzlich Projektmanagementprobleme.
- Die Bedeutung eines Zulieferers für den erfolgreichen Abschluss eines Projekts ist wesentlich höher als es seinem finanziellen Anteil entspricht.
  - Technisch /funktionale Abhängigkeiten
  - terminliche Abhängigkeiten (pünktliche Lieferung)
  - Kostendruck auf Zulieferer in Folge der Ausschreibung
- In komplexen Systemen vervielfältigen sich diese Risikofaktoren
  - Viele Zulieferer
  - Gegenseitige Abhängigkeiten

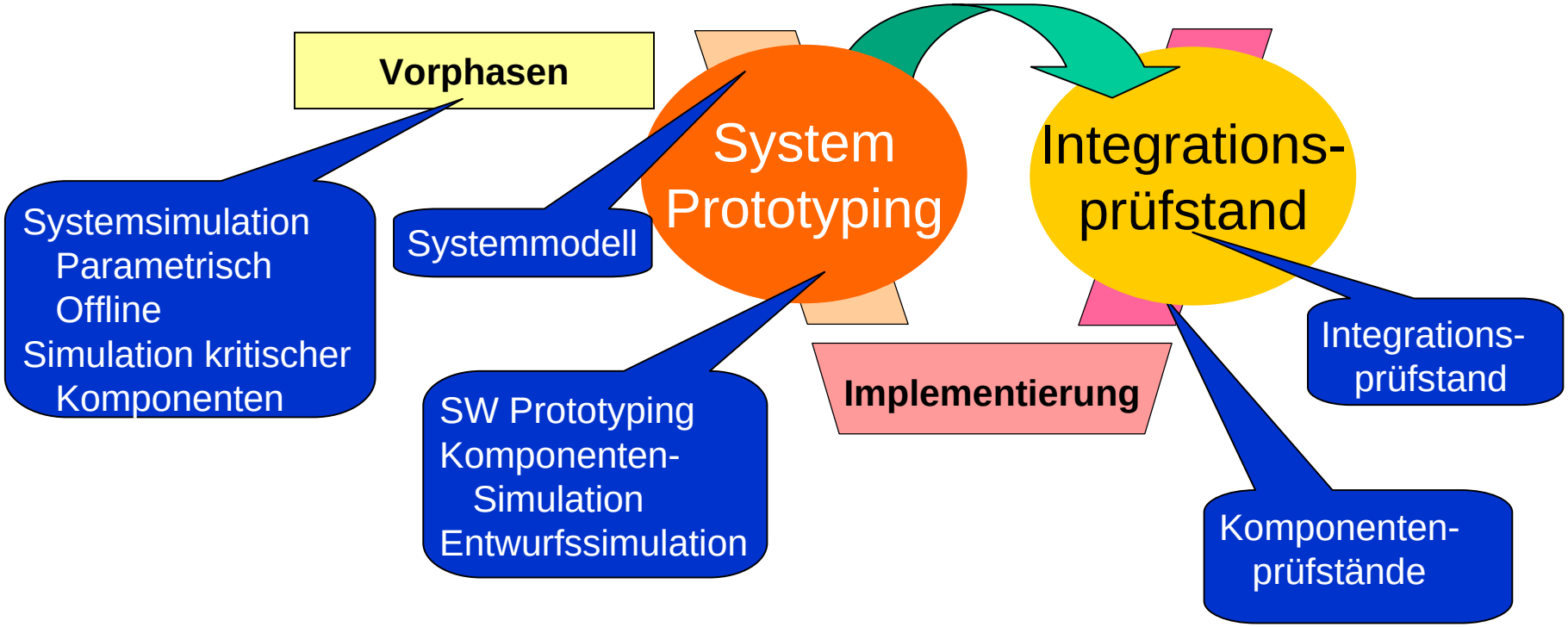
Vorgehensmodell



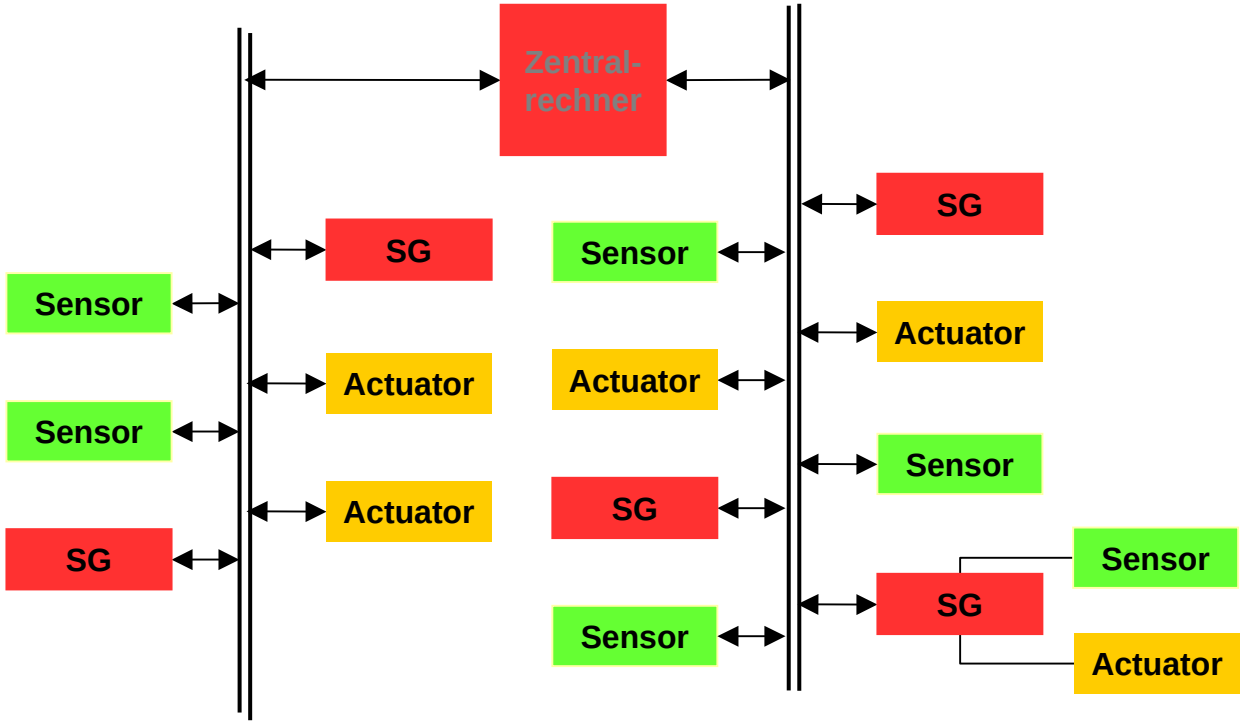
Prüfbare Spezifikationen

- Ziel ist es, die Richtigkeit von Funktionalität und Architektur zu verbessern.
- Ziel ist es, die Managementrisiken der Arbeitsteilung zu reduzieren
- Die modellbasierte Entwicklung ist nicht ausreichend um arbeitsteilige Entwicklungsprozesse hochvernetzter Multiprozessorsysteme zu unterstützen.
- Systems Engineering benötigt Werkzeuge um arbeitsteilige Entwicklungsprozesse komplexer Systeme kompetent zu unterstützen.

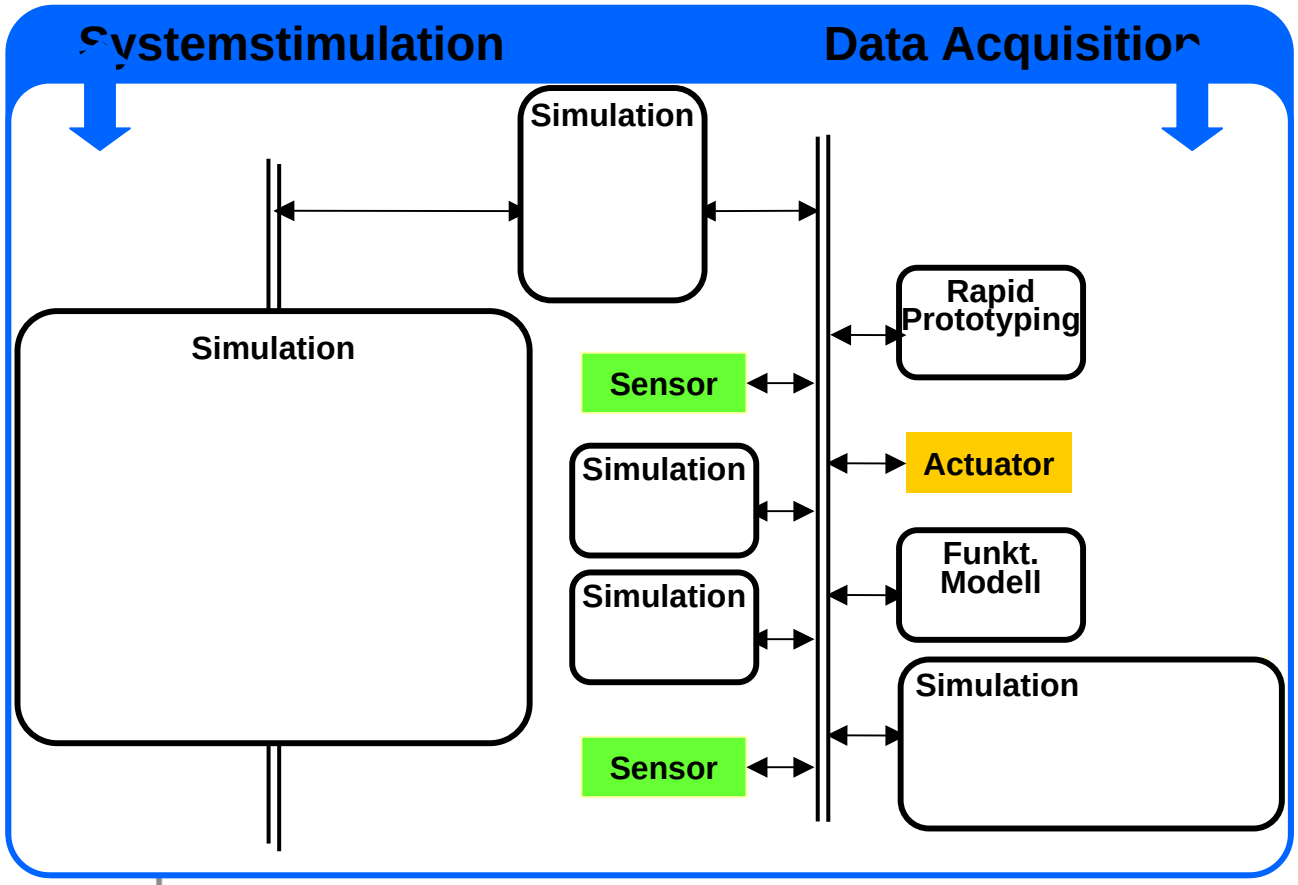
Existierende Werkzeuge



Zu realisierendes System



Systemdesign-Phase



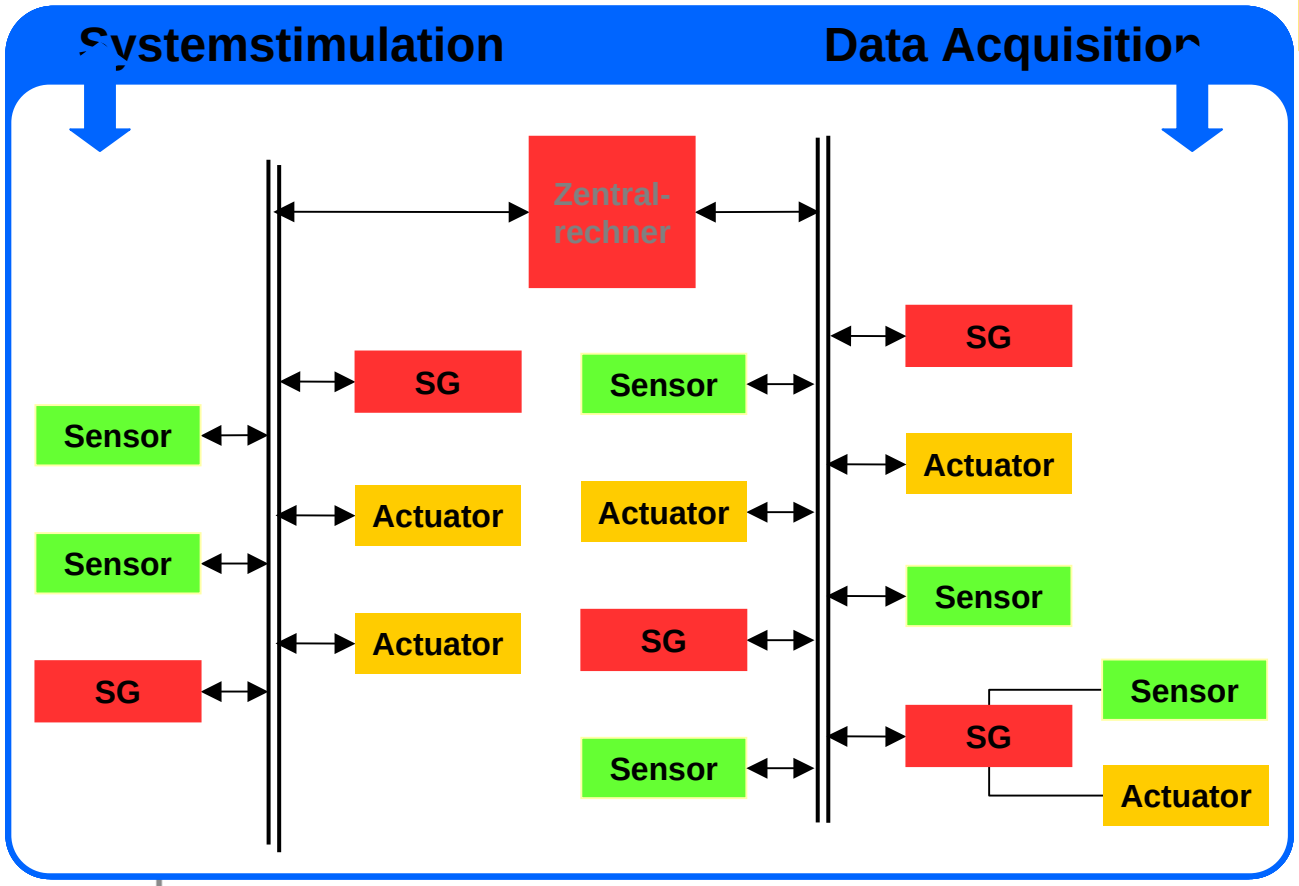
Systemprototyping:

Aufbau der realen Systemarchitektur
- **Busstruktur**
- **Datenleitungen**

Kopplung von
- **Simulationen**
- **funktionalen Modellen**
- **Realen Komponenten**
- **Rapid Prototyping**

Realisierung einer
- **Systemstimulation**
- **Data Acquisition**

Systemintegrations-Phase



Integrations-prüfstand:

Schrittweiser Ersatz der
- **Simulationen**
- **Modelle**

Durch reale
- **Komponenten**
- **Software**

Vorteile

- **Technisches Werkzeug für Systems Engineering**
- **Reduktion von Spezifikationsfehlern**
- **Überprüfbarkeit von**
  - **Systemleistungen (System-Design)**
  - **Spezifikationen, Schnittstellen**
  - **Änderungen während des Entwicklungsprozesses**
- **Reduzierung von Projektrisiken**
  - **Erlaubt Integrations- und Systemtests bei fehlenden Komponenten**
  - **Erwerb und Sicherung von Komponenten Know How auf Systemebene**
- **Integriertes Werkzeug für Design und Integration nutzt**
  - **Systemmodelle,**
  - **Komponentenmodelle (Matlab Simulink)**
  - **SW-Prototyping**
  - **MMI Simulation**

Resümee

Systemprototyping ist die natürliche und zwingende Ergänzung der modellbasierten Entwicklung in der heutigen Form

Die Alternative:

Standish Group International, Inc.

The MathWorks

MATLAB&SIMULINK®

Die Veränderung der Simulationsinfrastruktur von Simulink für die Modellierung physikalischer Systeme

Steve Miller

Application Engineer, Physical Modeling Tools

The MathWorks GmbH, Munich, Germany

Steve.Miller@mathworks.de

Agenda

- Intro to Presenter
- Intro to MATLAB/Simulink
- Modeling Mechanical Systems
- Solver Settings : Zero Crossing Detection
  - Improves Simulation Results
- Discretizing System
  - Shorten Simulation Time
- Questions

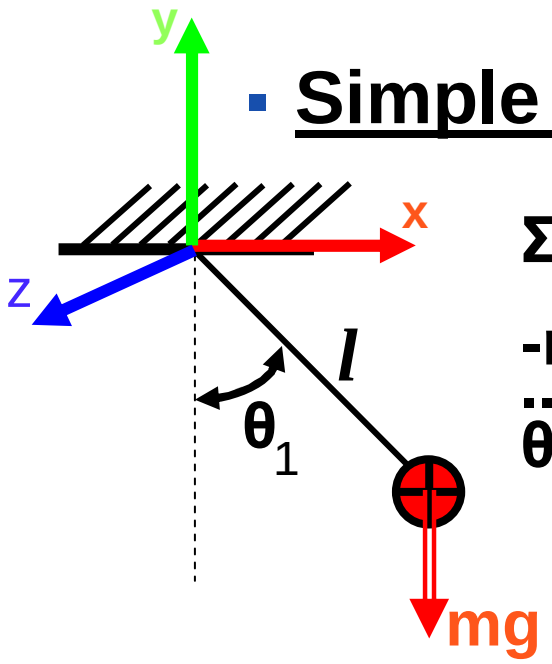
Presenter

- **Steve Miller**
  - Mechanical Engineer
    - BSME Cornell University
    - MSME Stanford University
  - Delphi Automotive
    - Braking Control Systems (2.5 years)
  - MSC.Software ADAMS Consulting (5 years)
    - Project Work (Ford, GM, Hyundai)
    - Onsite Support (BMW, Audi)
  - The MathWorks, GmbH, Munich (6 months)
    - Application Engineer, Electromechanical Systems

Intro to MATLAB/Simulink

- **MATLAB (Matrix Laboratory)**
  - High performance language for technical computing
  - Integrates computation, visualization and programming
  - Typical uses :
    - Math/Computation
    - Modeling/Simulation
    - Algorithm Development
    - Data Analysis
- **Simulink**
  - Software package for modeling, simulating and analyzing dynamic systems
  - Graphical environment

Mechanical Systems in Simulink

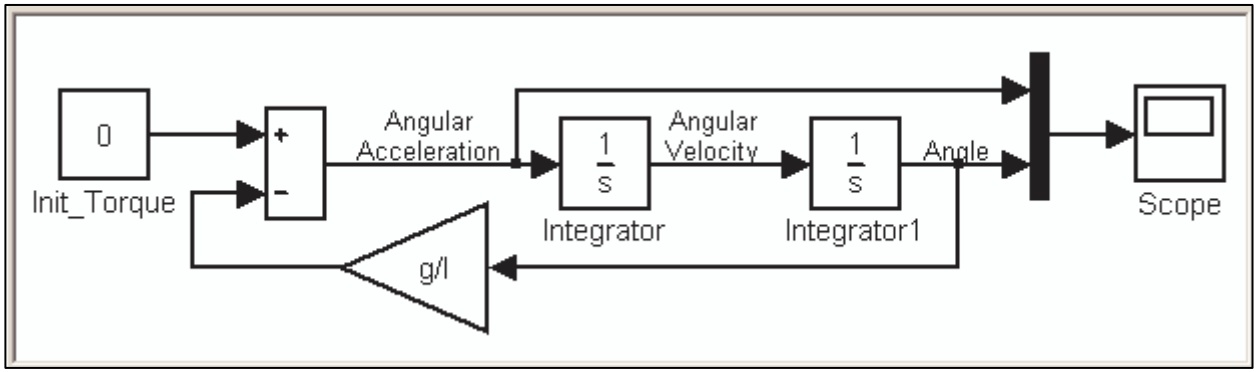


- **Simple Pendulum**

$$\Sigma T_z = I\ddot{\theta}_1$$

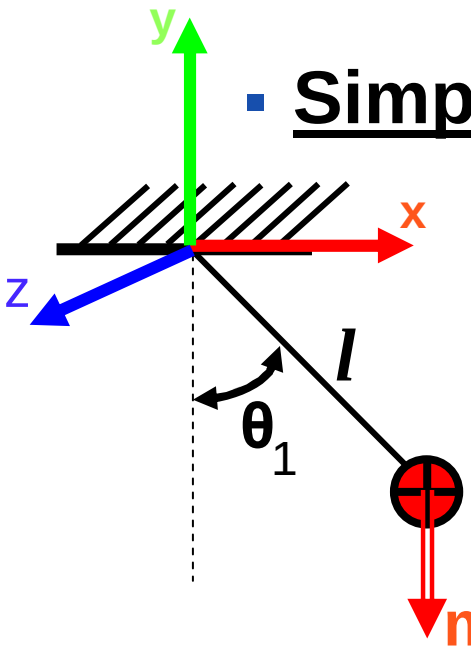
$$-mg*\sin(\theta_1) = ml^2\ddot{\theta}_1$$

$$\ddot{\theta}_1 = -\frac{g}{l}\sin(\theta_1)$$



Mechanical Systems in SimMechanics

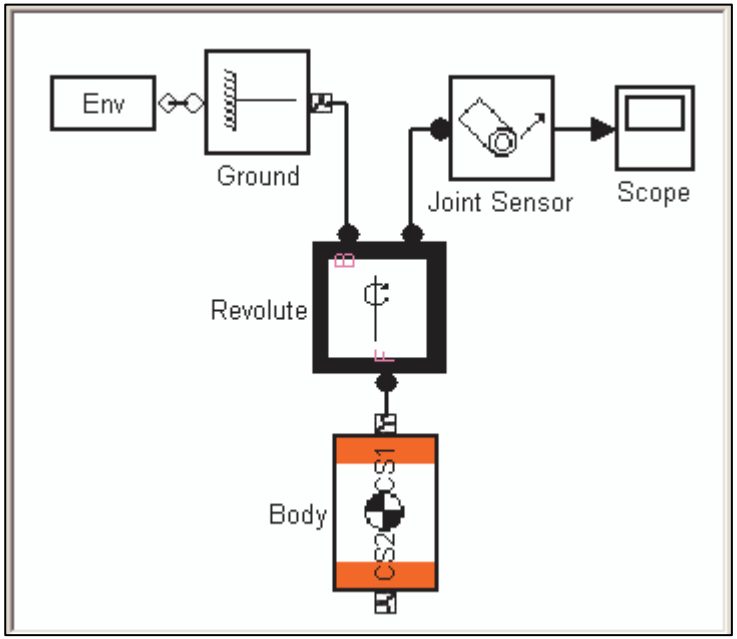
- **Simple Pendulum**



$$\Sigma T_z = I\ddot{\theta}_1$$

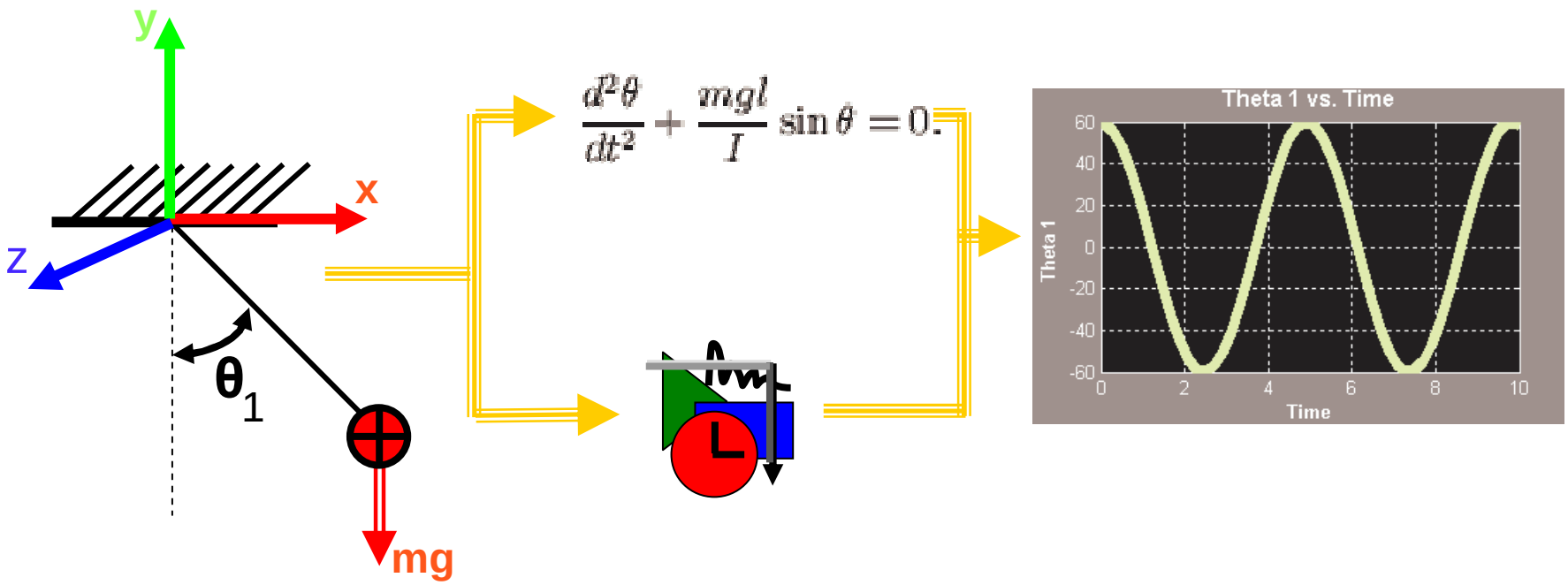
$$-mg^*\sin(\theta_1) = ml^2\ddot{\theta}_1$$

$$\ddot{\theta}_1 = -\frac{g}{l}\sin(\theta_1)$$

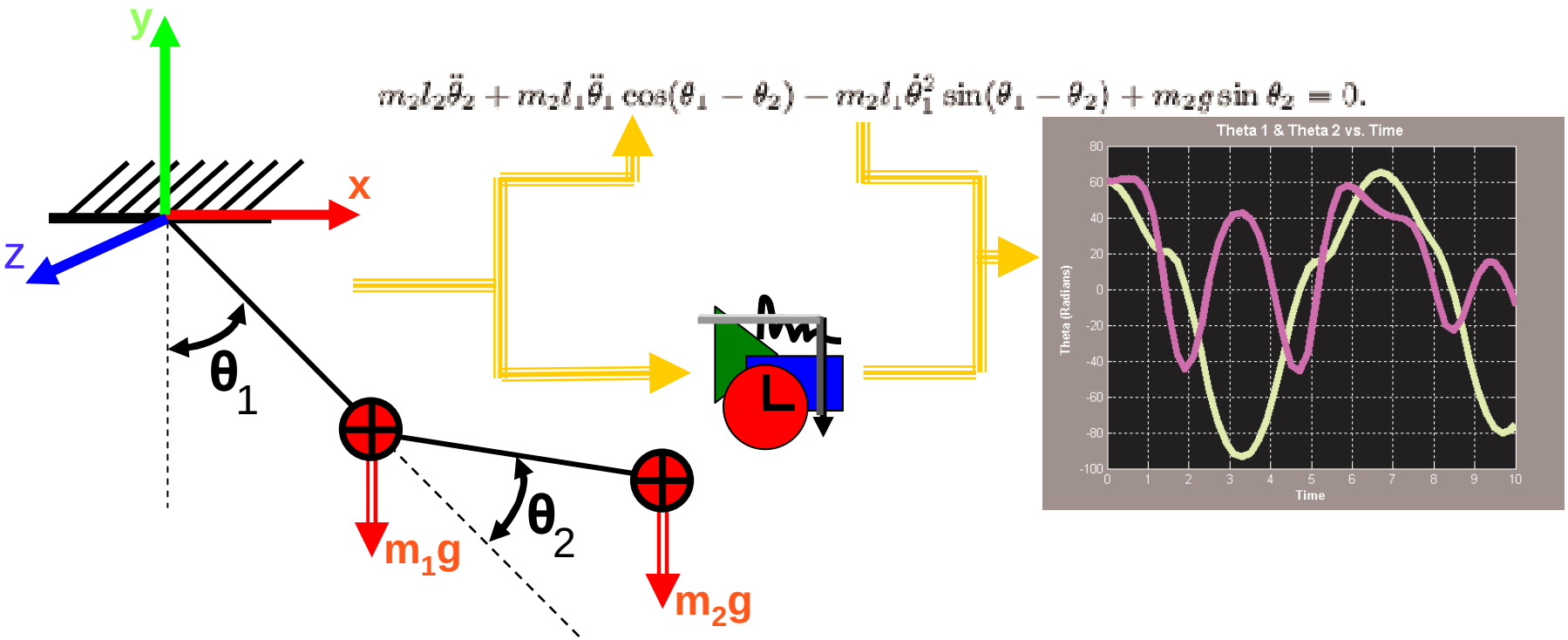


- Model is easier to read than equations
- Quicker to create
- More intuitive – easier to explain to other engineers

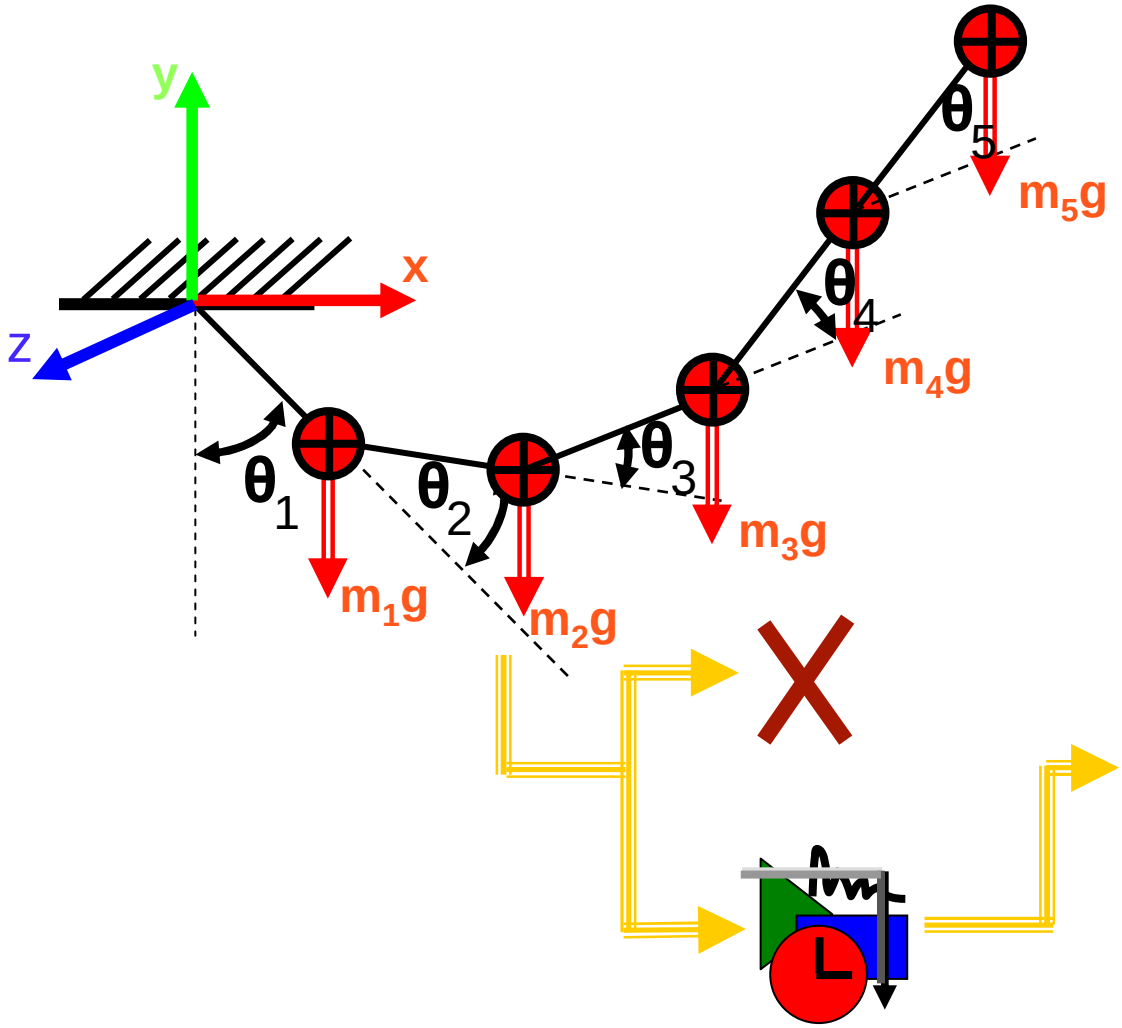
Mechanical Systems in SimMechanics

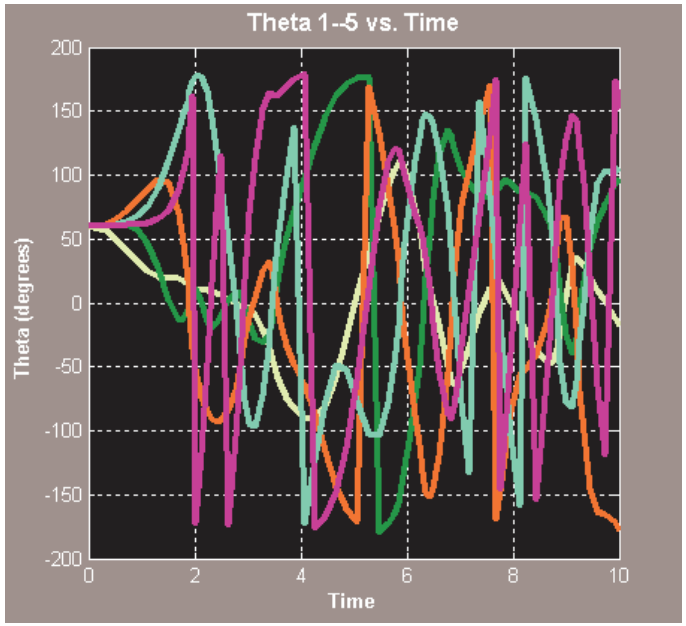


Mechanical Systems in SimMechanics



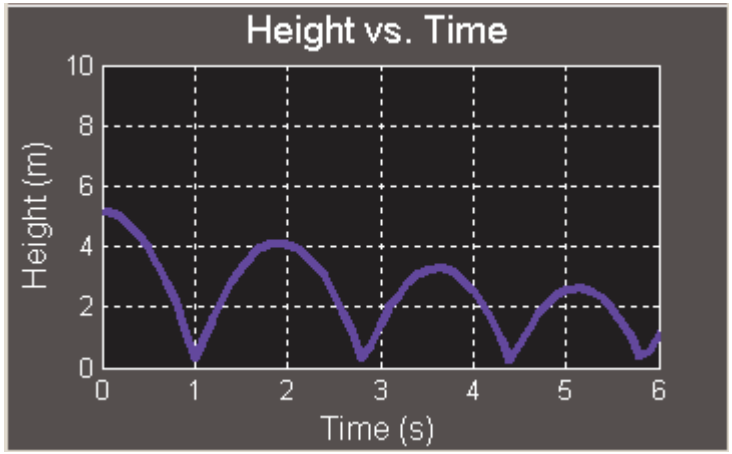
Mechanical Systems in SimMechanics

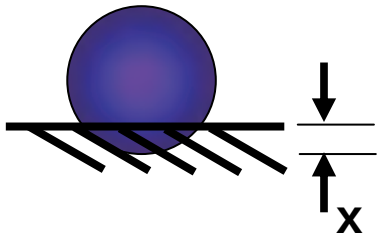




Zero Crossing Detection

Model :





Impact Force = k*x

Problem : Simulation does not accurately detect moment of impact

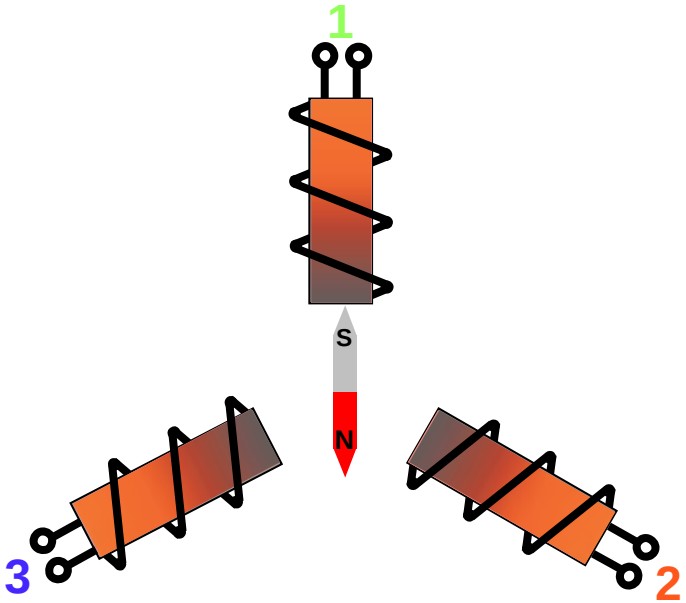
Solution : Enable Zero Crossing Detection to allow the solver to find the exact moment of impact

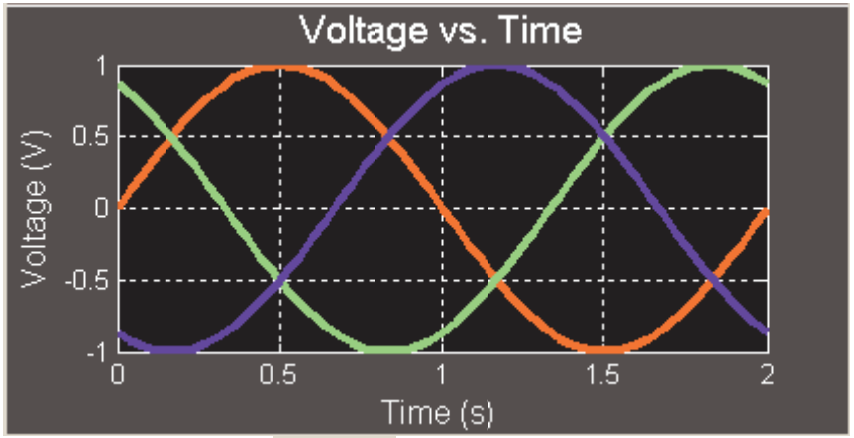
Simulating Electrical Systems

- **Simulation Challenges**
  - High frequencies require small step sizes
  - Many nonlinear elements (power electronic switches)
  - Results in long simulation times
- **SimPowerSystems Solution**
  - In SimPowerSystems, can discretize system using Tustin Method
  - System can be simulated using discrete solver
  - Simulation can run 10x – 100x faster

Simulation of Electrical Systems

- **Permanent Magnet Synchronous Motor**

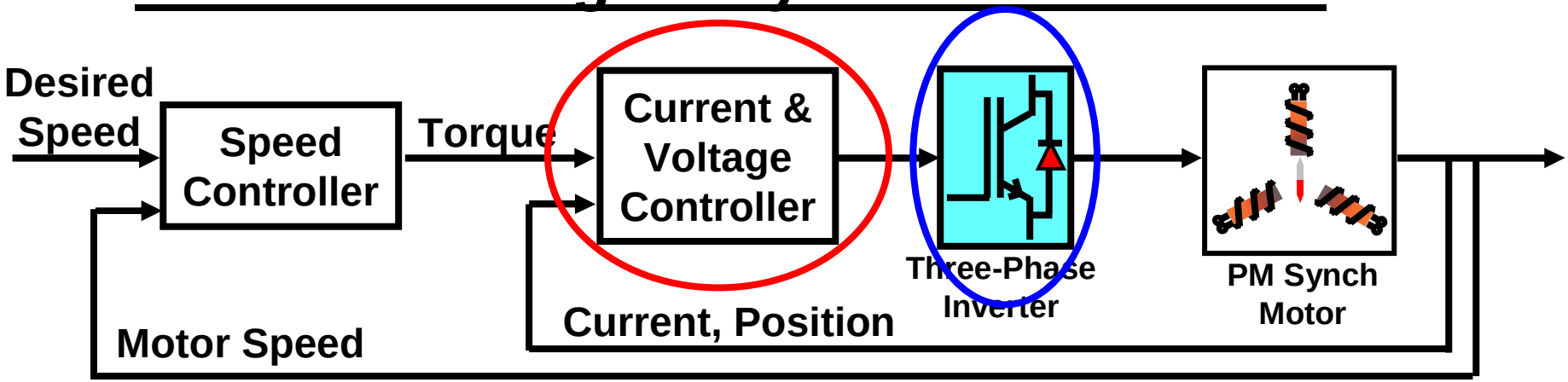




- Rotor is a permanent magnet
- Stator consists of three electromagnets
- Electromagnet voltage is a sine wave
- Signals are phased so that flux of stator and rotor are kept 90º out of phase for maximum torque

Simulation of Electrical Systems

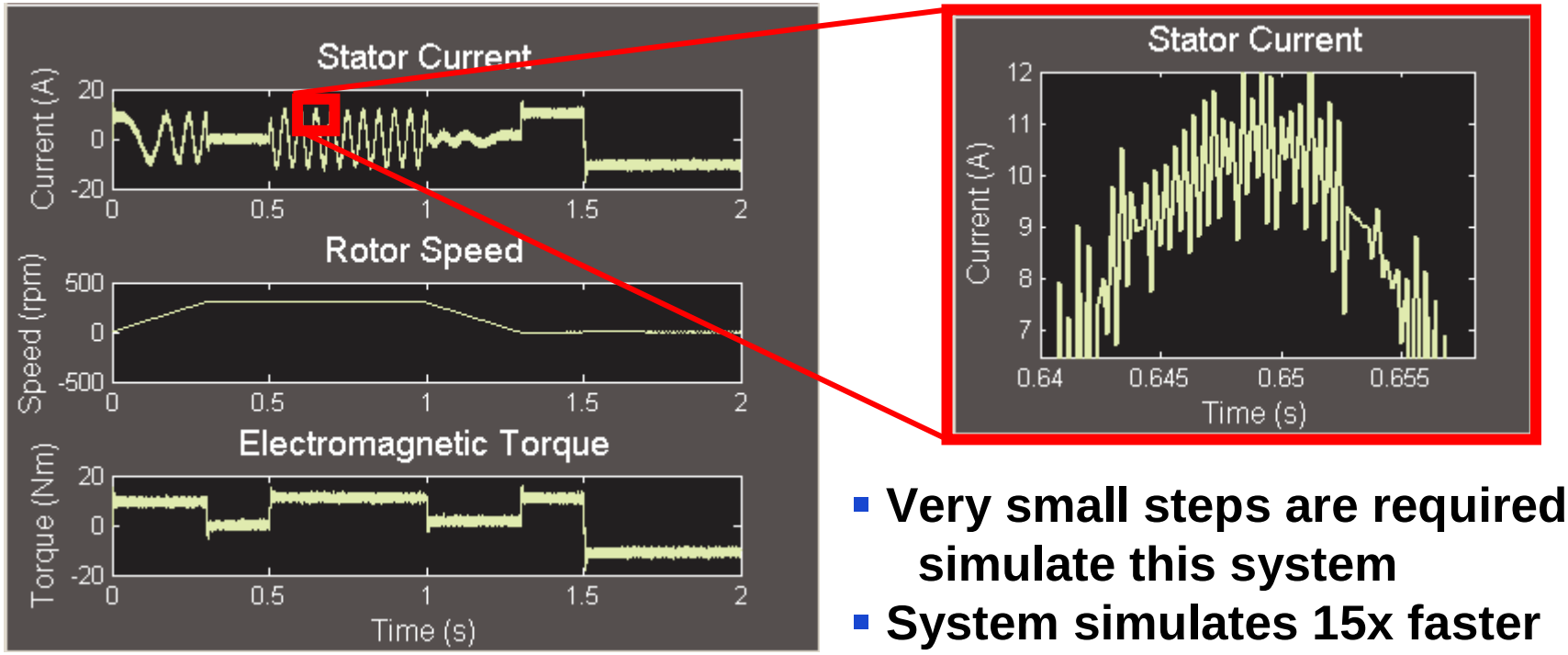
- **Permanent Magnet Synchronous Motor**



- Current Control is required to maintain phase and frequency of voltage signal
- Switching elements at electromagnetic terminals are very nonlinear

Simulation of Electrical Systems

- **Simulation**



- **Very small steps are required to simulate this system**
- **System simulates 15x faster when discretized**

Conclusion

- Modeling based on system instead of equations results in models that are easier to read and understand
- Using Zero Crossing Detection in Simulink can improve the results of a simulation
- Discretizing electrical systems models can shorten simulation times by a factor of 10 to 100

Questions?

Thank You For Your Attention!

Steve Miller

Application Engineer, Physical Modeling Tools

The MathWorks GmbH, Munich, Germany

Steve.Miller@mathworks.de

T-Systems / SSC-Embedded Systems
Durchgängiger modellbasierter Entwicklungsprozess mit Auto-Codegenerierung

T · · · Systems

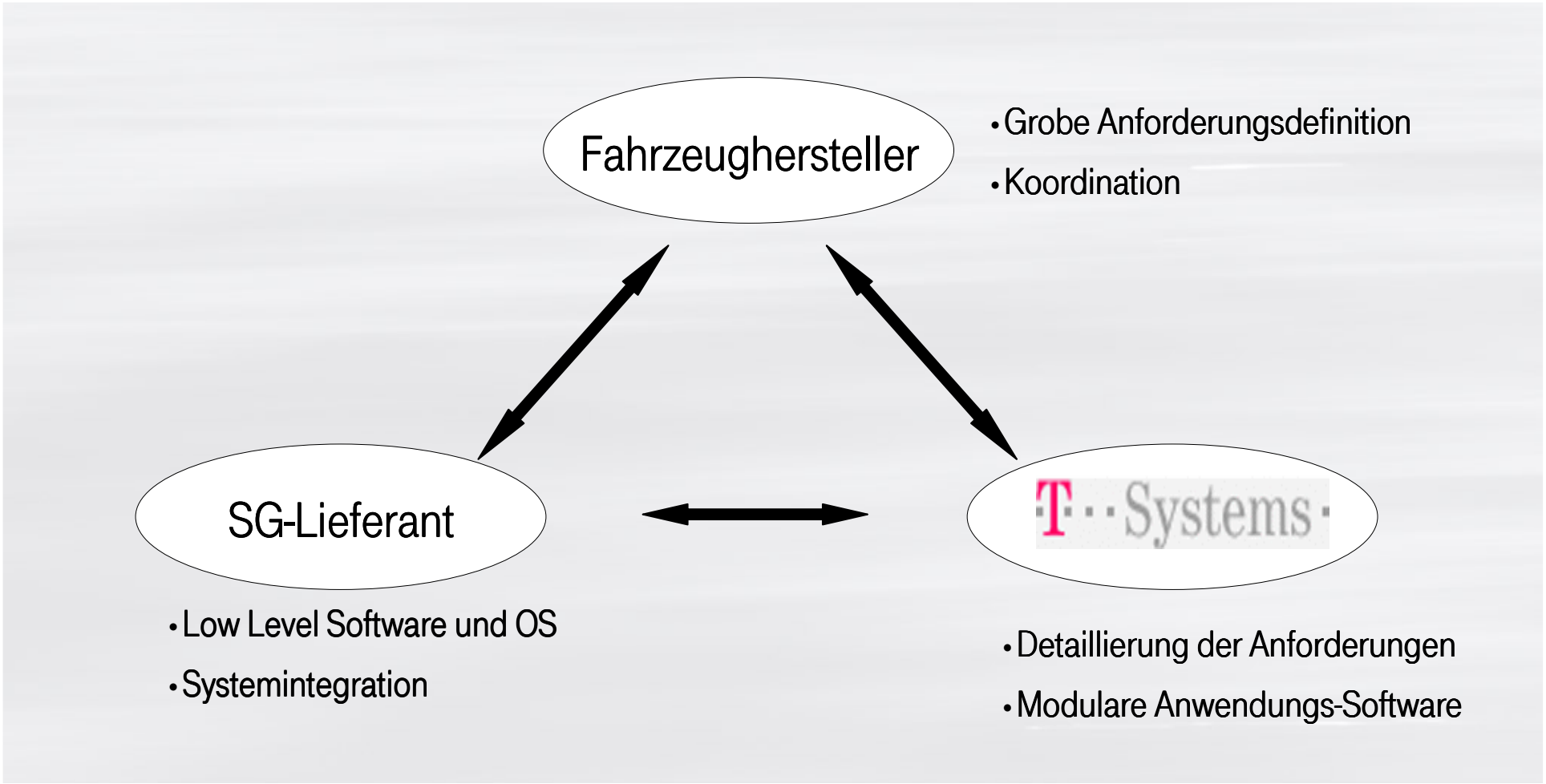
R. Moser / SSC - Embedded Systems
ASIM2006, 20.02.2006

Gliederung

Durchgängiger modellbasierter Entwicklungsprozess mit Auto-Codegenerierung

- Einleitung
- Anforderungen an den modellbasierten Entwicklungsprozess
- Prozessschritte des modellbasierten Entwicklungsprozesses
- Modularisierung und Qualitätskriterien
- Ausblick und geplante Erweiterungen
- Beispielprojekt Klimaregelung

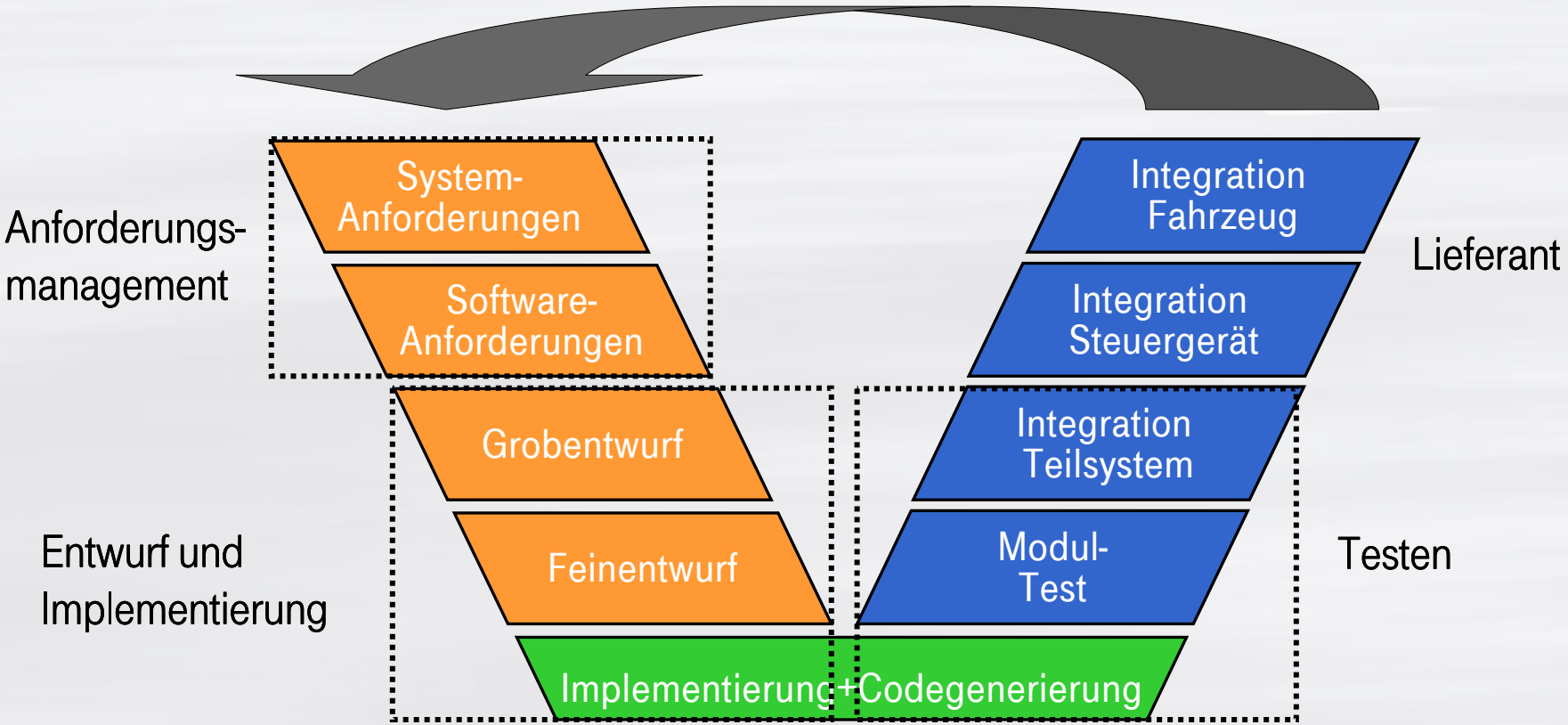
Einleitung
Entwicklung von Anwendungs-Software für eingebettete Systeme



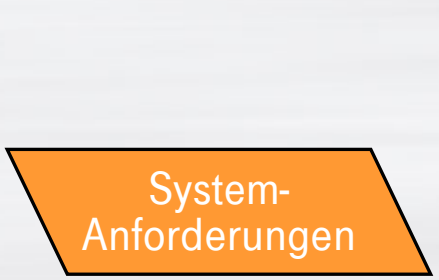
Mindestanforderungen an den Entwicklungsprozess

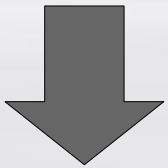
- Unterstützung bei der **Modularisierung**
- **Durchgängigkeit** im Entwicklungsprozess
- **Automatisierung** wiederkehrender Prozessschritte
- Definition und Prüfung der **Qualitätskriterien**
- Unterstützung von möglichst **standardisierten Schnittstellen**

Prozessschritte des Entwicklungsprozesses
Embedded Autocode for Software Systems development (EmbASSy)



Prozessschritte des Entwicklungsprozesses
Prozessschritt Anforderungsmanagement





Kunde+PM

Grobes
Lastenheft

Detailliertes
Lastenheft

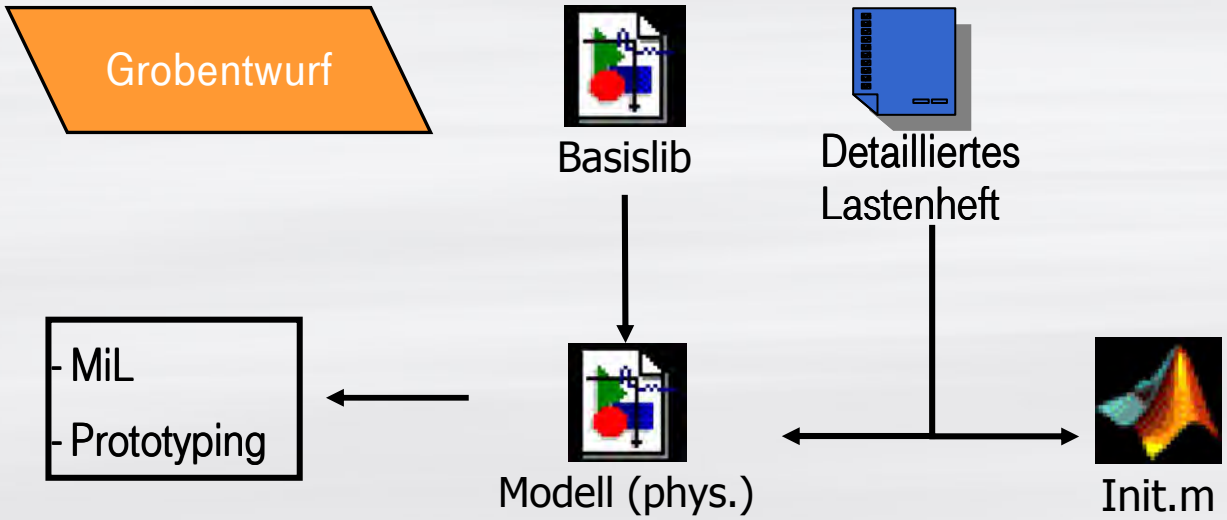
Ziele:

- Modularisierung des Systems
- Definition der Haupt-Anforderungen

Ziele:

- Verfeinerung der Anforderungen pro Modul/Komponente
- Definition von abprüfbaren Anforderungen

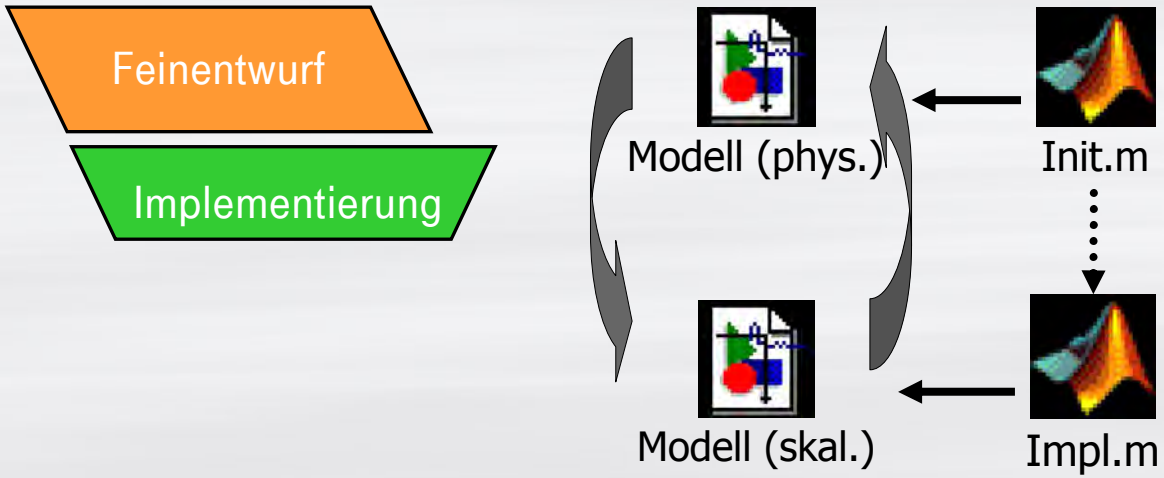
Prozessschritte des Entwicklungsprozesses
Prozessschritt Entwurf und Implementierung
Aktivität: Grobentwurf



Ziele:

- Entwurf physikalisches Modell
- Definition der Initialisierungswerte (Trennung von Modell und Daten)
- Prüfung der Funktionalität (MiL, Prototyping)

Prozessschritte des Entwicklungsprozesses
Prozessschritt Entwurf und Implementierung
Aktivität: Feinentwurf und Implementierung

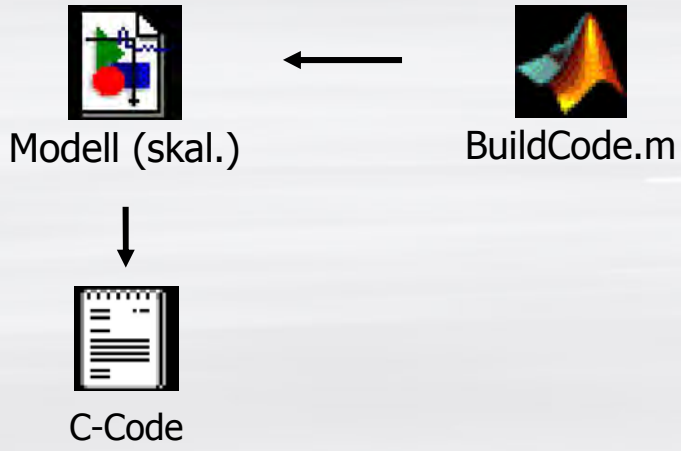


Ziele:

- Vorbereitung für die Code-Generierung
- Abbildung des physikalischen Modells auf das skalierte Modell (Ganzzahlarithmetik)
- Unterstützung für die Parameterverstellung (Applikation)
- Generierung der Schnittstellen

Prozessschritte des Entwicklungsprozesses
Prozessschritt Codegenerierung

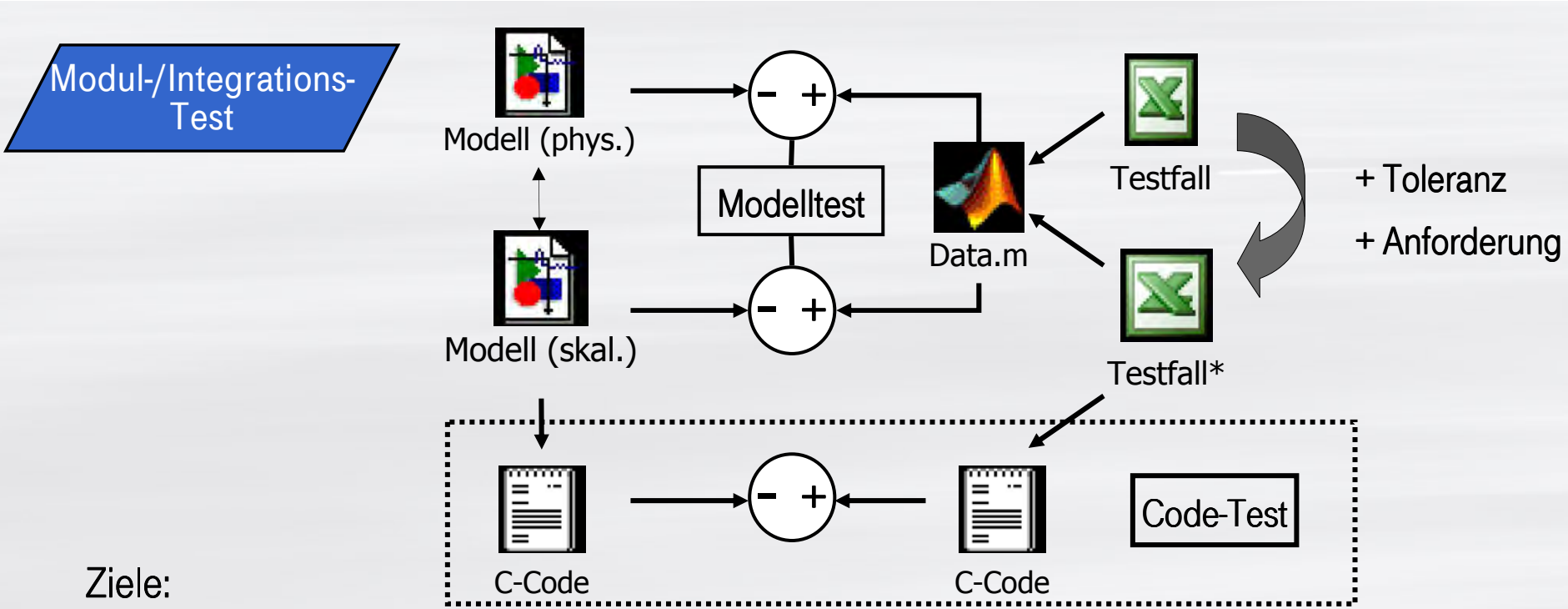
Codegenerierung



Ziele:

- Konfiguration für Codegenerator (Embedded Coder E-RTW) und Makefile
- Post-Build Optimierungen

Prozessschritte des Entwicklungsprozesses
Prozessschritt Testen



Ziele:

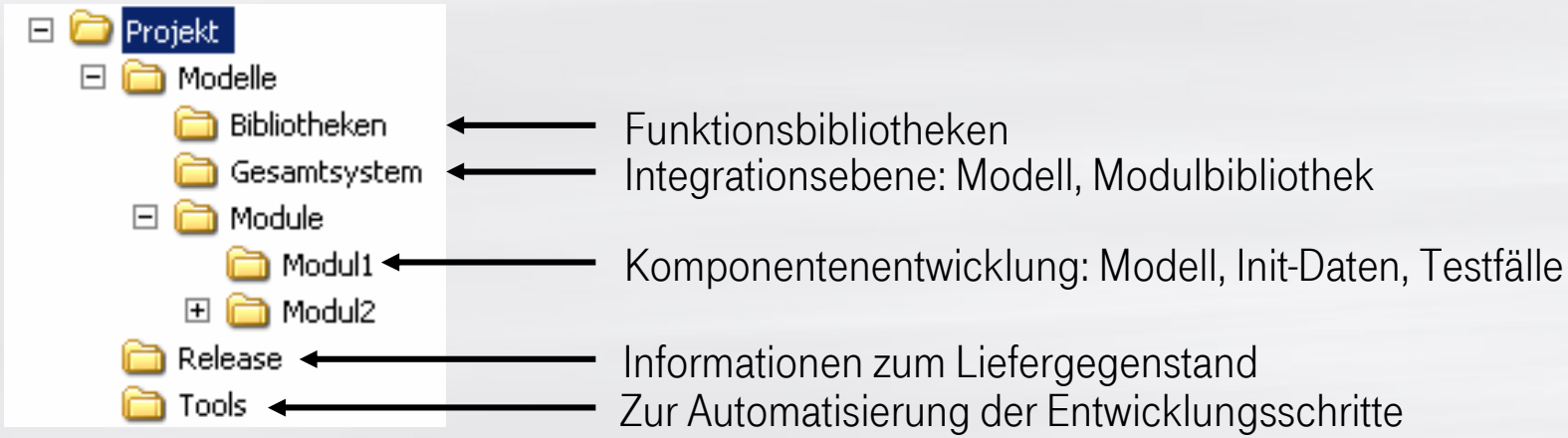
- Durchgängigkeit (Wiederverwendung der Testfälle)
- Je Entwurfsschritt ein Testschritt
- Automatische Testdurchführung und Testdokumentation

Prozessschritte des Entwicklungsprozesses
Prozessschritt Release (Delivery)

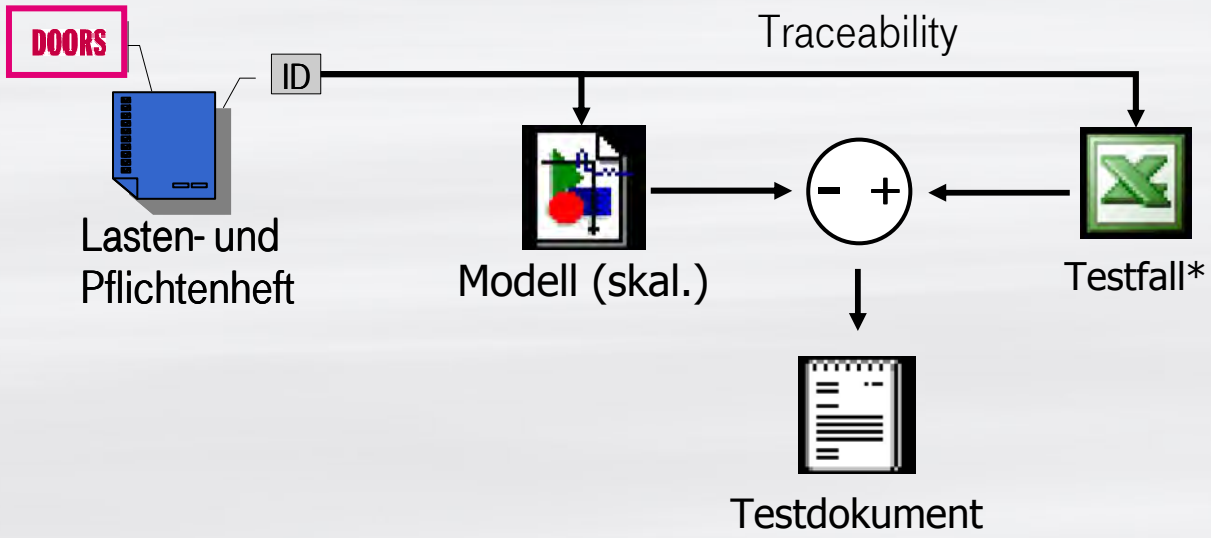
- Manuelle Erstellung der Konfiguration
- Skriptbasierte Erstellung der Lieferung
  - Abschlusstest auf allen Ebenen
  - Zusammenstellung und Dokumentation der Liefergegenstände

Modularisierung
Unterstützung durch den Entwicklungsprozess

- Hierarchische Ablage der Projektdaten
- Weiterentwicklung und Test auf Modulebene
- Integration in das Teilsystem über eine Modul-Bibliothek



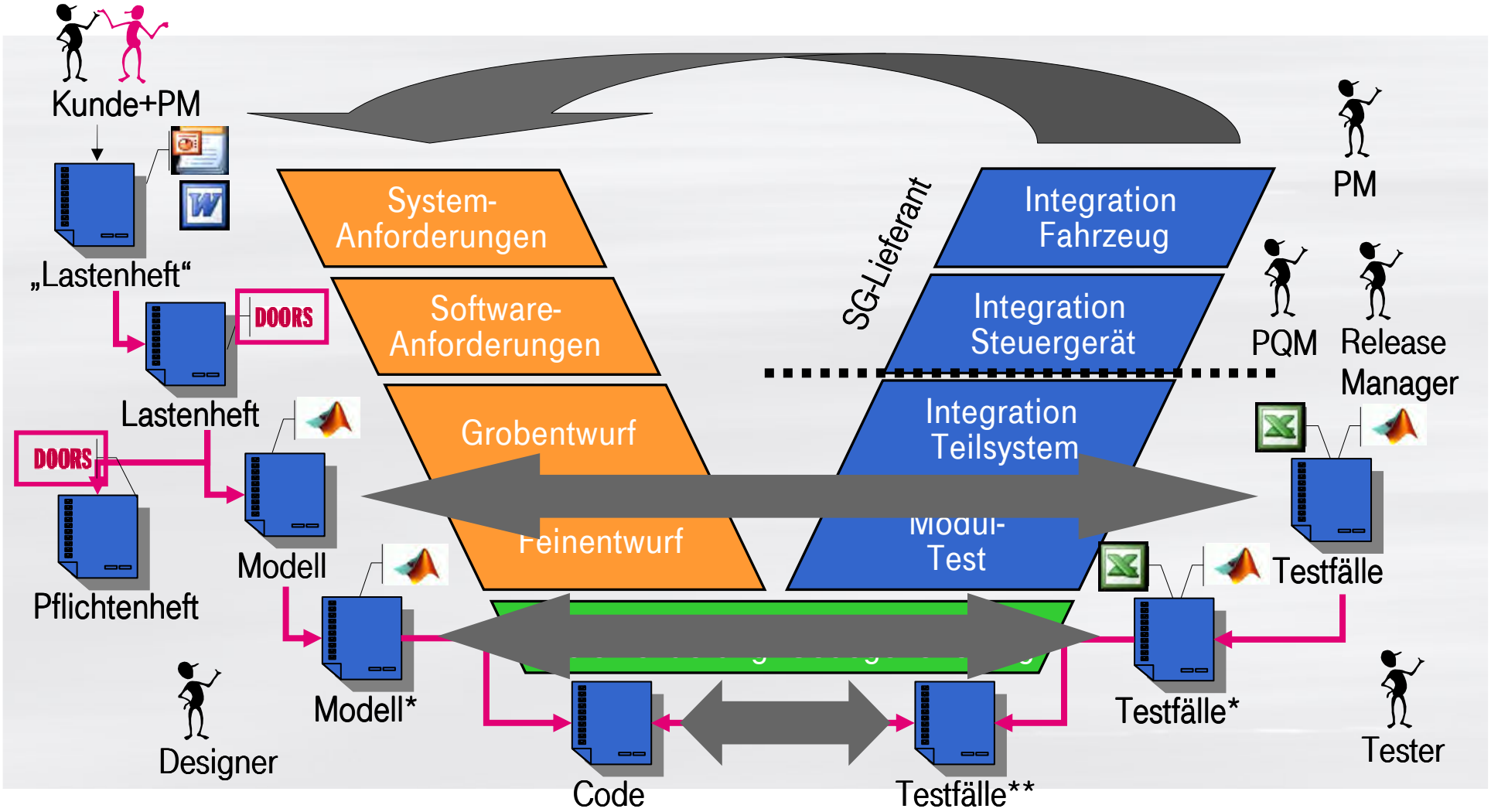
Qualitätskriterien
Unterstützung durch den Entwicklungsprozess



Ziele:

- Verbesserte Nachvollziehbarkeit der Anforderungen auf Modell und Test
- Erleichtertes Änderungsmanagement
- Abdeckung der Lasten/Pflichten als Qualitätsmaß (s. Testdokument)

Überblick von EmbASSy (Rollen)
Embedded Automotive Software System development (EmbASSy)



Weiterentwicklung von EmbASSy
Ausblick und Erweiterungen

- Ausdehnung der Schnittstellen von EmbASSy (Standards wie ASAM, AUTOSAR)
- Portierung auf aktuelles Matlab-Release
- Integration einer detaillierten Abdeckungsprüfung (Performance Toolbox bzw. Verification & Validation-Toolbox)
- Etablierung, Verankerung von EmbASSy im Unternehmen als Vorgehensmodell für modellbasierte SW-Entwicklung

R. Moser / SSC - Embedded Systems
ASIM2006, 20.02.2006

KNORR-BREMSE

Modellierung des Wärmehaushalts einer Scheibenbremse für Nutzfahrzeuge

Lösung eines thermischen Ersatznetzwerkes mit Dymola

ASIM/GI Fachgruppentreffen

Dipl.-Ing. Stephan Pitzing

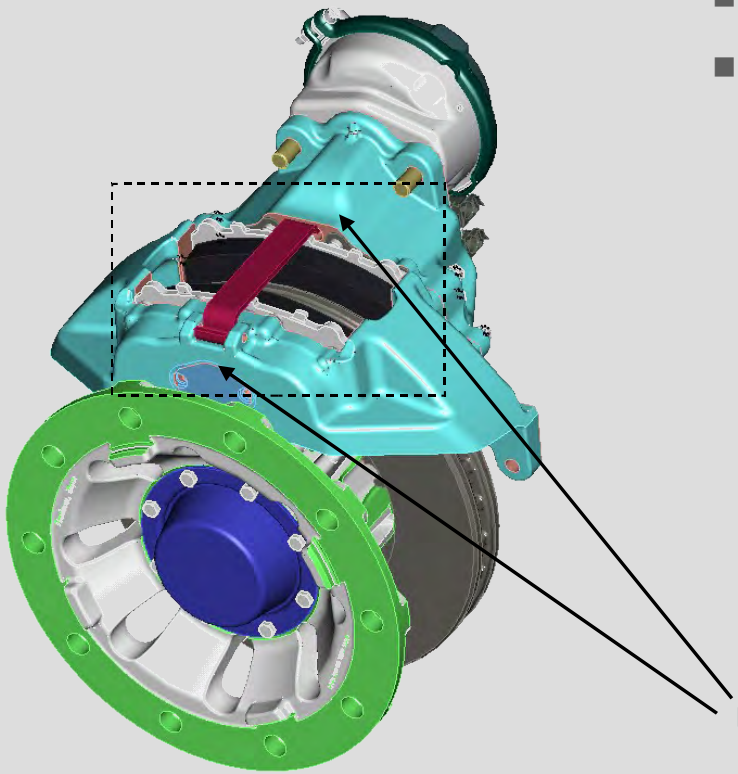
München, den 20.02.- 21.02.2006

Agenda

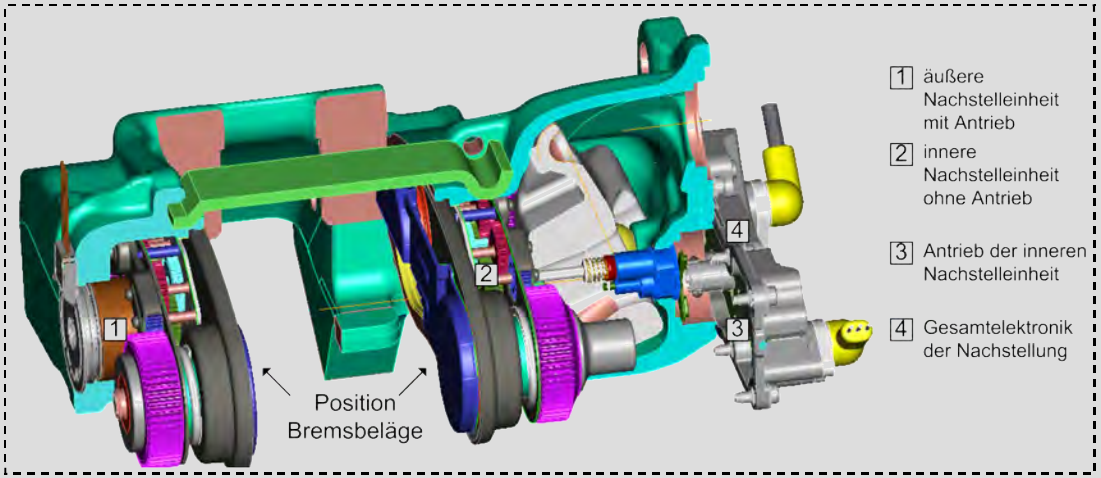
Motivation	■ Aufbau der Radbremse mit elektromechanischer Verschleißnachstellung ■ thermische Simulation im Vorentwicklungsprozess
Modellbildung und Ergebnisse	■ physikalischer Modellierungsansatz ■ Verknüpfung von finiten und konzentrierten Elementen ■ Ergebnisse der Simulation
Ausblick und Möglichkeiten	■ Integration in die Bremsensteuerung ■ Informationsgewinn ohne zusätzlichen Aufwand an Sensorik

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Aufbau der elektromechanischen Radbremse



- Druckluftbetätigung, Felgengröße 22,5 Zoll
- Festsattelprinzip (bewegliche Scheibe), Zuspannkraft > 270 kN
- Belagverschleiß max. 19 mm pro Seite; Ausgleich durch elektromotorisches Bewegen der Zuspannmechanik



- elektromechanische Verschleißnachstellung

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Ausgangspunkt und Problemstellung

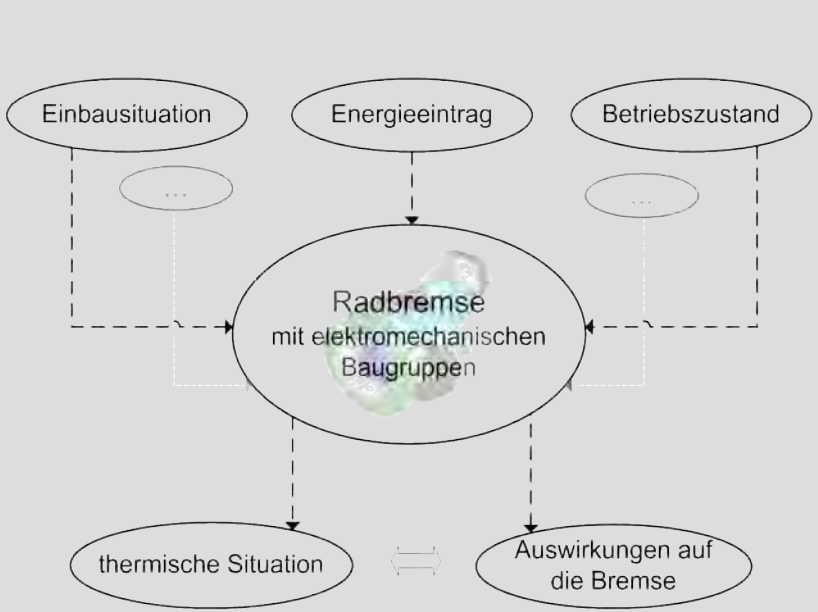
- großer Energieeintrag in die Radbremse
- Aufteilung der eingebrachten thermischen Energie nach geometrisch/physikalischen Zusammenhängen
- hohe thermische Belastung einzelner Baugruppen
- begrenzte Einsatzfähigkeit elektromechanischer Komponenten (elektromechanische Verschleißnachstellung) bei hohen Umgebungstemperaturen

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Thermische Simulation im Vorentwicklungsprozess

Entwicklungsbegleitende Simulation

- Analyse konstruktiver Änderungen auf ihren Einfluss, mögliche thermische Problemfälle schon vor dem Erstmusterstadium erkennen
- Reduzierung von Versuchsaufwand
- Ziel 1: quantitative Aussagen zu Temperaturen an sensiblen Baugruppen
- Ziel 2: vergleichende Aussagen zu verschiedenen Konstruktionen
- Wichtig: zeitlicher Verlauf von Temperatur und Wärmestrom



Ziele der Arbeit

Quantifizieren
Systematisieren

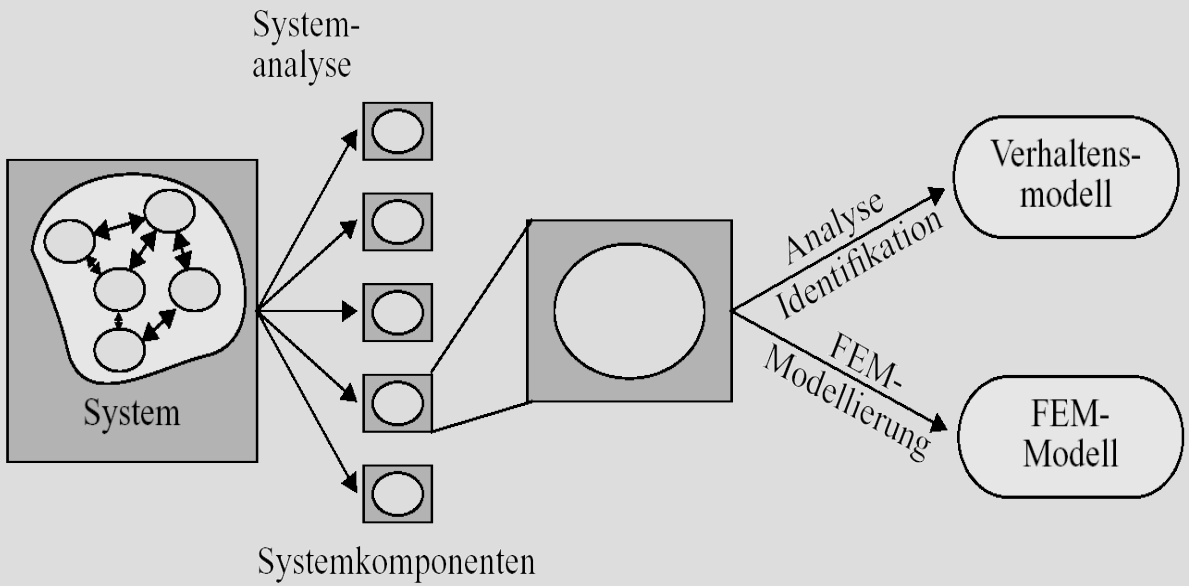
Beschreiben
Modellieren

Analysieren
Einflussmöglichkeiten
beurteilen

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Ansatz: Verknüpfung von Verhaltensmodellen mit Modellen finiter Elemente

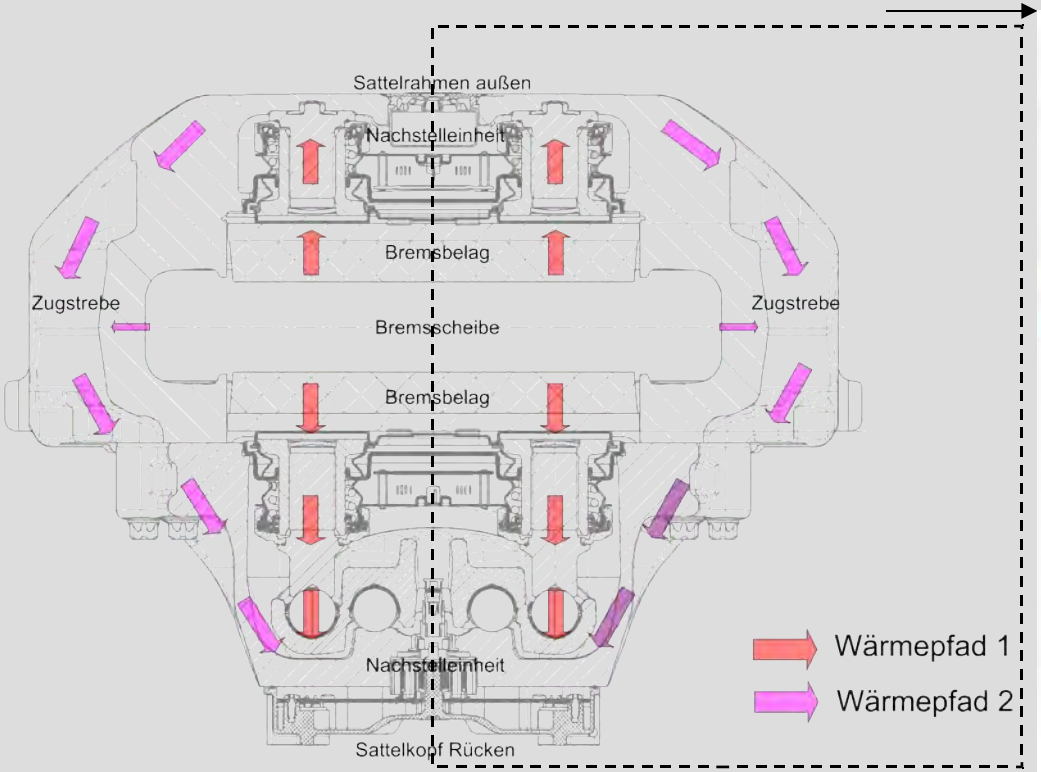
Systemsimulation mit Hilfe von konzentrierten Elementen

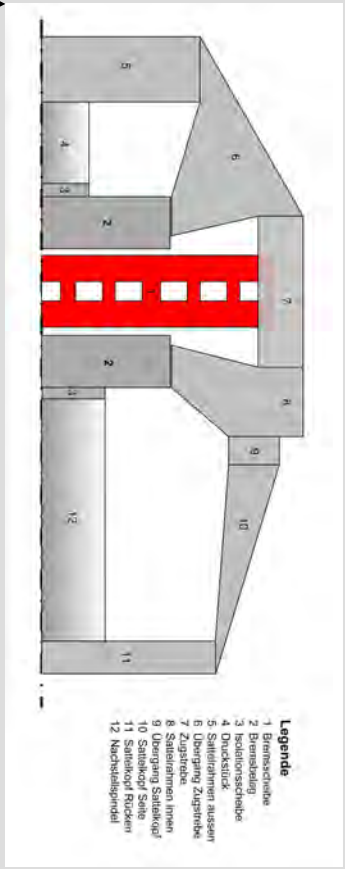


- Nachvollziehbarkeit der wärmeübertragenden Parameter von Bedeutung (Nutzung des VDI Wärmeatlas für konvektive Wärmeübergänge)
- Thermisches Ersatznetzwerk mit durch FE - Methoden bestimmten Parametern
- Energieflussorientierte Methode (Potential- und Flussgröße, Ausnutzung der Netzwerkanalogien)

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Wärmestromverteilung ist abhängig vom Bremsenzustand





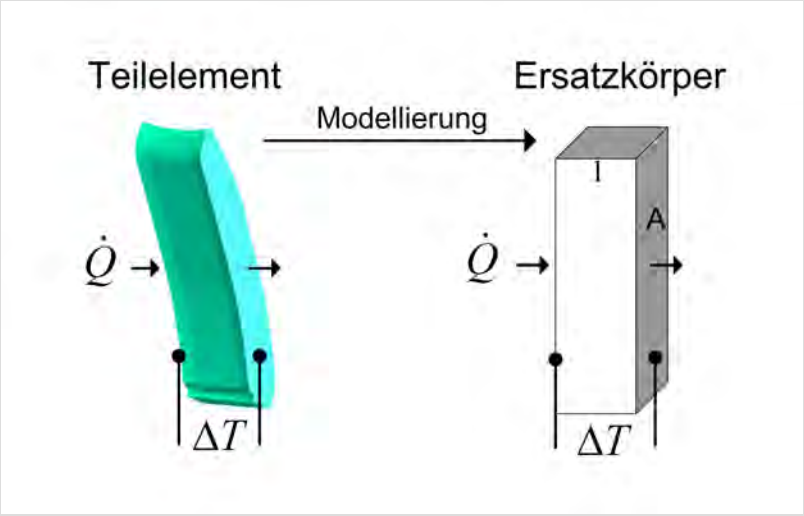
- Simulationsmodell muss verschiedene Betriebszustände der Radbremse abdecken können
- Änderung des Zustandes bewirkt gleichfalls eine Änderung des thermischen Ersatznetzwerkes
- Beschränkung des Modells auf Baugruppen entlang der Hauptwärmeströme (starke Vereinfachung)

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Verbindung von FEM und KEM zur Parameterbestimmung

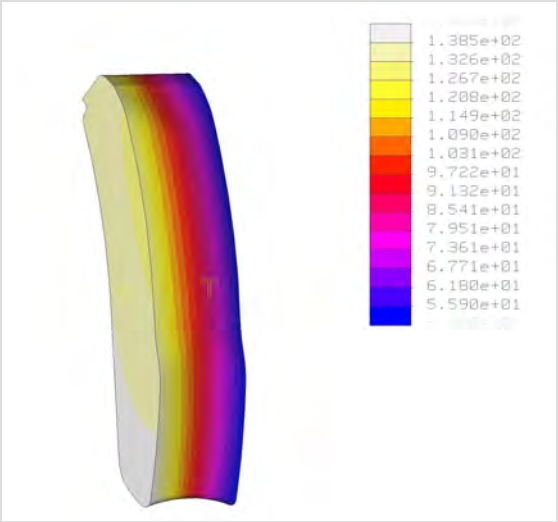
Methode: Geometrischer Ersatzkörper

$$R_{th} = \frac{l}{\lambda \cdot A}$$



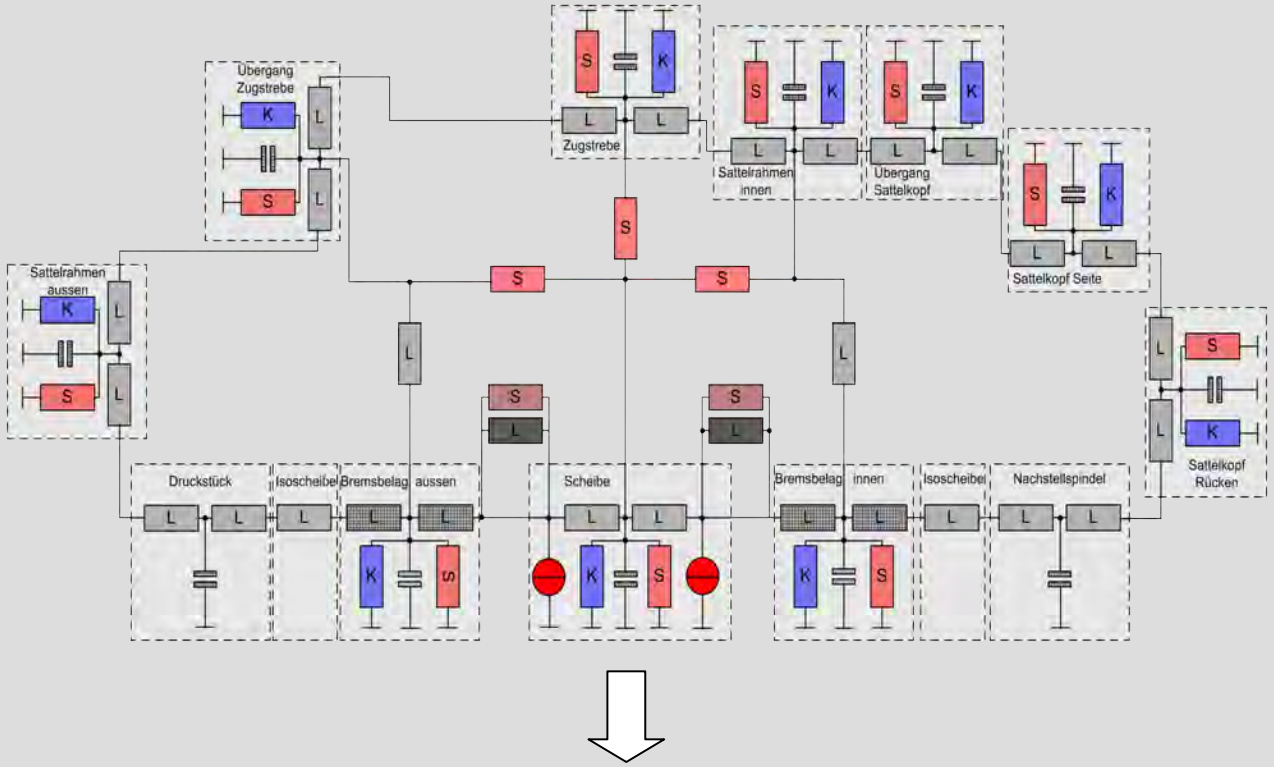
Methode: stationäre FEM Analyse

$$R_{th} = \frac{\Delta T}{\dot{Q}}$$



Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

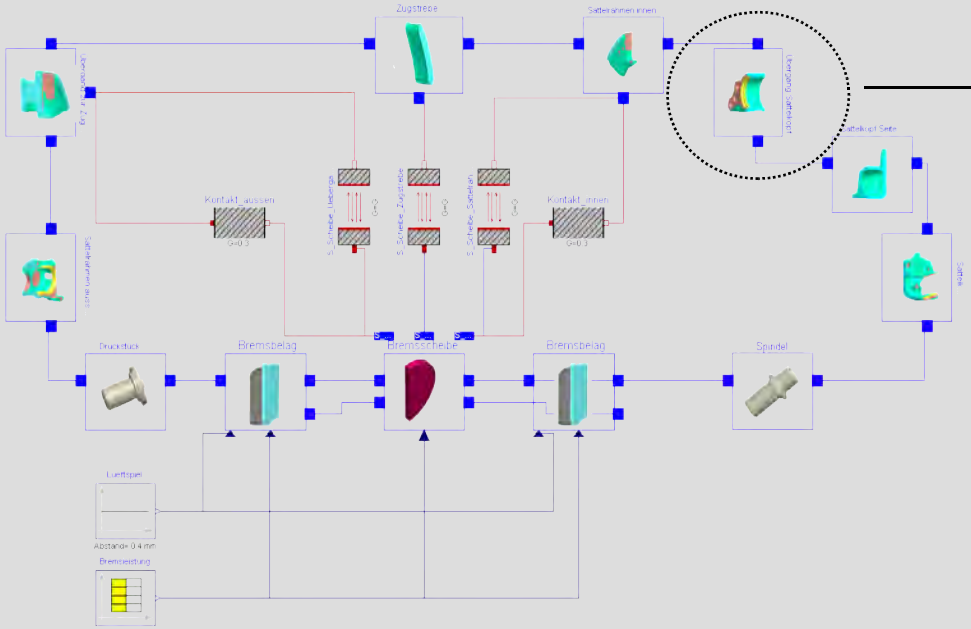
Thermisches Ersatznetzwerk der Radbremse



- Berücksichtigung von:
  - Temperaturabhängigkeit der Modellparameter
  - Bremsenzustand (Bremse angelegt, Bremse geöffnet) und weiteren Parametern wie dem Lüftspiel
  - Strahlung in die Umgebung sowie zwischen Bauteilen
- Erweiterbarkeit Richtung Radnabe / Radlager / Felge / Achse

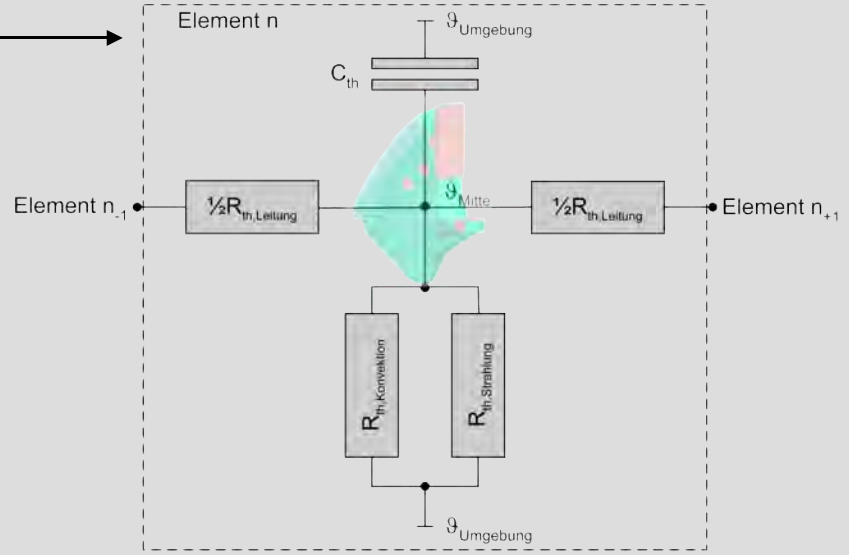
Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Implementierung in Dymola



- Aufbau des Netzwerks in Dymola mit zusammengefassten Elementen

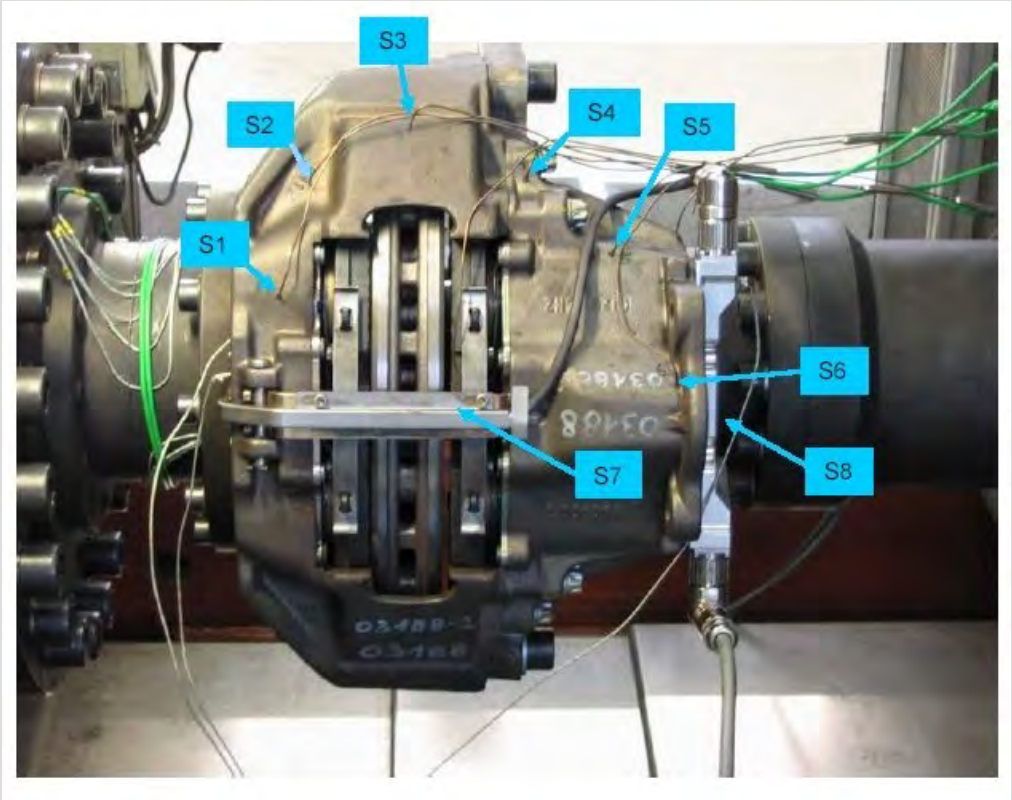
- Aufbau der einzelnen Elemente des Netzwerkes ähnlich



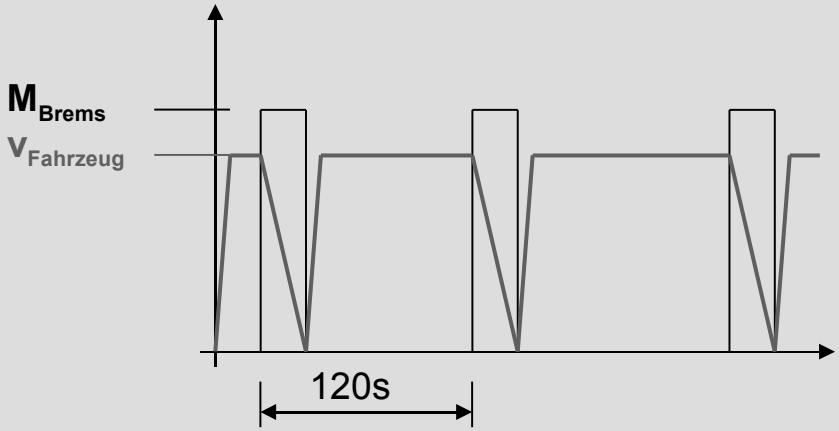
- Ausgabe einer Mittentemperatur, Wärmekapazität an Mittelpunkt angekoppelt

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Modellvalidierung am Schwungmassenprüfstand



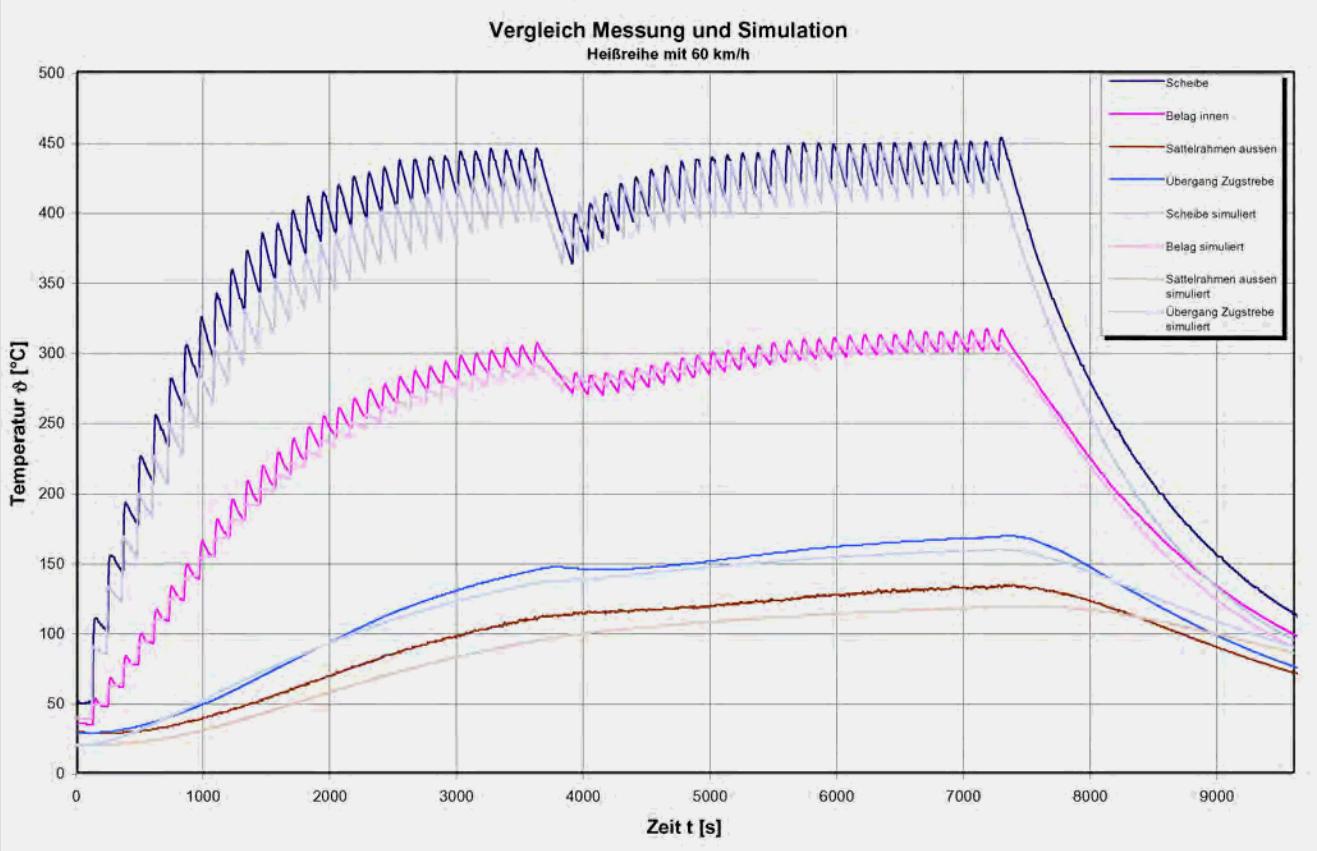
- Prüfprogramm „Heißreihentest“; 30 Stoppbremsungen pro Stunde



- Annahme: Gesamte kinetische Energie wird in Wärmeenergie überführt und in Bremsscheibe / Belag eingebracht
- Temperaturmessstellen platziert nach Modellvorgabe (ca. Elementmitte)

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

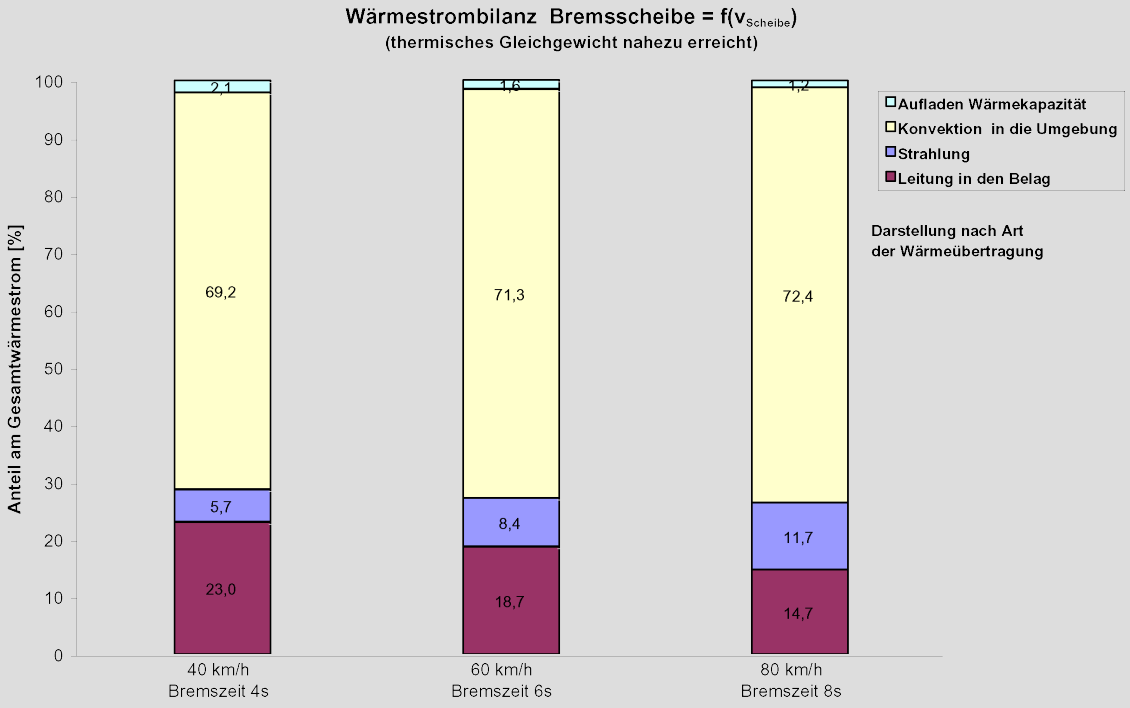
Vergleich zwischen Messung und Simulation



- gute Übereinstimmung der Temperaturverläufe
- Parameterveränderung durch Modellvalidierung erklärbar
- Abnahme der Genauigkeit bei scheibenfernen Bauteilen
- Ergebnisse für Abschätzung von Temperaturniveau hinreichend genau

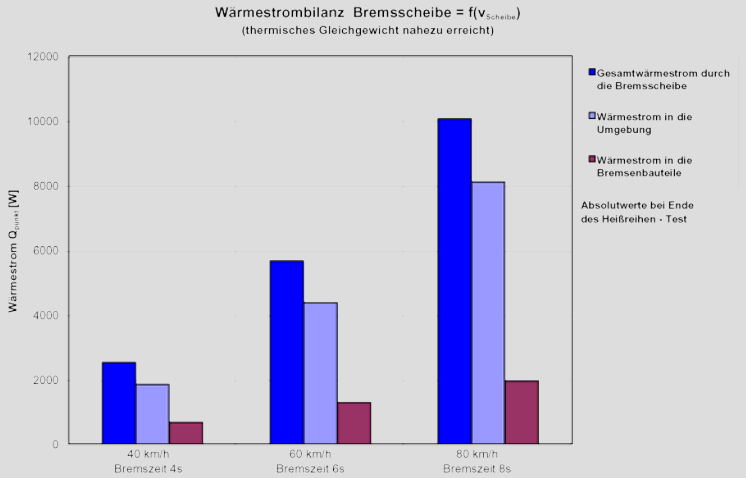
Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Aufteilung des Wärmestroms an der Bremsscheibe



relative Werte

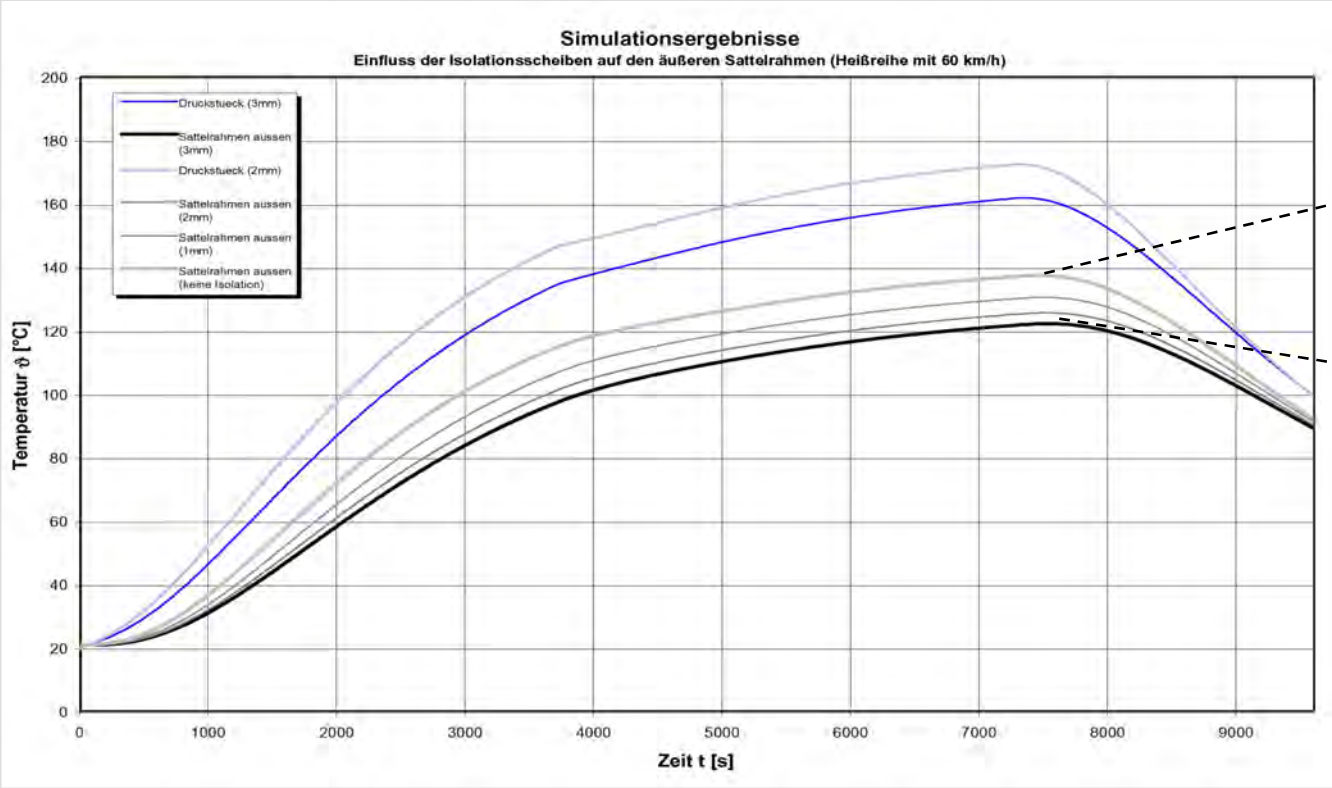
- Wärmestromaufteilung zeigt Bedeutung der Übertragungswege
- Strahlungsanteil steigt stark an
- Anteil der in den Belag fließenden Energie nimmt dagegen ab
- Konvektion bleibt der bestimmende Wärmeübertragungsmechanismus



absolute Werte

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Simulation von Isolationsmaßnahmen (Glimmerscheiben am Druckstück – Isolation der Mechanik vom heißen Bremsbelag)



- Isolationsstärke am Druckstück
  - keine Isolation
  - 1 mm
  - 2 mm
  - 3 mm
- Erkenntnis: Existenz eines Optimums
- Beachtung von Kosten / Bauraum und Isolationseffekt damit möglich

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

Anwendung des Modells im Fahrzeugbetrieb

- Integration eines möglicherweise vereinfachten Modells in das Bremsensteuergerät
  - Näherungsweise Berechnung von Bauteiltemperaturen während der Fahrt
  - Möglichkeit der Warnung des Fahrers vor Ausfall einer Komponente (vorgreifende Berechnung und Möglichkeit zur Abwendung eines Schadens im Vorfeld)
  - Dokumentationsmöglichkeit von Temperaturüberschreitungen (Garantiefall?)
- Vorteil: Informationsgewinn durch Softwarefunktion, alle benötigten Größen (Bremsdruck, Raddrehzahl) stehen zur Verfügung; keine zusätzliche Sensorik nötig
- Vision: automatischer Eingriff in Bremsensteuerung bei thermisch kritischen Situationen

Simulation des Wärmehaushalts einer Scheibenbremse für NfZ

weiteres Vorgehen - Ausblick

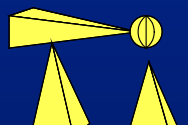
- Übertragung des Modells auf reale Fahrbedingungen
  - Parameteranpassung, physikalischer Bezug soll erhalten bleiben
  - Parameter müssen „erklärbar“ und „herleitbar“ sein
- Detaillierung des Modells mit Hilfe von verfeinerten Elementen
- Erweiterung des Modells und Ausbau der Simulation zur Systemsimulation
  - System „WheelEnd“ mit Bremse, Nabe, Radlager und Felge
  - Systembetrachtung bietet Vorteile

Vielen Dank für Ihre Aufmerksamkeit

KNORR-BREMSE SfN GmbH
Dipl.-Ing. Stephan Pitzing
Bereich Vorentwicklung Scheibenbremse T/DBS
Advanced Engineering Disc Brake T/DBS
Moosacher Str. 80
D-80809 München
Phone: +49 89 3547-1320
mailto: stephan.pitzing@knorr-bremse.com
http://www.knorr-bremse.com

Avatare in technischen Simulationen

Prof.-Dr.Ing. Dieter Wloka
Technische Informatik
Universität Kassel



Begriffe

vatar, virtual human, digital people, digital human, humanoide Modelle, digital character, digital actor, digital manikin, embodied human agents, synthespian, ...

efinition (N. Badler):

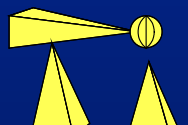
irtual humans are computer models of people

that can be used as substitutes for ``the real thing'' in ergonomic evaluations of computer-based designs for vehicles, work areas, machine tools, assembl lines, etc., prior to the actual construction of those spaces;

for embedding real-time representations of ourselves or other live participants into virtual environments.

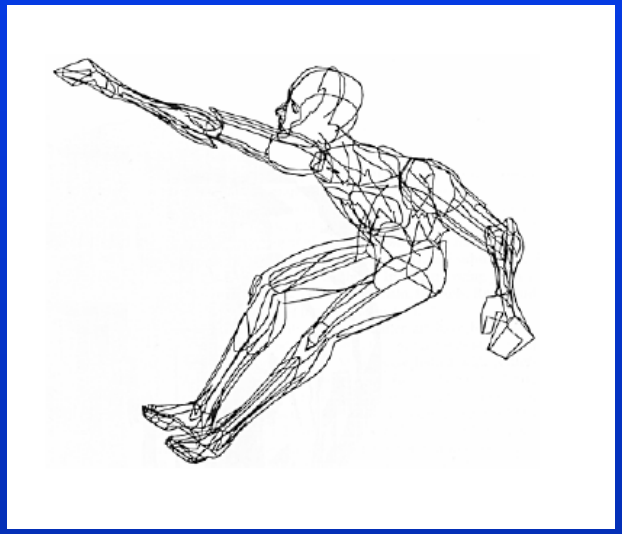
efinition digital manikin:

eometric model of the human body, has joints, allows typical human positions, can be created differently(gender, populations, antropometric data)



Historie

Fetter, 1960, Boeing
Pen Plotter



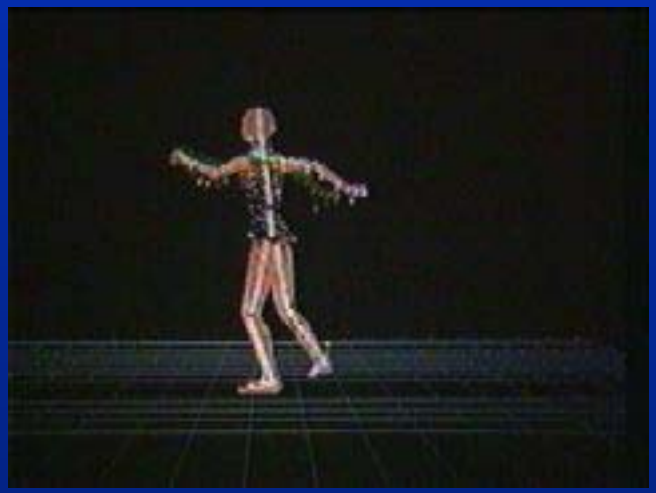
1984 „Max Headroom“
TV Serie

1987 ... Kleizer-Walzack
„Dozo“ : Don‘t touch
me

Aufbau eines Avatars

Kopfmodell

kinematisches
Körpermodell



Tools zum Aufbau von Avataren

3D-Studio Max

Maya

....

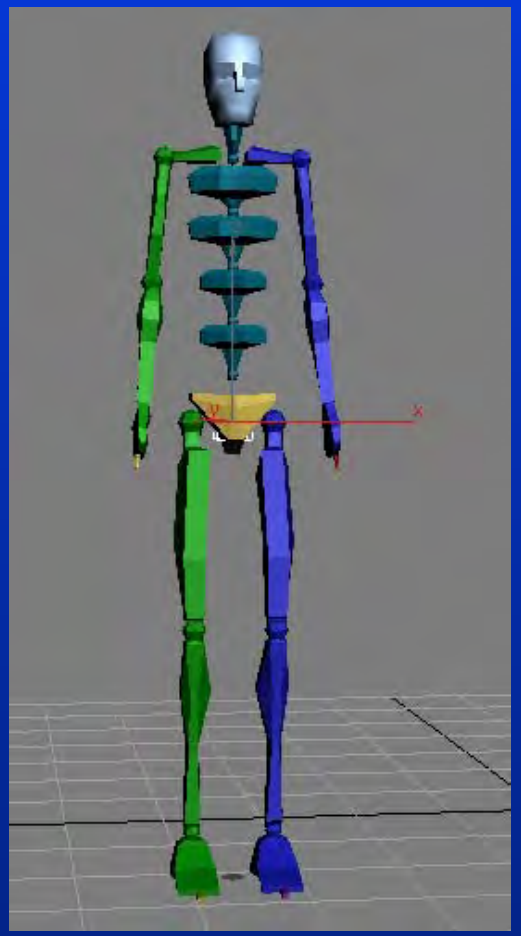
+ Plug-Ins

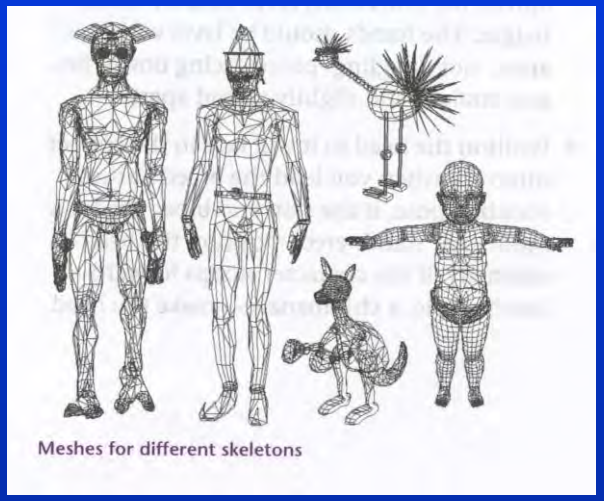
Skelett

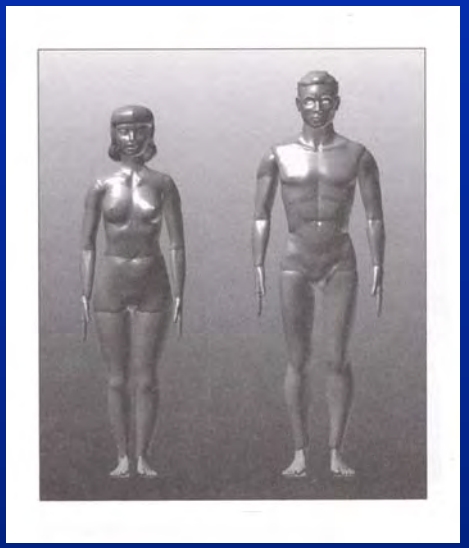
Hülle als mesh

Haut

Augen, Zähne, ...



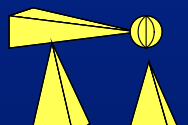




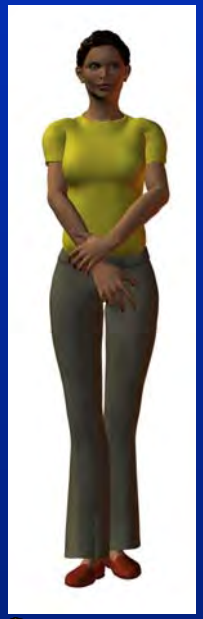
Körperhüllen

komplette Körper
Köpfe

Kleidung
+ Brillen, ...



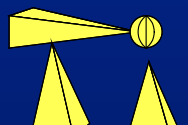
Poser



Dieter Wloka, Technische Informatik, Universität Kassel

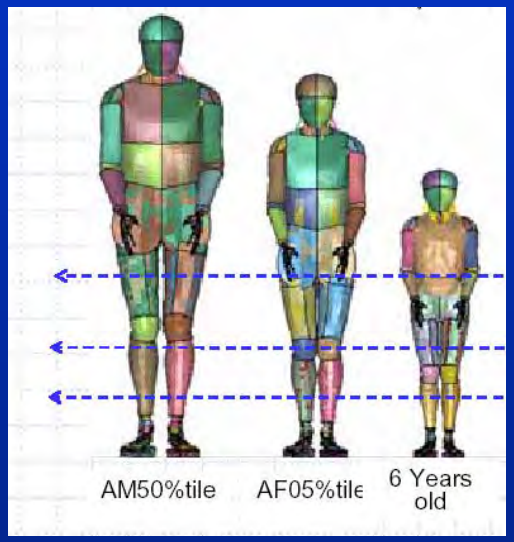
3D Scanner

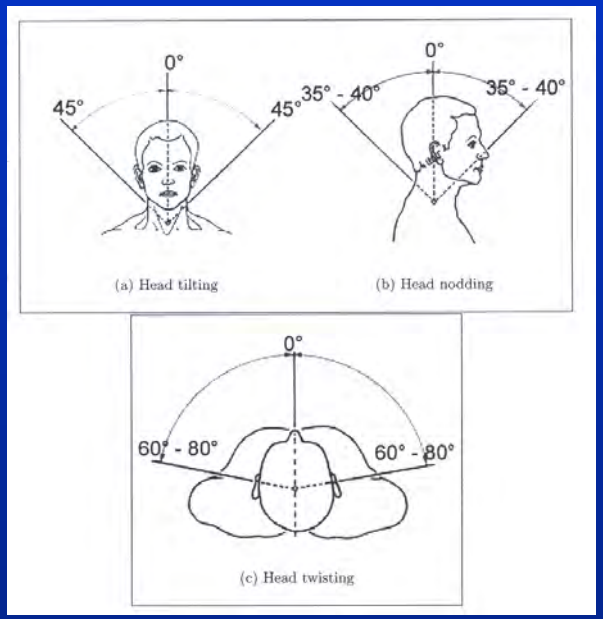
- Z.B. Cyberware

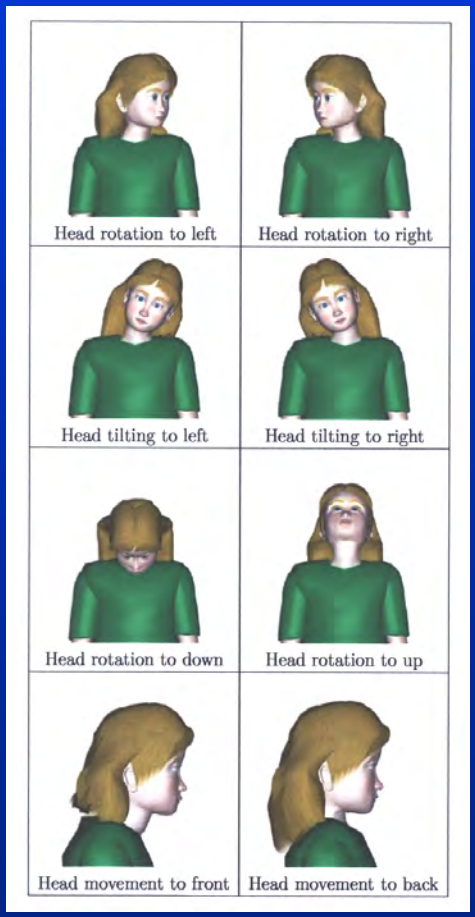


Anthropometrische Daten

Abmessungen, Bereiche
Basis für realistische Modelle

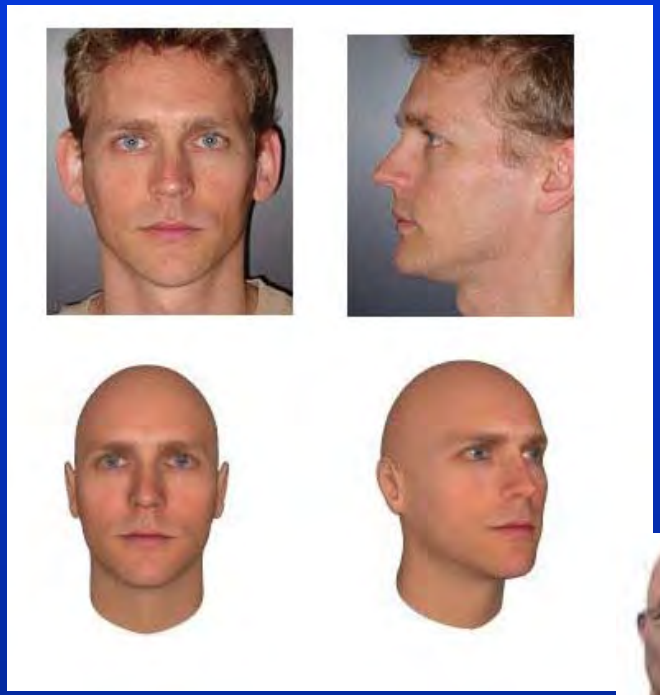






Aufbau 3D Modell des Kopfes

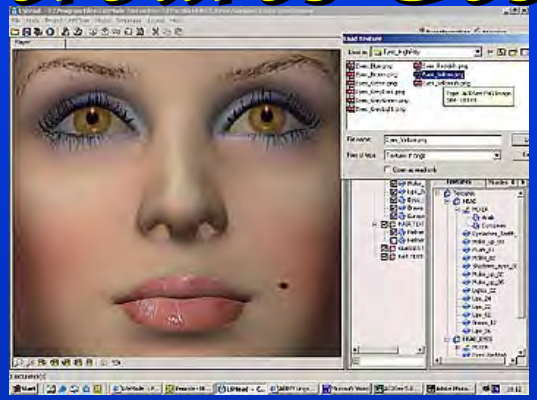
- Vorlage Photo
- alternativ Random-Prozess, Parameter werden per Zufall erzeugt - random face
- Geometrie (Kopfform), Rasse, Geschlecht, Alter, ...



Facegen

Facial Studio

animierbares Gesicht



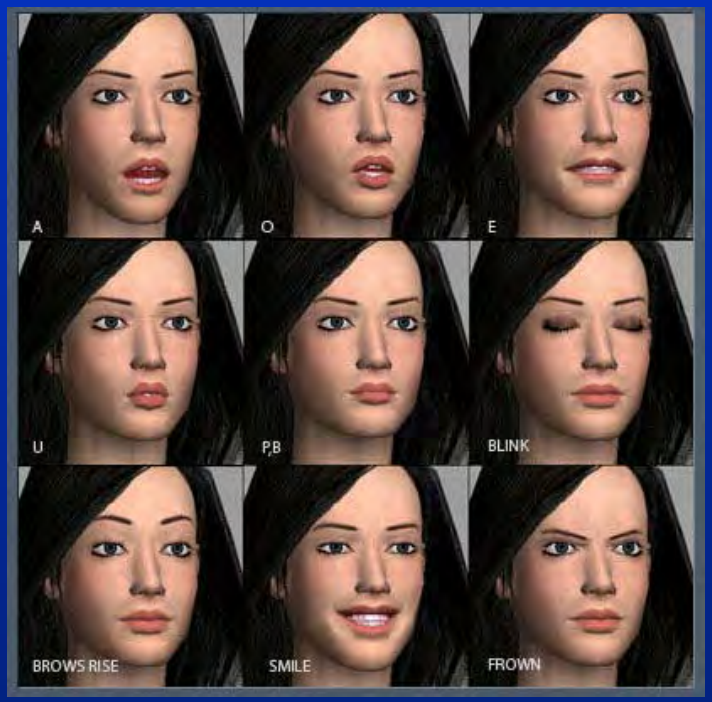
Hautdesign

Zähne, Zunge

Augenbewegungen

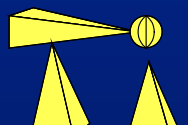
Gesichtsmimik

Sprache, TTS



Dieter Wloka, Technische Informatik, Universität Kassel

Haare ...



Bewegungen

Algorithmik -
Kinematik /
Inverse
Kinematik

SW-Bibliotheken,
z.B:

* Sega - Animanium
* Alias Human IK
Middleware

Synthese

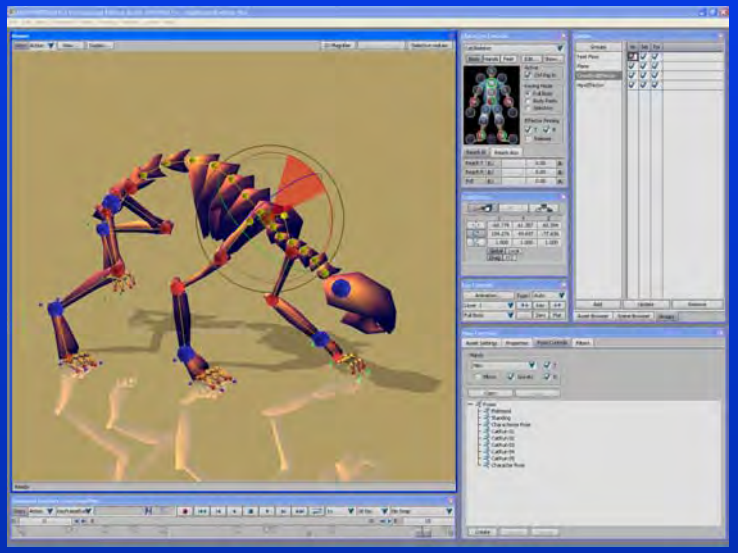
MOCAP

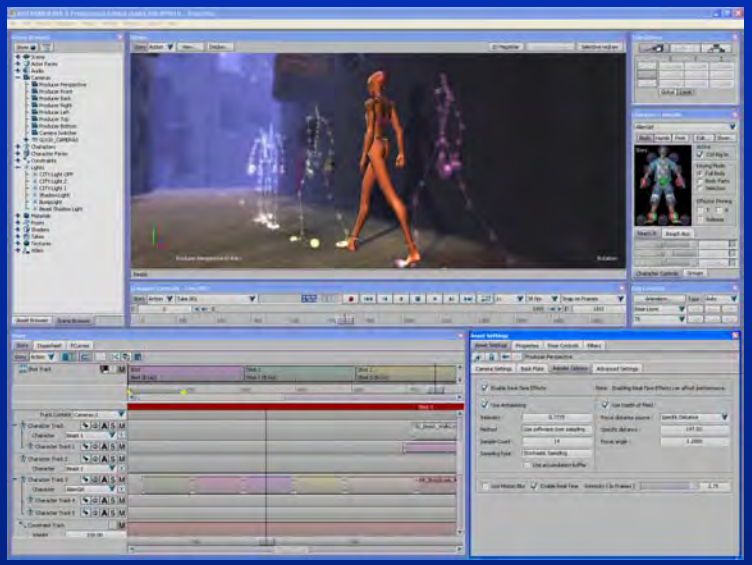
Bewegungsdaten-
bibliotheken via
MOCAP

z.B. VICON

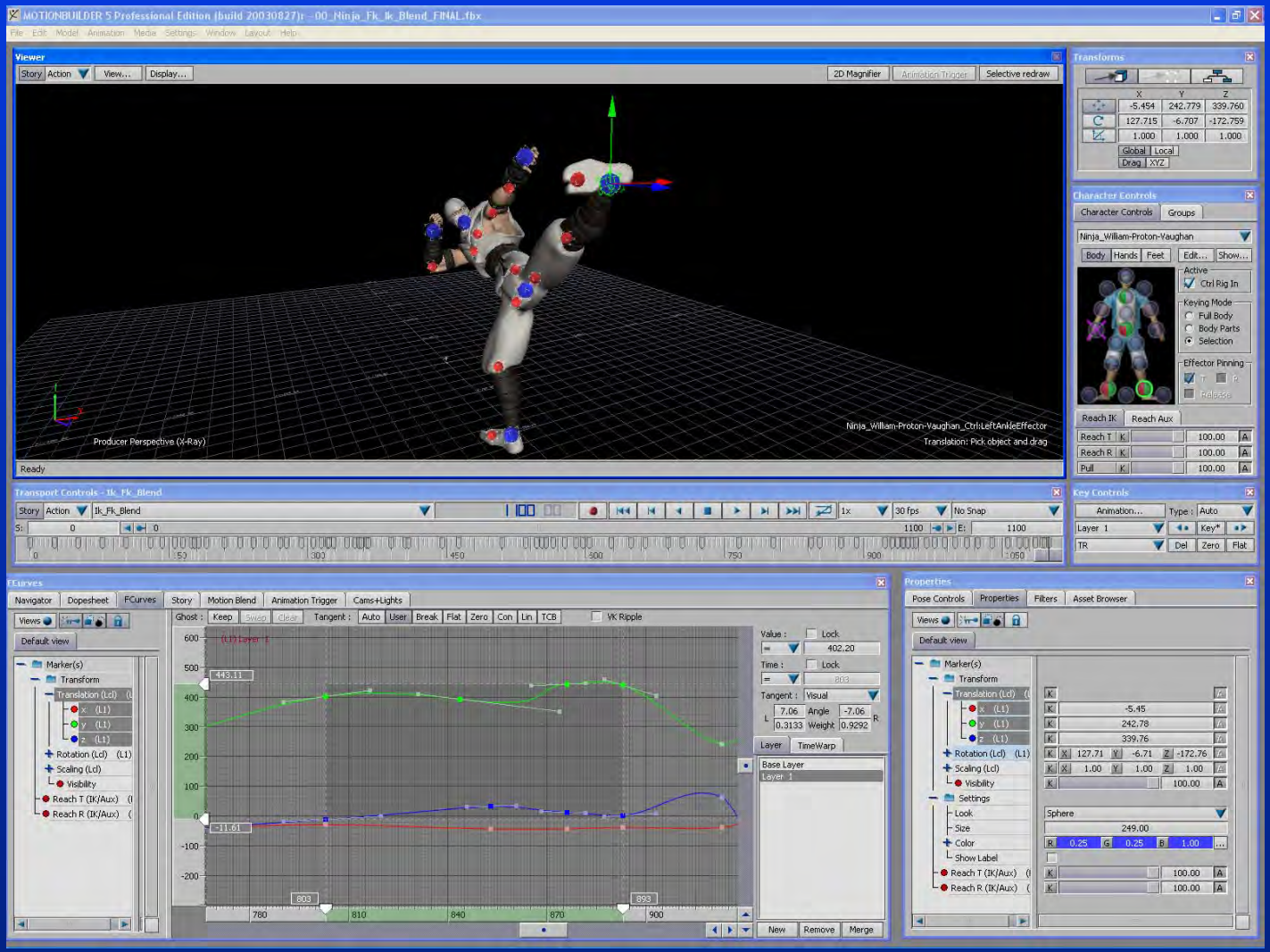
Motionbuilder

- Komposition von Bewegungen
- Keyframing Posen
- Editieren von MOCAP Daten
- Szenenkomposition („directors seat“)





Motionbuilder ..



Dieter Wloka, Technische Informatik, Universität Kassel

Kinematisch / Dynamisches Modell mit simulierten Muskeln

Modelle Fa. Anybody

FEM / Medizinisches Modell

detaillierte Modelle innerer Organe
Operationsplanung, Ausbildung

Crash-Test

Physikalisches Modell
Absorption Hitze (Auto), Strahlung,
...

Massive

Gruppen von Menschen

- Kampfsimulation
- Paniksimulation

fertige Avatare für bestimmte Anwendungsszenarien

z.B. Stadium Guy

Steuerung vs. Autonomie (KI)

autonome Agenten ....

Interaktion mit Avatar

Virtueller Schauspieler (Drehbuch), Hollywood

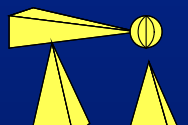
Computerspiele

Ausbildung, Training

Katastrophensimulation

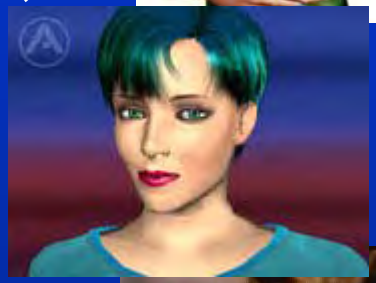
Virtueller Soldat

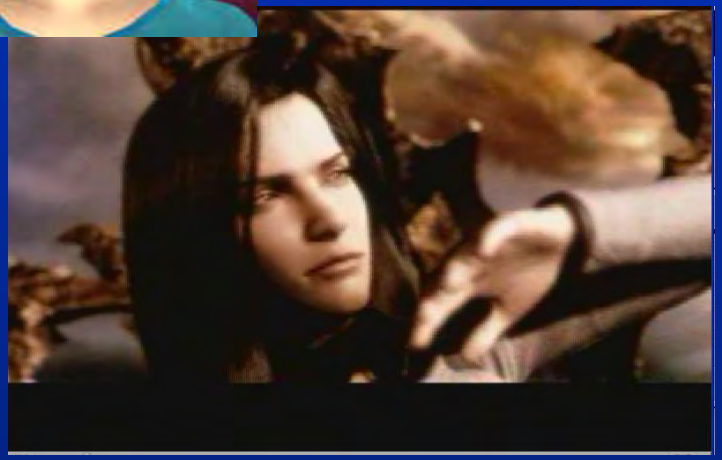
Einsatz von KI-Technologien: NN, XPS, Fuzzy, GA, AL, ...



Anwendungsbereiche

- Pop-Idol (Kyoko Date, E-Cyas)
- Werbung (Robert T. Online)
- Nachrichtensprecher (Ananova)
- Spiele (Lara Croft, ...)
- Film (Final Fantasy, Shrek, ...)





Stellvertreter in Internetapplikationen

Chat
virtual communities
z.B. blaxxun Avatar Studio

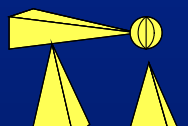
Präsentationen

Kombination mit Powerpoint
(sprechendes Gesicht)

Internet: 3D-Modell als Präsentator,
GUI

Weitere Bereiche

- Recht
- Visualisierung von Unfällen (USA)
- Sport - Bewegungsanalyse



Simulation mit kinematischem / dynamischem Modell

Typische Modellstruktur:

- Körperstruktur (2 Hände, 2 Beine) via Knochenhierarchie
- anthropometrische Daten
- Körper als Oberflächennetz
- Kinematik: FK,IK
- Dynamikberechnungen - Belastungen

Human-centered design

- Bemaßung (Fit), Komfort
- Einsteigen, Aussteigen (PKW)
- Sicht
- Kraftaufwand
- Erreichbarkeit, Greifen, Bedienen
- Bedienen von Fußpedalen

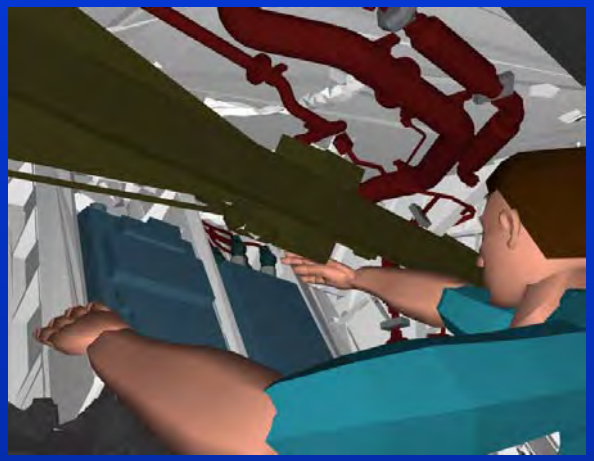
N. Badler - Jack / Transom Jack

- „Jack“ Avatarmodell entwickelt von N. Badler, Center for Human Modeling and Simulation (University of Pennsylvania) „Home of Jack“

Fabrikationsplanung

- Layout
- Simulation Arbeitsablauf
- manuelle Montage
- Materialhandhabung
- Umgang mit Werkzeugen

Transom Jack



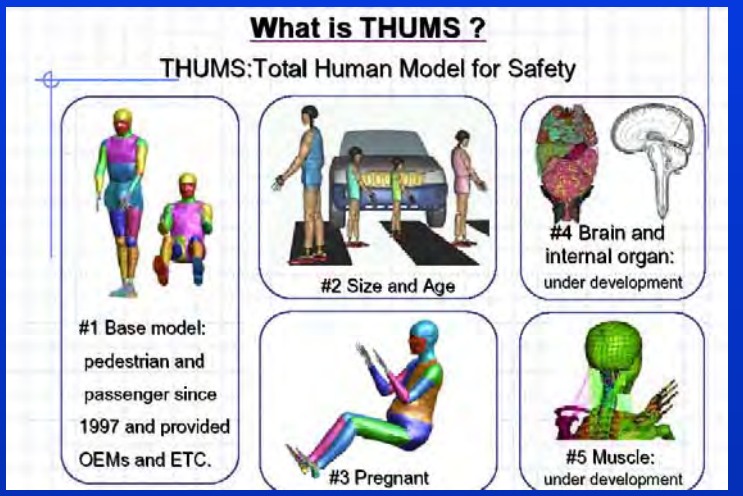
UGS eM-Human

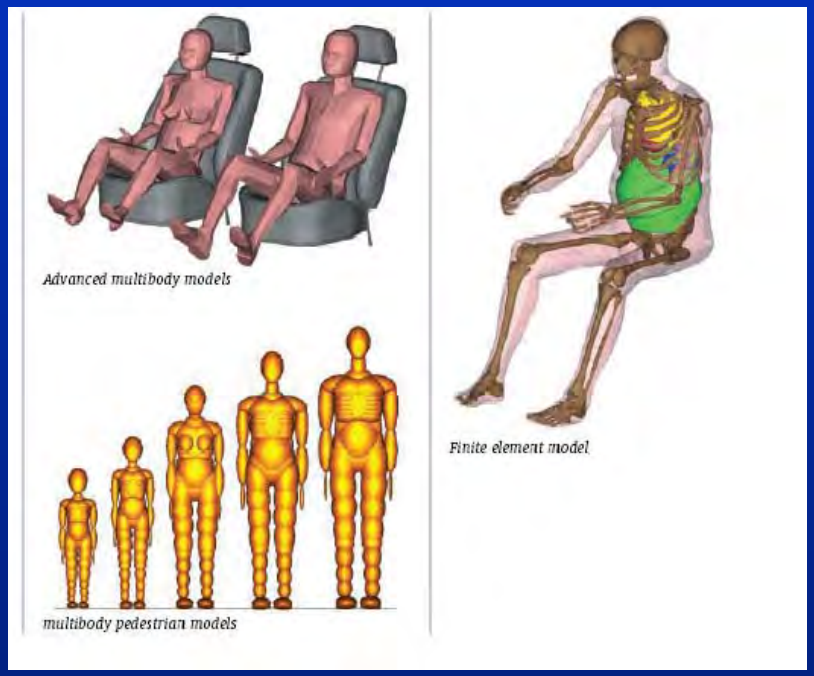
Planung von Wartungsaufgaben

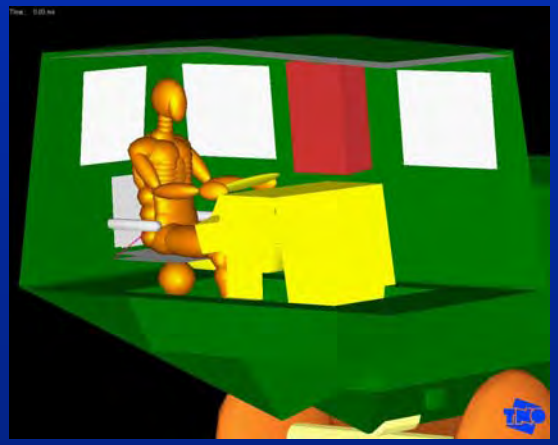
- Erreichbarkeit, Arbeitsräume
- Entfernen von Teilen und Ersatz
- Sinnhaftigkeit von Handbuchanweisungen
- Sichtbereiche
- Kraftaufwand
- Verletzungsrisiko

Crash-Tests

- Toyota THUMS
- TNO Madymo



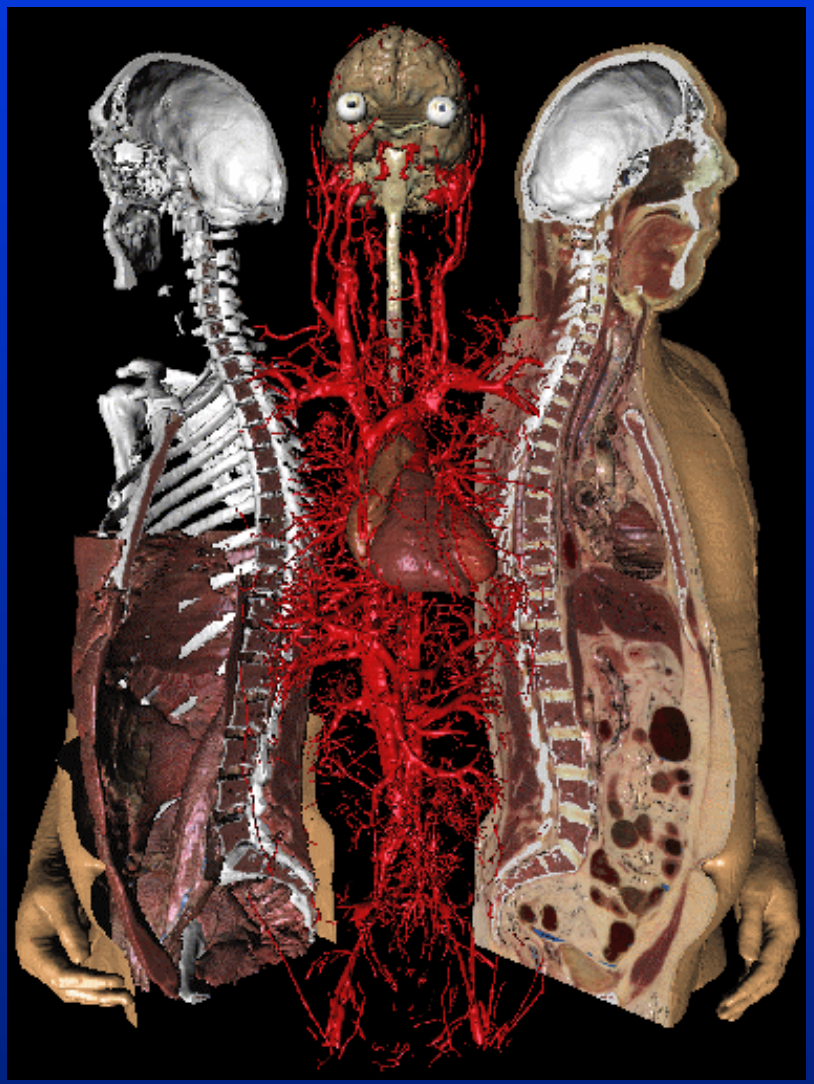




MEET - Man Project

Model zur Simulation von elektromagnetischen, elastomechanischen und thermischen Belastungen

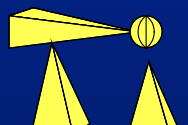
„MEET Man is a realistic, anatomical model of the human body extended by descriptions of physical attributes, eg electric conductivity and permitivity, magnetic permeability, thermal and elastomechanic properties. „



Simulations-basiertes Training

- Erstellen von Trainingsvideos
- Multimedia-Präsentationen

- VR-basierte Trainingsprogramme



U Iowa - Virtual Soldier Research

The Virtual Soldier Research (VSR) Team conducts research in the following areas:

- Human Modeling and Simulation
- Dynamic Simulation, Dynamics
- Physiology
- Muscle Modeling
- Hand modeling
- Posture and
- Motion Prediction
- Motion Capture
- Real-Time Visualization
- Biomechanics, Control
- Multi-Objective Optimization
- Clothing
- Human Performance Measures
- Electromyography
- Artificial Intelligence

Dieter Wloka, Technische Informatik, Universität Kassel

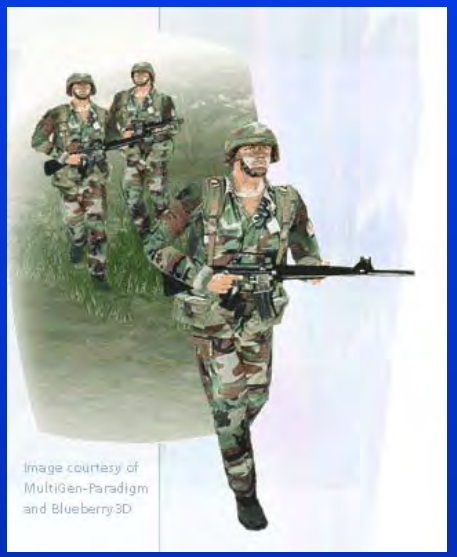
U Iowa Soldier SANTOS

Boston Dynamics

DI-Guy Scenario is a tool for creating interactive 3D training scenarios with lifelike human characters.

ts graphical user interface lets you work directly in the 3D scenario to create, place, and control realistic people and vehicles.

Once you have created scenarios, you can run them immediately in DI-Guy Scenario, or export them to run on desktop, laptop, or high-end computers.

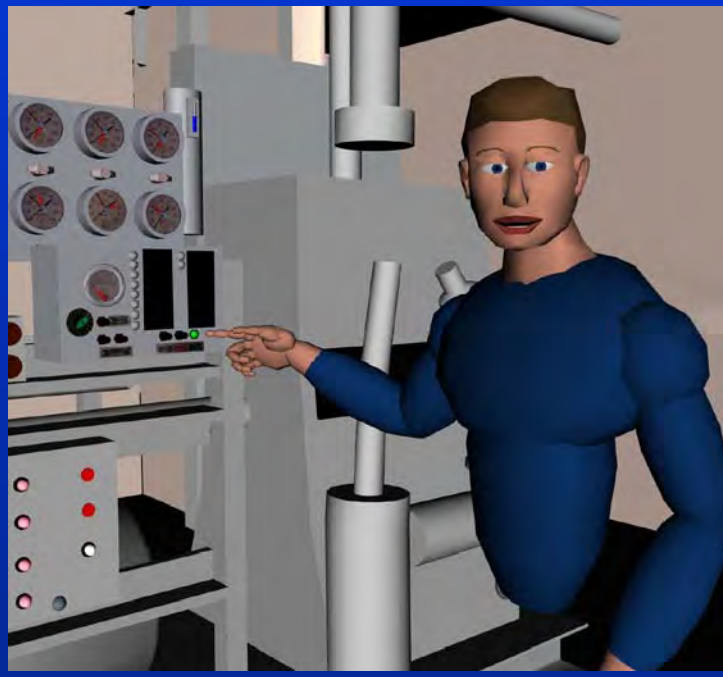


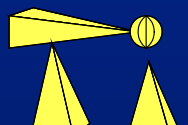
Pedagogical Agent Steve

- USC Information Sciences Institute

"USC / ISI is developing a pedagogical agent called Steve (Soar Training Expert for Virtual Environments) that supports the learning process. „

- can demonstrate skills to students,
- answer student questions,
- watch the students as they perform the tasks,
- and give advice if the students run into difficulties.





VR Training - Quest3D

VR Training als Ersatz für reales Training in Bereichen mit :

- großem Gefährdungspotential (Flugzeug)
- hohen Kosten (Ölplattform)
- schwieriger Erreichbarkeit (Weltraum)

Unfälle / Katastrophen

Fa. Animats

- realistische Simulation fallender Körper
- Computerspiele
- Unfallforschung

Fa. Naturalmotion

Software zur Synthese von Bewegungen

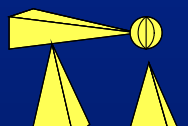
Computerspiele, Stunts in Filmen, Stürze, ...

Endorphin

endorphin *n.*

1. Next generation real-time animation tool.

GDC 2004
March 24-26 2004, San Jose
Booth #650



Swine Housing

Iowa State University is developing cutting-edge computational tools to design and evaluate swine housing systems in a 3-D virtual reality environment.“

U Kassel - RMS -Radfahren mit Multimedia-Software

www.rms-fahrradwelt.de

Dieter Wloka, Technische Informatik, Universität Kassel

Animation der Avatare

Laufen, Bewegen Arme, Fahrradfahren

Lsg: Keyframeanimation

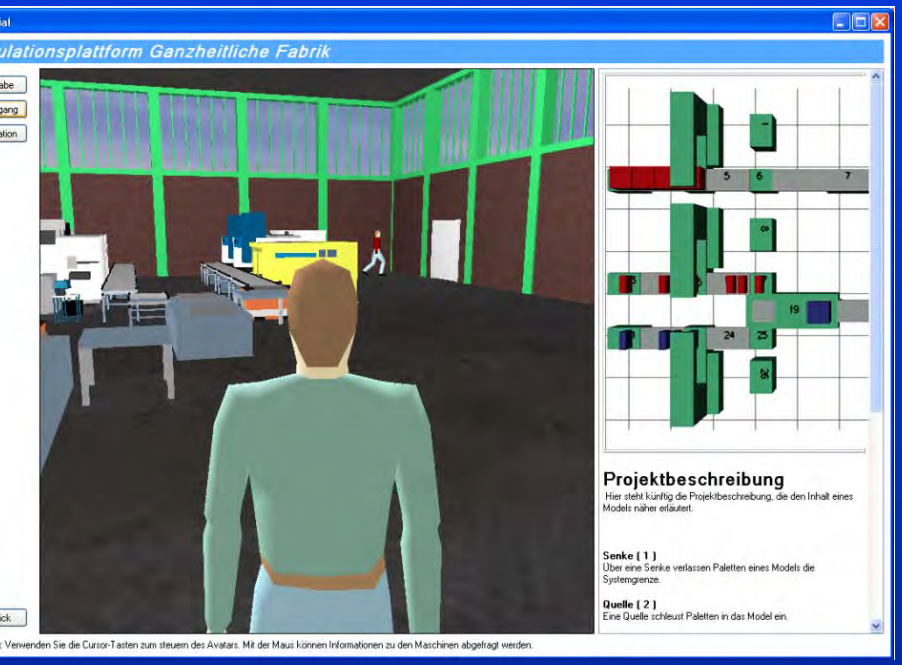
alternativ: MOCAP Daten

Videoanimationen im Stadtmodell

Digitale Fabrik

Dieter Wloka, Technische Informatik, Universität Kassel

Tutorial (Web-basiert)



Cave basierte Darstellung

Modellierung von Brennstoffzellen für die mobile Anwendung

Nicola Bundschuh

ASIM/GI-Fachgruppentreffen

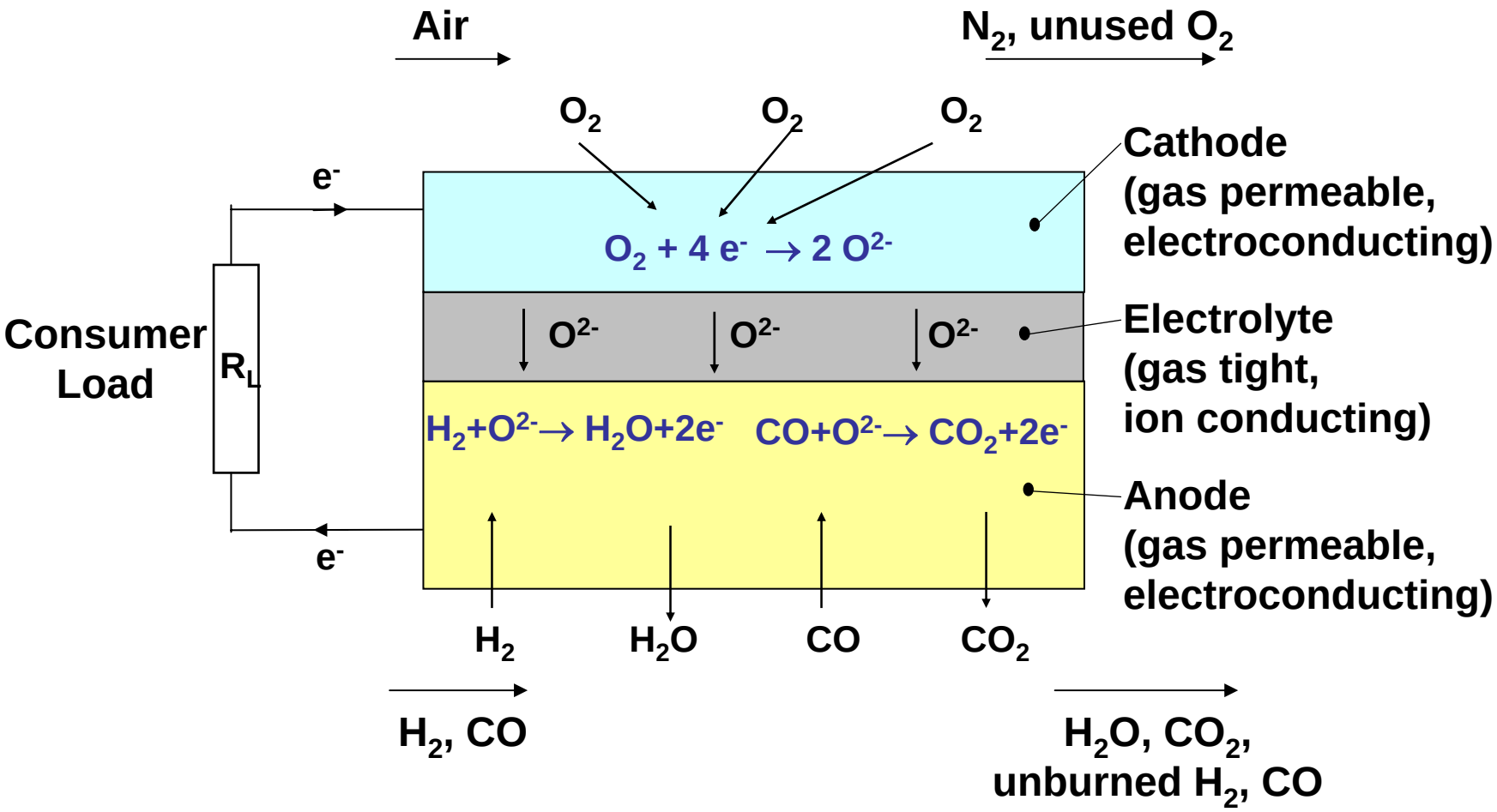
München, 20.-21.02.2006

1. Einleitung

- Brennstoffzellen wandeln chemische Energie in elektrische Energie um via kalter Verbrennung

- Brennstoffzellen benötigen eine Systemumgebung zur
  - Brennstoffaufbereitung
  - Luftversorgung
  - Wärmemanagement

1. Einleitung



2. Festlegung der Anforderungen

- Betrachtung der Leistungsanforderungen
  - Nennleistung
  - Dynamisches Leistungsspektrum
- Untersuchung möglicher Synergien
  - Stoffströme
  - Wärmeströme
  - Druckniveau
- Sicherheitsanforderungen
  - Redundanz

2. Festlegung der Anforderungen

- Betrachtung der Leistungsanforderungen:
  - Systemauslegung bzgl. Regelstrategien
  - „Nur“ Brennstoffzelle oder Hybridlösung?
  - Dynamik aus Brennstoffzelle oder Batterie?

- Untersuchung möglicher Synergien
  - Systemauslegung bzgl. Wärme- und Stoffströme
  - Kraftstoff?
  - Luft auf welchem Druckniveau?
  - Wärmebedarf im mobilen System?
  - Zusätzlicher Schub erforderlich?

2. Festlegung der Anforderungen

- Sicherheitsanforderungen :
  - Systemauslegung bzgl. Sicherheit
  - Zuverlässigkeit?
  - Redundanz?
  - Gefahrstoffverordnung
  - Explosionsschutz

3. Systembetrachtung

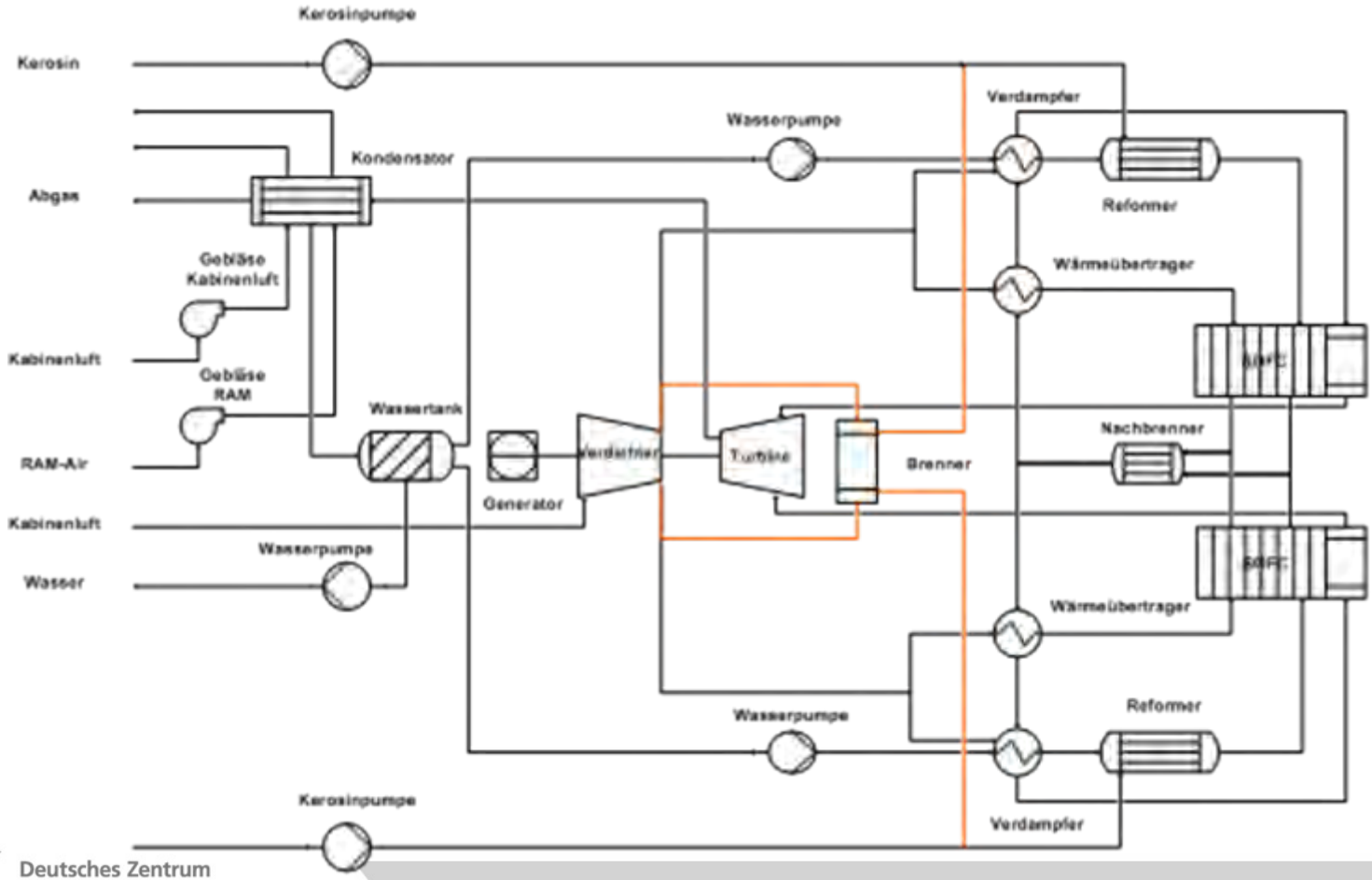
- Systemdefinition
  - Brennstoffzellentyp
  - Leistungsbereich
  - El. Spannung
  - Systemkomponenten (Reformer, Kompressor, Turbine, Speicherbatterie…)
- Systemgrenzen
  - Kraftstoffversorgung
  - Luftversorgung
  - Abgas

3. Systembetrachtung

- Systemintegration
  - Wassererzeugung
  - Kompressor + Turbine zur zusätzlichen Leistungssteigerung
  - Wärmeintegration
  - Kopplung Batterie oder zusätzlichen Antriebsaggregaten

- Systemauslegung
  - Auslegung des Gesamtsystems
  - Auslegung der Systemkomponenten
  - Verschaltung der Komponenten

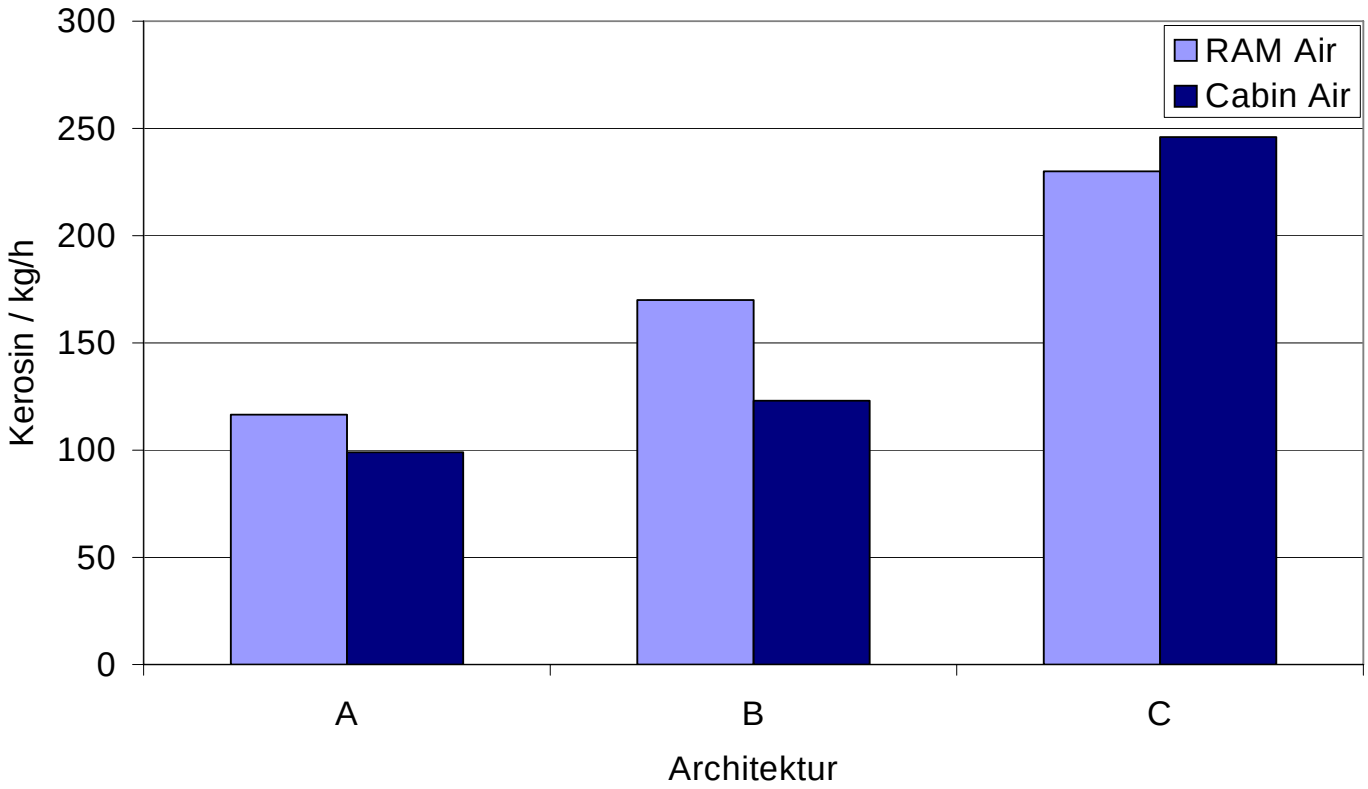
3. Systembetrachtung
Systemauslegung einer Flugzeug-APU mit SOFC und Kerosin



3. Systembetrachtung

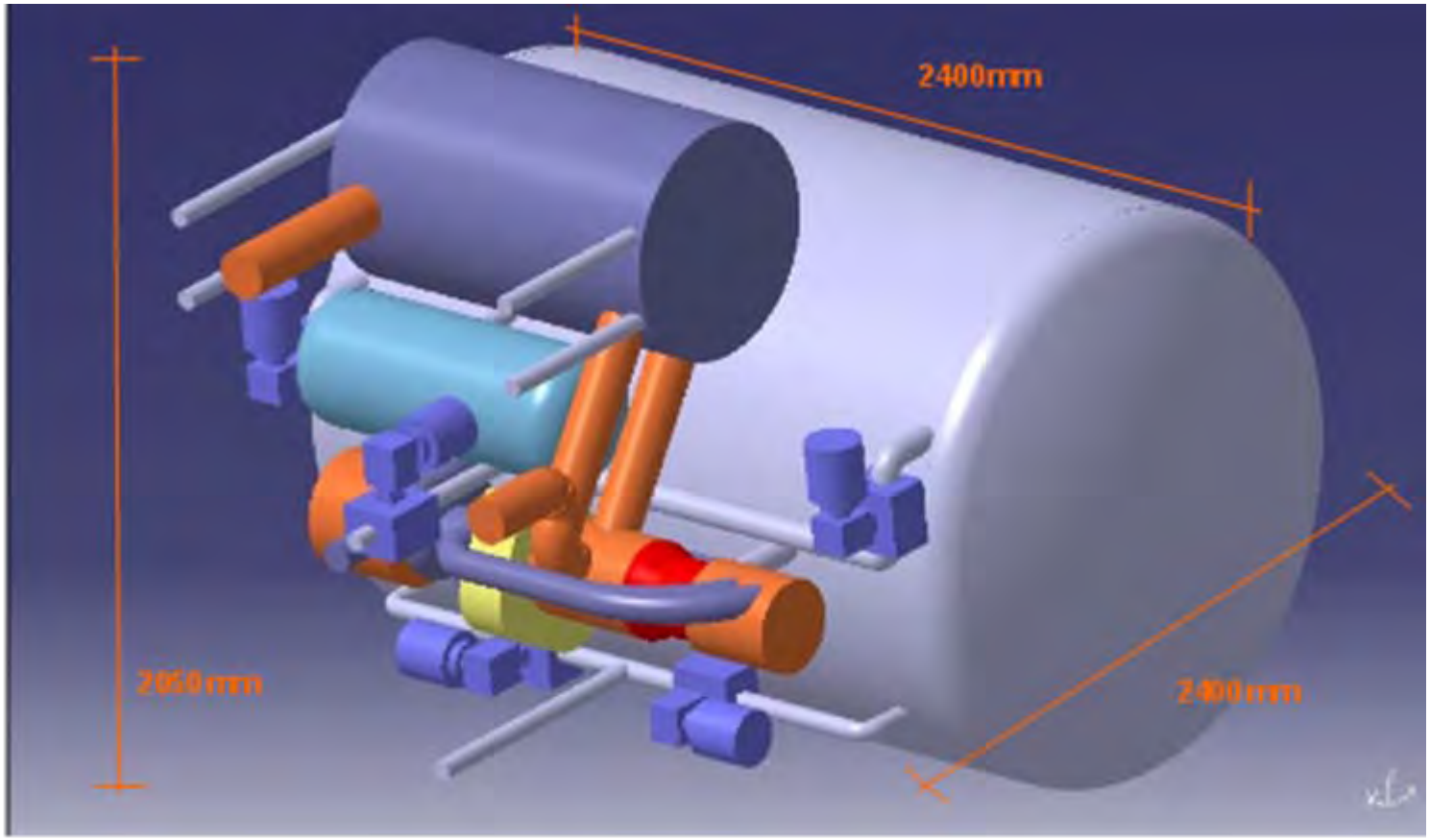
- Systemmodellierung
  - Stationäre Berechnung des Systemperformance
  - Dynamische Berechnung von Leistungszyklen unter Berücksichtigung der Dynamik aller Komponenten
- Systemdesign
  - „Reale“ Verschaltung und Anordnung der Systemkomponenten
  - Adaption an den Bauraum
  - Berücksichtigung der Systemmasse

3. Systembetrachtung
Kerosinverbrauch, versch. Systeme und Integrationen, 39000 ft, Mach 0,8



Bundschuh, Nicola; Dollmayer, Jürgen; Carl, U. B.; “Fuel Weight Penalty due to Fuel Cells and Generators as Energy Source of the All-Electric Airplane”; Deutscher Luft- und Raumfahrtkongress, 2005

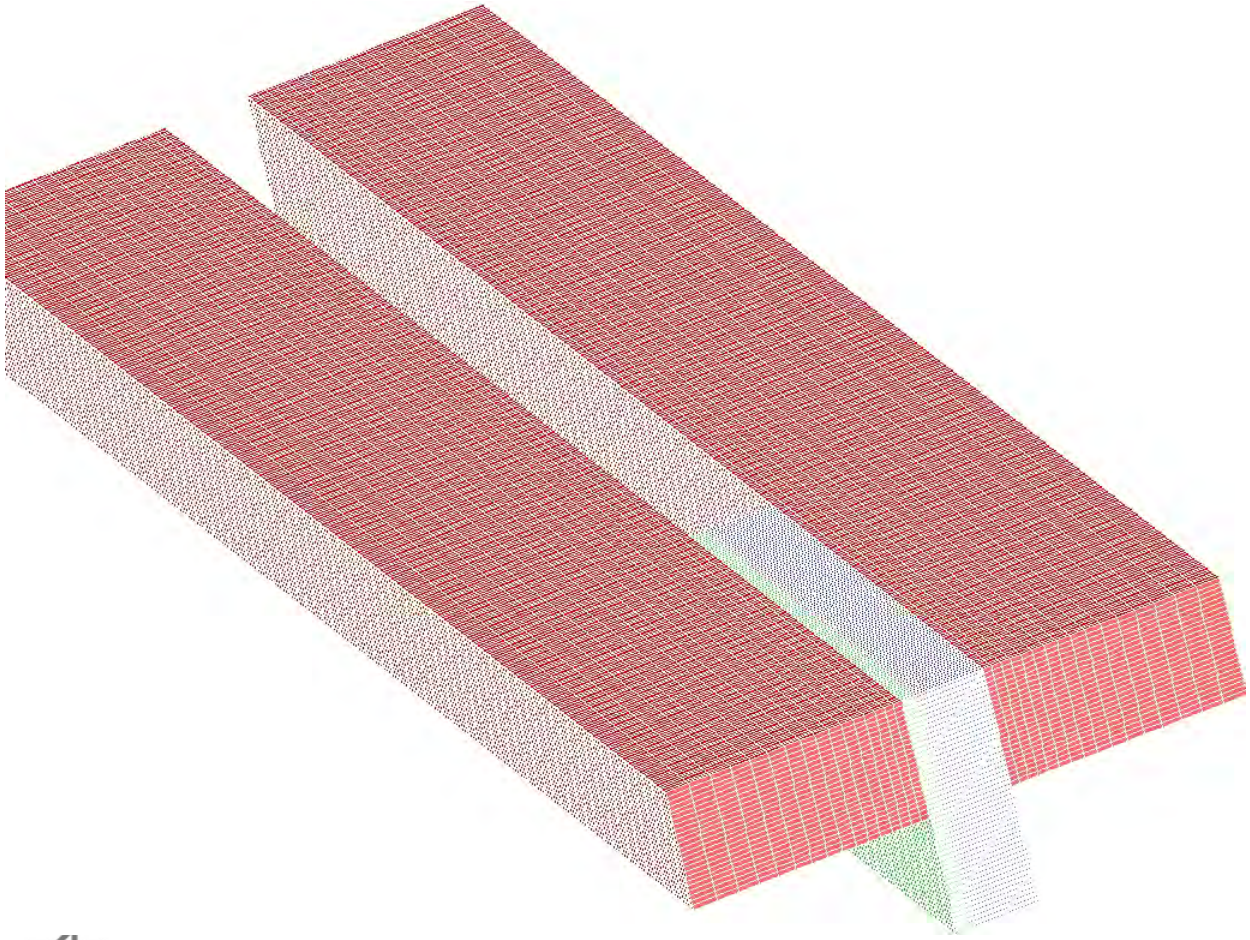
3. Systembetrachtung
Systemdesign einer Flugzeug-APU mit SOFC und Kerosin, 1 MW



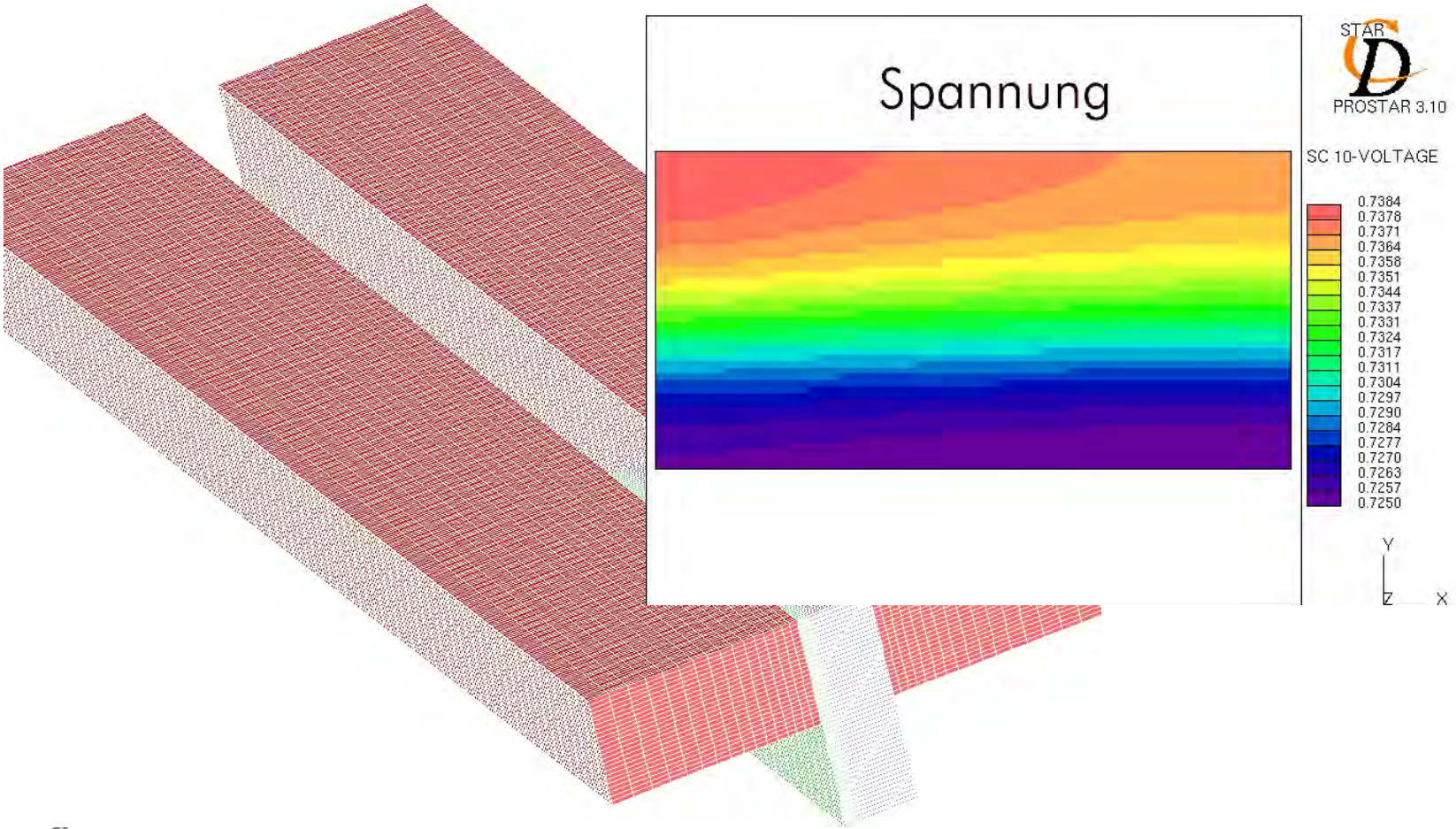
4. Detailuntersuchungen

- Detaillierte Betrachtung von Systemkomponenten mit CFD
- Untersuchung des dynamischen Verhaltens neuer Komponenten
- Errechnen dynamischer Kenngrößen für die Systemsimulation

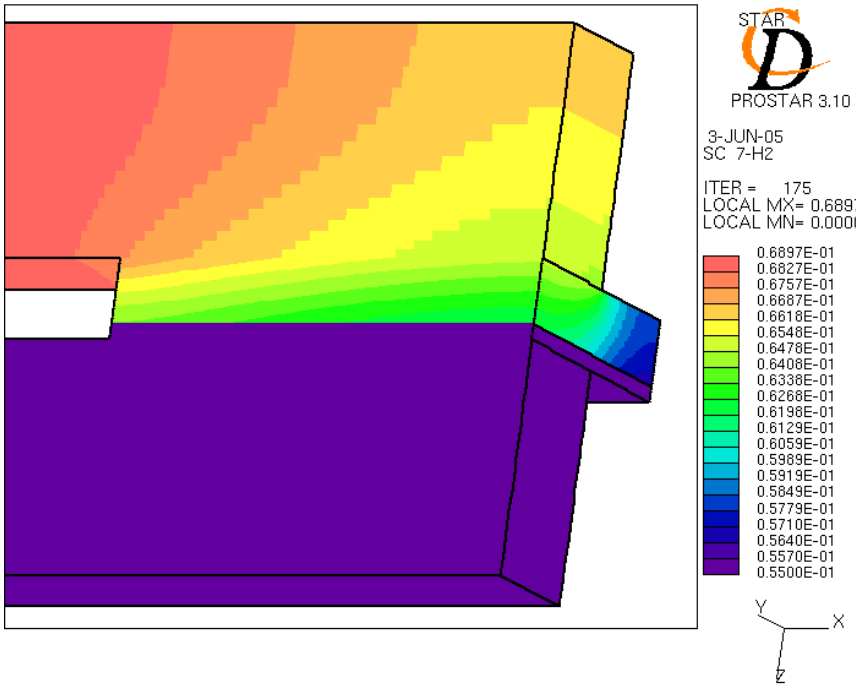
4. Detailuntersuchungen eines Brennstoffzellensegments

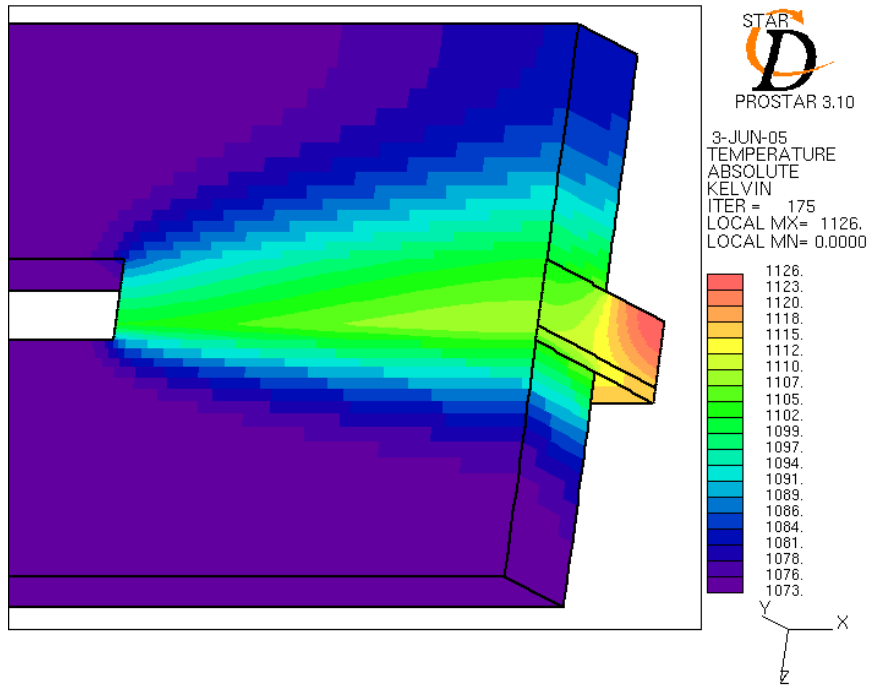


4. Detailuntersuchungen eines Brennstoffzellensegments



4. Detailuntersuchungen eines Brennstoffzellensegments





5. Zusammenfassung

- Zur Systemsimulation sollten alle Anforderungen bekannt sein
- Einfluss des Designs auf die Performance
- Detaillierte Modellierung einzelner Komponenten mit CFD zur Abbildung der Dynamik

6. Kontakt

TTI GmbH
an der Universität Stuttgart

Rechtlicher Rahmen
Betreuung in Vorgründungsphase

TGU Sim-BEL
Nicola Bundschuh
Florian Leucht
Max Eschenbach

Forschungskontakte
Mentoring

Deutsches Zentrum für Luft- und Raumfahrt
Institut für Technische Thermodynamik

Forschungskontakte
Infrastruktur

Universität Stuttgart
Institut für Thermodynamik und Wärmetechnik

6. Kontakt

Gefördert durch:

eXIST
Existenzgründungen aus Hochschulen

Technologie Transfer Initiative

Bundesministerium für Bildung und Forschung

EUROPÄISCHE UNION
Europäischer Sozialfonds

Projektträger Jülich
Forschungszentrum Jülich GmbH

6. Kontakt

Nicola Bundschuh

TTI GmbH - TGU Sim-BEL
An der Universität Stuttgart
Institut für Thermodynamik und Wärmetechnik
Pfaffenwaldring 10
70550 Stuttgart
mail: bundschuh@visimbel.de

Methodik der impedanzbasierten Modellierung für Batterien

ASIM Fachgruppentreffen, München, 20./21. 2. 2006

Dipl.-Ing. Julia Schiffer

Fachgruppe Elektrochemische Energiewandlung und Speichersystemtechnik
Prof. Dr. Dirk Uwe Sauer

Institut für Stromrichtertechnik und Elektrische Antriebe
Rheinisch-Westfälische Technische Hochschule Aachen
Univ.-Prof. Dr. ir. R. W. De Doncker

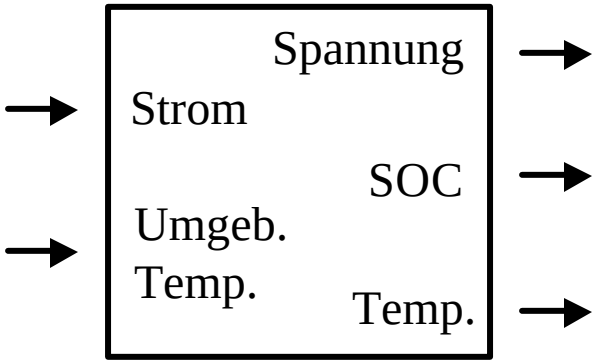
Motivation

- Warum brauchen wir Simulationsmodelle von Batterien?
  - Gestiegene Anzahl der Verbraucher im Fahrzeug und verbesserte Ausnutzung des Brennstoffs
  - Simulationen helfen bei der Auslegung von Systemen und Strategien
  - ➔ Geeignete Modelle sind nötig (schnelle Berechnung und relativ einfache Struktur)

- Welche Simulationsmodelle von Batterien gibt es?
  - **Physikalisch / chemisch basiert**: genaue Modelle, aber hoher Berechnungs- und Parametrierungsaufwand
  - **Empirische Modelle (Black Box)**: begrenzter Gültigkeitsbereich
  - **Elektr. Ersatzschaltbilder**: leichte Implementierung und Parametrierung

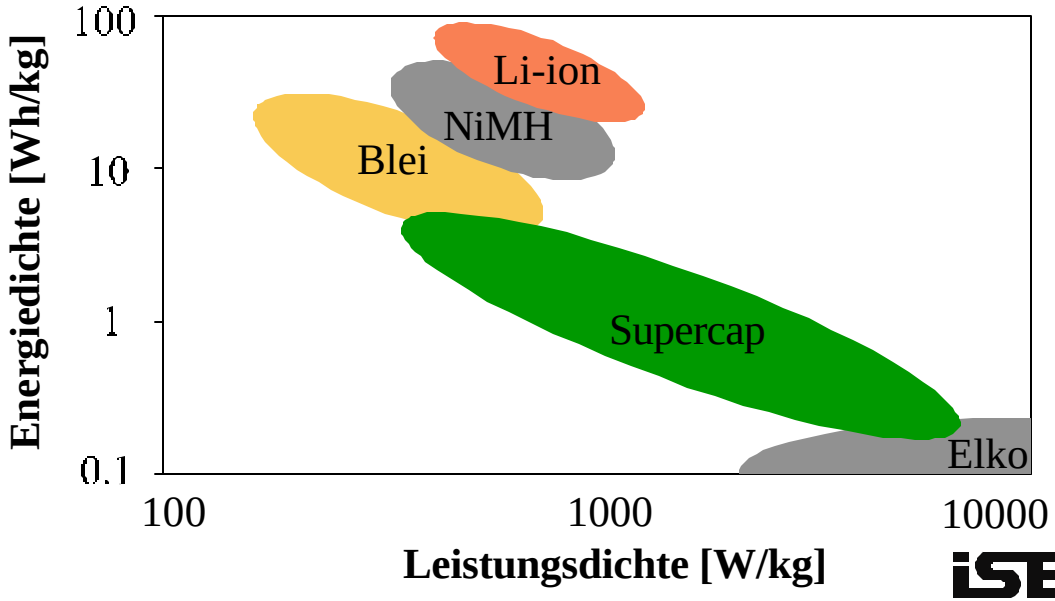
Impedanzbasierte Modellierung

- Speichertechnologie-übergreifendes Konzept
- Dynamische Modelle, z.B. für Start/Stop Betrieb
- Physikalisch-basierte Ersatzschaltbilder
- Parametrierung mittels elektrischer Messungen



Modellierte Systeme:

- Superkondensatoren
- Li-Ionen Batterien
- Bleibatterien
- NiMH Batterien

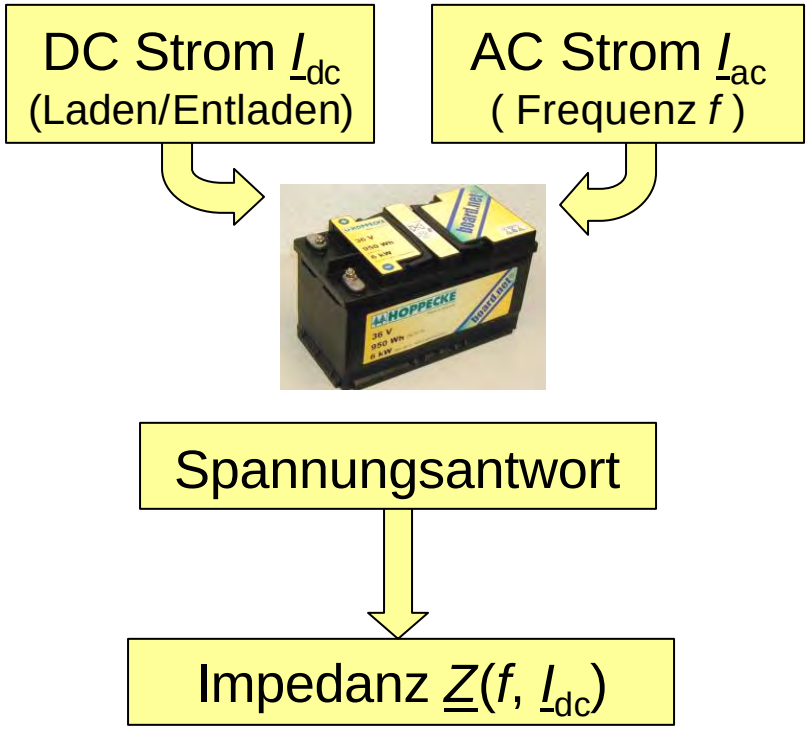


Gliederung

- Konzept der Impedanzbasierten Modellierung
  - Beispiel Bleibatterie
- Implementierung und Ergebnisse
- Grenzen der impedanzbasierten Modellierung
- Ergebnisse mit anderen Batterietypen
- Zusammenfassung

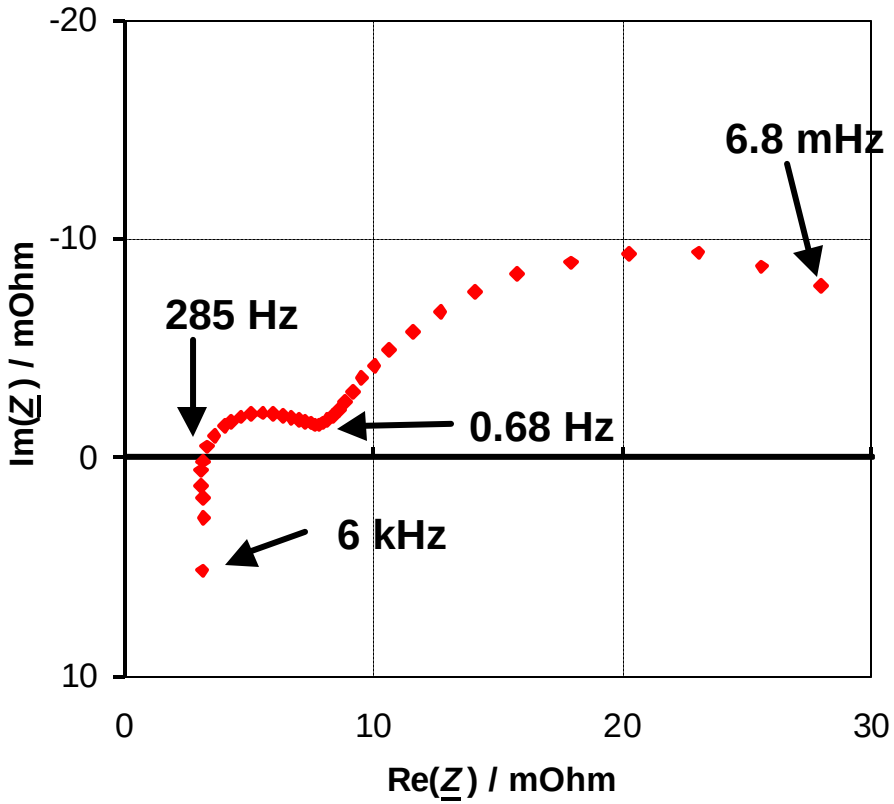
Elektrochemische Impedanzspektroskopie (EIS)

- Messmethode:
  - Einprägung eines kleinen Wechselstromsignals mit definierter Frequenz
  - ggf. Überlagerung eines Gleichstroms
  - Messung der resultierenden Spannung
  - Bestimmung der komplexen Impedanz
    $\underline{Z}(f, \underline{I}_{dc}) = \underline{U}_{ac} / \underline{I}_{ac}$



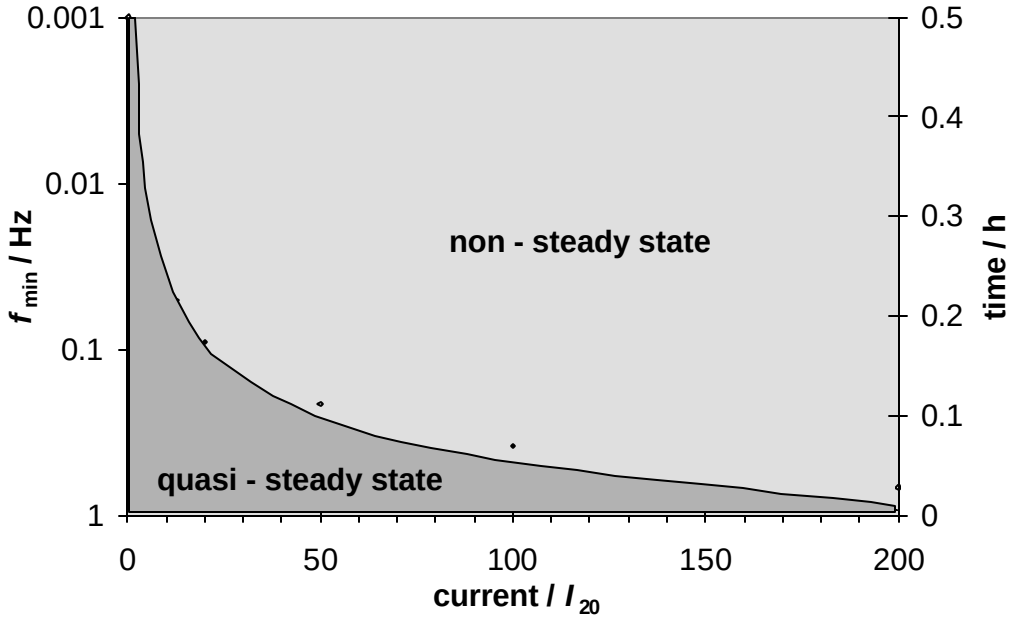
Elektrochemische Impedanzspektroskopie (EIS)

- Messmethode:
  - Einprägung eines kleinen Wechselstromsignals mit definierter Frequenz
  - ggf. Überlagerung eines Gleichstroms
  - Messung der resultierenden Spannung
  - Bestimmung der komplexen Impedanz
    $\underline{Z}(f, \underline{I}_{dc}) = \underline{U}_{ac} / \underline{I}_{ac}$
  - Start bei hohen Frequenzen



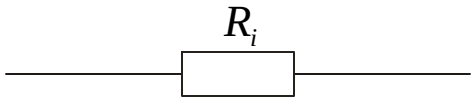
Elektrochemische Impedanzspektroskopie (EIS)

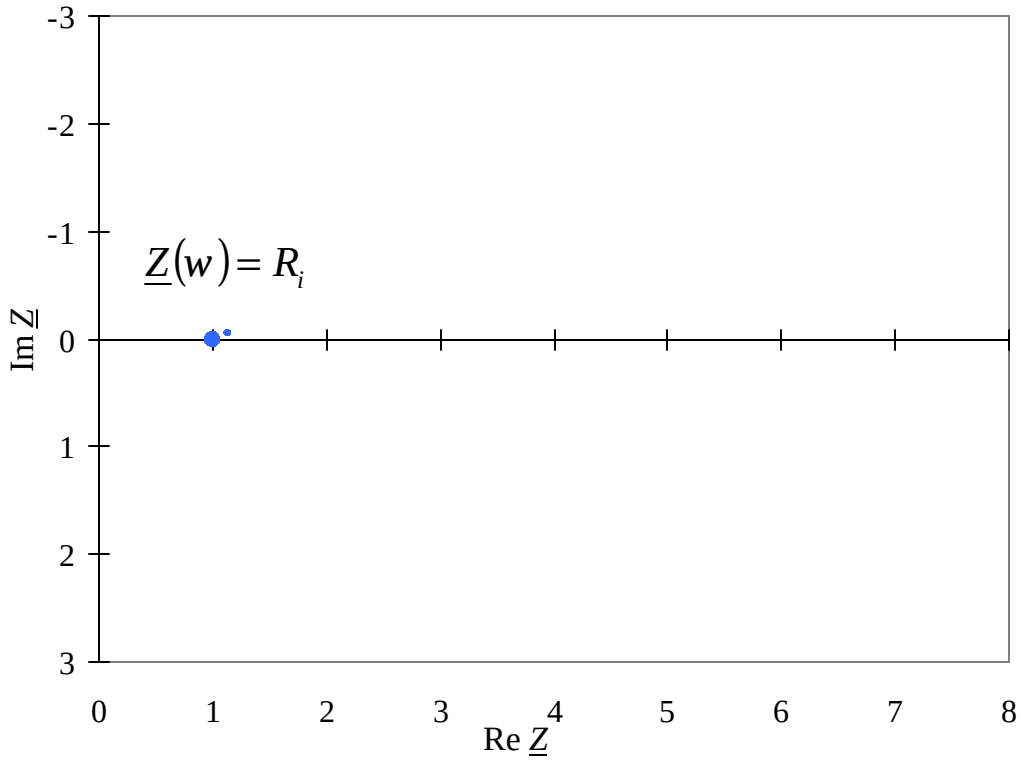
- Sicherung des quasi-stationären Zustands während der Messung:
  - **DSOC < 5%** halten
  - I_{dc} begrenzt die Messdauer und die minimale Frequenz



Konzept der Impedanzspektroskopie

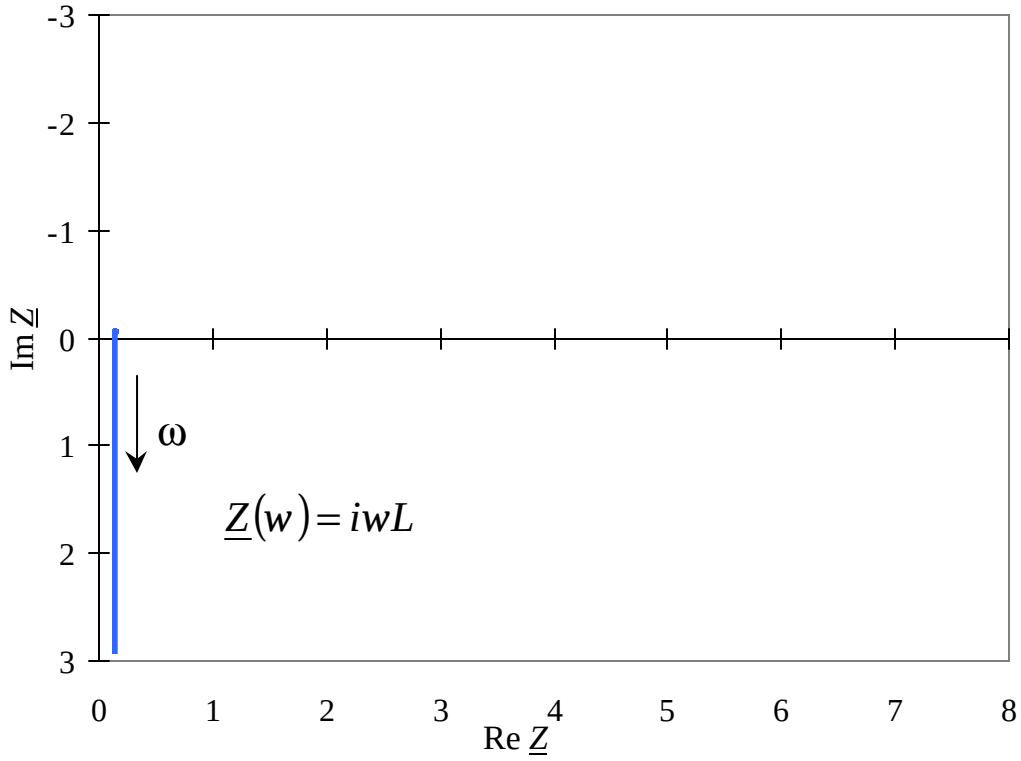
- Widerstand = Punkt





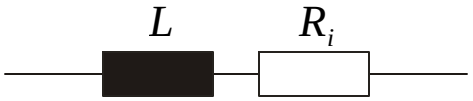
Konzept der Impedanzspektroskopie

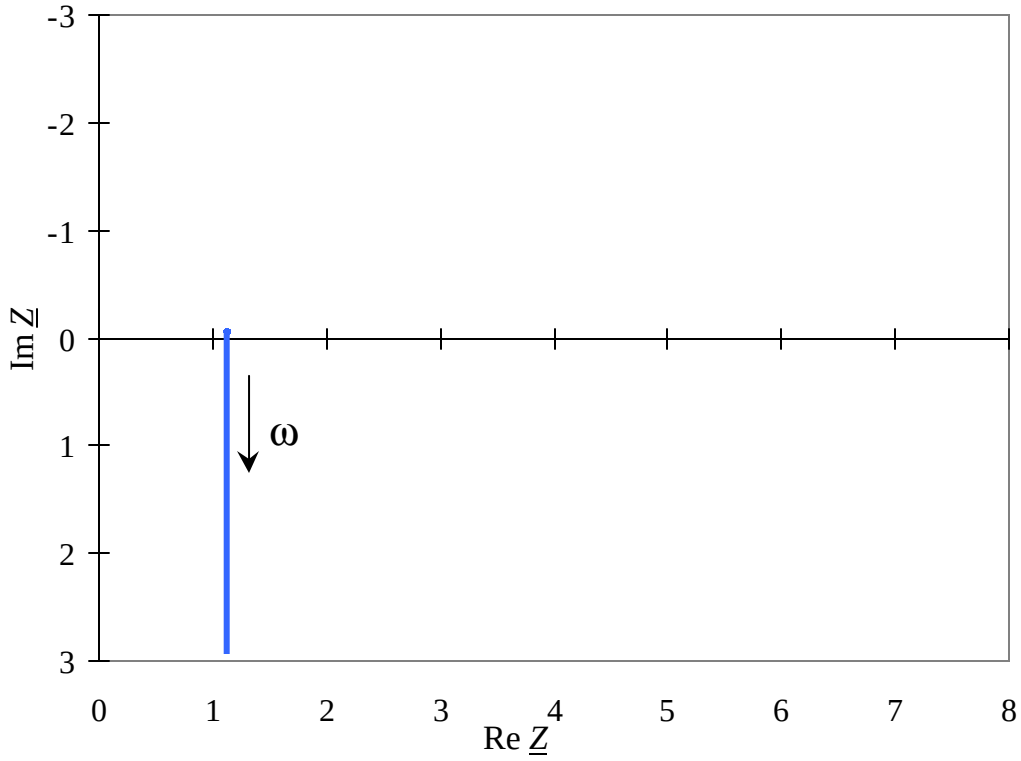
- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil



Konzept der Impedanzspektroskopie

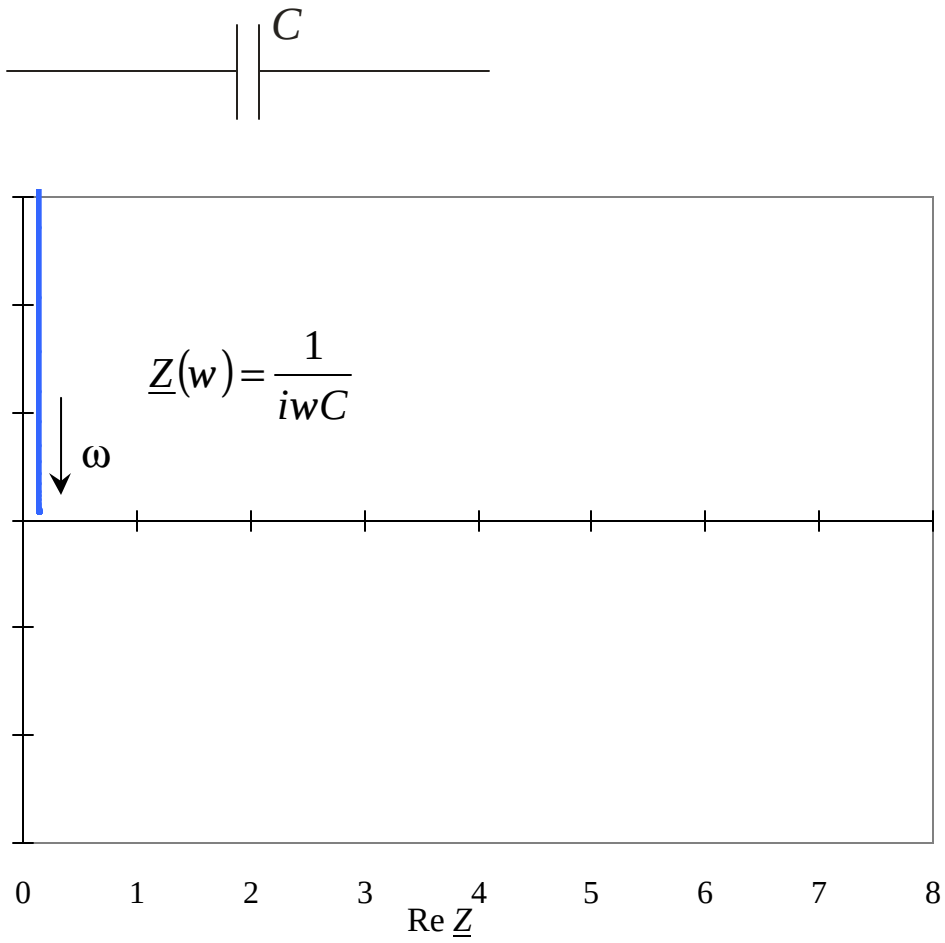
- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil





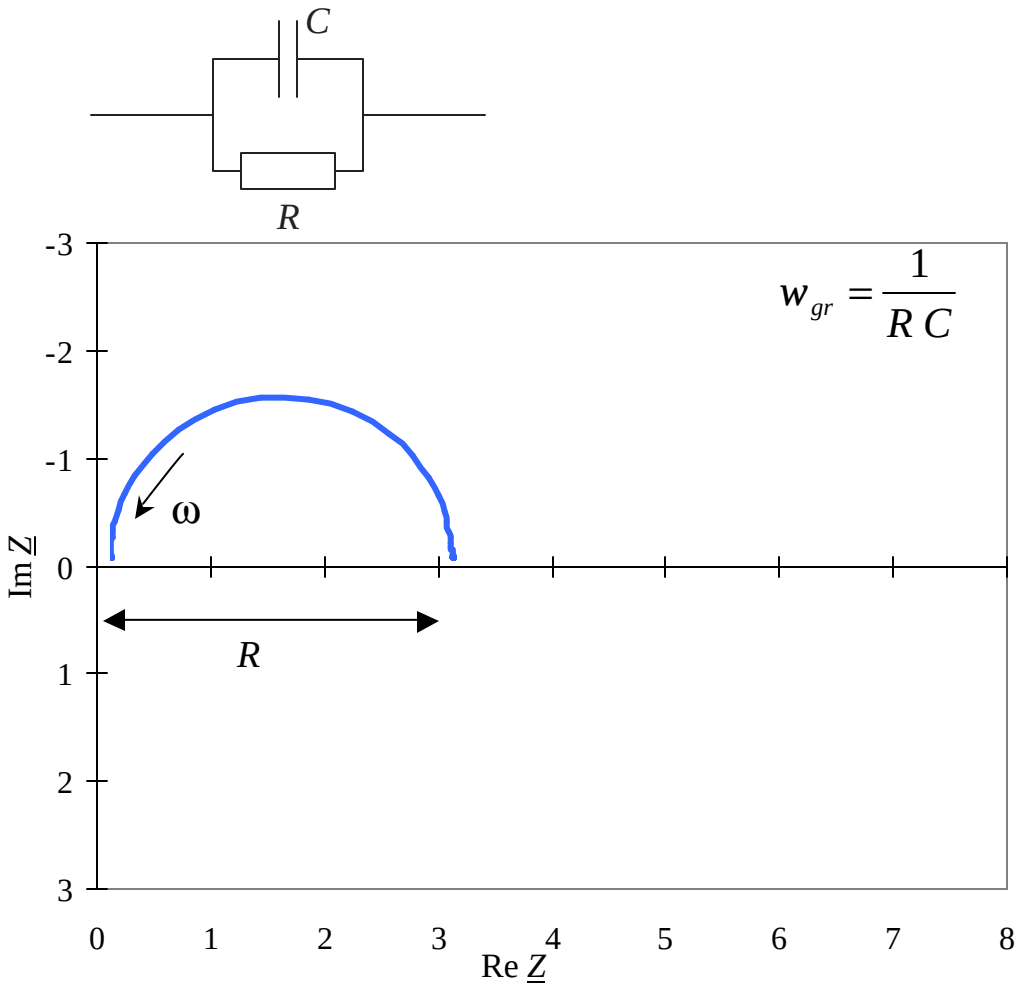
Konzept der Impedanzspektroskopie

- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil
- Kapazität = senkrechte Gerade mit neg. Imaginärteil



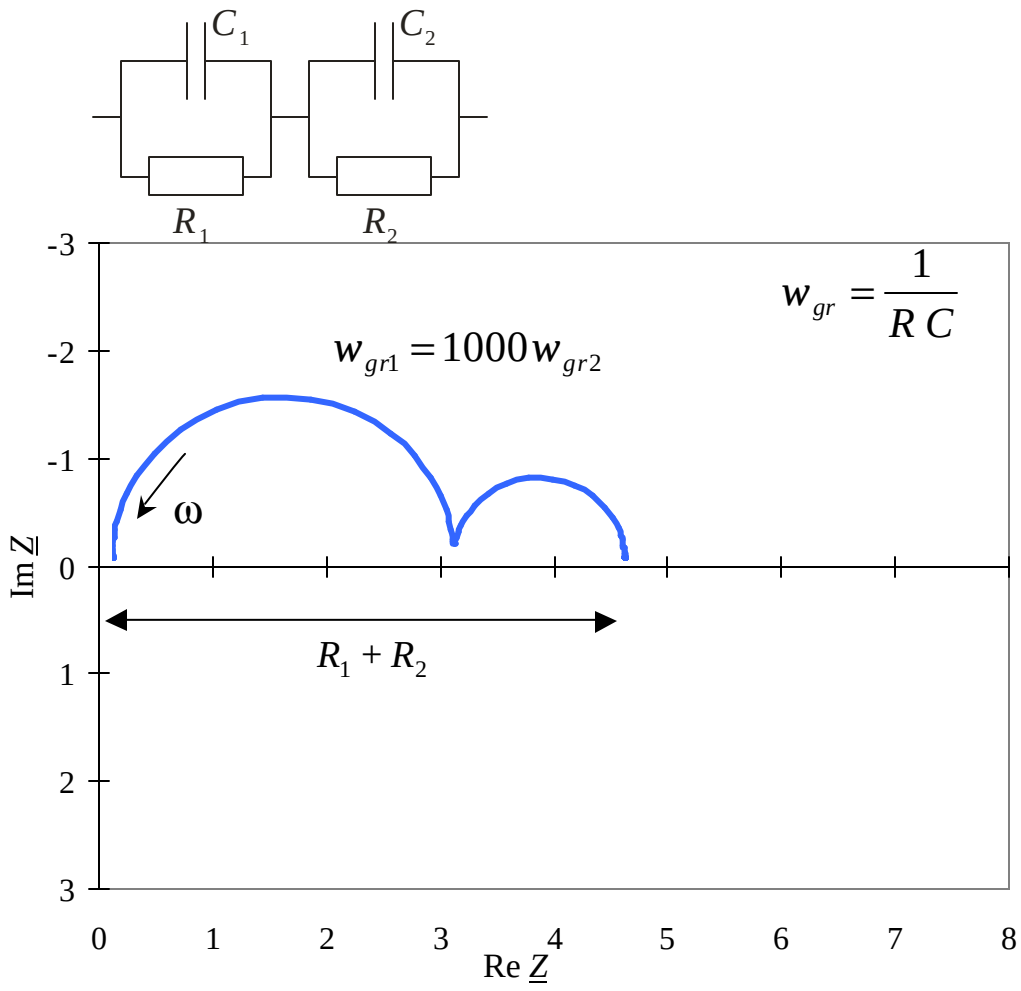
Konzept der Impedanzspektroskopie

- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil
- Kapazität = senkrechte Gerade mit neg. Imaginärteil
- Widerstand parallel Kapazität = Halbkreis mit neg. Imaginärteil



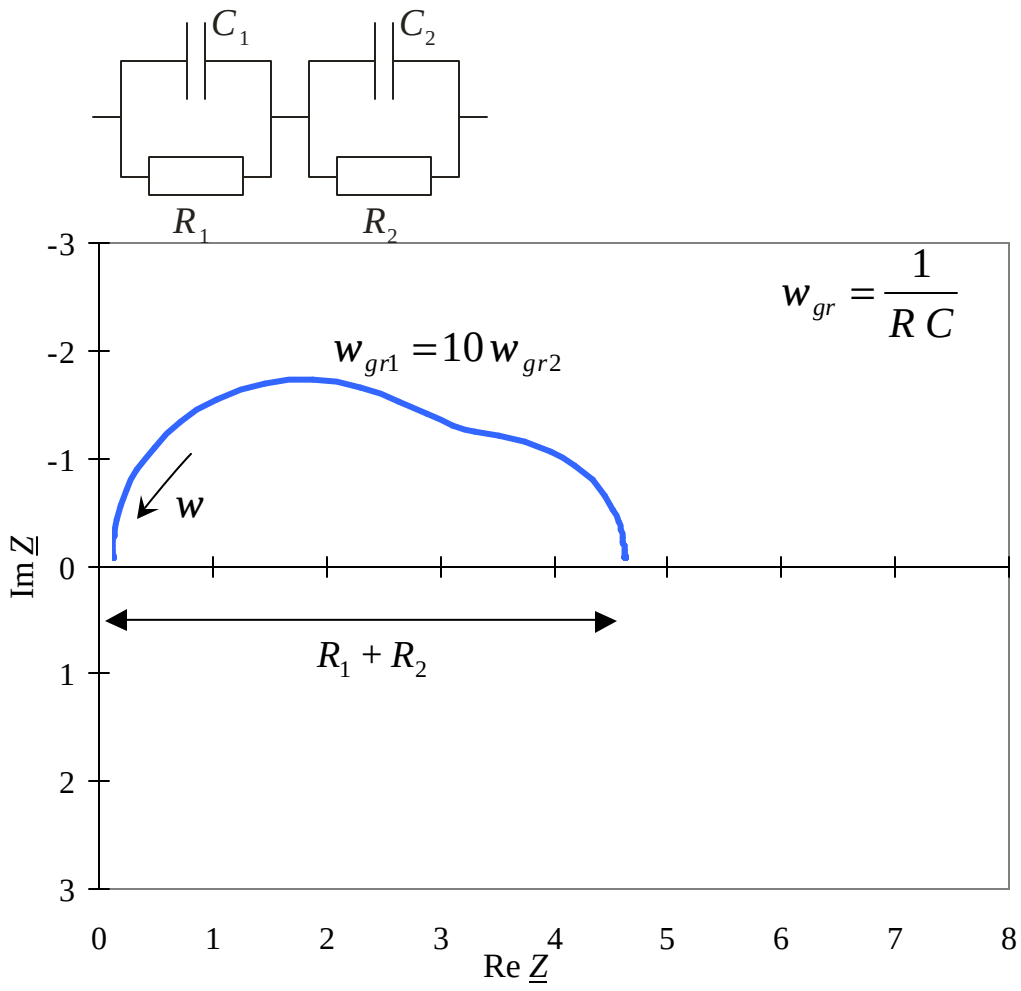
Konzept der Impedanzspektroskopie

- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil
- Kapazität = senkrechte Gerade mit neg. Imaginärteil
- Widerstand parallel Kapazität = Halbkreis mit neg. Imaginärteil



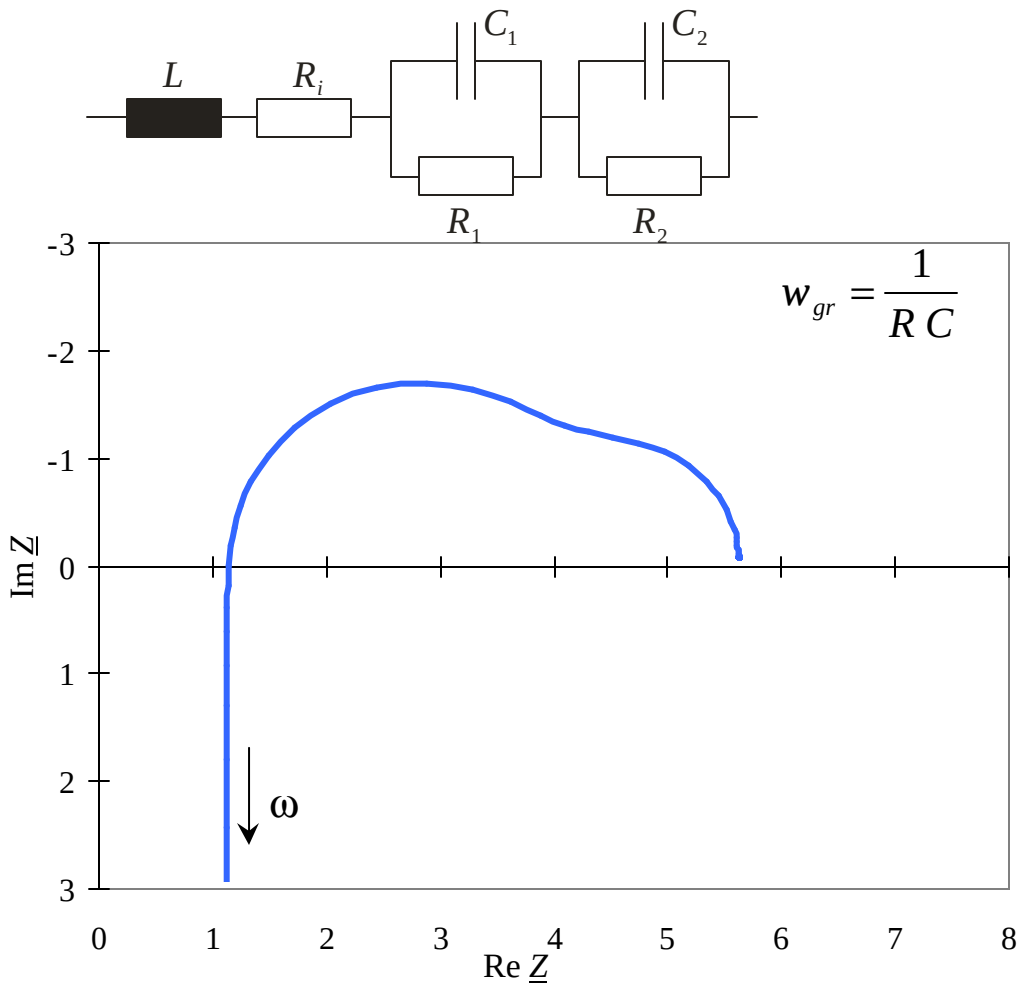
Konzept der Impedanzspektroskopie

- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil
- Kapazität = senkrechte Gerade mit neg. Imaginärteil
- Widerstand parallel Kapazität = Halbkreis mit neg. Imaginärteil



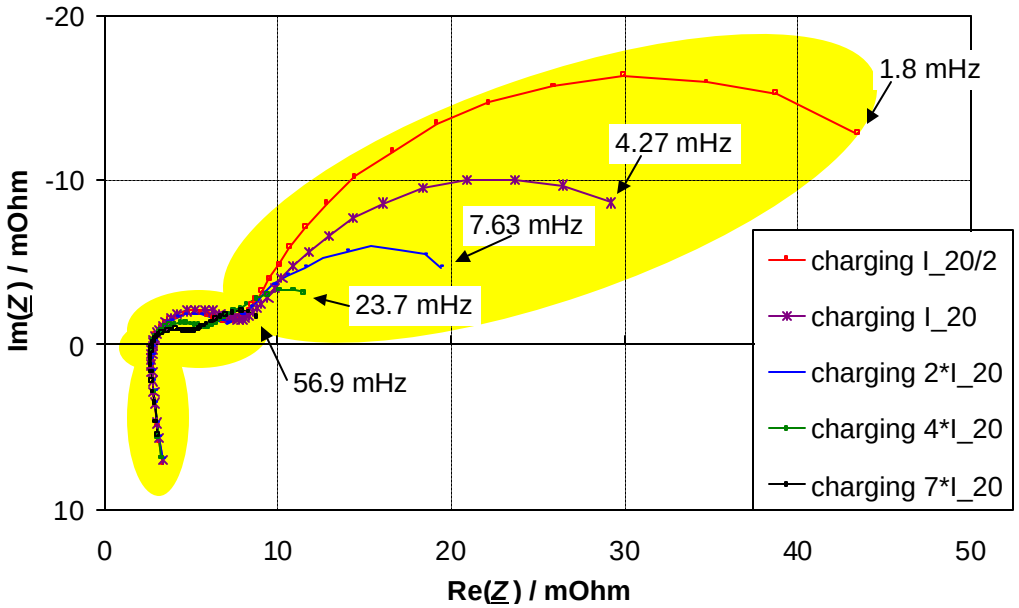
Konzept der Impedanzspektroskopie

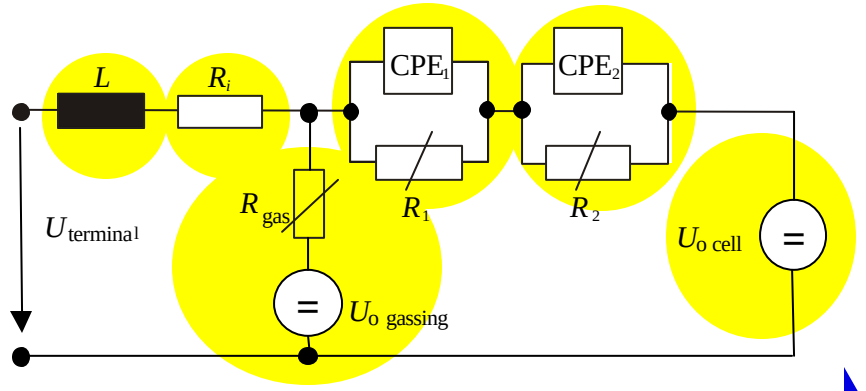
- Widerstand = Punkt
- Induktivität = senkrechte Gerade mit pos. Imaginärteil
- Kapazität = senkrechte Gerade mit neg. Imaginärteil
- Widerstand parallel Kapazität = Halbkreis mit neg. Imaginärteil



Impedanz von Bleibatterien

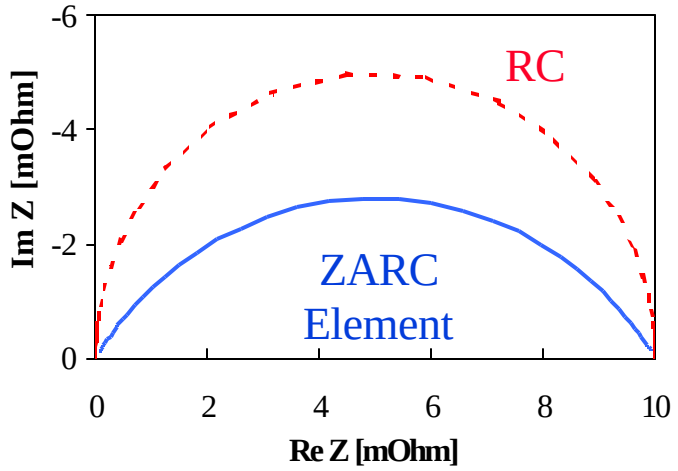
- Ladespektren
- Elektrisches ESB
  - Induktivität (induktiver Ast)
  - Ohm'scher Widerstand
  - zwei generalisierte RC-Glieder ("eingedrückte" Halbkreise)
- Gasungsreaktion (Nebenreaktion)
- Leerlaufspannung (OCV)

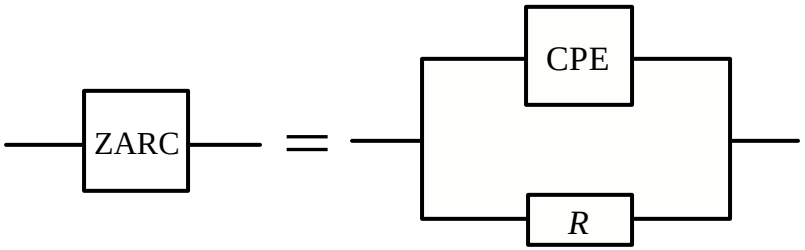




ZARC Element für eingedrückte Halbkreise

- CPE (constant phase element) statt Kapazität
- Parallelschaltung mit Widerstand ergibt ZARC Element





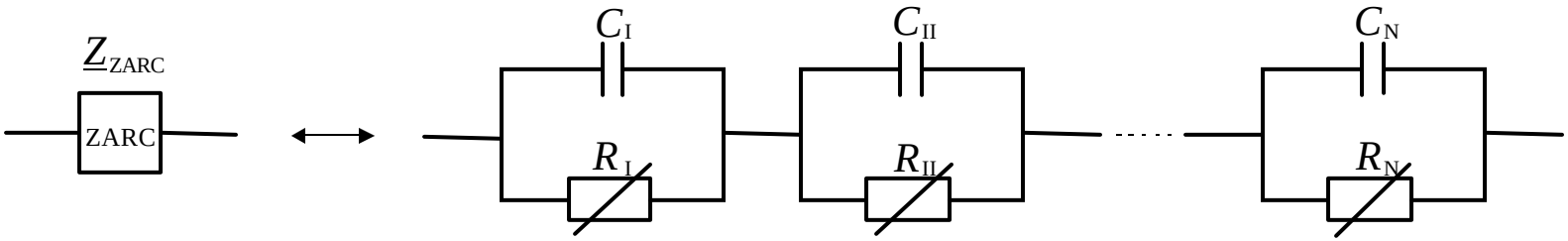
$$\underline{Z}_{\mathrm{CPE}} = A \cdot (\mathrm{j}w)^{-e}$$
$$0 < e \leq 1$$

Gliederung

- Konzept der Impedanzbasierten Modellierung
  - Beispiel Bleibatterie
- Implementierung und Ergebnisse
- Grenzen der impedanzbasierten Modellierung
- Ergebnisse mit anderen Batterietypen
- Zusammenfassung

Matlab/Simulink Implementierung

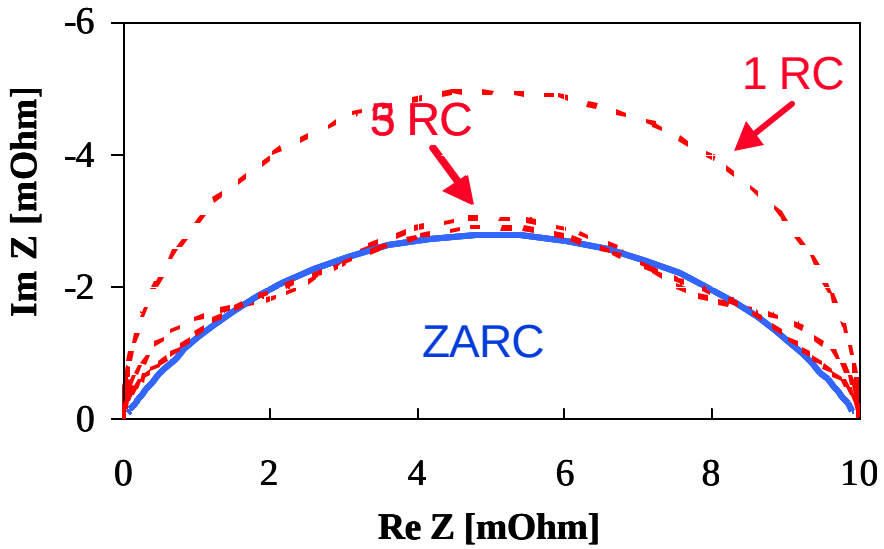
- Parameter R, L, $C = f\,(I_{dc}, SOC, J)$ problemlos zu implementieren
- Implementierung elektrochemischer Impedanzen (ZARC)
  - ZARC Element → Reihenschaltung nicht-linearer RC Glieder



  - RC Glieder durch Parameter R_I, C_I, ... R_N, C_N beschrieben
  - R_I, C_I, ... R_N, C_N sind **mathematische** Parameter
  - Algorithmus zur Offline-Ermittlung aus ZARC Parametern R, A, e
  - Anzahl der RC Glieder: Kompromiss zwischen Genauigkeit ↔ Simulationszeit
  - Varianten: z.B. 1 RC Glied, 3 RC Glieder, 5 RC Glieder

Matlab/Simulink Implementierung

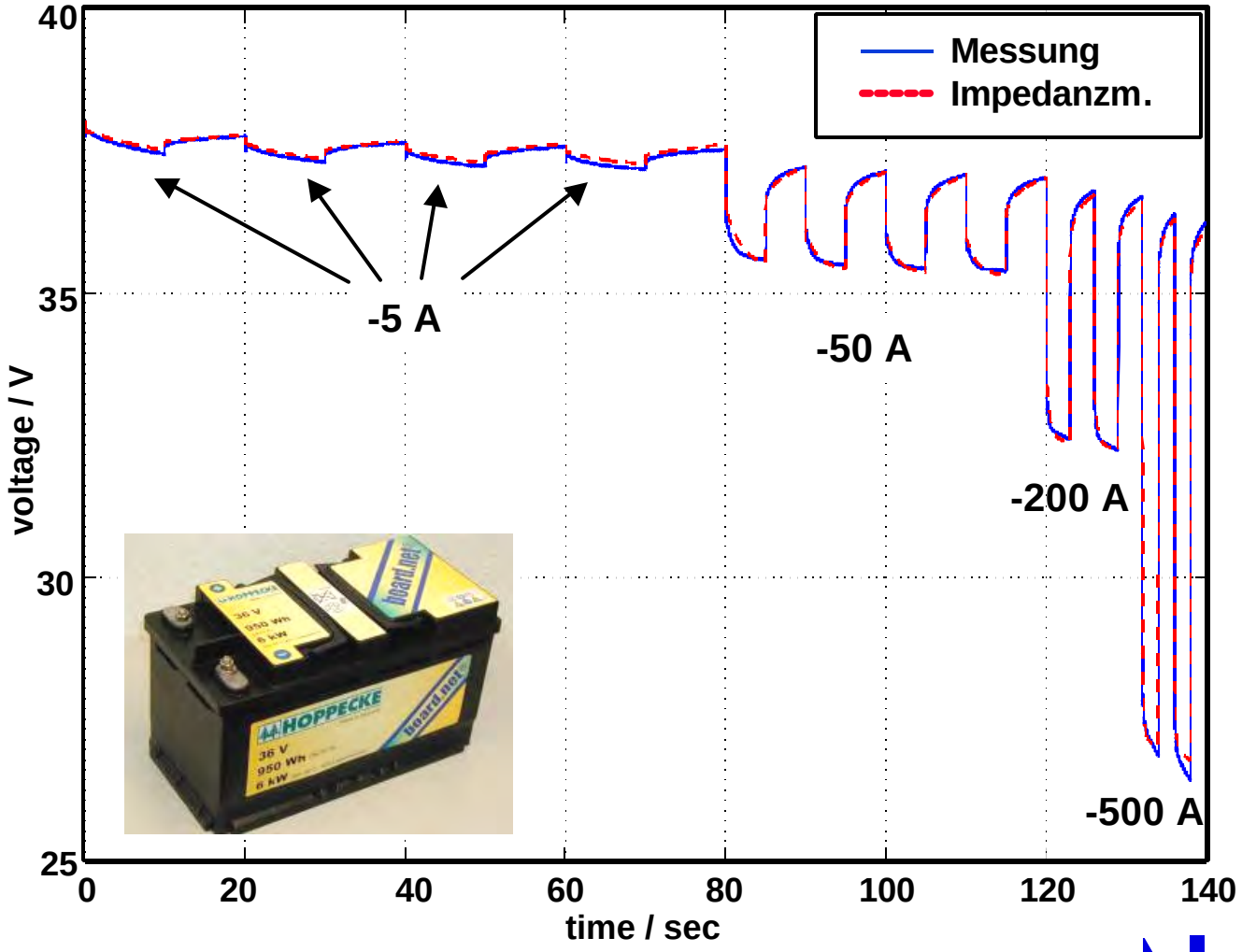
- Parameter R, L, $C = f\,(I_{dc}, SOC, J)$ problemlos zu implementieren
- Implementierung elektrochemischer Impedanzen (ZARC)
  - ZARC Element → Reihenschaltung nicht-linearer RC Glieder



  - Standard-Implementierung: 3 RC Glieder

Vergleich zwischen Messung und Modellierung

- Validierung durch hochdynamisches Stromprofil
- 36V, VRLA/AGM
- C_N = 27,5 Ah
- sehr gute Übereinstimmung zwischen Messung und Impedanzmodell

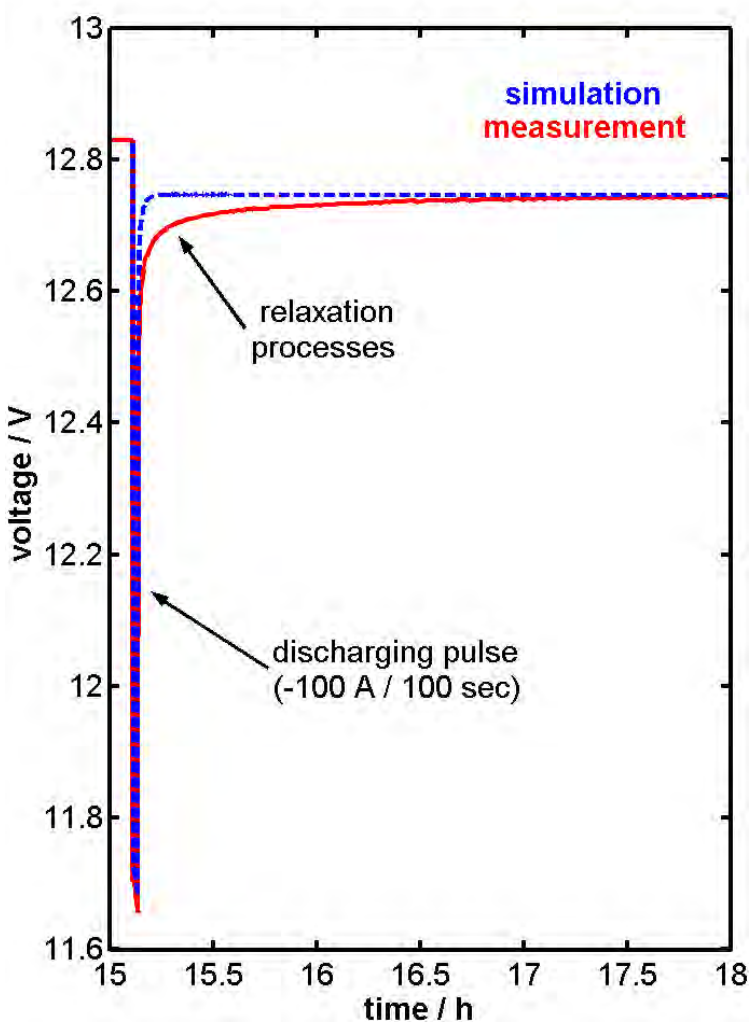


Gliederung

- Konzept der Impedanzbasierten Modellierung
  - Beispiel Bleibatterie
- Implementierung und Ergebnisse
- Grenzen der impedanzbasierten Modellierung
- Ergebnisse mit anderen Batterietypen
- Zusammenfassung

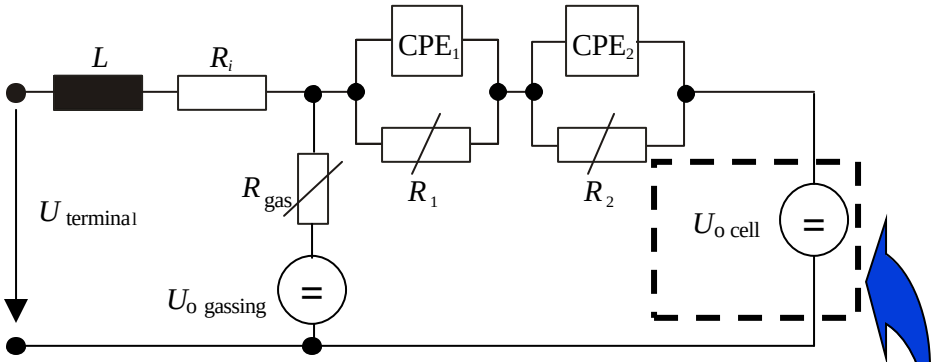
Grenzen

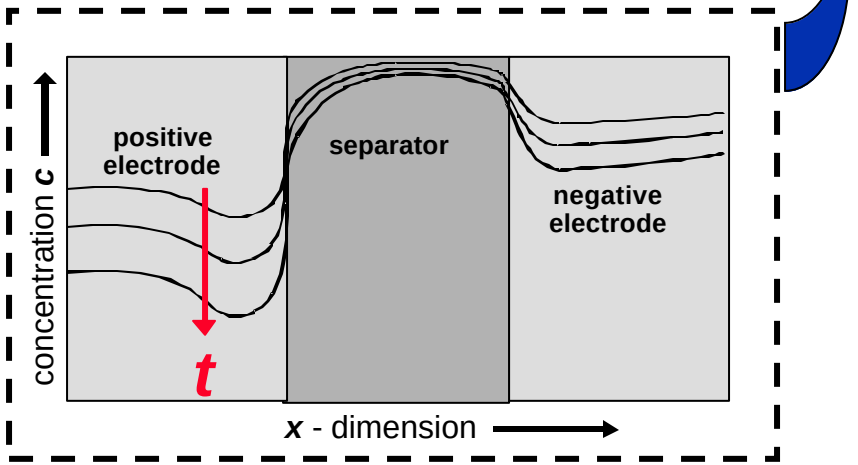
- Langzeitverhalten mit EIS nicht messbar
- Relaxationsprozesse mit Zeitkonstanten im Stundenbereich können nicht nachgebildet werden
- Modellverbesserung: Berücksichtigung von Elektrolyt-Transport „Hybrid-Modell“



Lokale Elektrolytkonzentration ***c*(*x*,*t*)**

- Gleichspannung entspricht dem Unterschied der Elektrodenpotentiale
- Potentiale sind säurekonzentrationsabhängig
- rein impedanzbasiertes Modell:
  - eine mittlere Säurekonzentration (SOC-abhängig)
- in Realität:
  - Säurekonzentration orts- und zeitabhängig



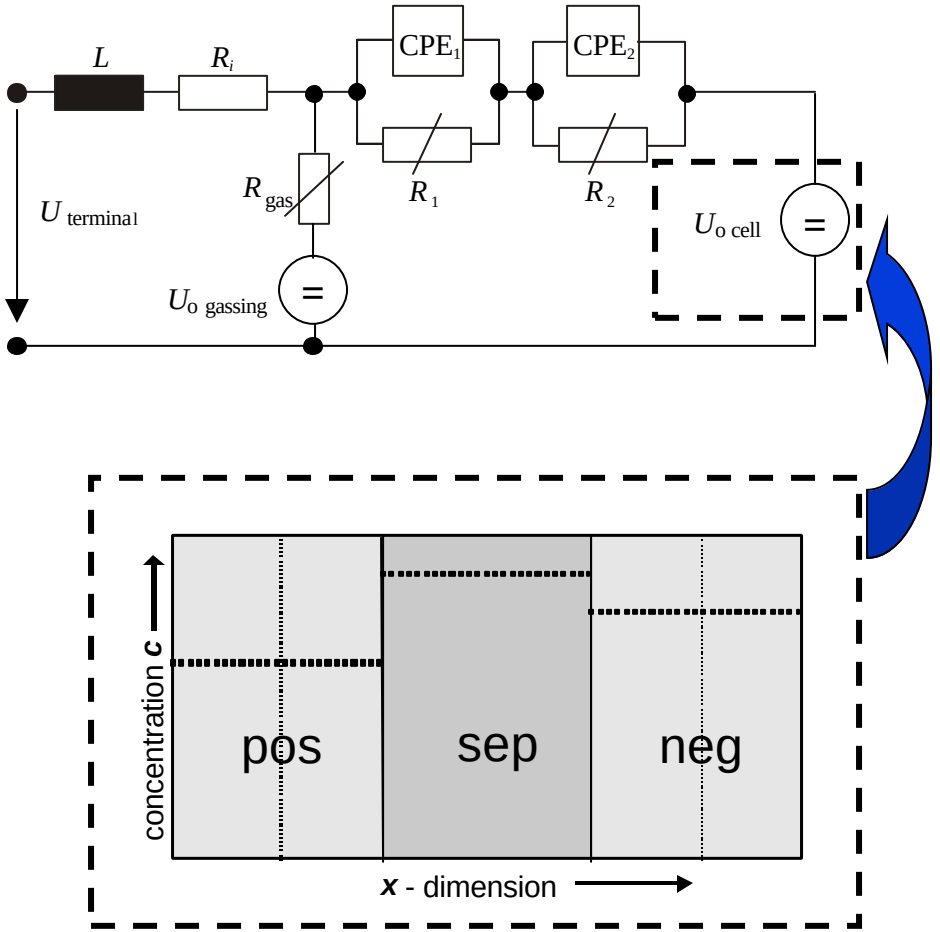


Vereinfachtes Elektrolyt-Transport-Modell

- Modellierung:
  - geringe räumliche Auflösung
  - drei mittlere Säurekonzentrationen

  ➔ geringer Rechenaufwand und einfache Implementierung

- “Hybrid-Modell”:
  - rein impedanzbasiertes Modell + Elektrolyt-Transport-Modell

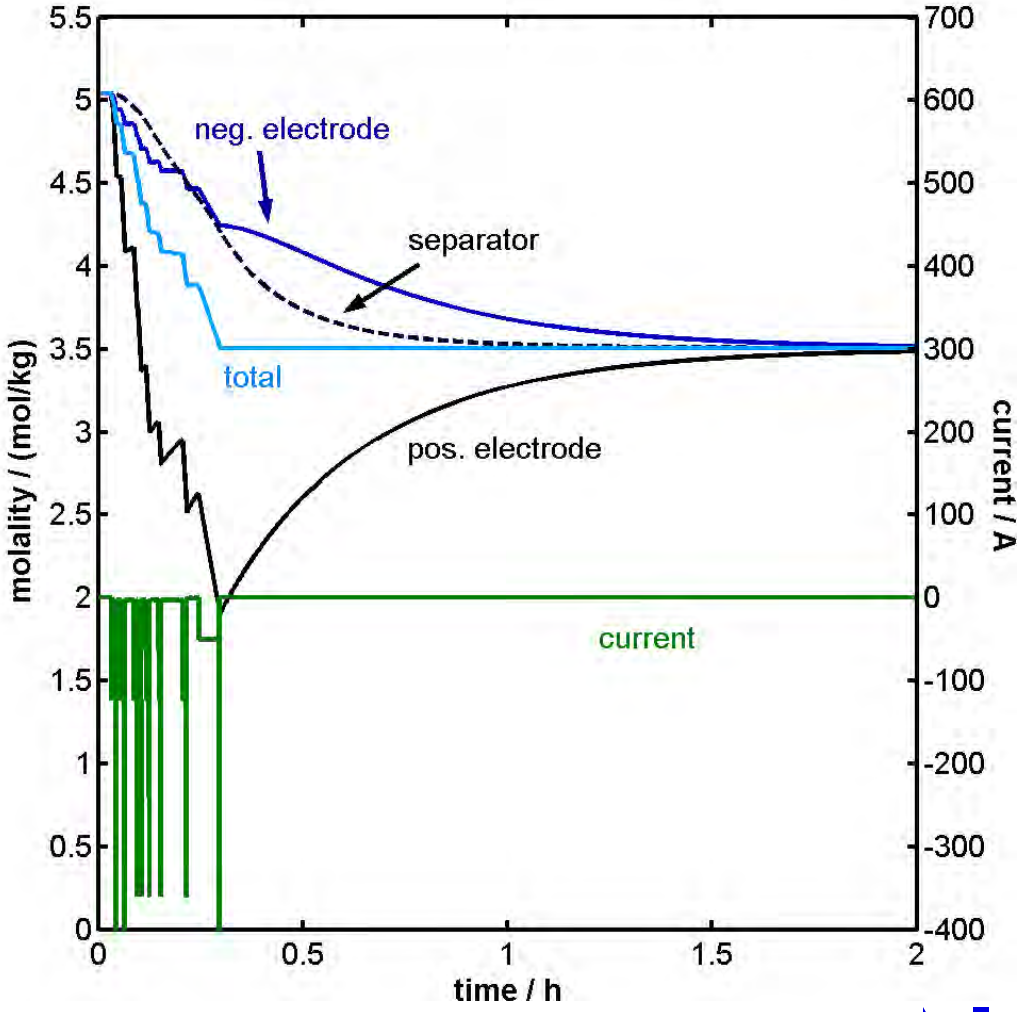


Vereinfachtes Elektrolyt-Transport-Modell

- Modellierung:
  - geringe räumliche Auflösung
  - drei mittlere Säurekonzentrationen

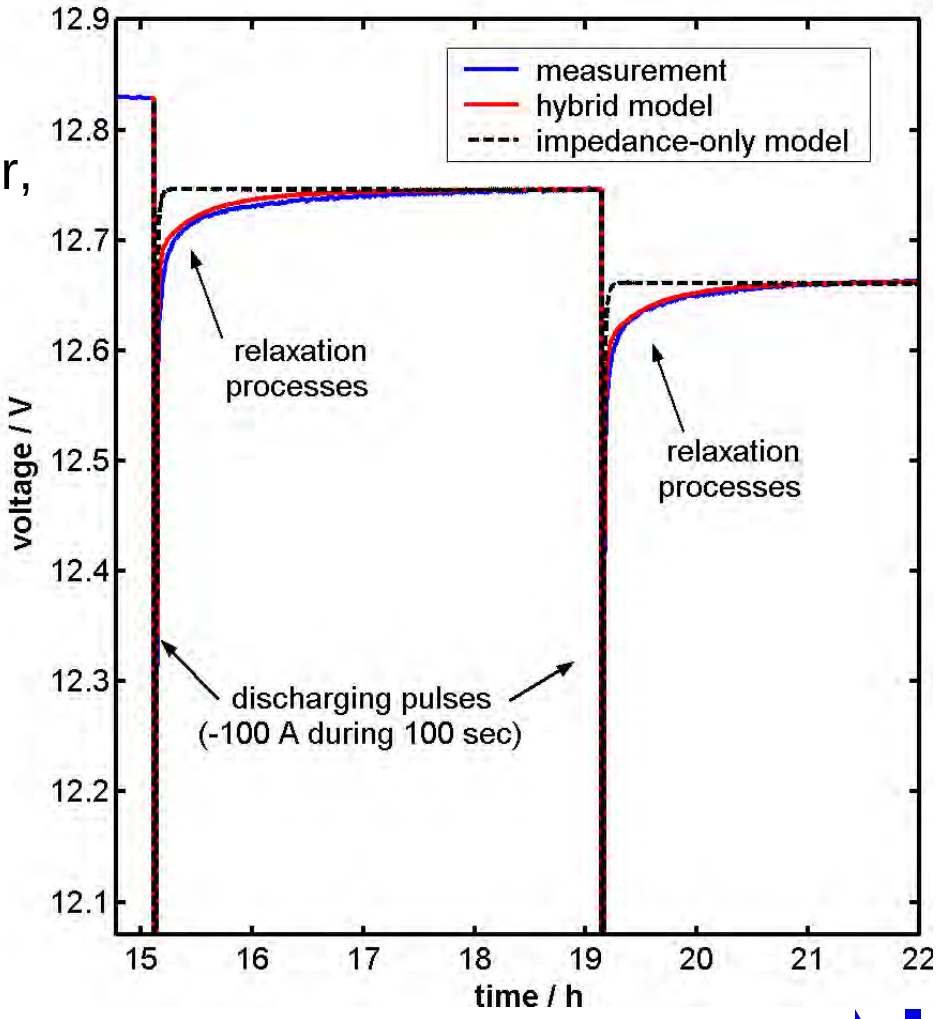
  ➔ geringer Rechenaufwand und einfache Implementierung

- “Hybrid-Modell”:
  - rein impedanzbasiertes Modell + Elektrolyt-Transport-Modell



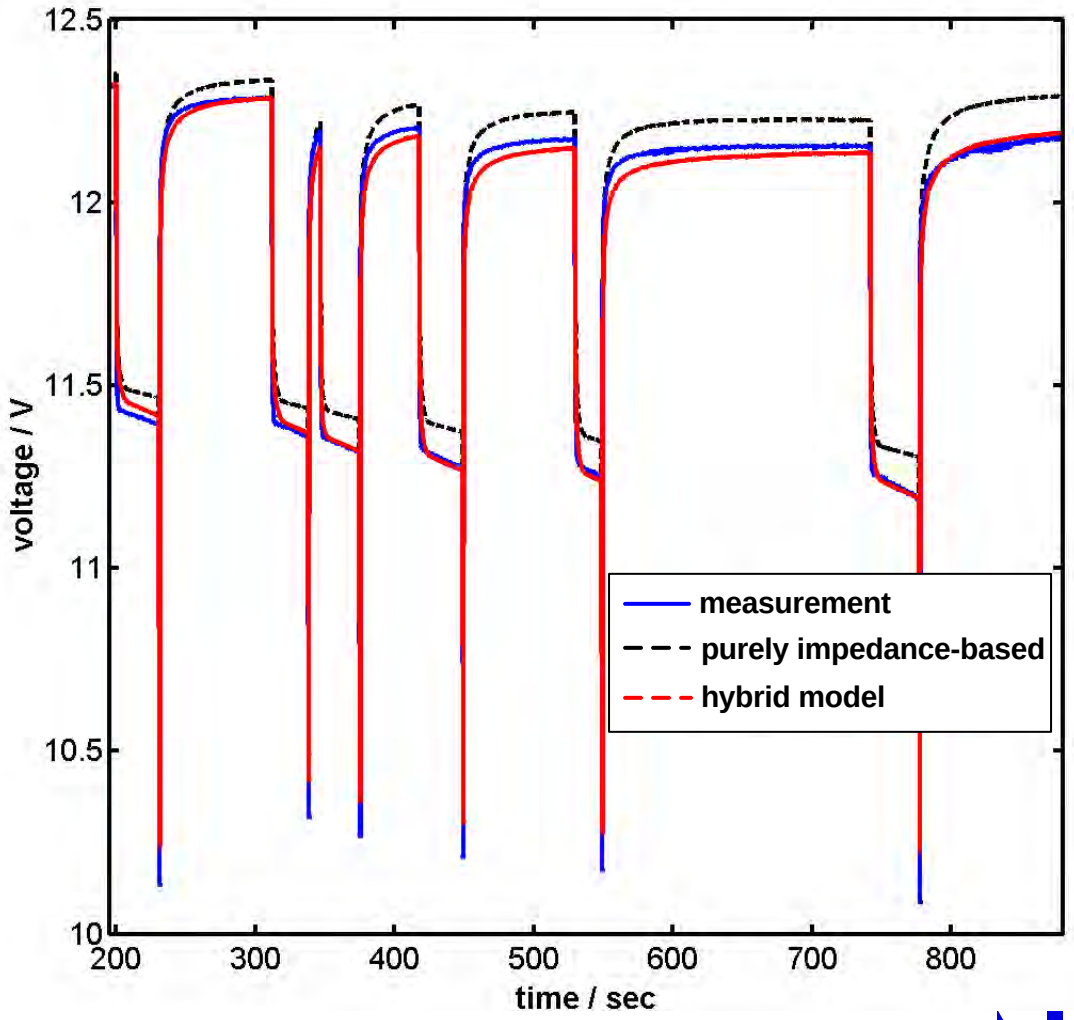
Parametrierung

- Geometrieparameter:
  - Dicke von Elektroden und Separator, geometrische Oberfläche
  - Porosität
- Parametrierung im Zeitbereich
  - Sprungantwort: Hochstrom-Entladepuls (100 A, 100 sec)
  - einmalige Anpassung der Diffusionskonstante $D_{diff,eff}$
  - keine weitere Anpassung nötig



Validierung des Hybrid-Modells

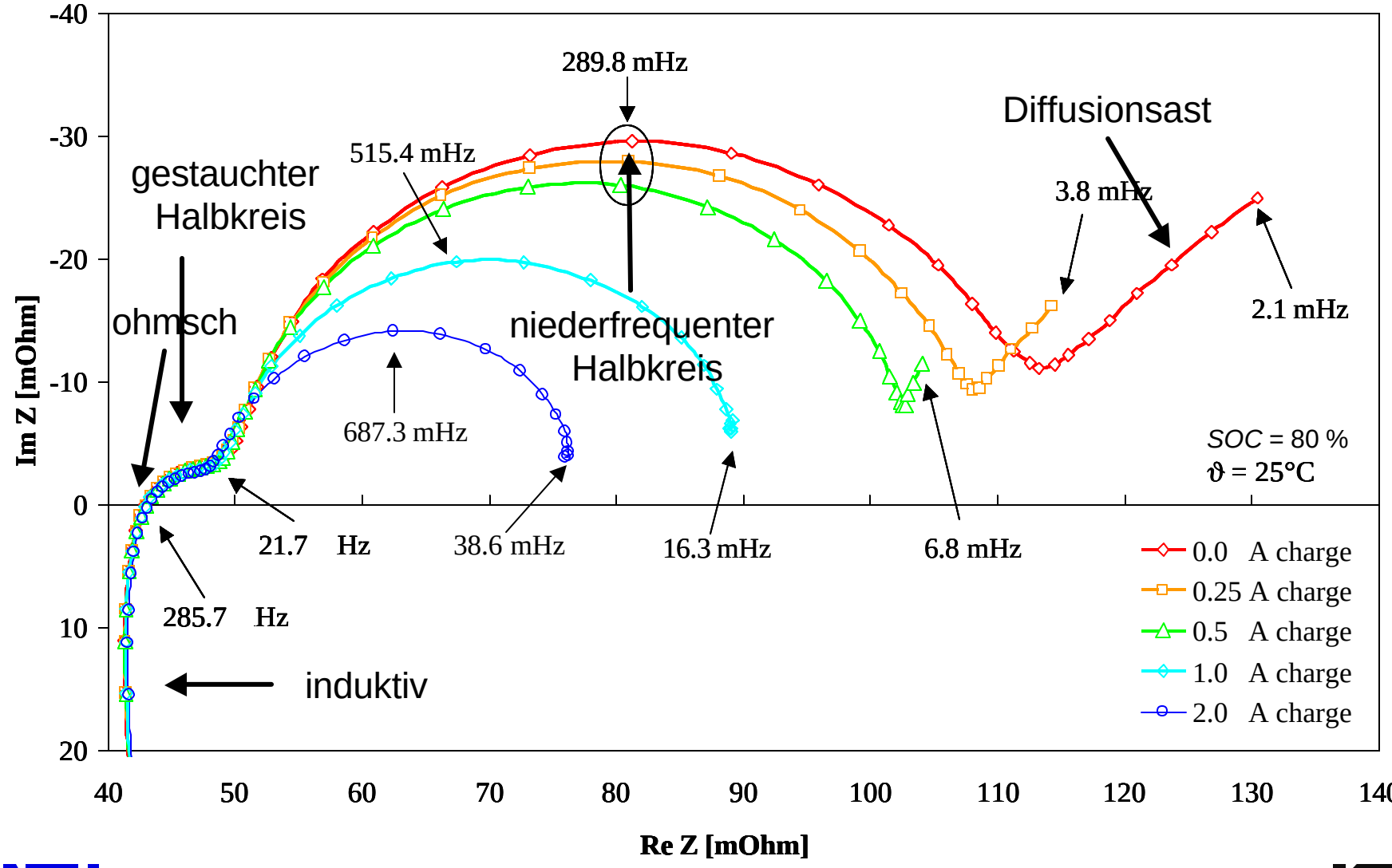
- Lastprofil / Start-Stop PKW (engine idle-off)
- 12V, VRLA/AGM, 44 Ah
- Ah Durchsatz: > 20 % C_N



Gliederung

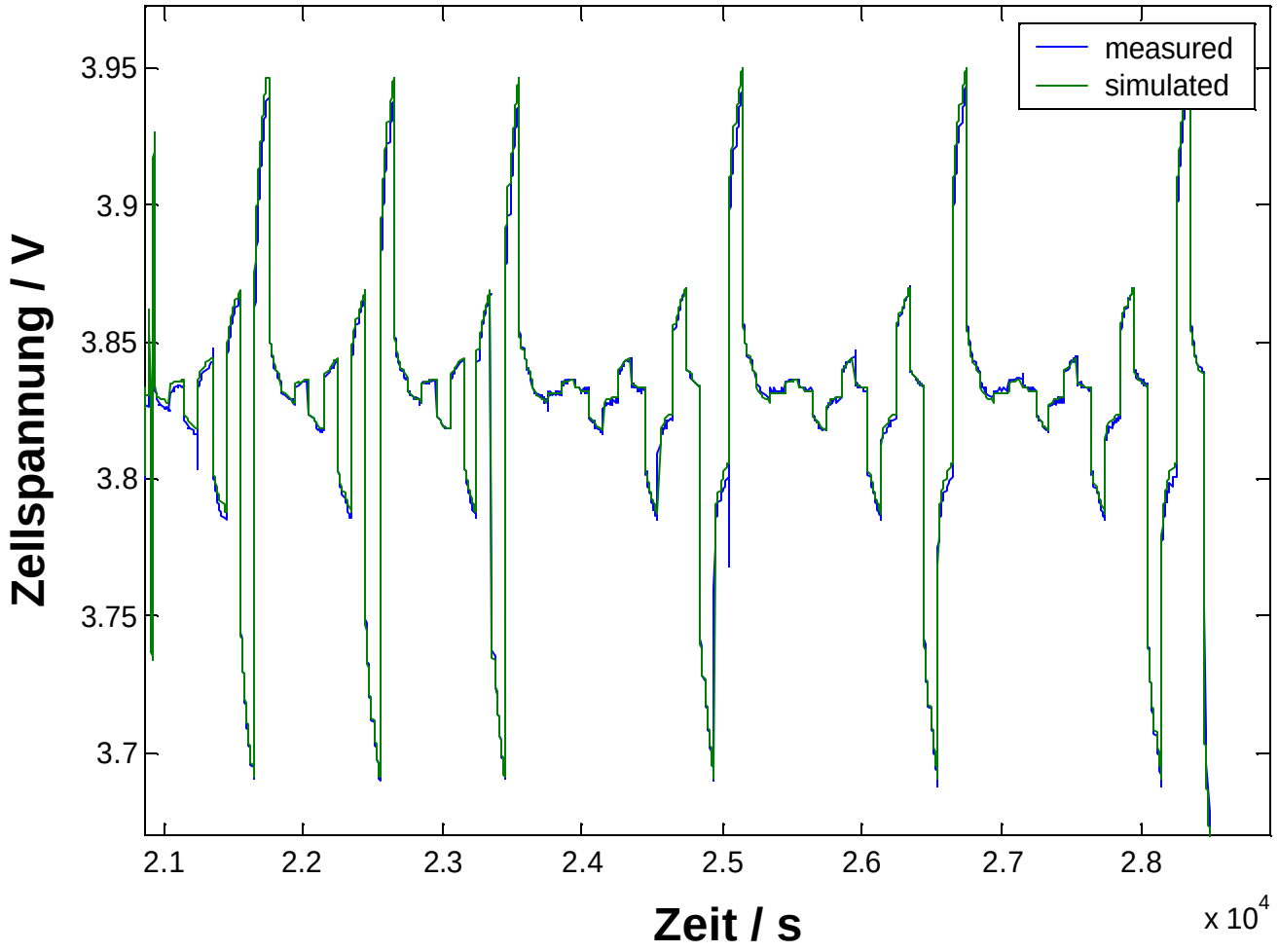
- Konzept der Impedanzbasierten Modellierung
  - Beispiel Bleibatterie
- Implementierung und Ergebnisse
- Grenzen der impedanzbasierten Modellierung
- Ergebnisse mit anderen Batterietypen
- Zusammenfassung

Typische Impedanzspektren von Li-Ion-Batterien



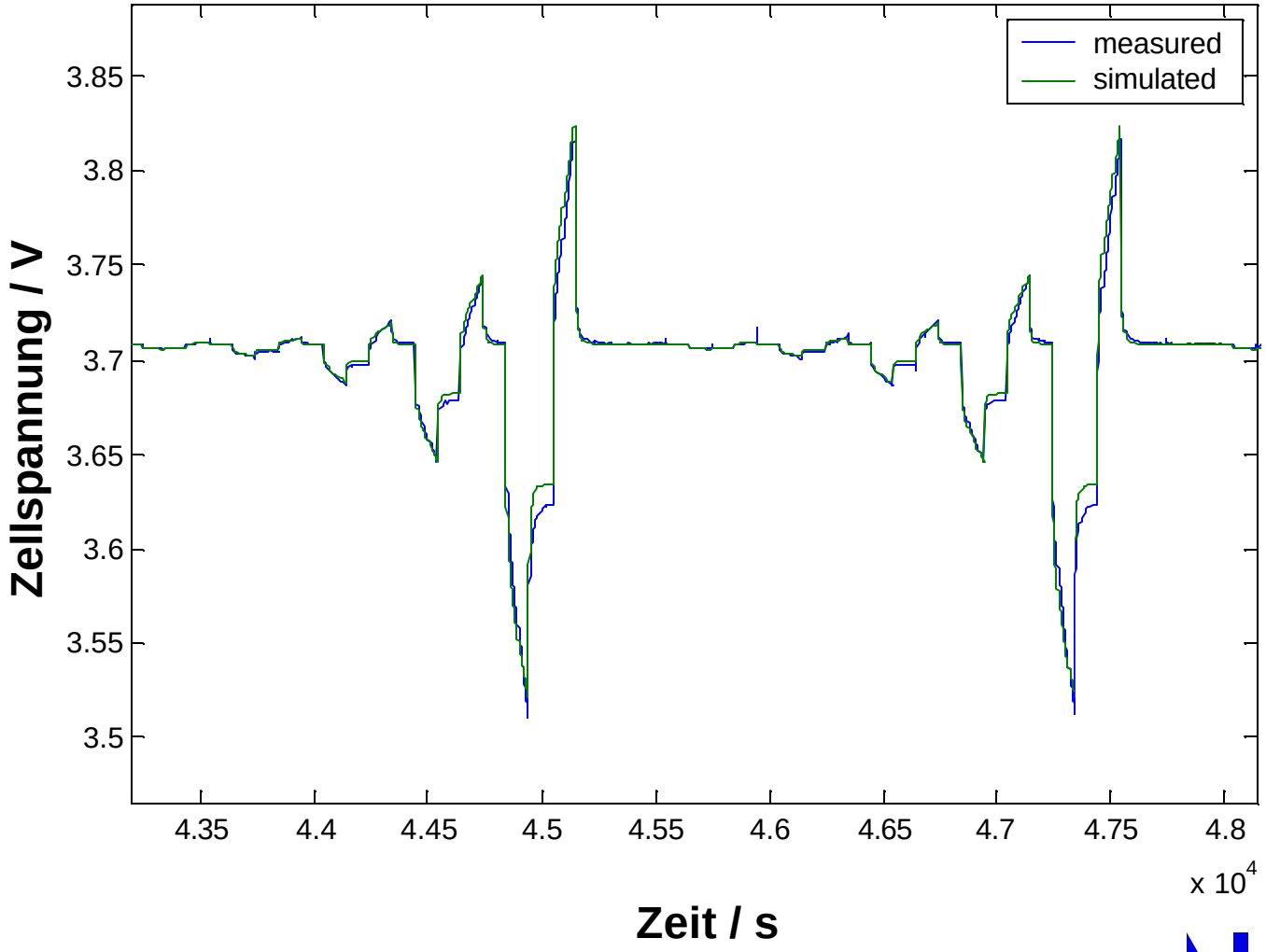
Verifikation - Lithium @ -18°C

- Ströme 100 mA bis 40A
- C_{Nenn} = 6,5 Ah



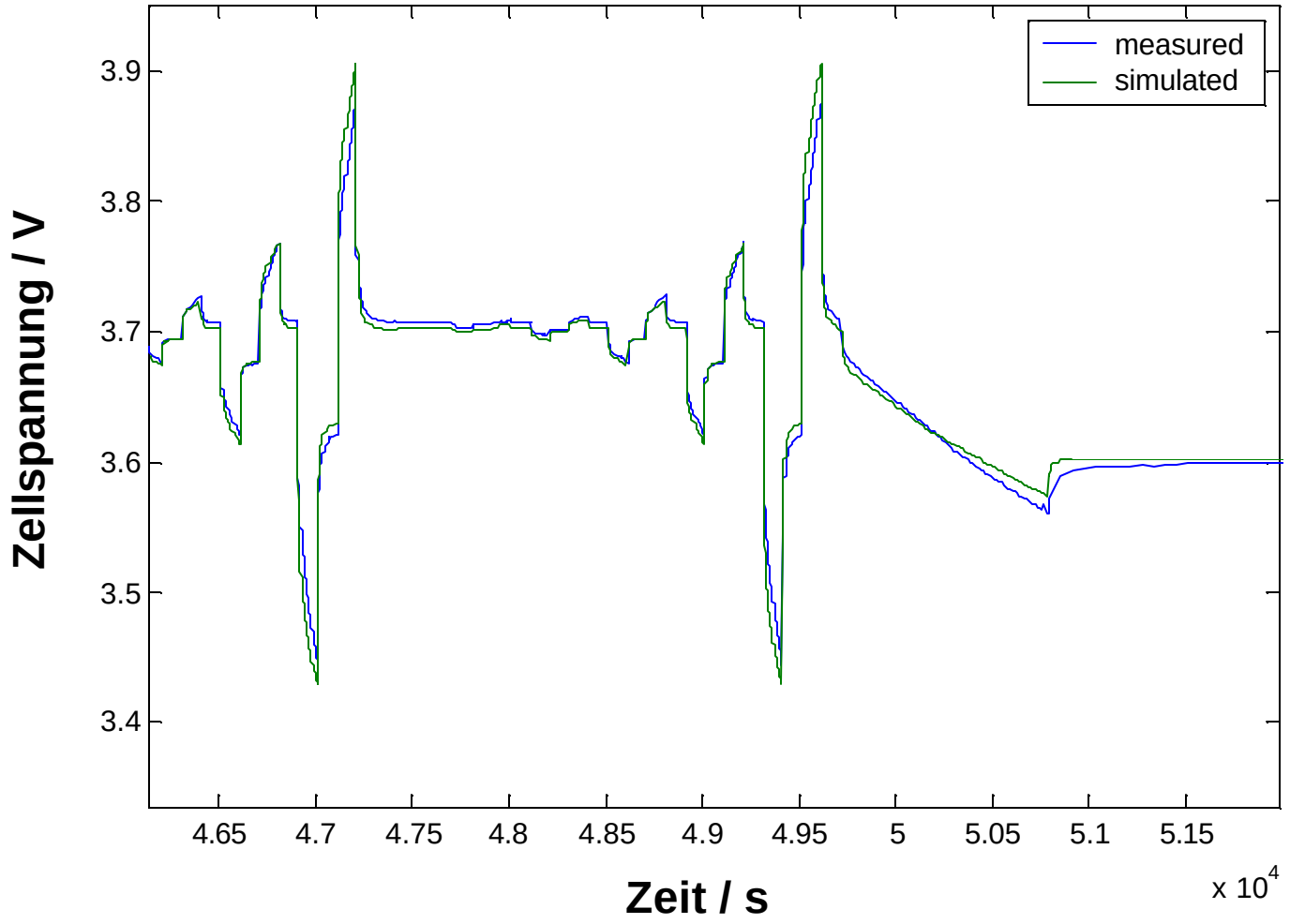
Verifikation - Lithium @ 40°C

- Ströme 100 mA bis 40A
- C_{Nenn} = 7,5 Ah



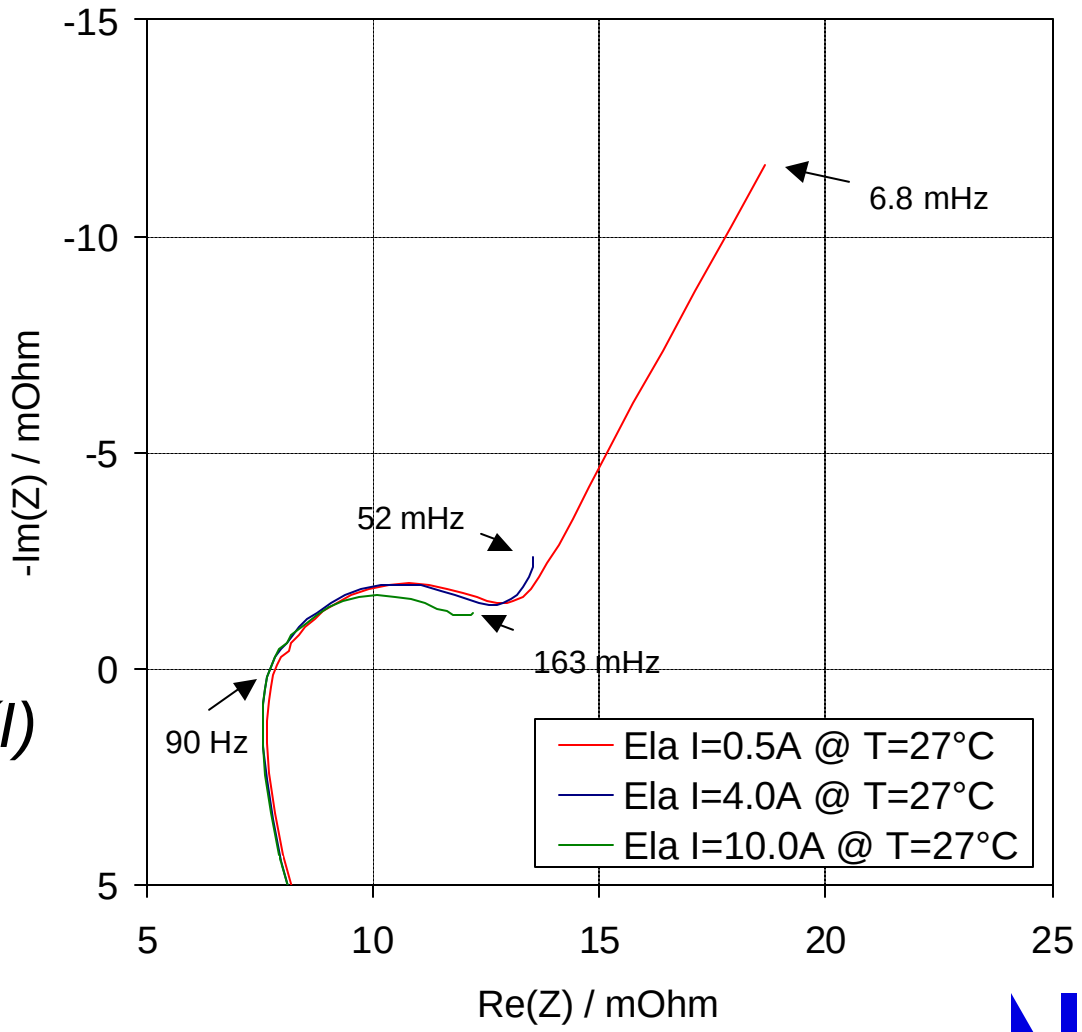
Verifikation - Lithium @ 23°C

- Ströme 100 mA bis 40A
- C_{Nenn} = 7,5 Ah



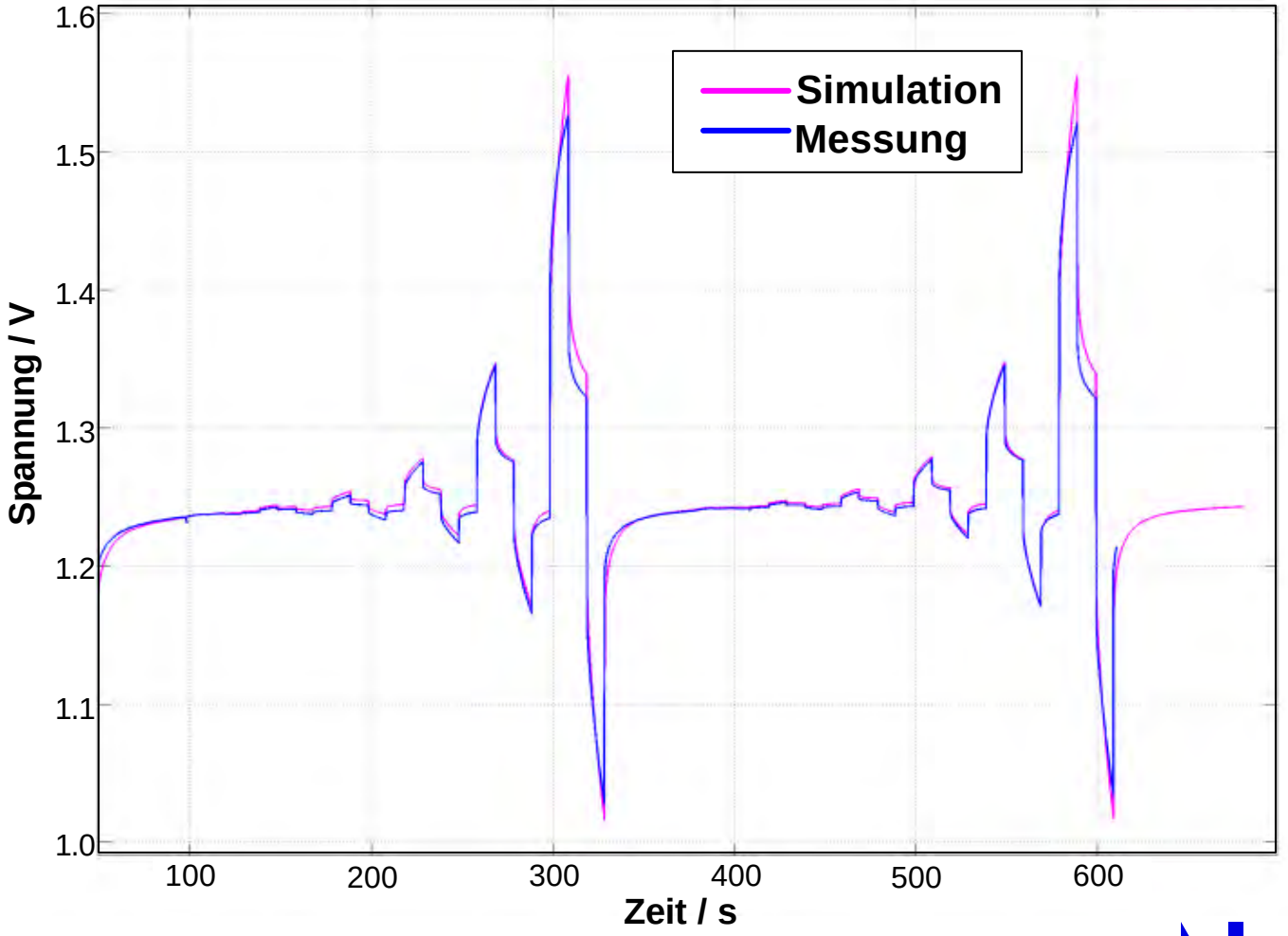
Impedanzspektren von NiMH Batterien

- Entladespektren bei 70% SOC und T=27°C
- Hochfrequenter Bereich: $R_i \neq f(I)$, $L \neq f(I)$
- "erster" Halbkreis:
  -> ZARC - Element = *f(I)*
- tiefe Frequenzen:
  -> Diffusionselement Z_p= *f(I)*



Verifikation NiMH

- Alternierende Ströme bis ±100A bei 20% SOC und Raumtemperatur



Zusammenfassung

- Impedanzspektroskopie ist ein hilfreiches Werkzeug für die Batteriemodellierung
- Einfache Implementierung und Parametrierung von elektrischen Ersatzschaltbildern
- Übertragbar auf alle Arten von Energiespeichern, außerdem: Brennstoffzellen
- Ergänzende Zeitbereichsmessungen nötig, um den Gültigkeitsbereich zu erweitern (Ruhespannung, Elektrolyttransport, Gasung, ...)

Methodik der impedanzbasierten Modellierung für Batterien

ASIM Fachgruppentreffen, München, 20./21. 2. 2006

Dipl.-Ing. Julia Schiffer

Fachgruppe Elektrochemische Energiewandlung und Speichersystemtechnik
Prof. Dr. Dirk Uwe Sauer

Institut für Stromrichtertechnik und Elektrische Antriebe
Rheinisch-Westfälische Technische Hochschule Aachen
Univ.-Prof. Dr. ir. R. W. De Doncker

Implementierung eines Stoffmodells für die Modellierung von Brennstoffzellen in Modelica und Simulink: ein Vergleich

Jörg Ungethüm <joerg.ungethuem@dlr.de>

Institut für Fahrzeugkonzepte

Deutsches Zentrum für Luft- und Raumfahrt e.V. in der Helmholtz-Gemeinschaft

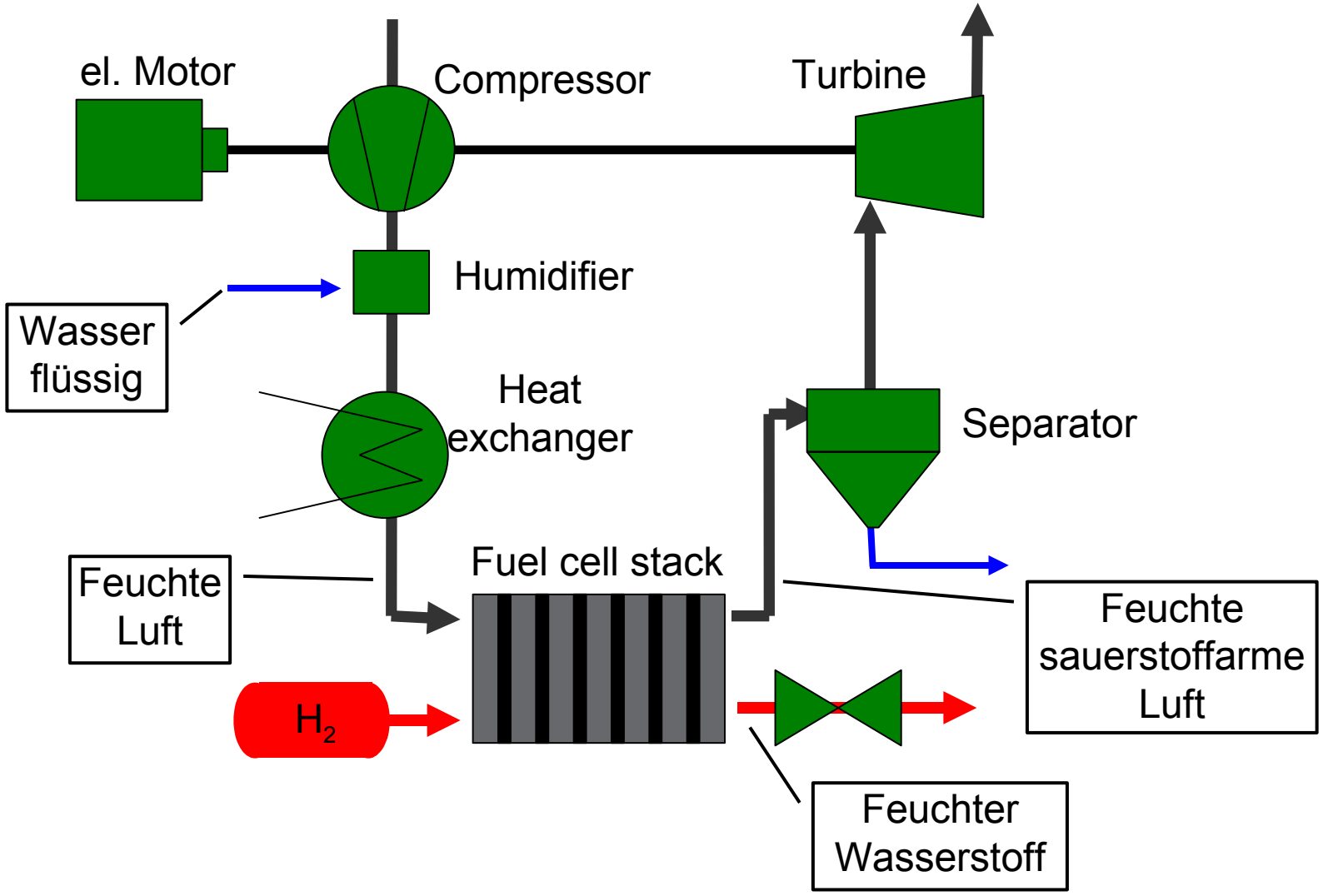
ASIM Fachgruppentreffen STS München, 20./21.2.2006

Implementierung eines Stoffmodells für die Modellierung von Brennstoffzellen in Modelica und Simulink: ein Vergleich

Überblick

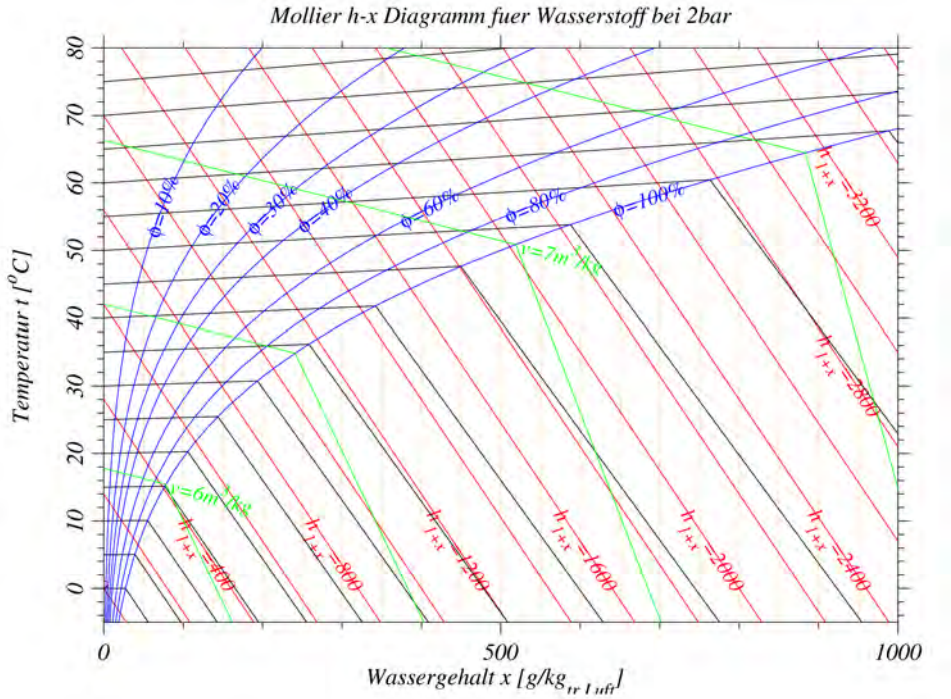
- Problemstellung: Was soll modelliert werden?
- Physikalische Beschreibung des Stoffmodells
- Implementierung in Modelica
- Anwendungsbeispiel Brennstoffzellensystem
- Implementierung in MATLAB/Simulink
- Anwendungsbeispiel Druckausgleich, Vergleich mit Modelica
- Zusammenfassung und Ausblick

PEM-Brennstoffzellensystem



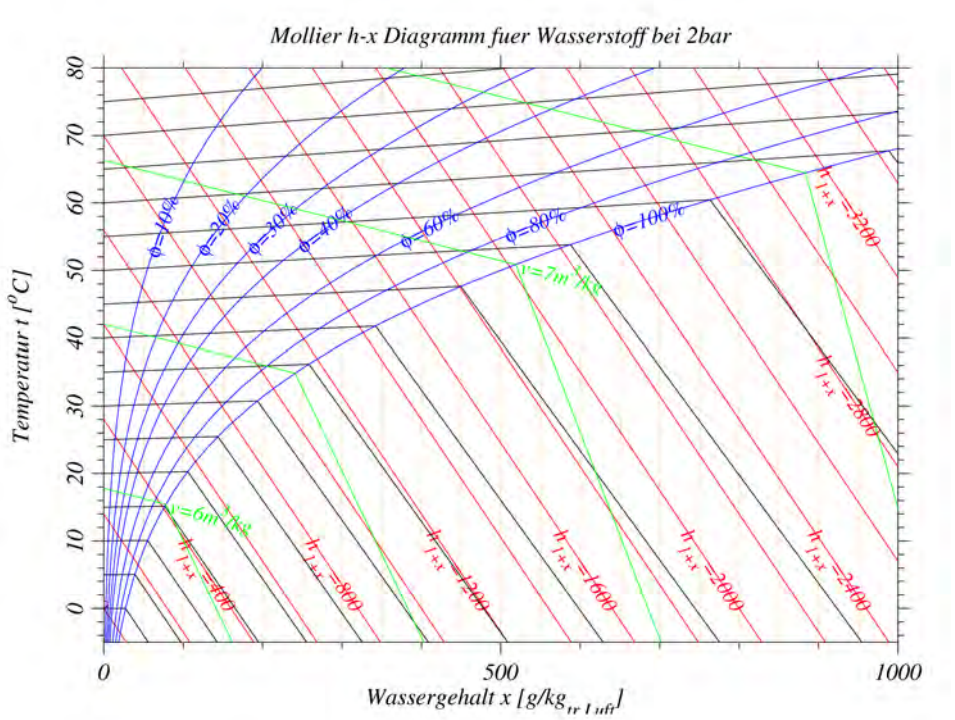
Anforderungen an das Stoffmodell

- Temperaturbereich: -25°C – 150°C
- Druckbereich: 1 bar – 4 bar
- Variable Komposition:
  - Kathode: Sauerstoff, Stickstoff, Wasser
  - Anode: Wasserstoff, Wasser
- Phase



Formulierung des Stoffmodells

- Homogenes Fluid (Nebel, keine stehende Flüssigkeit)
- Ideales Gas + ideale Flüssigkeit
- Volumen der Flüssigkeit wird vernachlässigt
- Wärmekapazität der Komponenten ist konstant
- Adaption des „feuchte Luft“ – Modells aus der Klimatechnik



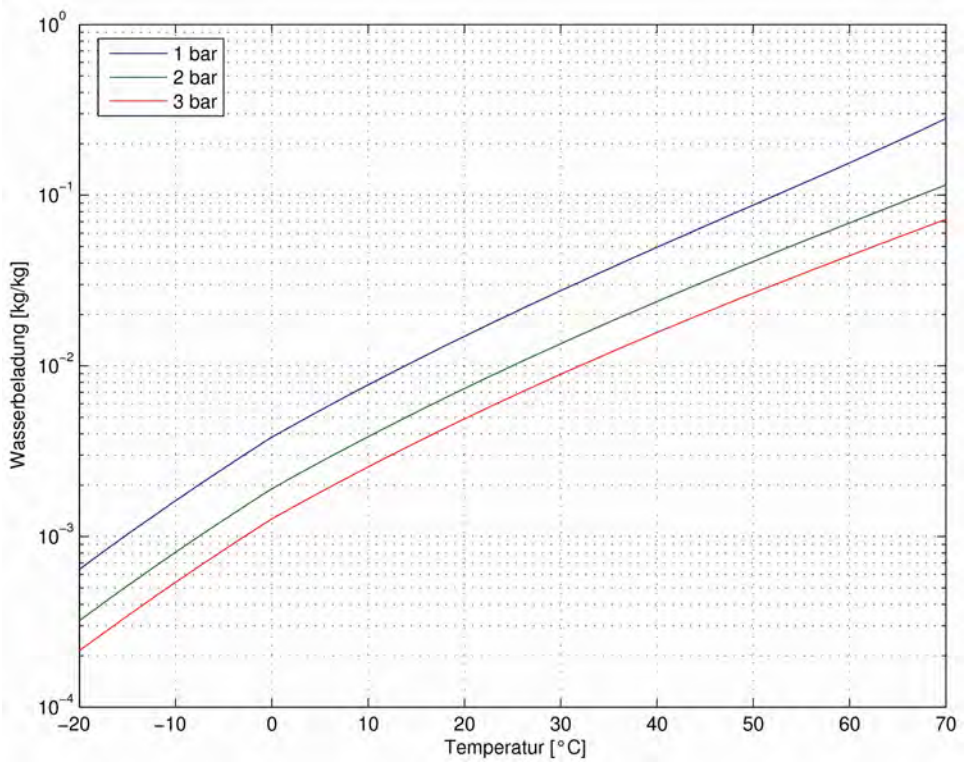
Thermische Zustandsgleichung

$$v_{1+x} = (R_{dg} + x_{vap} R_w) \frac{T}{p}$$

$$x = \frac{m_w}{m_{dg}}$$

$$x_{sat} = x_{sat}(t, p)$$

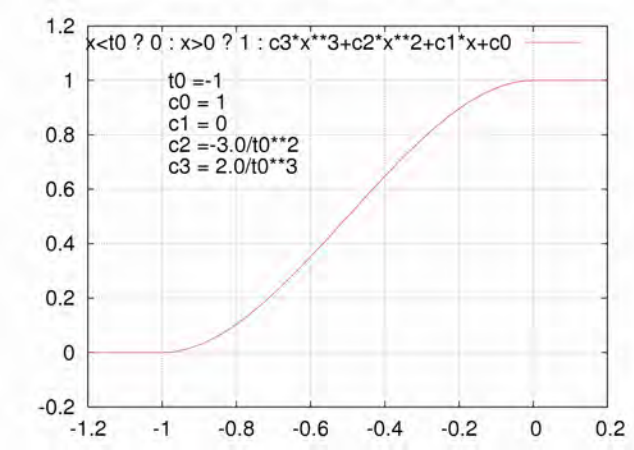
$$x_{vap} = \min(x, x_{sat})$$



Kalorische Zustandsgleichung

$$x_{liq} = \xi(t)\left(x - x_{vap}\right)$$

$$x_{ice} = x - x_{vap} - x_{liq}$$



$$h_{1+x} = h_{dg} + x_{vap}h_{vap} + x_{liq}h_{liq} + x_{ice}h_{ice}$$

$$h_{dg} = c_{p,dg}t$$

$$h_{vap} = c_{p,vap}t + r_0$$

$$h_{liq} = c_{liq}t$$

$$h_{ice} = c_{ice}t - r_e$$

$$c_{p,dg} = \sum_{i=2}^{n} x_i c_{p,i}$$

$$c_{p,i} = const$$

$$c_{liq} = const$$

$$c_{ice} = const$$

Zustandsgleichungen

Thermische Zustandsgleichung

$$v_{1+x}(t, p, \vec{x}) = (R_{dg} + x_{vap} R_w) \frac{(t + 273.15)}{p}$$

Kalorische Zustandsgleichung

$$h_{1+x}(t, p, \vec{x}) = h_{dg} + x_{vap} h_{vap} + x_{liq} h_{liq} + x_{ice} h_{ice}$$

$$h_{dg} = c_{p,dg} t$$

$$h_{vap} = c_{p,vap} t + r_0$$

$$h_{liq} = c_{liq} t$$

$$h_{ice} = c_{ice} t - r_e$$

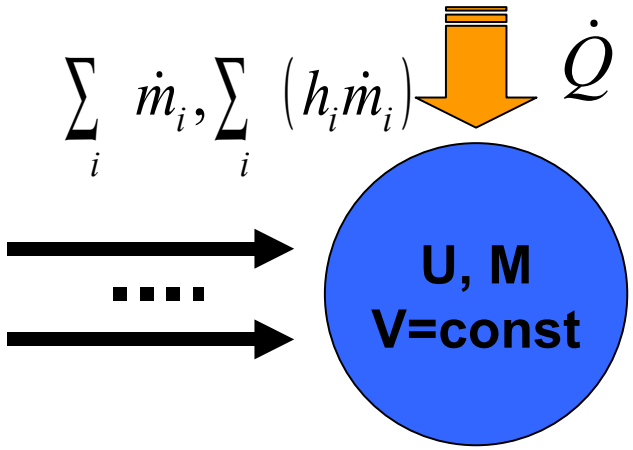
$$c_{p,dg} = \sum_{i=2}^{n} x_i c_{p,i}$$

$$c_{p,i} = const$$

$$c_{liq} = const$$

$$c_{ice} = const$$

Bilanzgleichungen in Grundform



$$\text{Massenbilanz} \quad \frac{dM}{dt} = \sum_i \dot{m}_i$$

$$\text{Energiebilanz} \quad \frac{dU}{dt} = \sum_i (h_i \dot{m}_i) + \dot{Q}$$

Bilanzgleichungen enthalten keine Stoffwerte

$$\frac{d\rho}{dt} = \frac{1}{V} \sum \dot{m}_i$$

$$\frac{du}{dt} = \frac{1}{M} \left(\sum_i (h_i \dot{m}_i) + \dot{Q} \right) - \frac{1}{\rho} \frac{d\rho}{dt}$$

Enthalpie: h = u + p*v
Wärmestrom = f (Temperatur)
Massenstrom = f (Druck)

Stoffwerte müssen i.d.R. iterativ berechnet werden

Transformation der Bilanzgleichungen

z.B. Dichte und innere Energie → Druck und Temperatur

$$\rho = \rho(T,p) \Rightarrow \left(\frac{\partial \rho}{\partial p}\right)_T \frac{dp}{dt} = \frac{d\rho}{dt} - \left(\frac{\partial \rho}{\partial T}\right)_p \frac{dT}{dt}$$

$$u = u(T,p) \Rightarrow \left(\frac{\partial u}{\partial T}\right)_p \frac{dT}{dt} = \frac{du}{dt} - \left(\frac{\partial u}{\partial p}\right)_T \frac{dp}{dt}$$

Differentiale müssen im Stoffmodell berechnet werden!

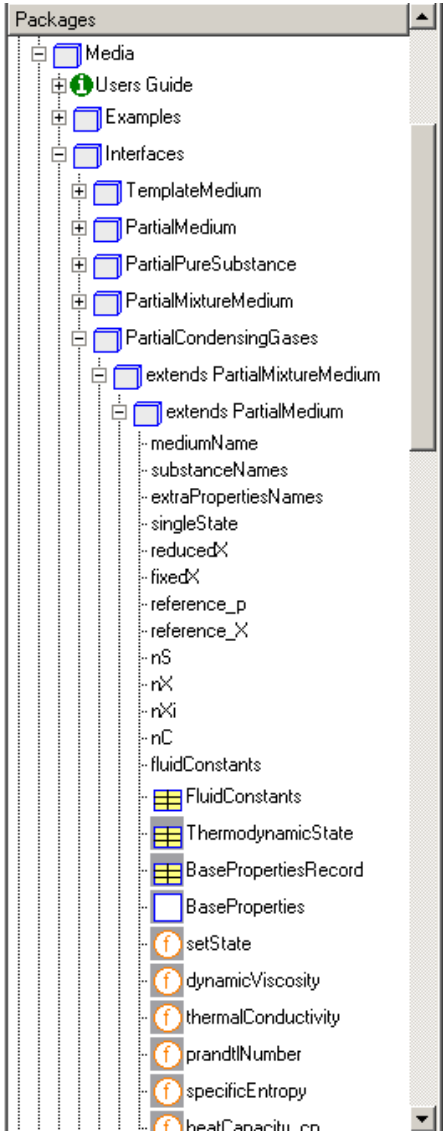
- Berechnung durch formales Differenzieren der Zustandsgleichungen
- Berechnung durch automatisches symbolisches Differenzieren

Die Basisbibliothek Modelica.Media

- Modelica.Media enthält ca. 1250 Stoffmodelle
- Davon 1241 Modelle idealer Gase (reine Gase)
- Mehrere Wasser-Modell nach IAPWS97 (p/h, p/T, einzelne Regionen)
- Ein Modell für „feuchte Luft“
- Mehrere Modelle für inkompressible Flüssigkeiten (u. a. Wasser)
- Modelle für Mischungen idealer Gase (Verbrennungsabgas)

- Keine Realgasmodelle
- Keine Modelle für Kältemittel
- Idealgasmodelle häufig zu komplex T = 200 … 6000K

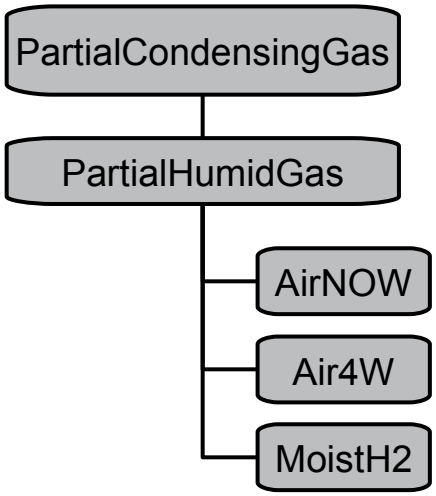
Architektur Modelica.Media



- Jedes Stoffmodell ist ein replaceable package
- Jedes Stoffmodell enthält die Größen
  p, T, ρ, u, h, X_1,…X_n
- Davon sind 2+(n-1) Zustandsgrößen
- Für die anderen 4 enthält das Stoffmodell Gleichungen
- Alle anderen Größen (s, c_p, a, Transportgrößen) sind Funktionen der Zustandsgrößen

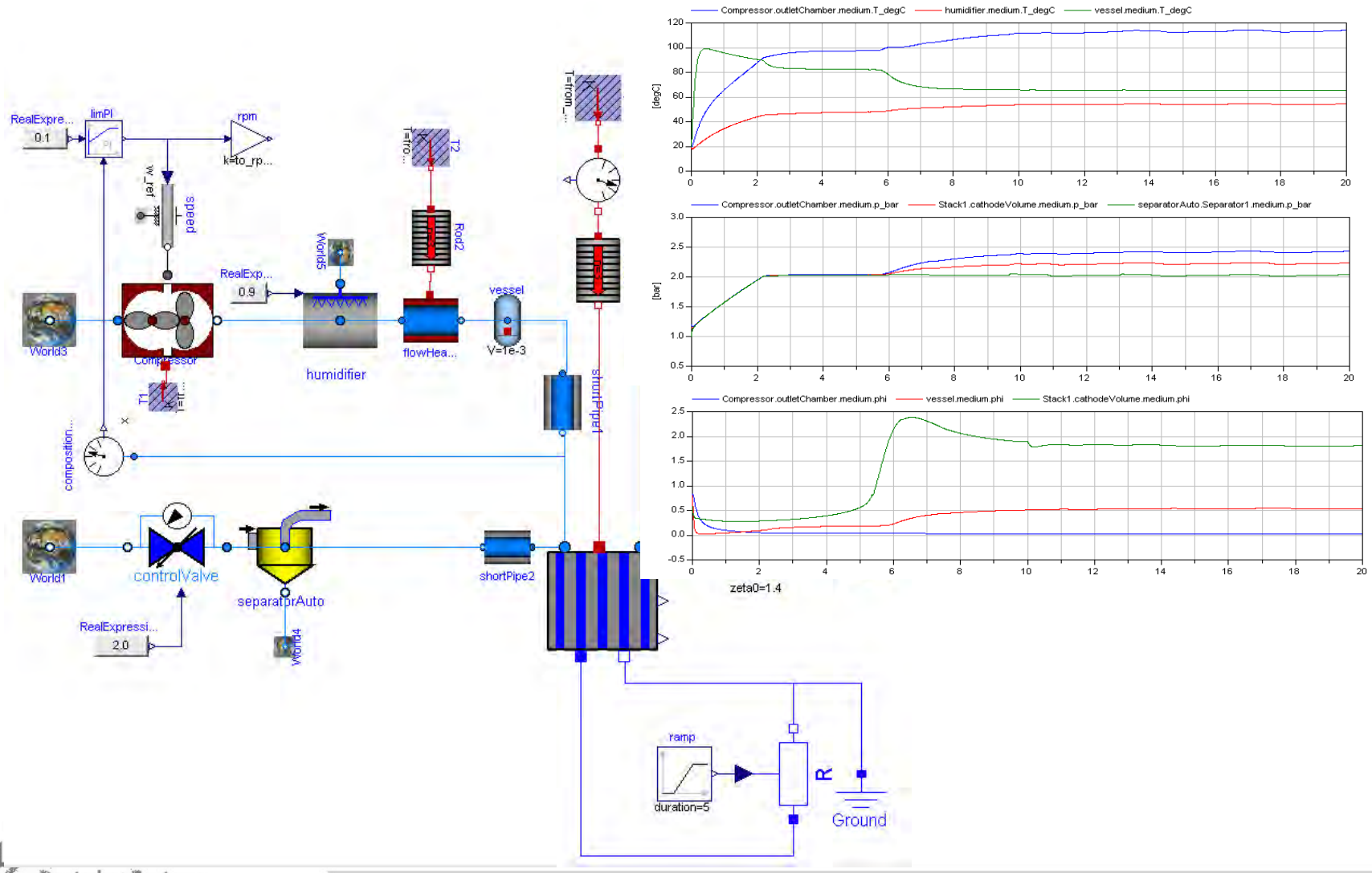
- Im Stoffmodell werden die geeigneten Zustandsgrößen gewählt!
- Notwendige partielle Ableitungen werden automatisch generiert!

Implementierung des Stoffmodells in Modelica




```
partial package PartialHumidGas
extends
Modelica.Media.Interfaces.PartialCondensingGases;
  ...
redeclare replaceable model extends
BaseProperties(
     T(stateSelect=StateSelect.prefer),
     p(stateSelect=StateSelect.prefer),
     Xi(stateSelect=StateSelect.prefer))

 h = h_pTXi(p, T, Xi);
 d = densityOfGasPhase(state);
 u = h - p/d;
  …
end BaseProperties;

redeclare function extends saturationPressure …
redeclare function extends heatCapacity_cp …
…
```

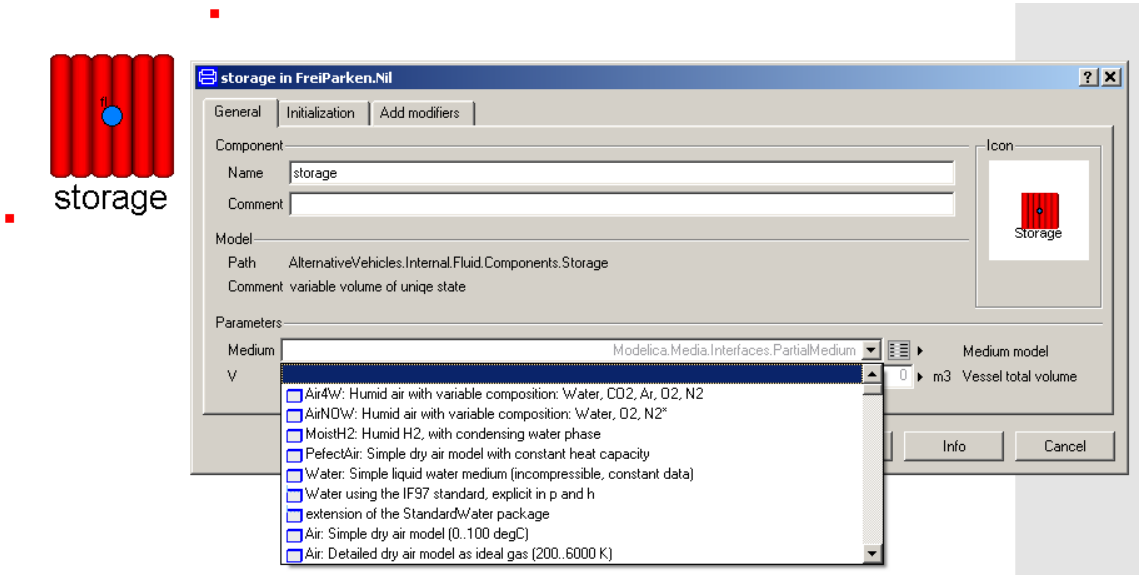
Anwendungsbeispiel Brennstoffzellensystem



Vorteile der Modelica-Implementierung

- Weitere Modelle für feuchte Gase lassen sich durch Vererbung sehr leicht erzeugen: z.B. Rauchgas
- Stoffmodell lässt sich durch weitere Funktionen ausbauen z.B. Wärmeleitfähigkeit, Zähigkeit, Prandtlzahl

- **Komponentenmodelle sind unabhängig vom Stoffmodell z.B. Kontrollvolumen, Drossel, Ventil, …**



Probleme mit der Modelica-Implementierung

- Implementierung ist sehr aufwändig
- Viele Formalitäten müssen eingehalten werden
- Verbindung verschiedener Stoffmodelle erfordert spezielle Komponenten

- Symbolische Differenzierung von Funktionen schlägt fehl
  Problem: h = h_pTXi(p, T, Xi)
- stateSelect=StateSelect.prefer wird ignoriert
  → Es werden andere Zustände gewählt (i.A. M und U)
  → Erzeugung impliziter Gleichungen

Implementierung in MATLAB/Simulink

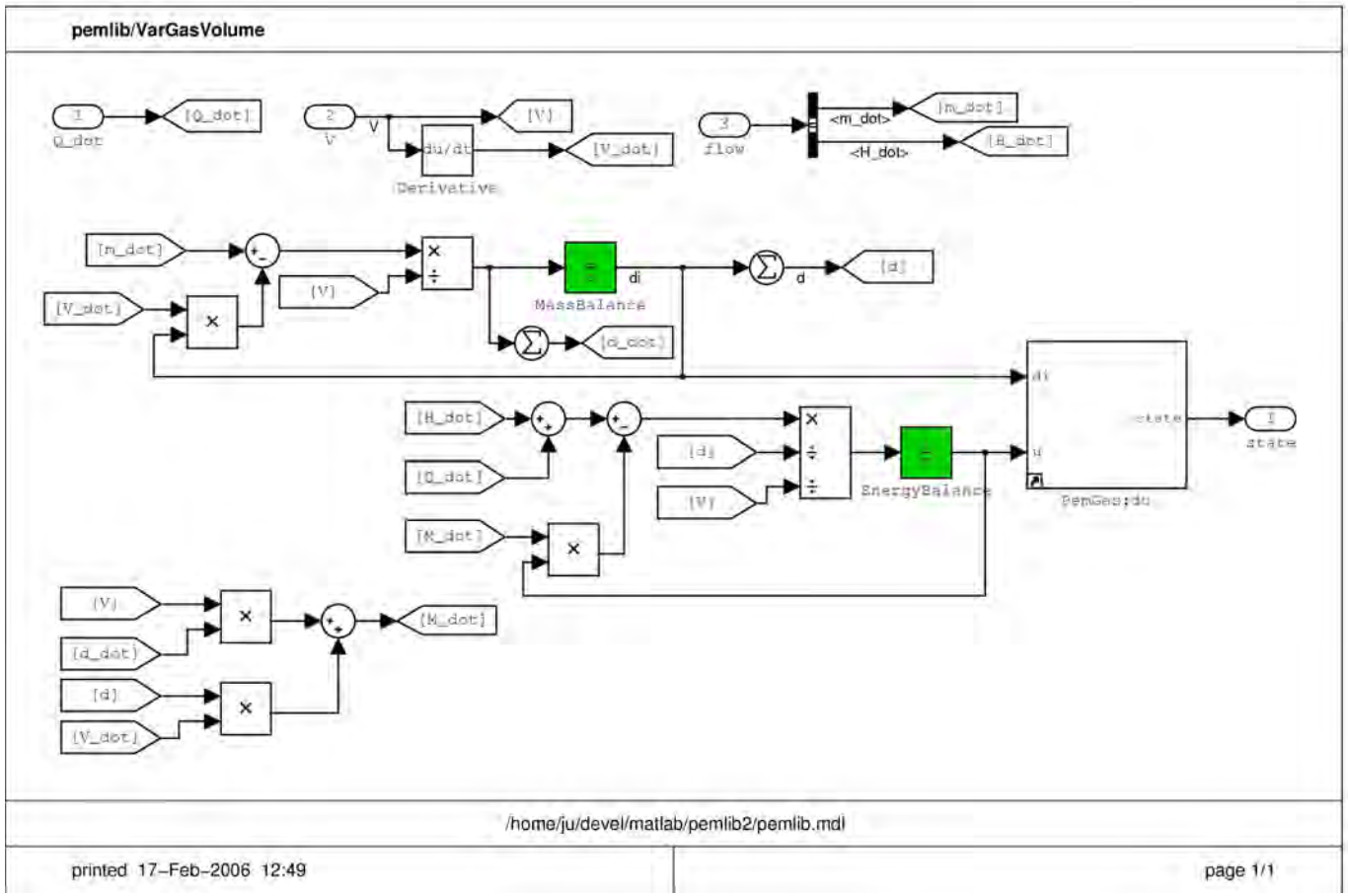
- Bilanzgleichungen werden in der einfachsten Form implementiert

$$\frac{d\rho}{dt} = \frac{1}{V}\sum \dot{m}_i \qquad \frac{du}{dt} = \frac{1}{M}\left(\sum_i \left(h_i \dot{m}_i\right) + \dot{Q}\right) - \frac{1}{\rho}\frac{d\rho}{dt}$$

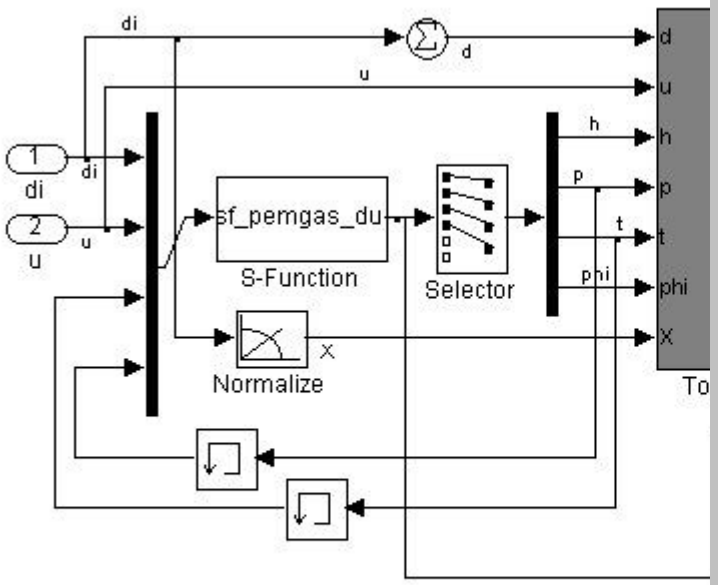
- Dichte und innere Energie sind Zustandsgrößen
- Stoffwertroutinen sind implizite Gleichungen!
- Auswertung erfordert Lösung von 2 verkoppelten impliziten Gleichungen
- Stoffwertroutinen sind fest mit den Komponentenmodellen verbunden

- Implementierung als S-Function mit eigenem Sublöser
- Verwendung der Primitivform des gedämpften Newton-Verfahrens
- Konvergenz i.d.R. in weniger als 5 Iterationen

Implementierung des Kontrollvolumens

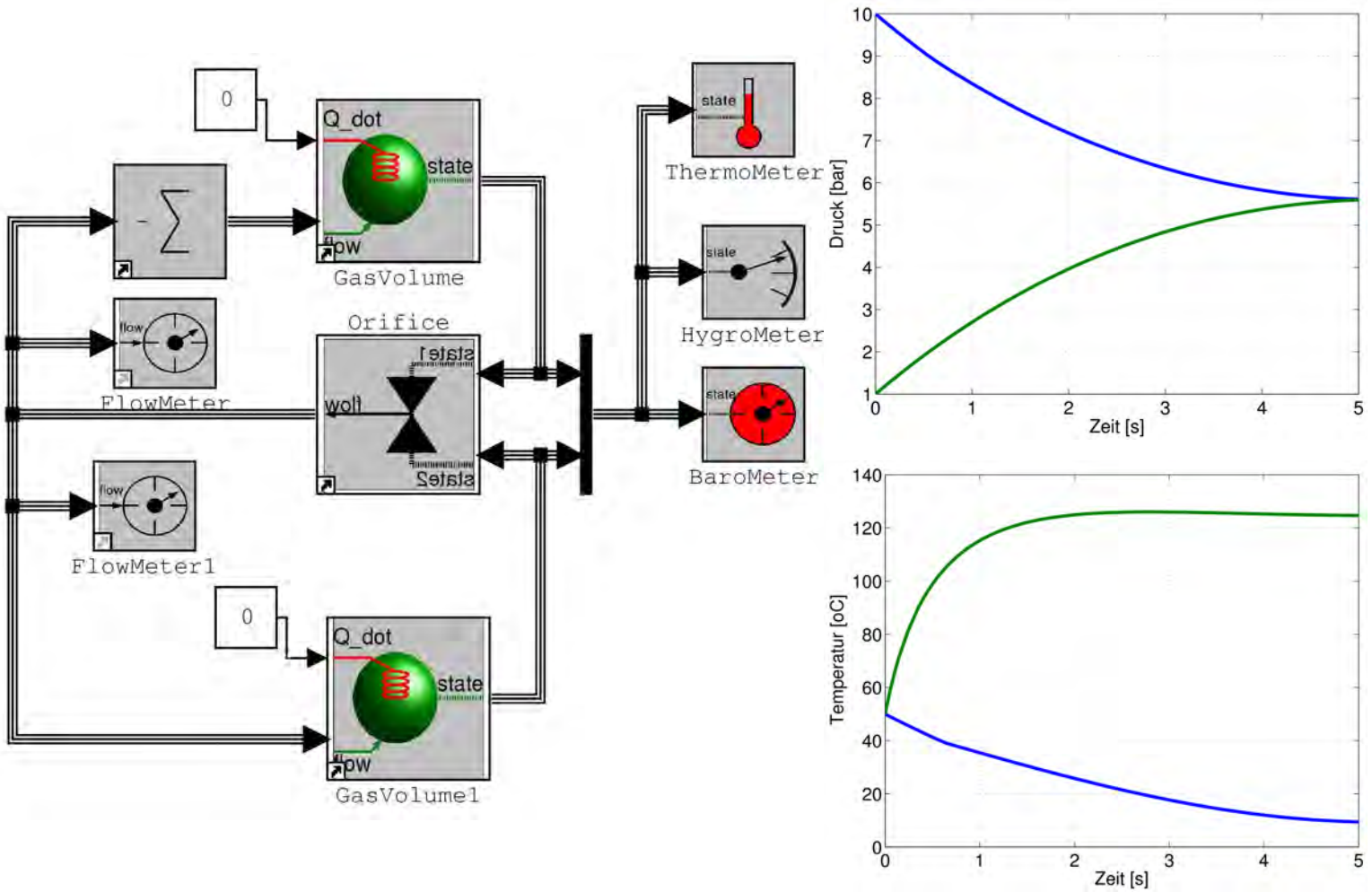


Implementierung der Stoffwertfunktion



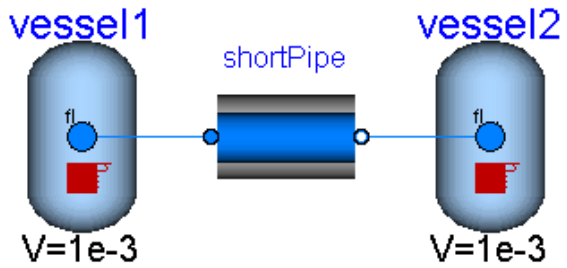

```
      maxit = 100
      kmax = 4
      eps  = 1.0d-12
      do 100 i=1,num_comp
         density_i(i) = u(i)
100   continue
      innerEnergy = u(num_comp+1)
      xx(1) = u(num_comp+2)
      xx(2) = u(num_comp+3)
      call smnewd(func0_pt,n_eq,
     $  maxit,ierr,kmax,itape,iupd,eps,
     $  rnorm2,ff,xx,df,n_eq,iwork,work)

      y(1) = enthalpy
      y(2) = xx(2)
      y(3) = xx(1)
      y(4) = humidity
      y(5) = maxit
      y(6) = ierr
```

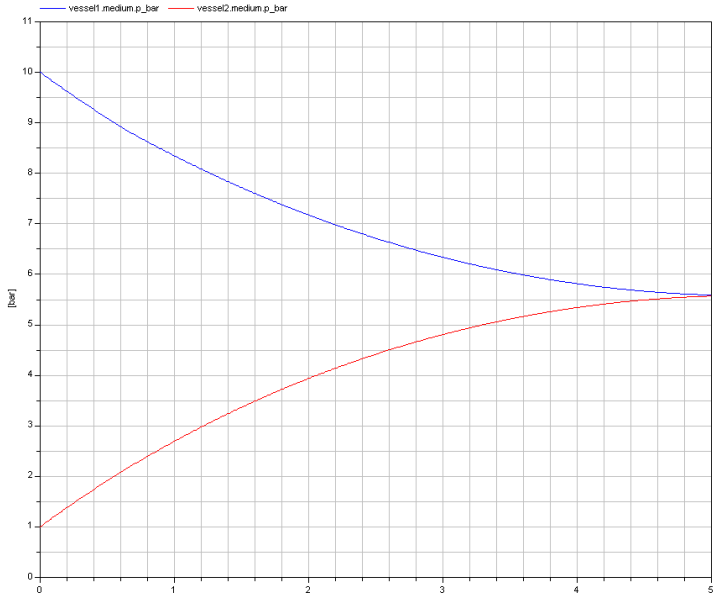
Anwendungsbeispiel Druckausgleich

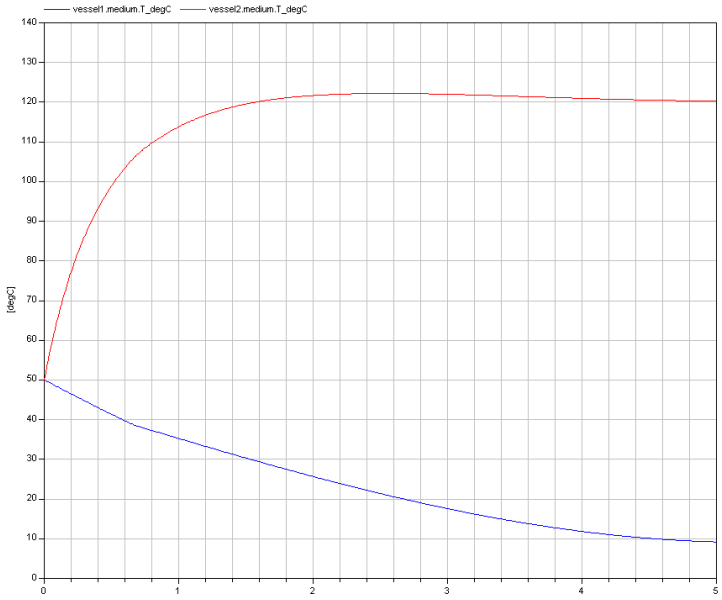


Cpu-time: ca. 0.9 Sekunden (Pentium M 1400MHz)

Zum Vergleich: Druckausgleich in Modelica







Cpu-time: ca. 1.2 Sekunden (Pentium M 1400MHz)

Initialisierung

- Modelica
  - Sehr komplexe Initialisierung durch ‘initial equation‘
  - Initialisierung der Zustandsgrößen oder deren Ableitungen
  - Steady-state Initialisierung (theoretisch) möglich

- MATLAB/Simulink
  - Einfache Initialisierung mit MATLAB Skripten
  - Nur direkte Initialisierung der Zustandsgrößen möglich

Zusammenfassung und Ausblick

- Objektorientierter Ansatz und symbolische Gleichungsbehandlung in Modelica bringt wesentliche Vorteile für die Modellierung
  - Stoffunabhängige Formulierung der Komponenten
- Der aktuelle Stand der symbolischen Gleichungsbehandlung ist verbesserungsfähig, insbesondere bei der Ableitung von Funktionen (Dymola)
- Die Implementierung Modelica erfordert erheblichen Aufwand

- Die Implementierung in MATLAB/Simulink ist relativ einfach
- Der Zugriff auf einen externen Sublöser (gedämpftes Newton-Verfahren) ist unproblematisch
- Modelle sind weniger flexibel verwendbar

Ideales Gas nach Janaf-Tabellen

$$p = \rho R T$$

$$u = c_v(T)(T - T_0)$$

1-D Iteration zur Berechnung der Temperatur

NBS/NRC Steam Tables 1984

$$p = p(T, \rho)$$

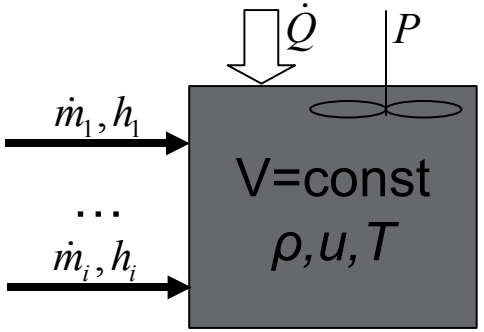
$$u = u(T, \rho)$$

Implementierung „PROST“ (O. Bauer, O. Engel)
-Umkehrfunktionen durch Newton-Iteration
-z.B. p/h, p/s, t/p, …
-Teilweise 2-D Iteration notwendig

Bilanzgleichungen

Massenbilanz

$$V\frac{d\rho}{dt} = \sum_i \dot{m}_i$$



Energiebilanz

$$V\left(u\frac{d\rho}{dt} + \rho\frac{du}{dt}\right) = \dot{Q} + P + \sum_i \dot{m}_i h_i$$

Transformation u □ T

$$\left.\frac{\partial u}{\partial T}\right|_{\rho}\frac{dT}{dt} = \frac{du}{dt} - \left.\frac{\partial u}{\partial \rho}\right|_{T}\frac{d\rho}{dt}$$

Ableitungen 2. Ordnung müssen vom Stoffmodell bereitgestellt werden

Thermodynamische Stoffmodelle

- Thermische Zustandsgleichung

$$v = v(T, p) \quad \text{oder} \quad p = p(T, v)$$

- Kalorische Zustandsgleichung

$$h = h(T, p) \quad \text{oder} \quad u = u(T, v)$$

- Entropie-Zustandsgleichung

$$s = s(T, p) \quad \text{oder} \quad s = s(T, v)$$

Thermodynamische Stoffmodelle

Fundamentalgl.	$f(T,v)$	$g(T,p)$
Thermische ZG	$p(T,v) = -\left(\partial f / \partial v\right)_T$	$v(T,p) = \left(\partial g / \partial p\right)_T$
Kalorische ZG	$u(T,v) = f - T\left(\partial f / \partial T\right)_v$	$h(T,p) = g - T\left(\partial g / \partial T\right)_p$
Entropie ZG	$s(T,v) = -\left(\partial f / \partial T\right)_v$	$s(T,p) = -\left(\partial g / \partial T\right)_p$

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Durchgängige Open-Loop-Testverfahren für Kfz-Elektronik im Labor und Fahrversuch

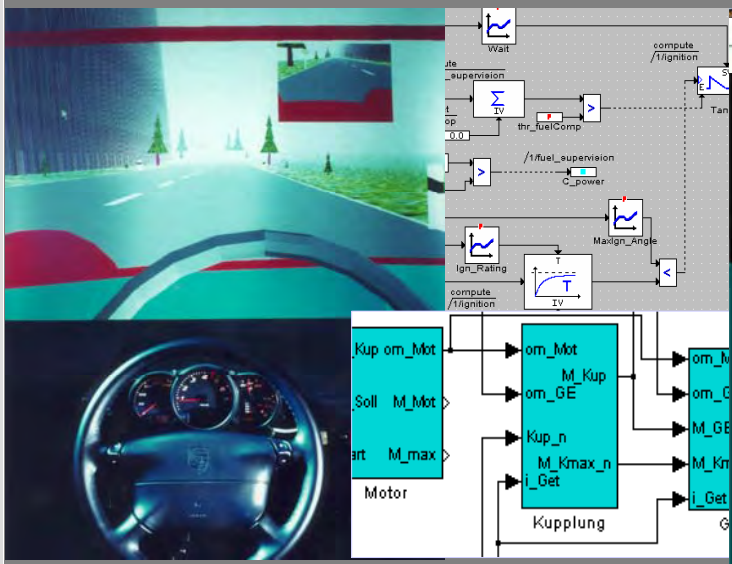
Gerd Baumann
Michael Brost
Hans-Christian Reuss

ASIM 2006
München, 20./21. Februar 2006

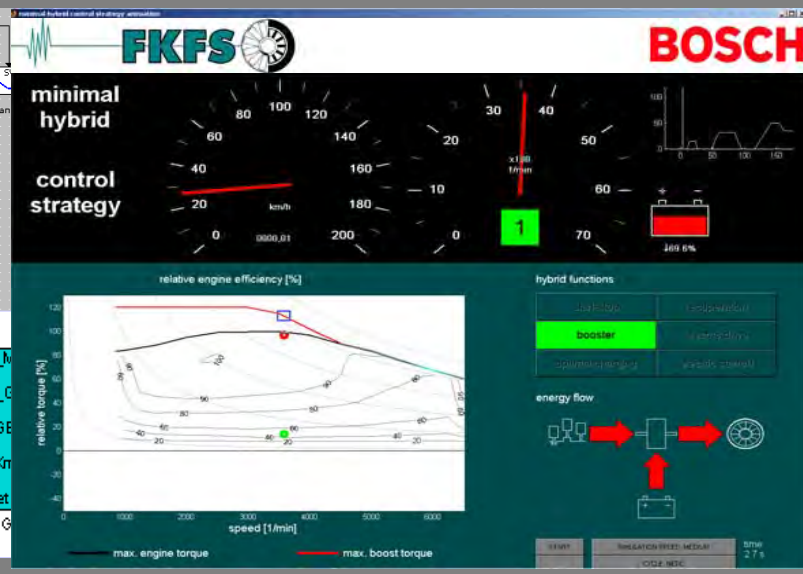
Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

- Stiftung, Gründung 1930
- Forschung und Entwicklung
- Lehre (mit Uni Stuttgart)
- Motorentechnik, Akustik, Fahrdynamik, Aerodynamik, Elektronik und Software
- 140 MA, Umsatz 15 M€ / a

Arbeitsgebiet Kfz-Mechatronik (Beispiele):



ECU-Software-Entwicklung, Test und Diagnose



Antriebs- und Bordnetz-Management, Hybridantrieb

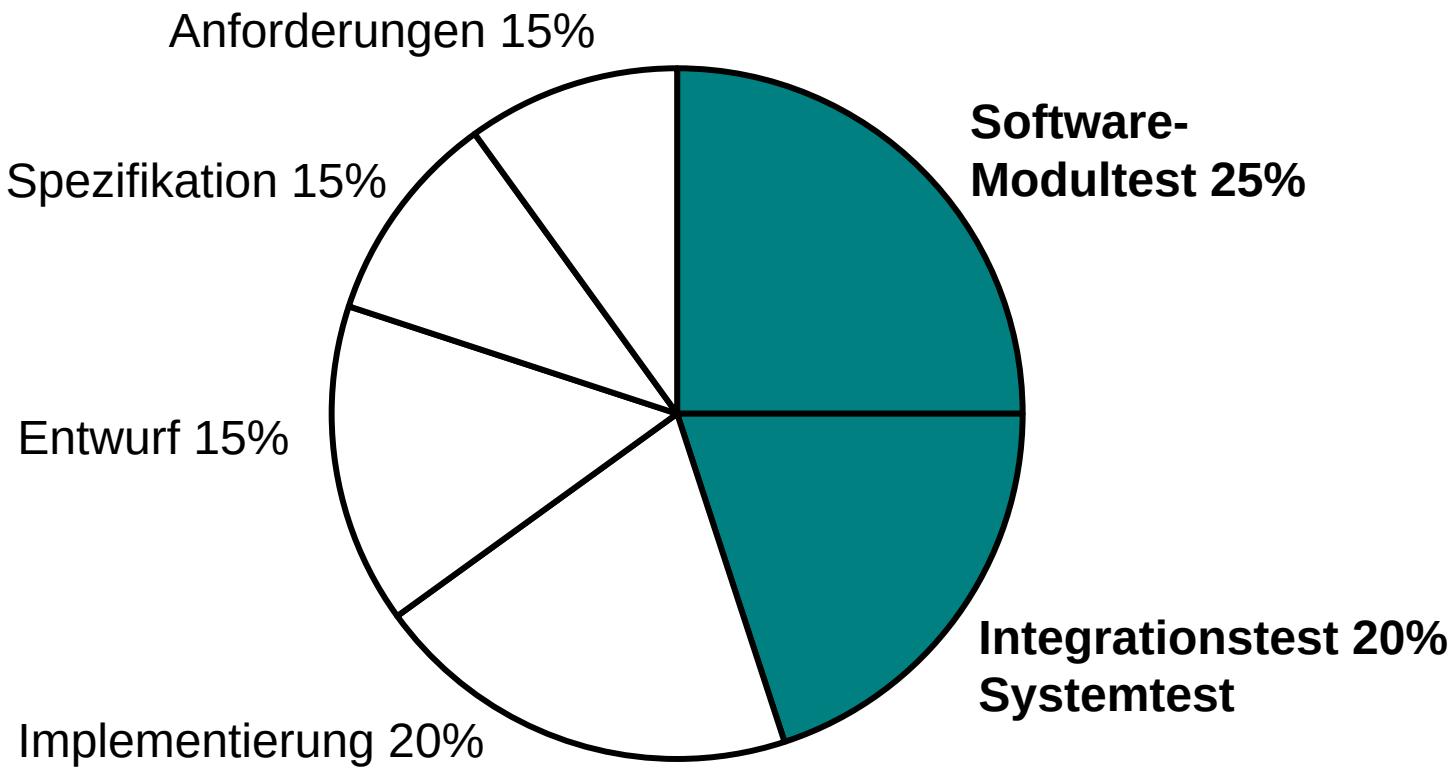
Folie 2

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Kostenverteilung beim Software-Engineering

- SW-Engineering: fast 50% der Kosten entfallen auf den Test
- Steigender SW-Anteil im Fahrzeug
- Testaufwand nähert sich dem des SW-Engineering an

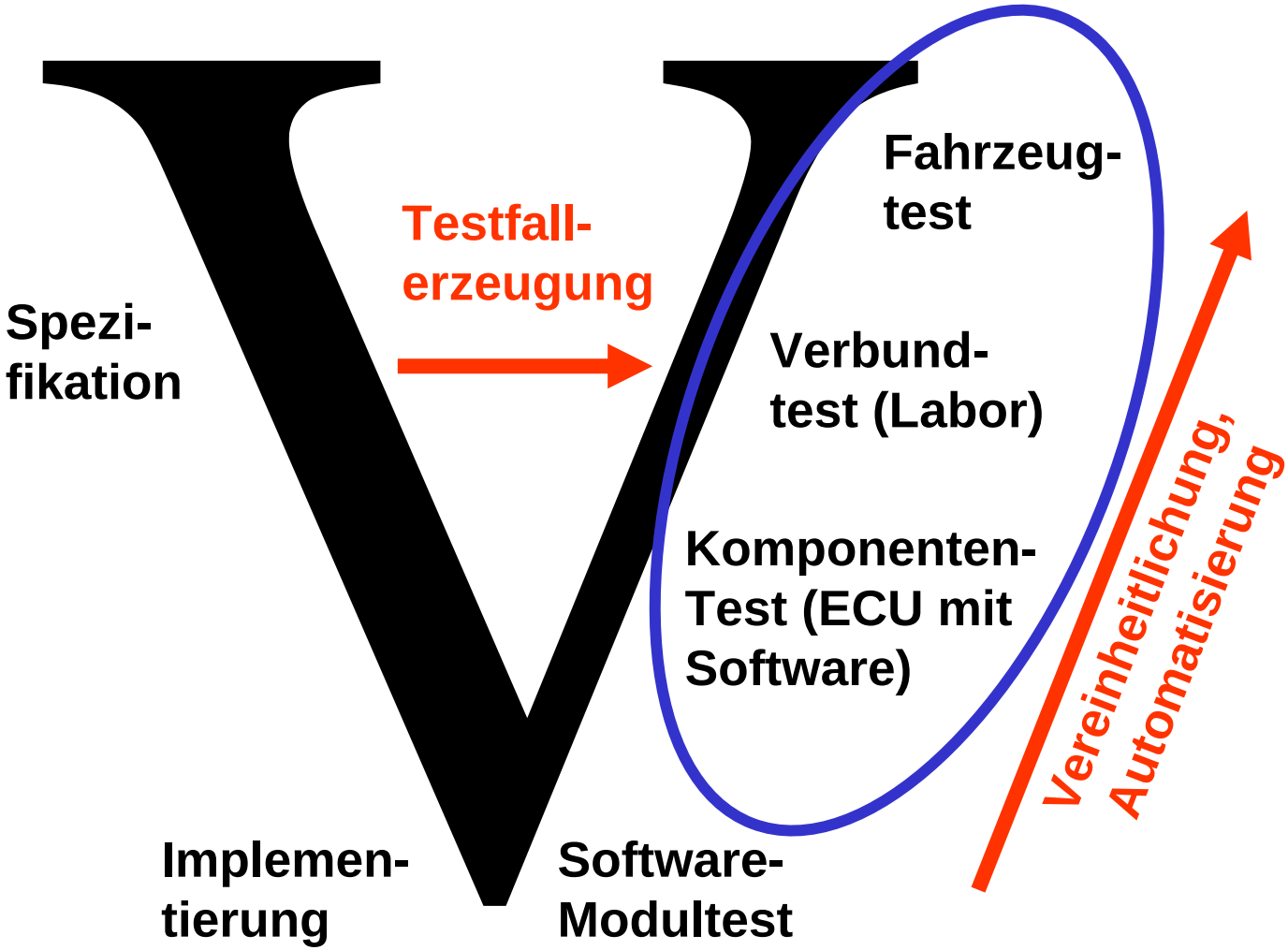


Quellen: [Vigenschow.04] und [Ebert.05]

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Zielsetzung:



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Heutige Situation:

- Uneinheitliche Testverfahren und Testbeschreibungen für Softwaretest, ECU-Test im Labor und Fahrzeugtest.
- Häufig manuelle Testfallerzeugung
- Sehr hoher Hardware- und Software- Aufwand für HIL-Verbundtests

These:
Die Zunahme der Komplexität der Elektronik, Software und Vernetzung erfordert eine Vereinfachung der Testverfahren !

Lösungsansätze:

- Open-Loop-Verfahren und quasistationäre Tests als Ergänzung zum HIL-Verfahren, insbesondere für den Karosserie- und Innenraumbereich
- „Universaltester“: Entkopplung von ECU und Testsystem
- Teilparalleler Testaufbau statt vollparallelem Testaufbau

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

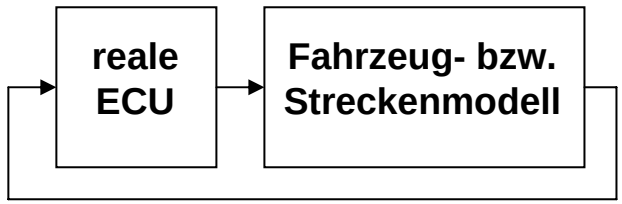
Closed Loop vs. Open Loop Testverfahren

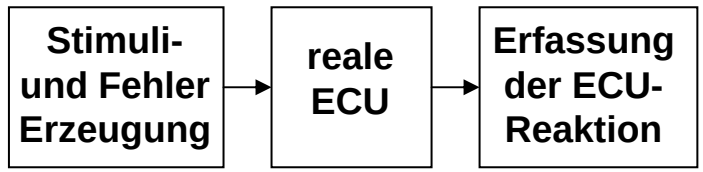
Hardware-in-the-Loop-Simulator

Foto: Fa. dSpace

Ergänzung

Open-Loop-Tester





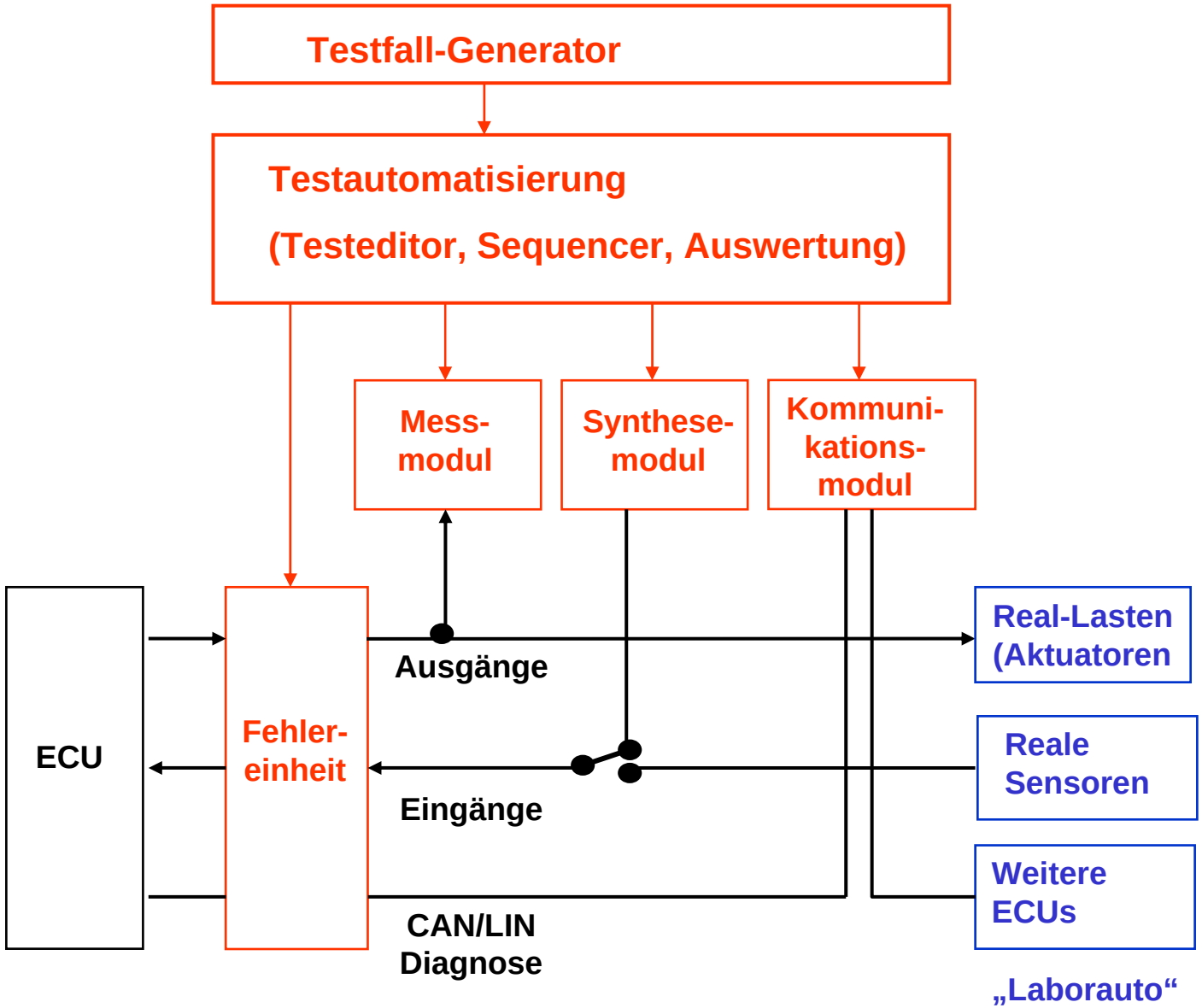
- Closed loop
- Dynamische Betriebszustände
- Fahrzeug- bzw. Strecken-modell notwendig
- Komplexe Signalverläufe i.d.R. manuelle Auswertung
- Typische Anwendungen: Motor, Antriebsstrang
- Einsatz im Labor

- Open loop
- Quasistationäre Betriebszustände
- Direkte Abbildung der Spezifikation in Testsequenzen
- Einfache Signalverläufe -> automatische Auswertung
- Typische Anwendungen: Karosserie und Innenraum
- Einsatz im Labor und Fahrzeug

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Prinzipieller Aufbau eines Open Loop-Testsystems für eine einzelne ECU



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für
Kraftfahrwesen und
Fahrzeugmotoren
Stuttgart

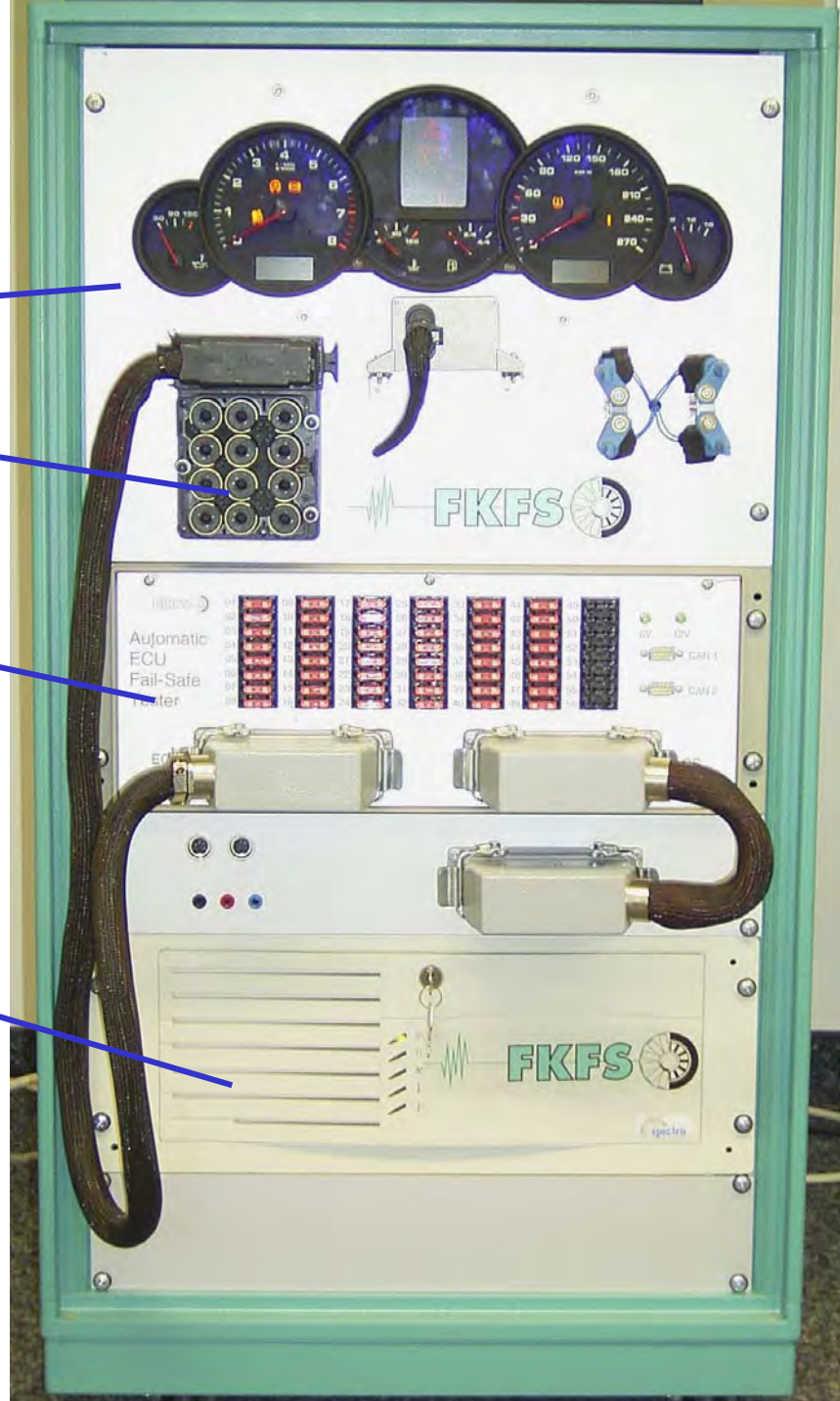
Open Loop-Testsystem P.A.T.E

„Laborauto“

ECU (Prüfling)

Fehlereinheit,
Mess/Synthesemodul,
Kommunik. modul

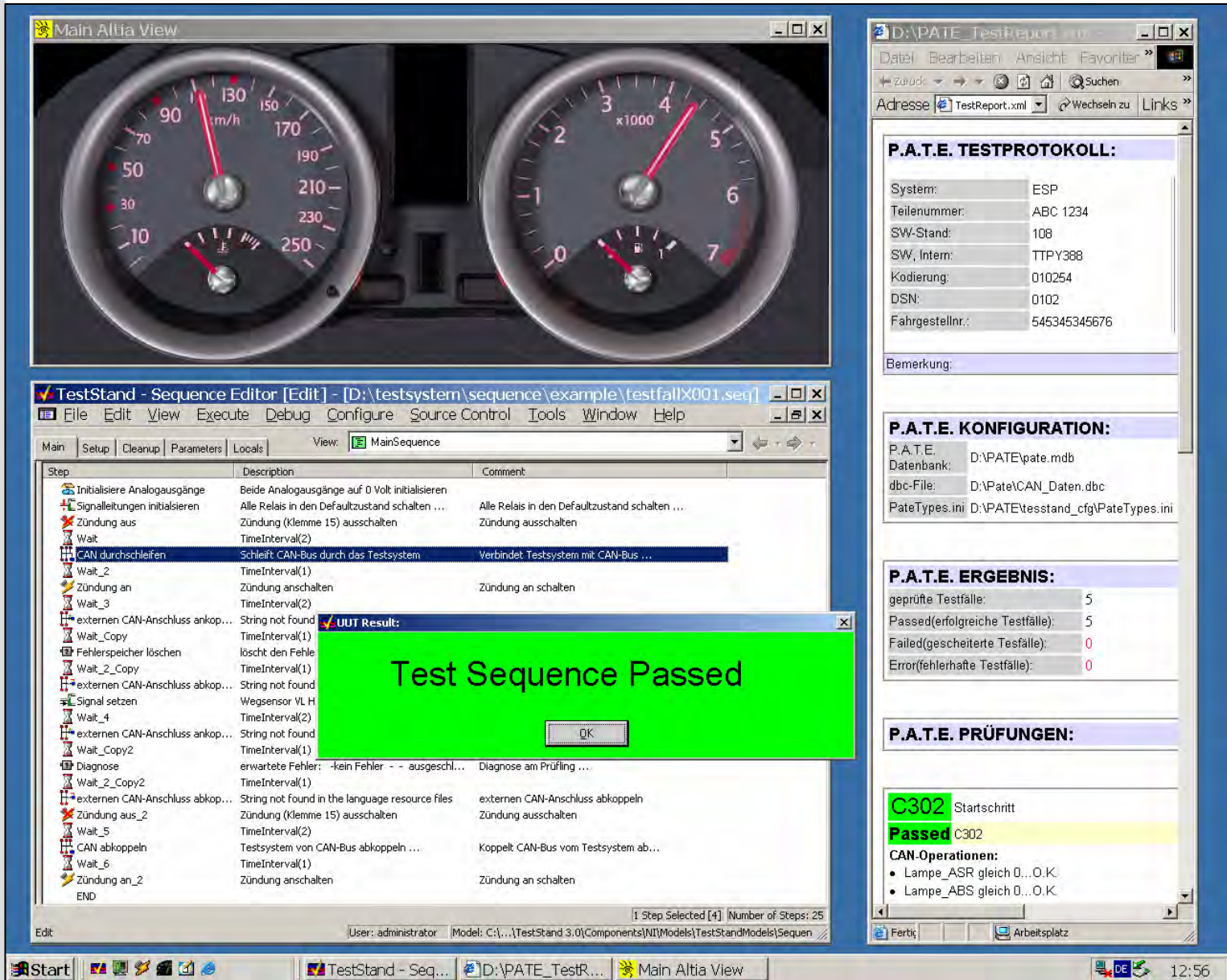
Steuerrechner
Testautomatierung



G. Baumann
M. Brost
H.-C. Reuss

P.A.T.E Software

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart



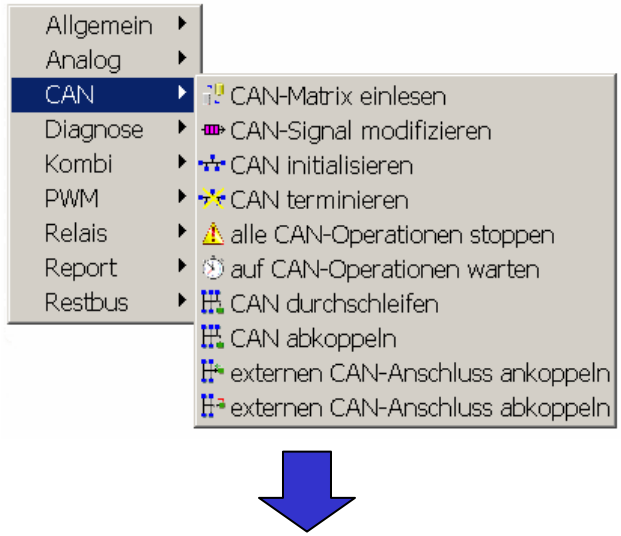
Folie 9
G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

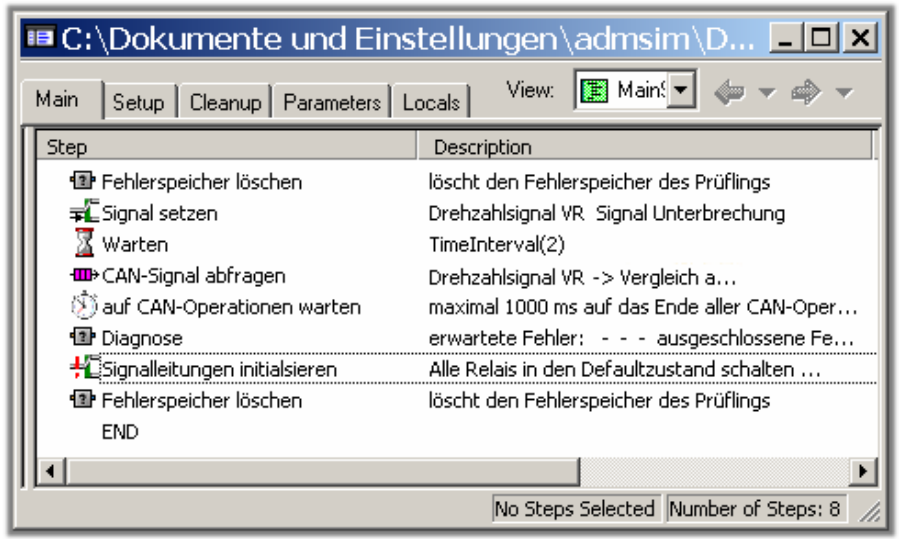
P.A.T.E Software

- Vollständig menügeführte Testerstellung ohne Skriptsprache
- Mehrstufige Testhierarchie

Auswahlmenü für Testschritte



Darstellung eines Testfalls im Editor

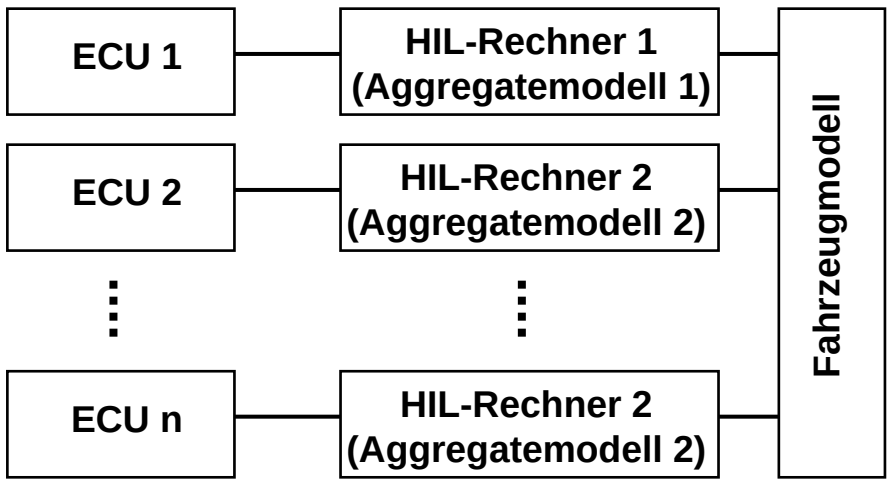


G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Ansätze beim Verbundtest

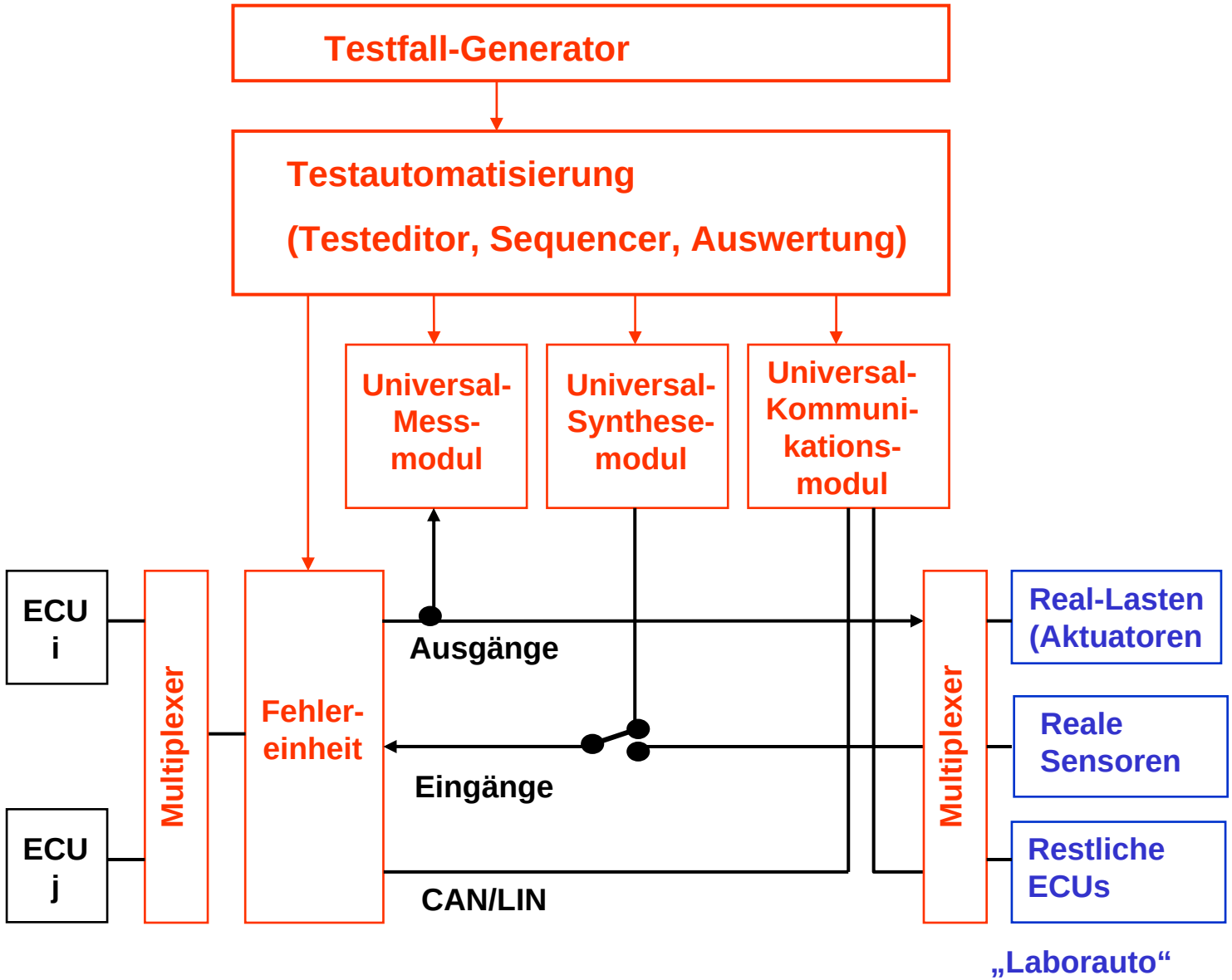
- Heute verbreitet: häufig „massiv paralleler Verbundtest“ durch Parallelbetrieb aller ECUs an einem „Vernetzungs-HIL“, i.d.R. mit vollständiger Substitution der Sensoren
- Vorteil: theoretisch 100 % Nachbildung des realen Systems möglich
- Nachteil: Hoher Investitions- und Pflegeaufwand



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Prinzipieller Aufbau eines Open Loop-Testsystems für einen Steuergeräte-Verbund (vereinfacht)

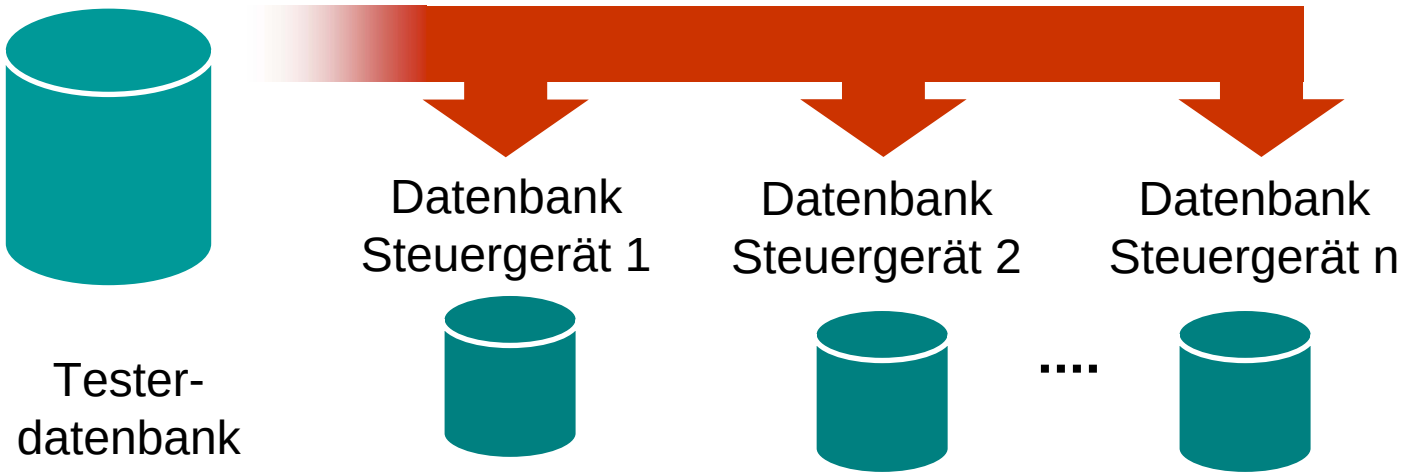


G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Universaltester für Verbundsyteme

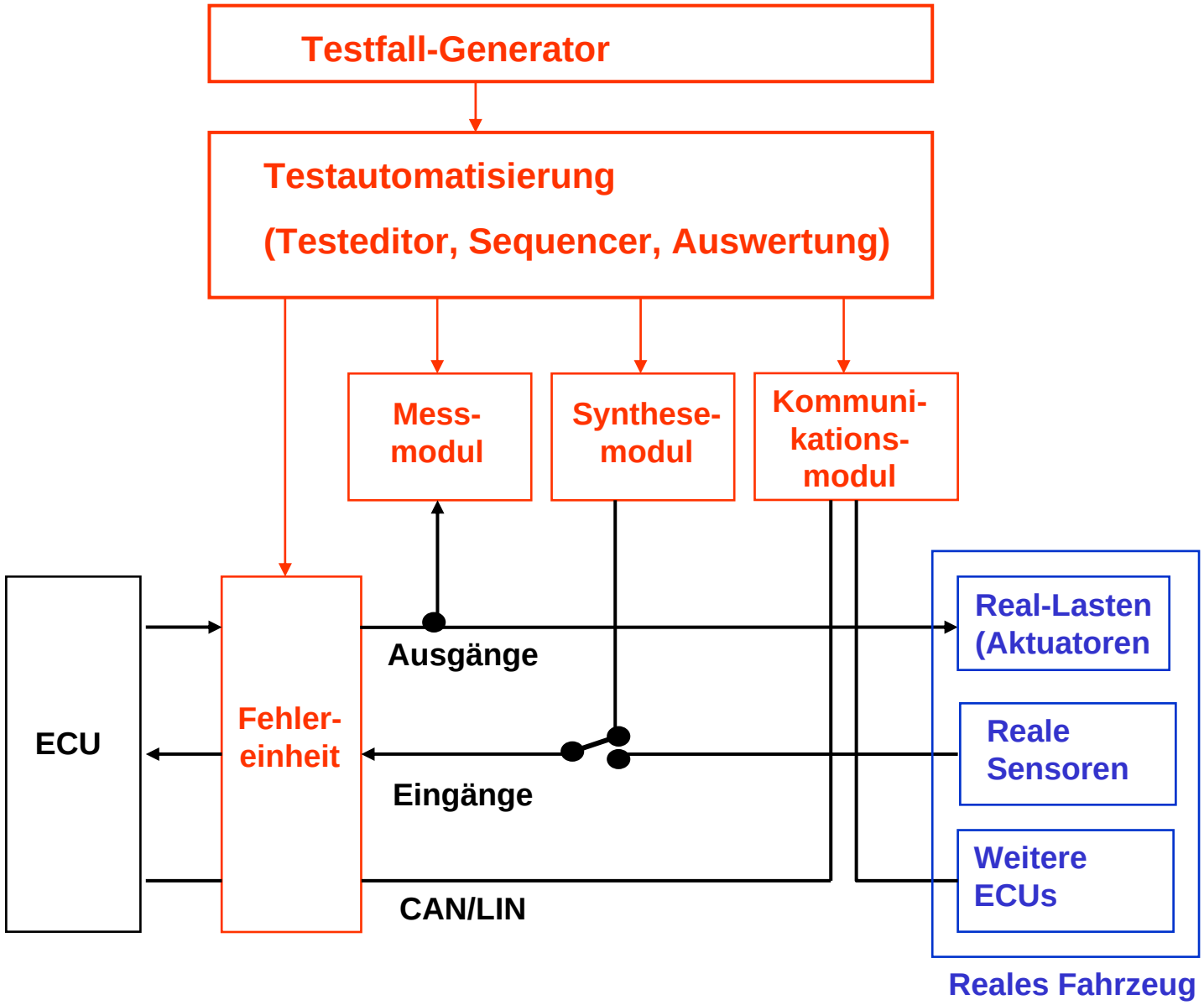
- Vollständig datenbankgestützte Beschreibung des Testaufbaus
- Tester-Datenbank beschreibt das Testsystem, z.B. Anzahl und Typ der verfügbaren CAN-Kanäle, Messkanäle, etc.
- Steuergeräte-Datenbanken beschreibt jeweils ein Steuergerät, z.B. Pinbelegung, verwendete Diagnose-Codes usw.
- Weitere Datenbanken für systemweite Daten (CAN-Kommunikationsmatrix, verwendetes Diagnose-Protokoll, etc.)
- Beschreibung eines konkreten Testaufbaus (z.B. Verbindung eines physikalischen ECU-Pins mit einem physikalischen Tester-Kanal) erfolgt durch Verknüpfung der Datenbanken



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Testsystem-Einsatz im Versuchsfahrzeug



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

P.A.T.E.

mobile

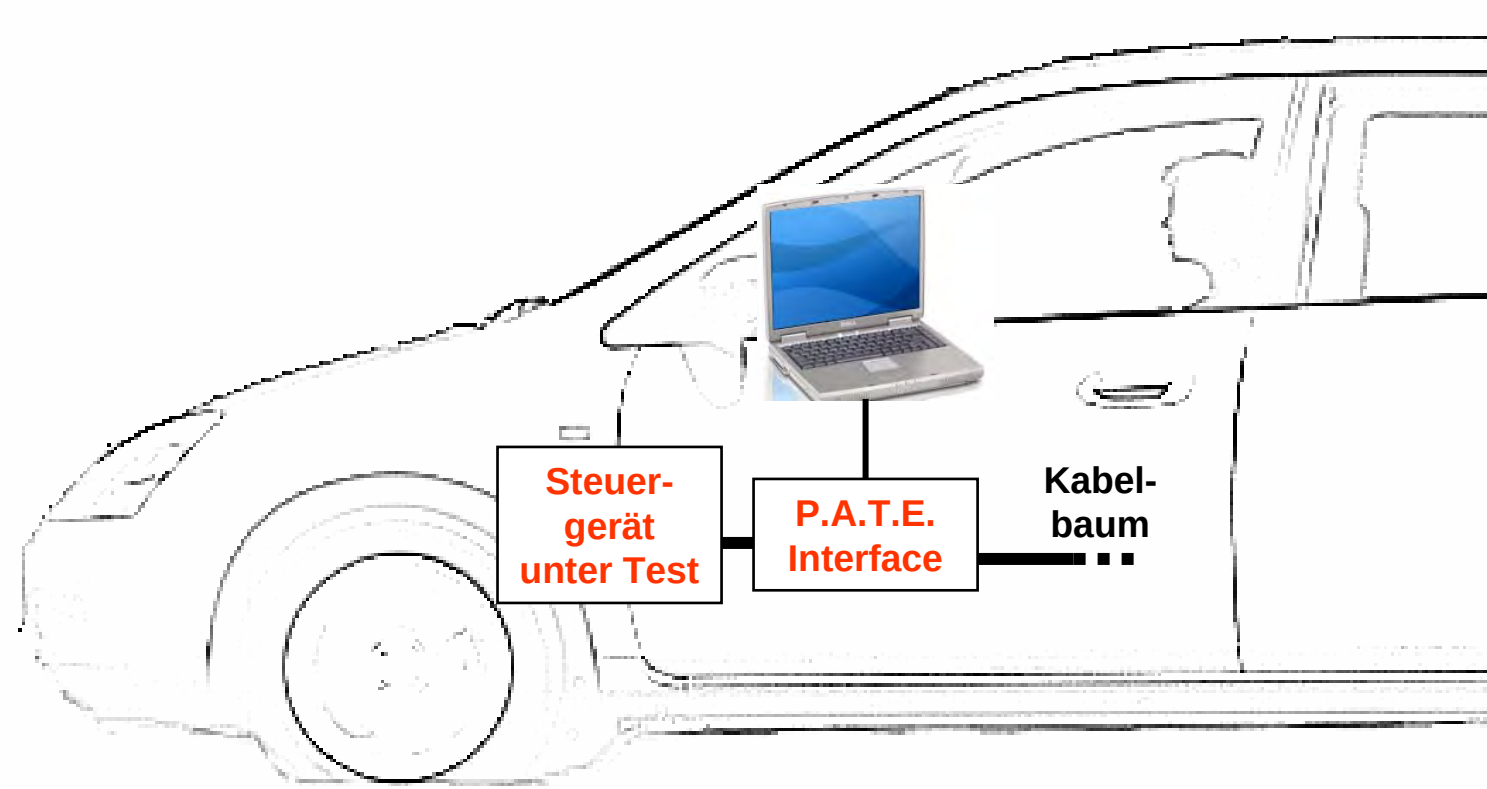
(Prototyp)

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

P.A.T.E. *mobile*

- Kompakte, fahrzeuggerechte Version des P.A.T.E. Interface wird im Fahrzeug montiert und in den Kabelbaum zum Steuergerät geschaltet.
- P.A.T.E. Software wird auf einem Notebook ausgeführt.
- Test- und Analysevorgänge vollautomatisch im Fahrbetrieb



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Folie 17

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Zusammenfassung:

- P.A.T.E. System wird seit 2003 bei einem OEM eingesetzt
- Domänen: Fahrwerk, Komfort, Bordnetz (bisher 6 Anlagen)
- Ziele: Serienfreigabe, Regressionstest, Varianten-Management
- Nachweisbare, signifikante Ressourcen-Einsparung

Ausblick:

- FKFS-Forschungsschwerpunkt „Testökonomie“ (Neue Forschungsprojekte mit einem OEM und einem Tier1):
- Automatische Generierung von Testvektoren aus der Funktionsspezfikation
- Vereinheitlichung der Testverfahren für ECU-Software, Hardware und Verbund
- Fachtagung AutoTest am 25./26. Oktober in Stuttgart

G. Baumann
M. Brost
H.-C. Reuss

arsenal research
Ein Unternehmen der Austrian Research Centers.

Simulation von elektrischen Antrieben im automotiven Bereich mit der Smart Electric Drives (SED) Library in Dymola

J. Gragger, C. Kral, F. Pirker
Geschäftsfeld
Monitoring, Energie- und Antriebstechnik arsenal research
2006.02.16

Überblick

- Anwendungen der SED im automotiven Bereich
- Allgemeines zur Simulation von elektrischen Antrieben im automotiven Bereich
- Aufbau der Smart Electric Drives (SED) Library
- Spezielle Features und deren Nutzen
- Beispiele

Anwendungen der SED Library im Automotiven Bereich

- Simulation kompletter Antriebsstränge von alternativen Antriebskonzepten (HEV)
- Simulation von elektrifizierten Nebenaggregaten
- Minimierung des Treibstoffverbrauchs in Hybridkonzepten
- Analyse der elektrischen Energieflüsse des gesamten Fahrzeuges
- Simulation dynamischer Effekte in elektrischen Komponenten
- Entwicklung und Optimierung von Betriebsstrategien

Wichtigste Funktionen der SED Library

- Die Modelle
  - Asynchronmaschinen, Permanentmagnet-Synchronmaschinen, Gleichstrommaschinen
  - Regelungen (Feldorientierte Regelung, Brushless DC, Direct Torque Control…)
  - Stromrichter (verschiedene Abstraktionsstufen)
  - Energiequellen (Akkus, Supercaps, Brennstoffzellen)
- Anwendungen
  - HEVs mit SED- und PowerTrain Library
  - Starter-Generatorkonzepten
  - Elektrifizierte Wasserpumpen
  - Elektrifizierte Ölpumpen
  - Elektrifizierte Klimaanlagen
  - Etc.

Applikationsspezifisches Antriebsdesign

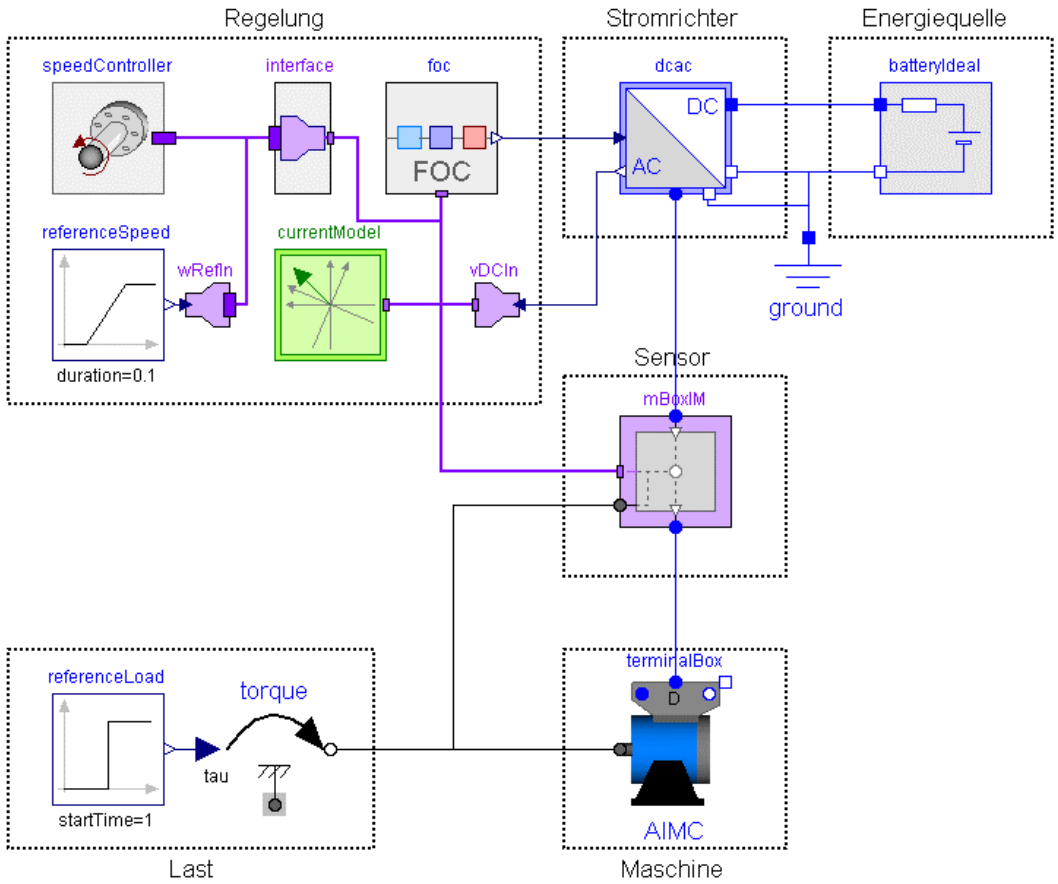
- Gemischte Systeme – Mechanisch und Elektrisch
  - Gleichzeitige Simulation von mechanischen und elektrischen Systemteilen
  - Übersichtliches Handling von Simulationstools
- Hoher Rechenaufwand
  - Definition von Abstraktionsstufen
- Kurze Arbeitszeiten für die Simulation und Analyse von Systemen
  - Automatisierung von Entwicklungsschritten

Applikationsspezifisches Antriebsdesign

- Verschiedene Technologien (z.B. Akkus, Supercaps, etc.)
- Abstimmung der Komponenten (Bauteilspezifikationen)
- Maximierung der Effizienz
- Ganzheitliche Analyse von dynamischen Effekten
- Bauteilschutz (Überstrom, Überspannung, etc.)
- Regelungseinstellungen (Dynamische- und statische Abweichungen)

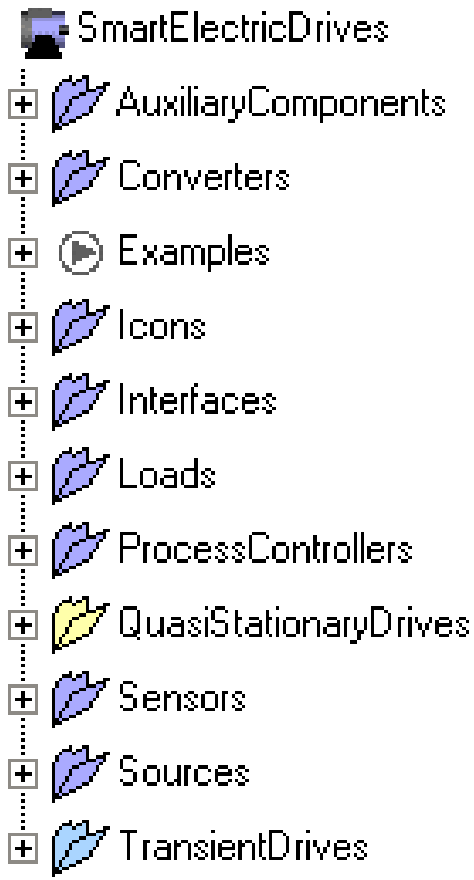
Komponenten von elektrischen Antrieben

- Energiequellen
- Stromrichter
- Elektrische Maschinen
- Sensoren
- Regelungsalgorithmen
- Mechanische Lasten



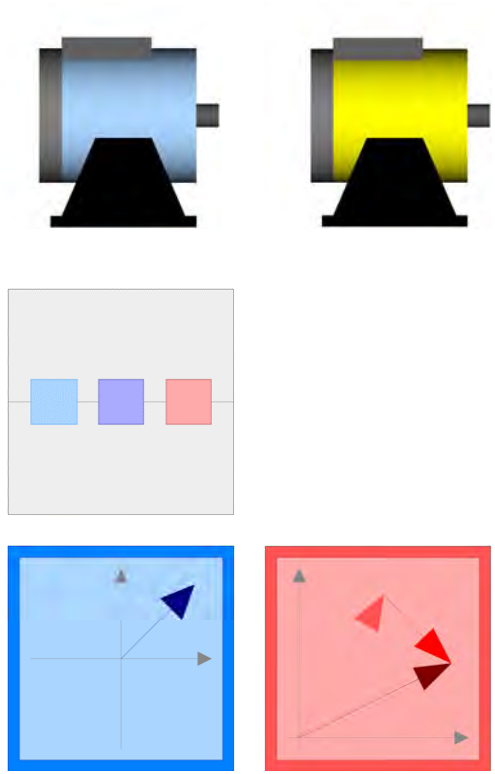
Aufbau der SED Library

- **„Ready to use“ - Modelle** zur Automatisierung des Simulationsaufbaus
- Verschiedene **Abstraktionsstufen** für unterschiedliche Analysen
- **Signalbusse** für übersichtliches Design

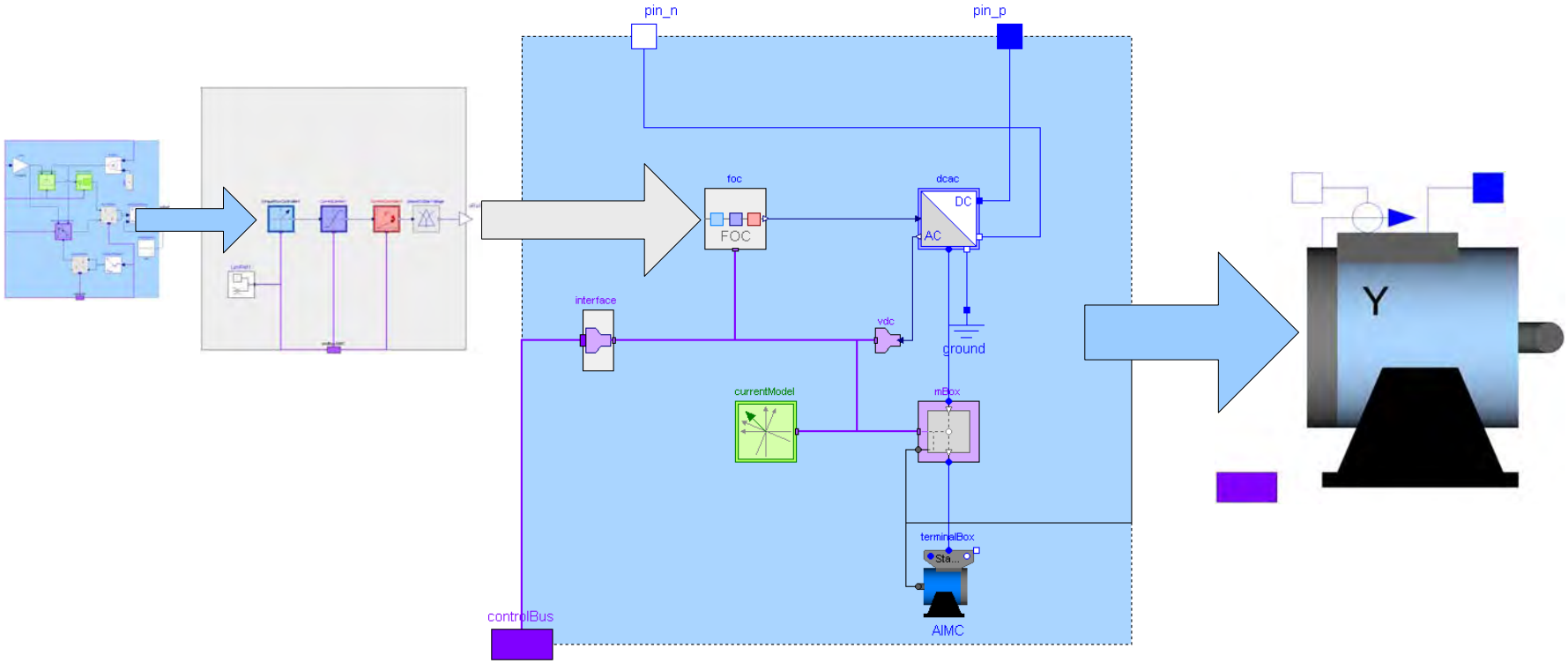


„Ready to use“ - Modelle

- Modelle von geregelten Maschinen
- Modelle von kombinierten Regelungsblöcken
- Modelle von elementaren Regelungsblöcken

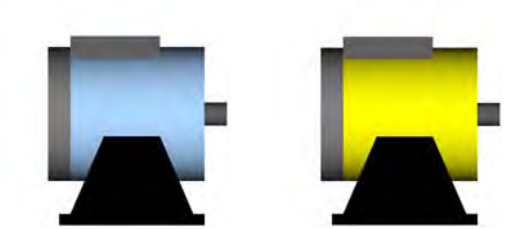


ASM mit feldorientierter Regelung



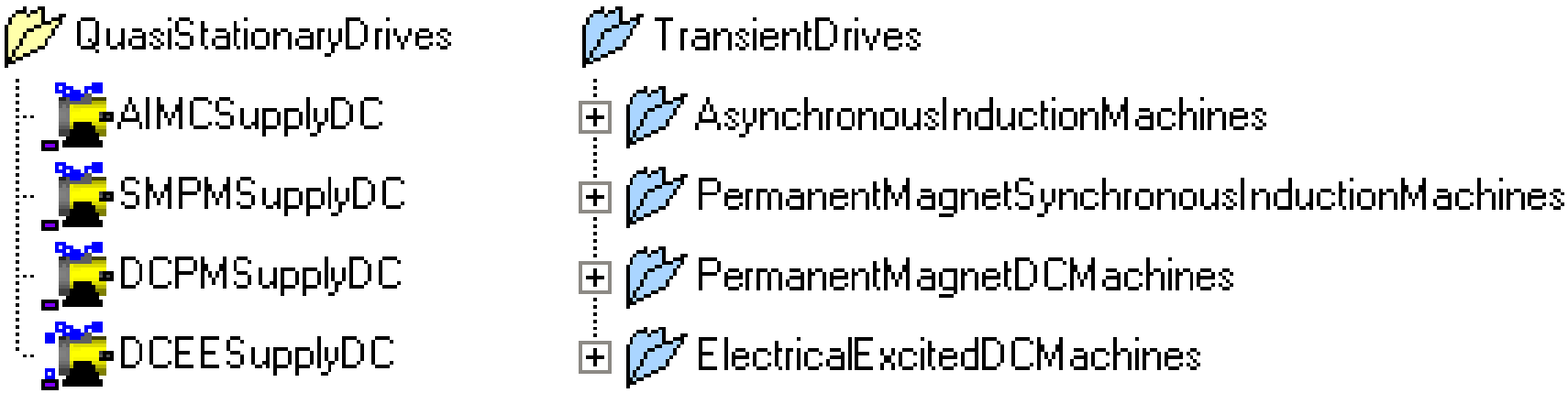
Verschiedene Abstraktionsstufen

- Regelungen und Maschinen
  - Transiente- und quasi stationäre Modelle

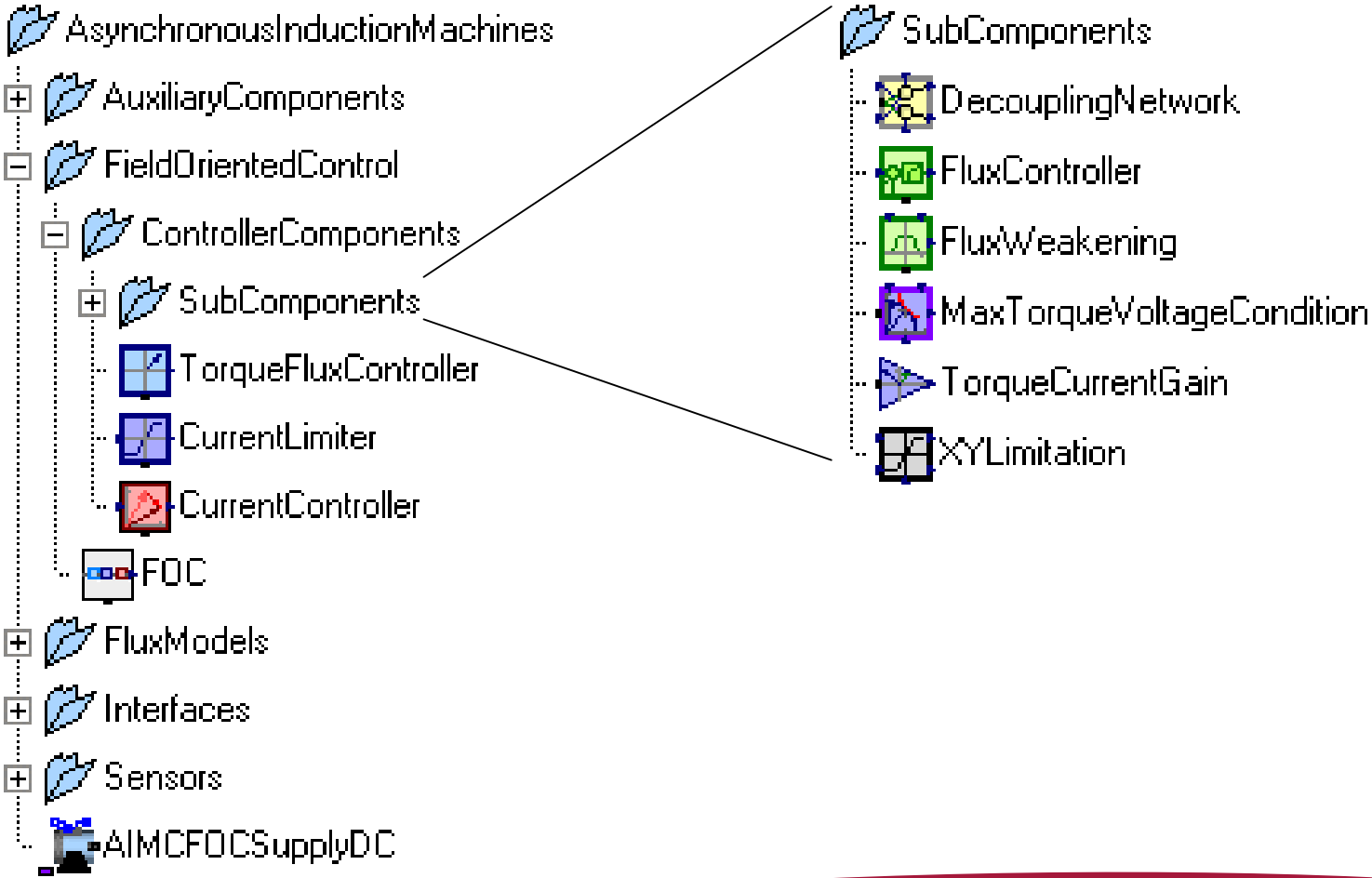


- Stromrichter
  - Leistungsbilanz
  - Ideale Thyristoren, Schalter,…
  - (zukünftig reale Schalter)

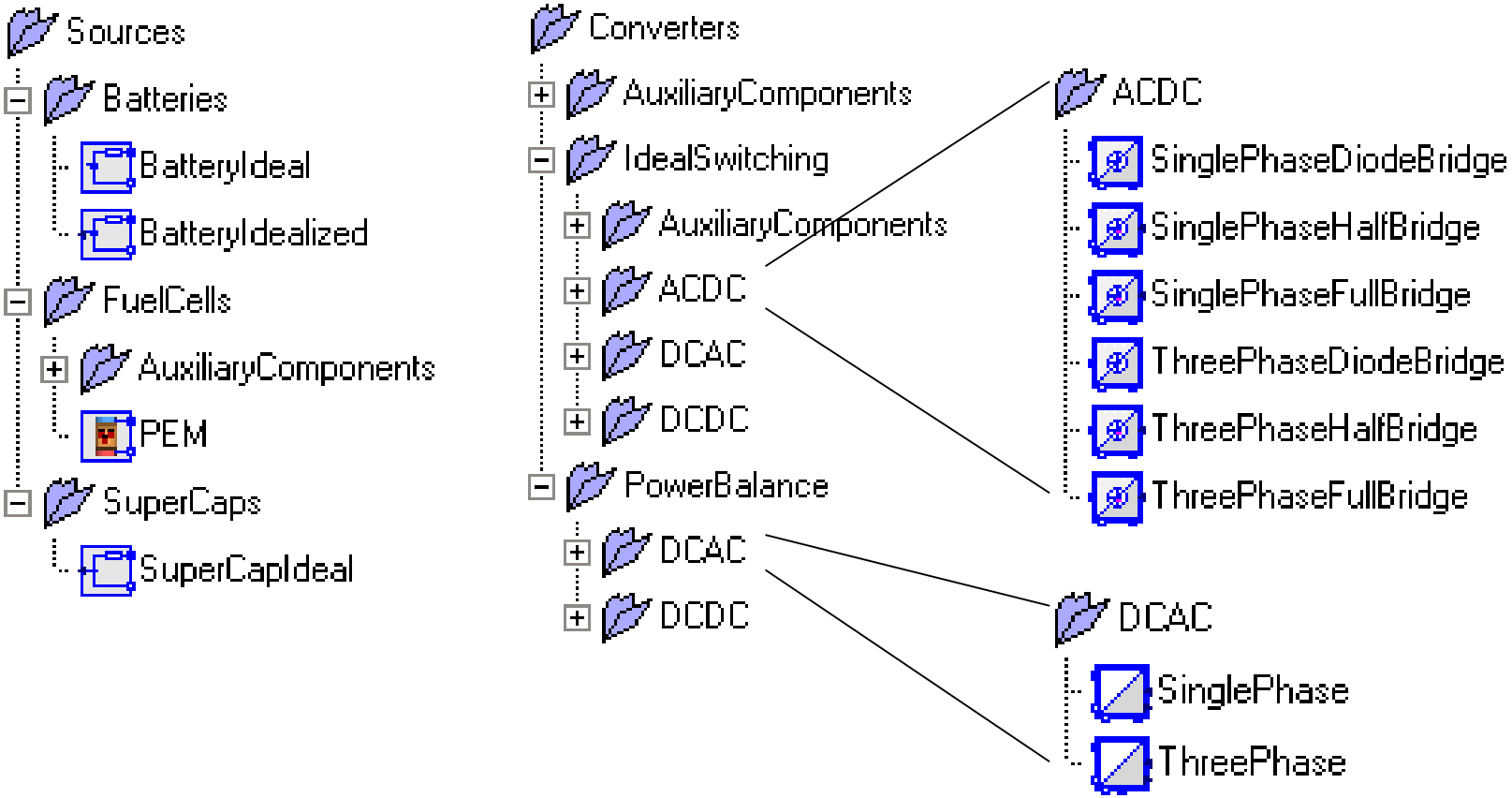
Wichtige Packages – Maschinen und Regler



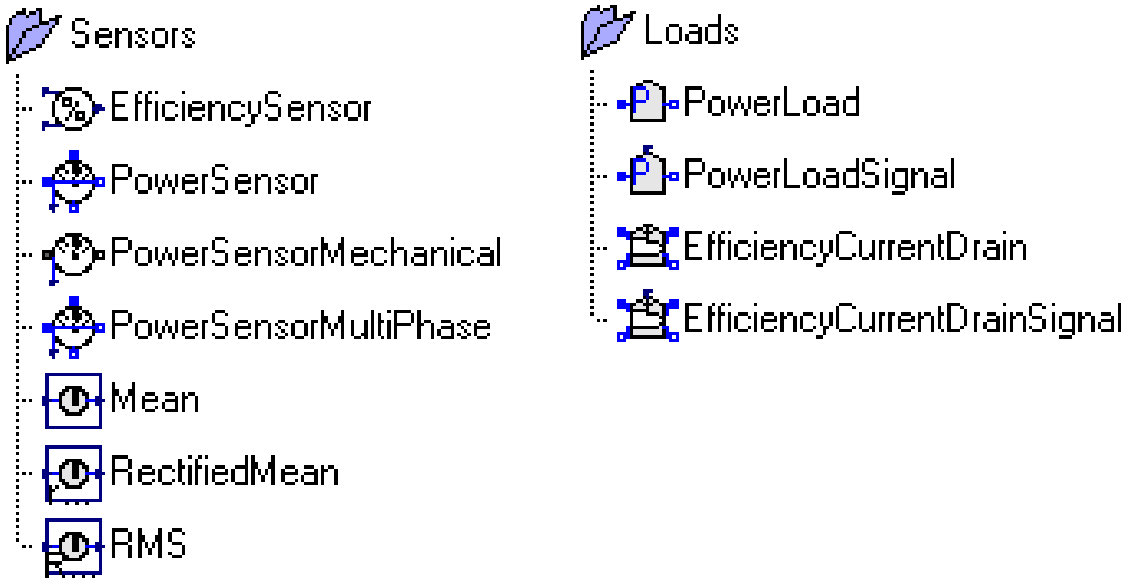
Wichtige Packages – Maschinen und Regler



Wichtige Packages – Quellen und Stromrichter



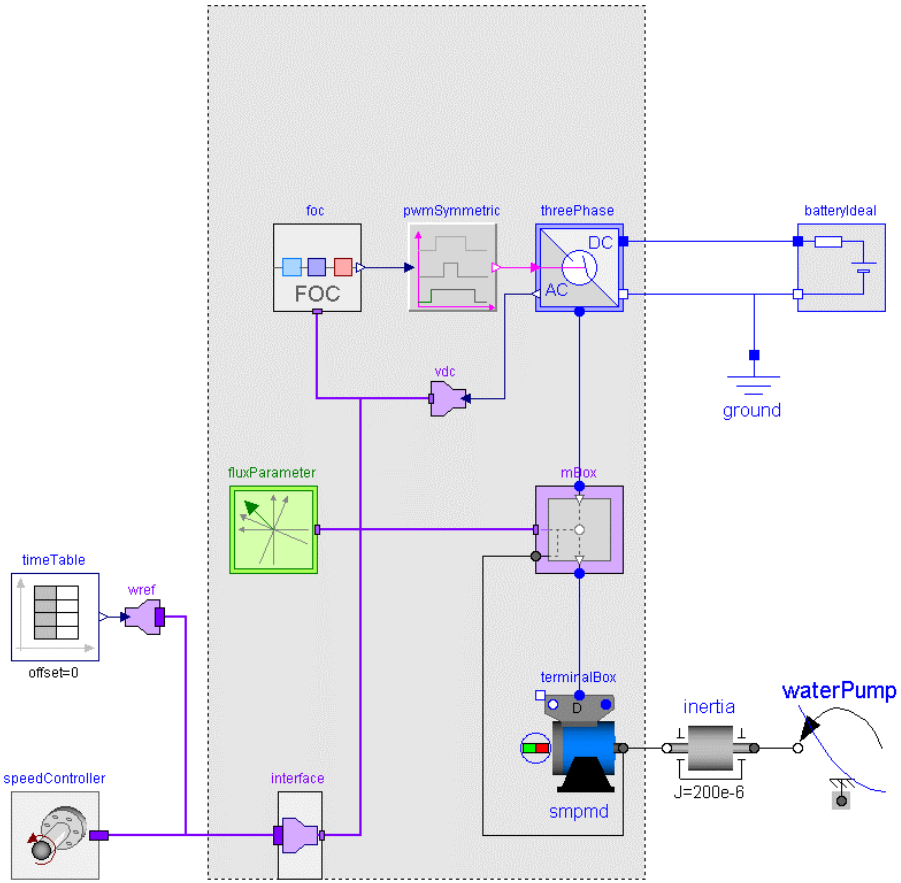
Wichtige Packages – Sensoren und Lasten

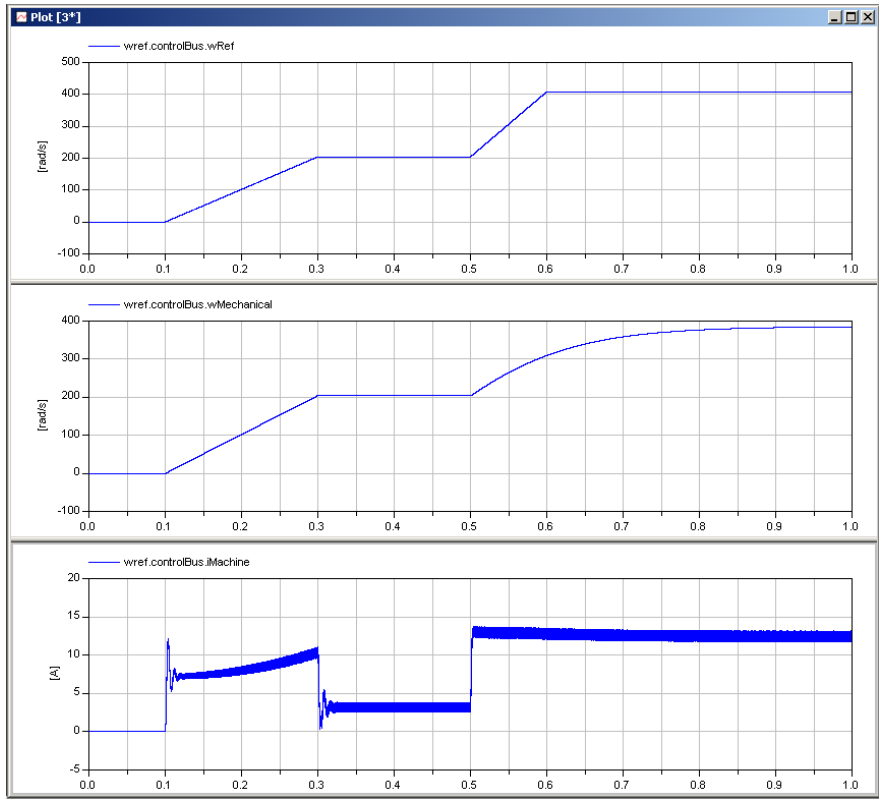


Beispiele

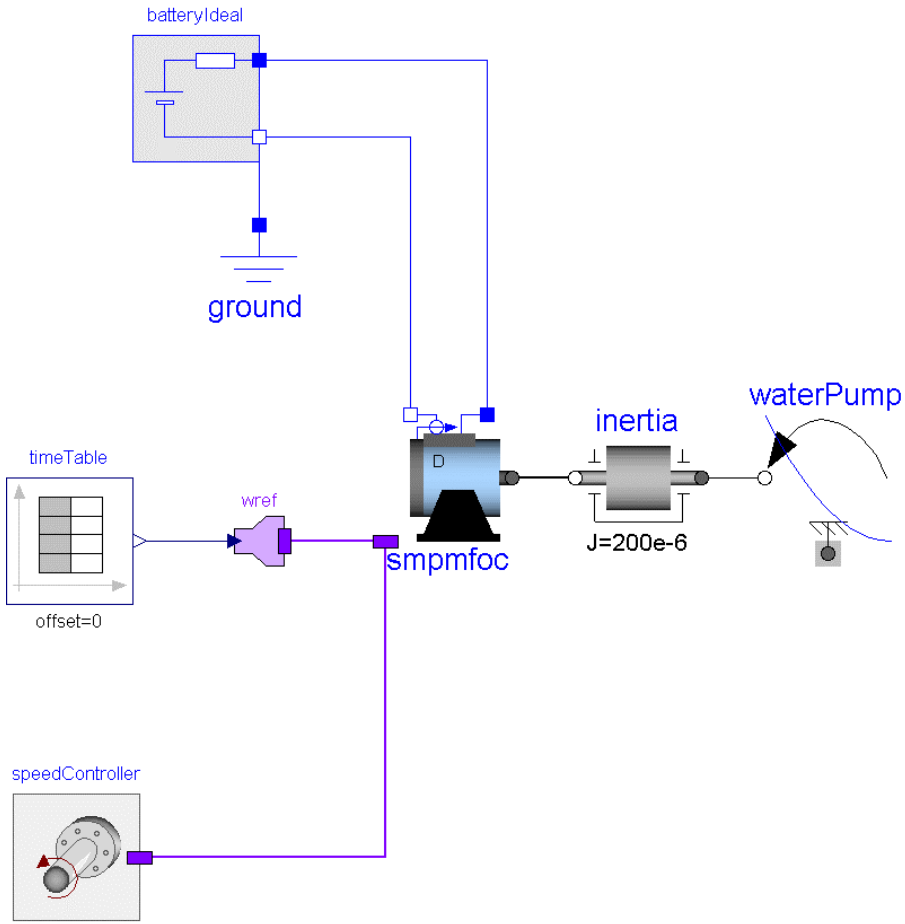
- Drehzahlregelung eines PMSM Pumpenantriebs
  - 4-polig in Dreieckschaltung
  - V_0 = 5.26 / 9.11 V
  - $f_{Nominal}$ = 130 Hz
  - $I_{Nominal}$ = 22 / 12.7 A
  - J_r = 72e-6
- Mit verschachtelten Modellen
- In verschiedenen Abstraktionsstufen

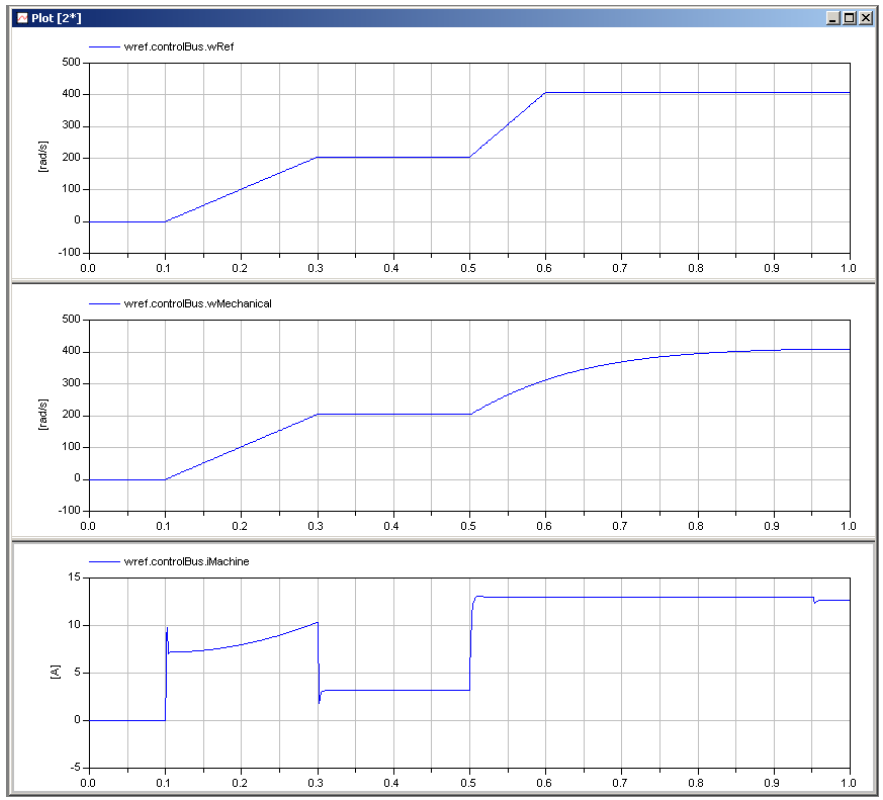
Detaillierte Simulation – Transient mit Schaltern



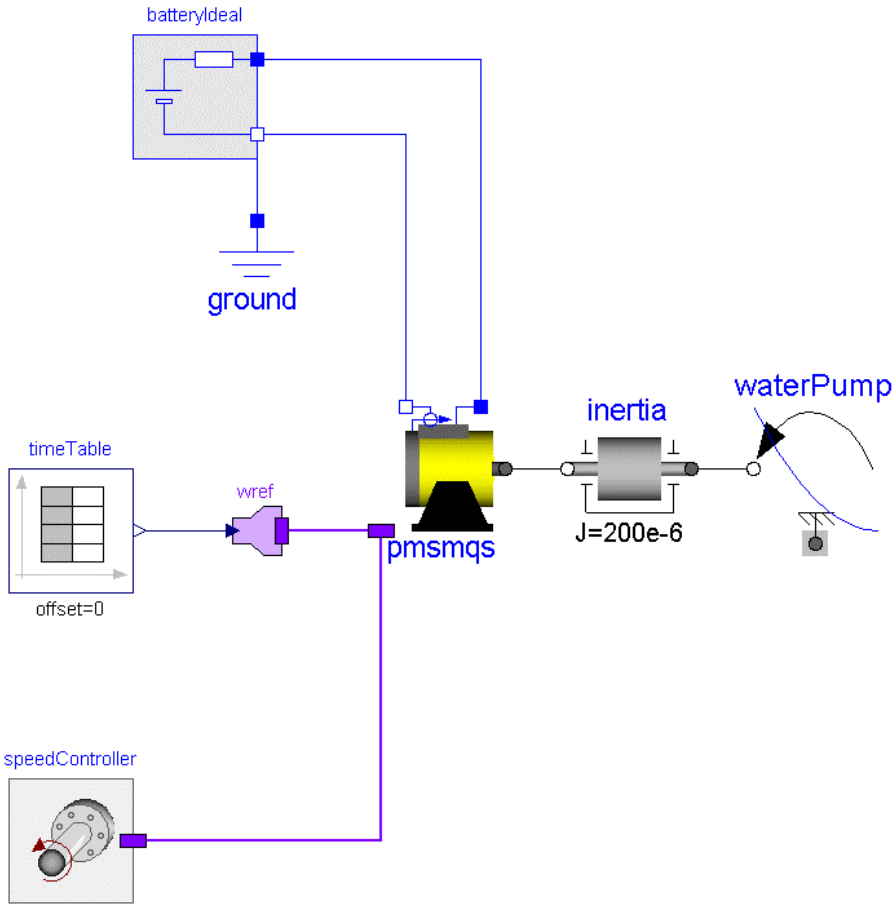


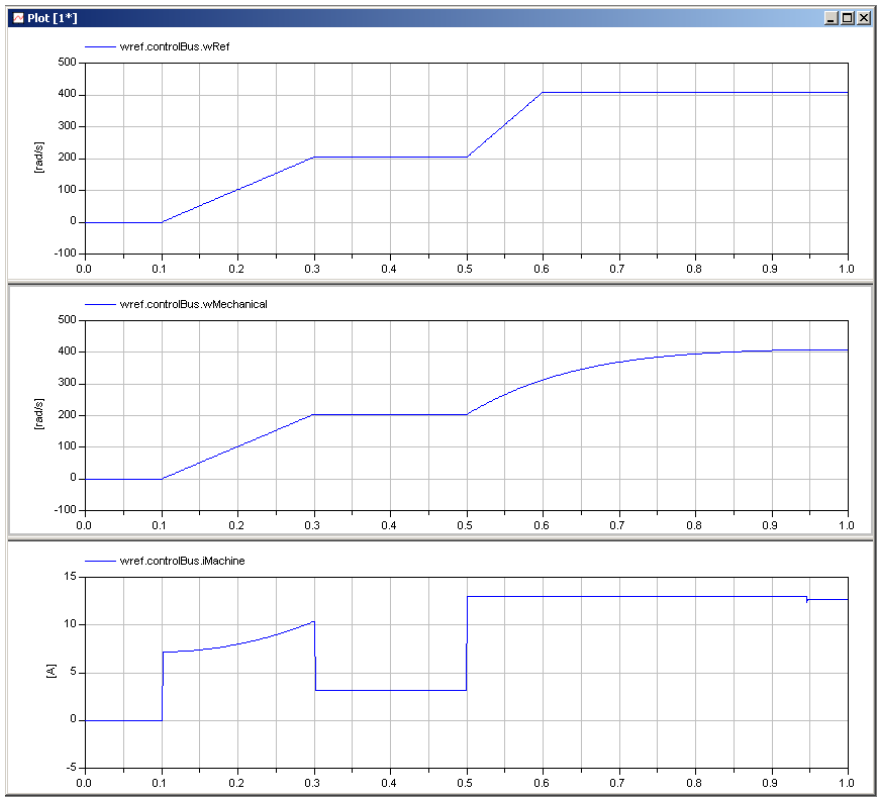
Simulation – Transient ohne Schalteffekte





Quasi-stationäre Simulation





Vergleich der Simulationen

- Modelltiefe beeinflusst die Simulationszeit erheblich
- QS-Modell t < 1sec
- Transientes Modell ohne Schalter t ~ 1.5sec
- Transientes Modell mit PWM t ~ 5min 30sec

Zusammenfassung

- „Ready to use“ - Modelle erleichtern den Aufbau von Simulationen
- Verschiedene Abstraktionsstufen verkürzen die Rechenzeiten
- Alle Komponenten für Antriebe sind in der SED Library vorhanden
- Schneller Simulationsaufbau und Parametrierung durch entsprechende Dokumentation und Berechnungsfunktionen ist möglich

arsenal research
Ein Unternehmen der Austrian Research Centers.

Vielen Dank für die Aufmerksamkeit

Gragger Johannes V.
arsenal research
Johannes.gragger@arsenal.ac.at
www.arseanl.ac.at

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Durchgängige Open-Loop-Testverfahren für Kfz-Elektronik im Labor und Fahrversuch

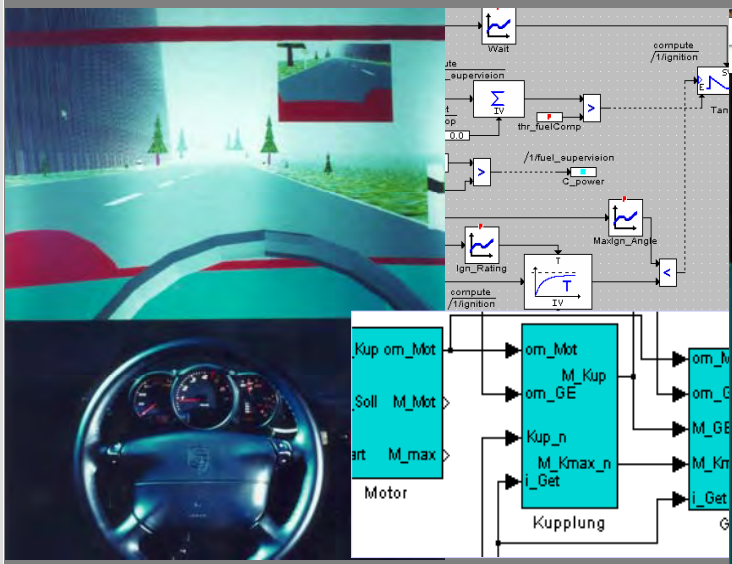
Gerd Baumann
Michael Brost
Hans-Christian Reuss

ASIM 2006
München, 20./21. Februar 2006

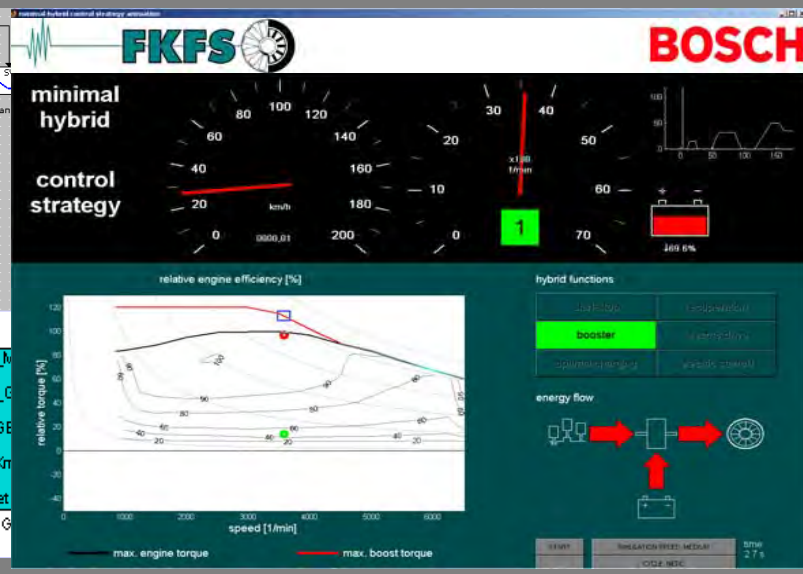
Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

- Stiftung, Gründung 1930
- Forschung und Entwicklung
- Lehre (mit Uni Stuttgart)
- Motorentechnik, Akustik, Fahrdynamik, Aerodynamik, Elektronik und Software
- 140 MA, Umsatz 15 M€ / a

Arbeitsgebiet Kfz-Mechatronik (Beispiele):



ECU-Software-Entwicklung, Test und Diagnose



Antriebs- und Bordnetz-Management, Hybridantrieb

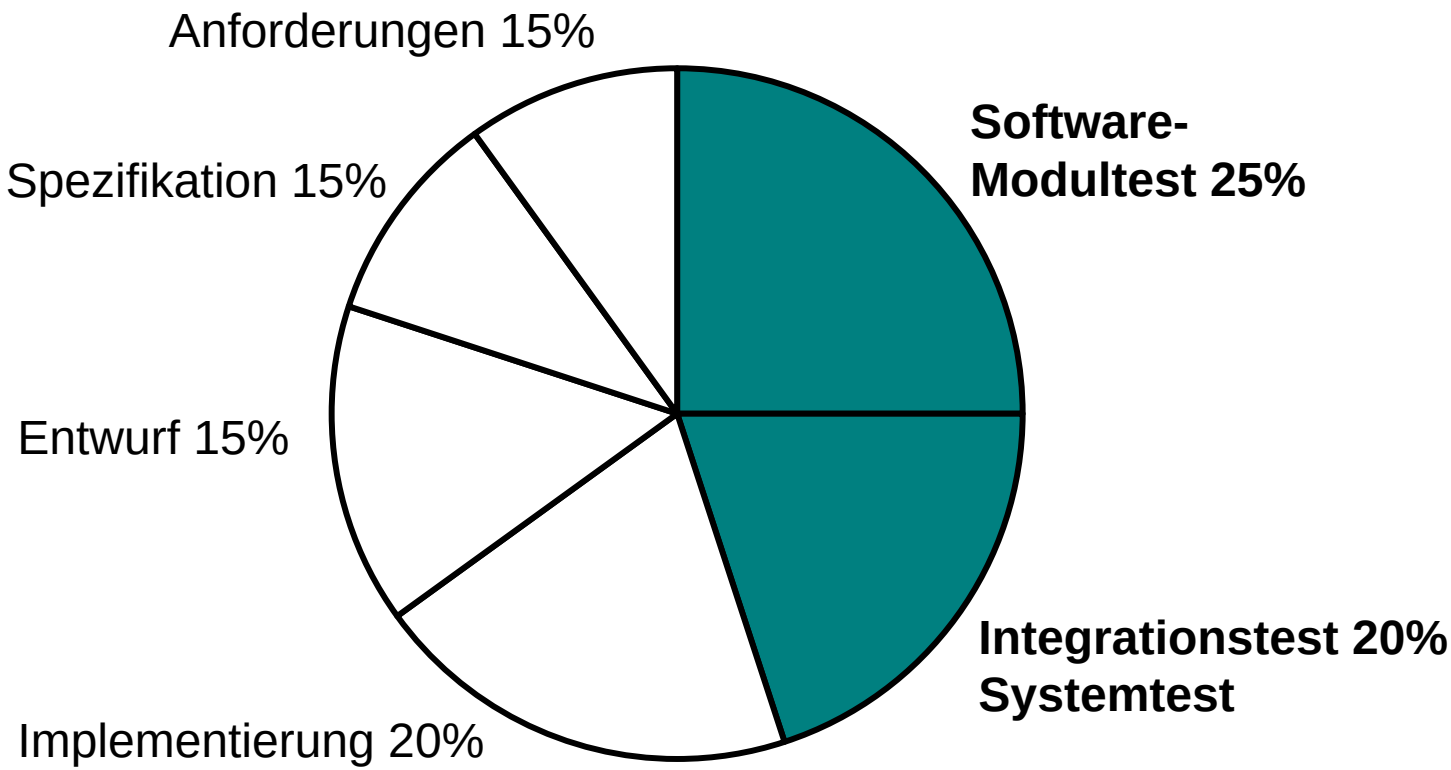
Folie 2

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Kostenverteilung beim Software-Engineering

- SW-Engineering: fast 50% der Kosten entfallen auf den Test
- Steigender SW-Anteil im Fahrzeug
- Testaufwand nähert sich dem des SW-Engineering an

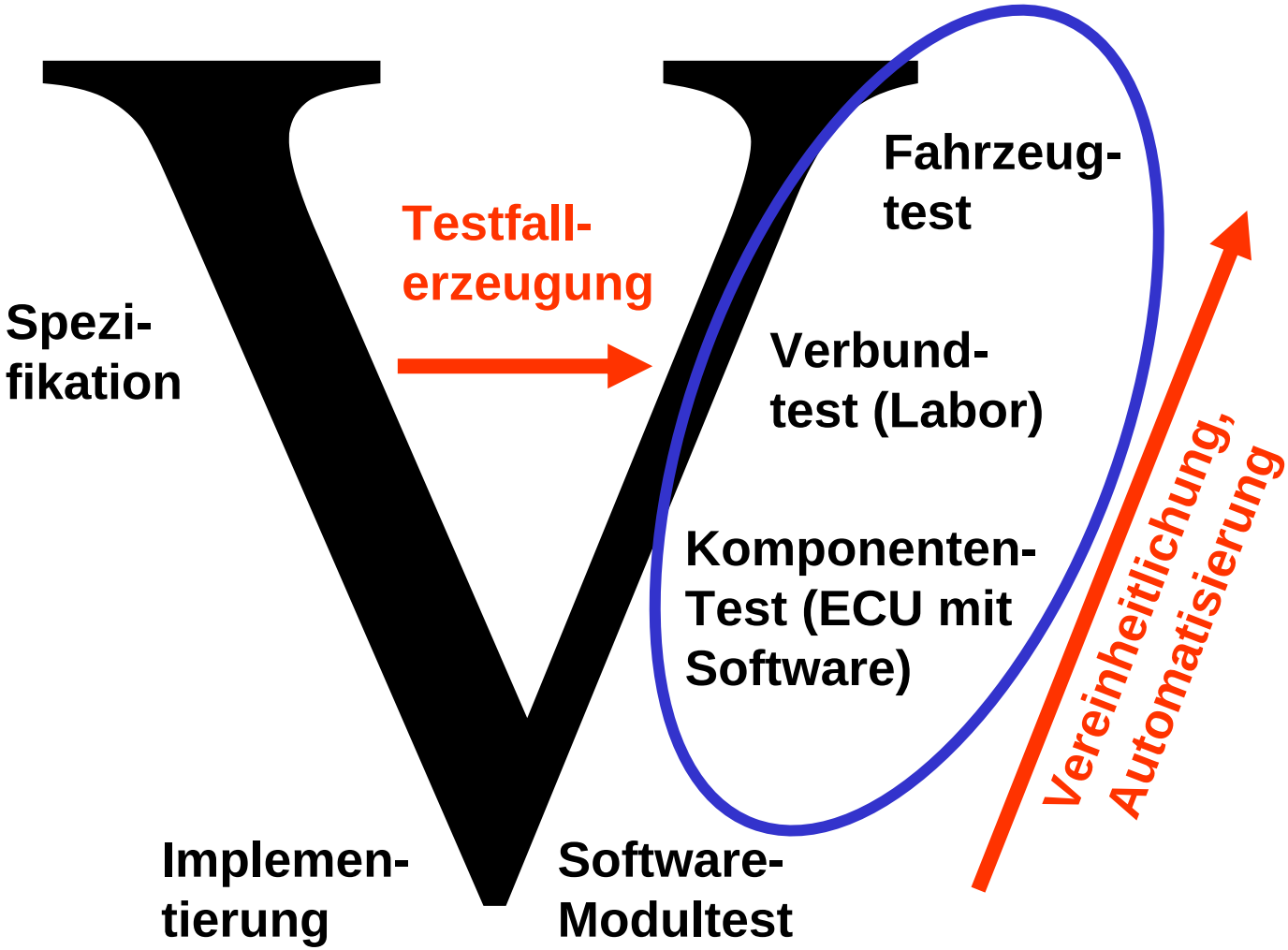


Quellen: [Vigenschow.04] und [Ebert.05]

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Zielsetzung:



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Heutige Situation:

- Uneinheitliche Testverfahren und Testbeschreibungen für Softwaretest, ECU-Test im Labor und Fahrzeugtest.
- Häufig manuelle Testfallerzeugung
- Sehr hoher Hardware- und Software- Aufwand für HIL-Verbundtests

These:
Die Zunahme der Komplexität der Elektronik, Software und Vernetzung erfordert eine Vereinfachung der Testverfahren !

Lösungsansätze:

- Open-Loop-Verfahren und quasistationäre Tests als Ergänzung zum HIL-Verfahren, insbesondere für den Karosserie- und Innenraumbereich
- „Universaltester“: Entkopplung von ECU und Testsystem
- Teilparalleler Testaufbau statt vollparallelem Testaufbau

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

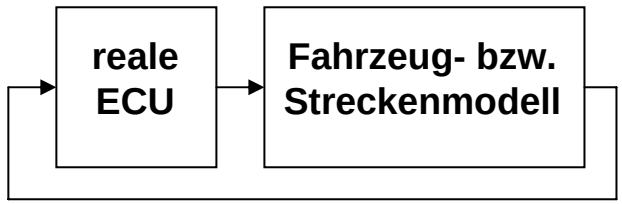
Closed Loop vs. Open Loop Testverfahren

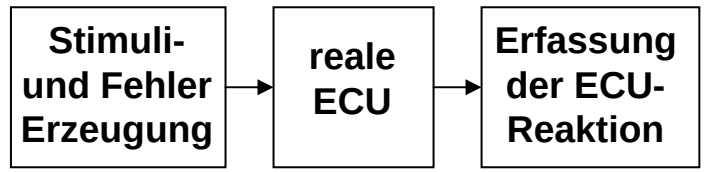
Hardware-in-the-Loop-Simulator

Foto: Fa. dSpace

Ergänzung

Open-Loop-Tester





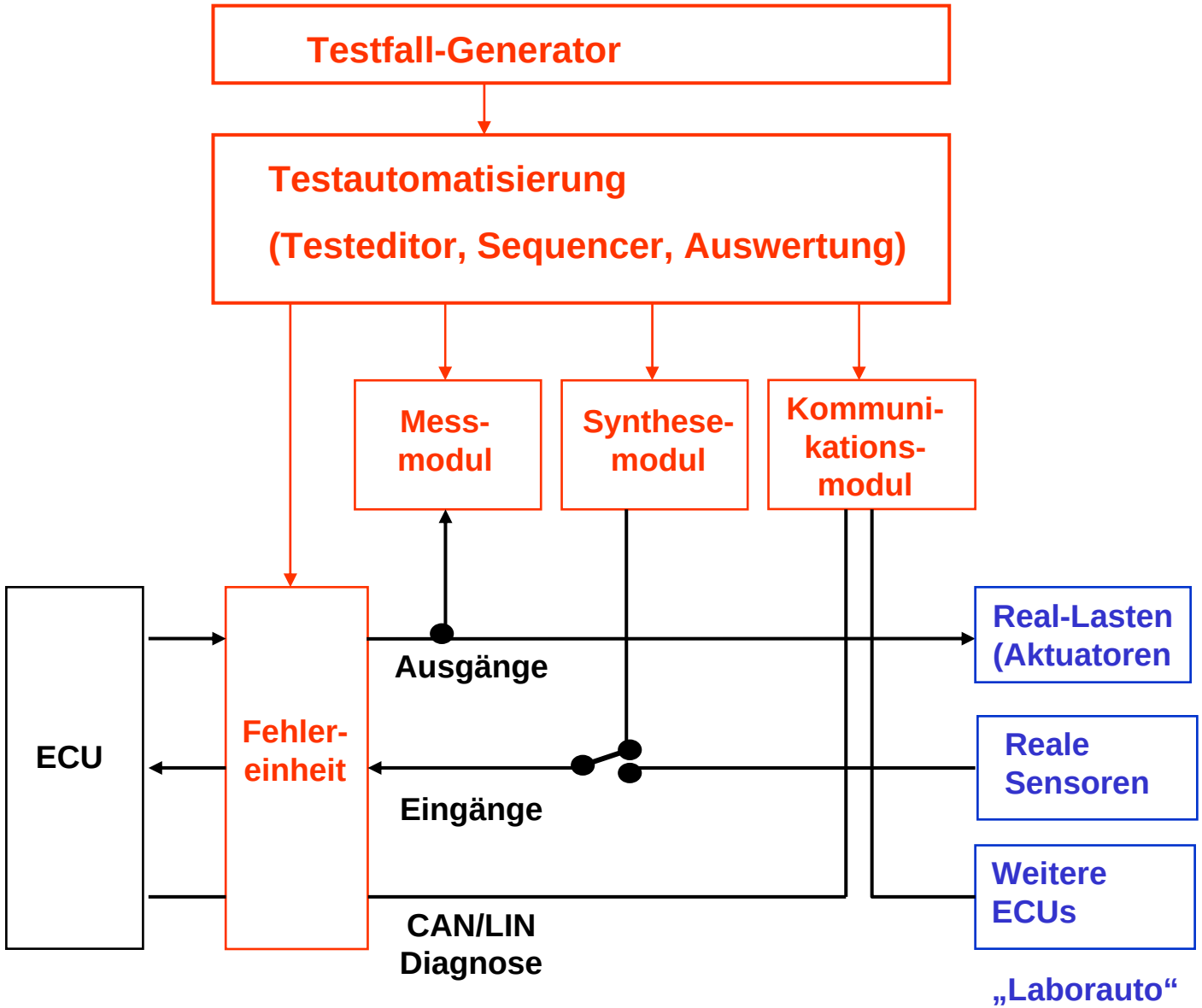
- Closed loop
- Dynamische Betriebszustände
- Fahrzeug- bzw. Strecken-modell notwendig
- Komplexe Signalverläufe i.d.R. manuelle Auswertung
- Typische Anwendungen: Motor, Antriebsstrang
- Einsatz im Labor

- Open loop
- Quasistationäre Betriebszustände
- Direkte Abbildung der Spezifikation in Testsequenzen
- Einfache Signalverläufe -> automatische Auswertung
- Typische Anwendungen: Karosserie und Innenraum
- Einsatz im Labor und Fahrzeug

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Prinzipieller Aufbau eines Open Loop-Testsystems für eine einzelne ECU



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für
Kraftfahrwesen und
Fahrzeugmotoren
Stuttgart

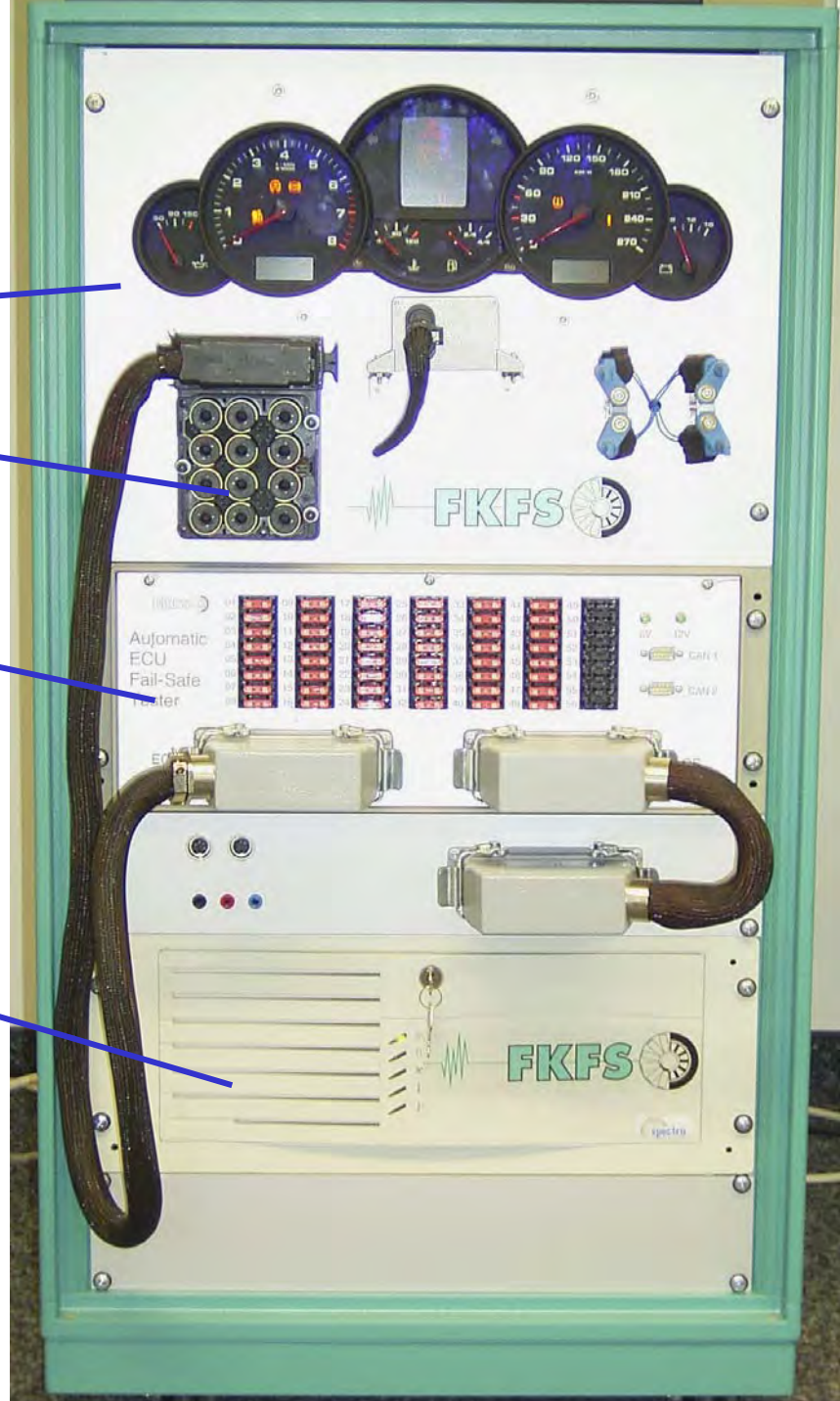
Open Loop-Testsystem P.A.T.E

„Laborauto“

ECU (Prüfling)

Fehlereinheit,
Mess/Synthesemodul,
Kommunik. modul

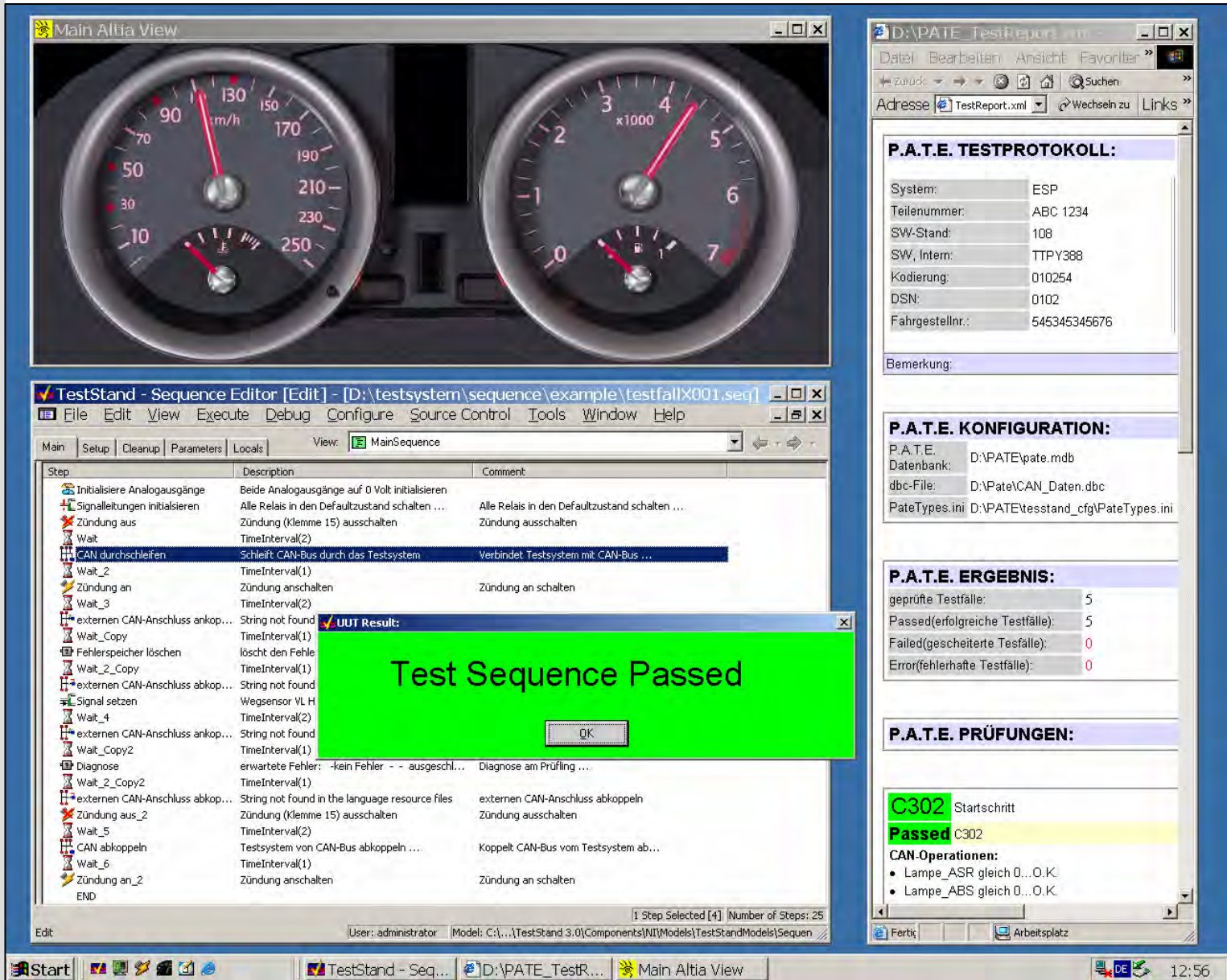
Steuerrechner
Testautomatierung



G. Baumann
M. Brost
H.-C. Reuss

P.A.T.E Software

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart



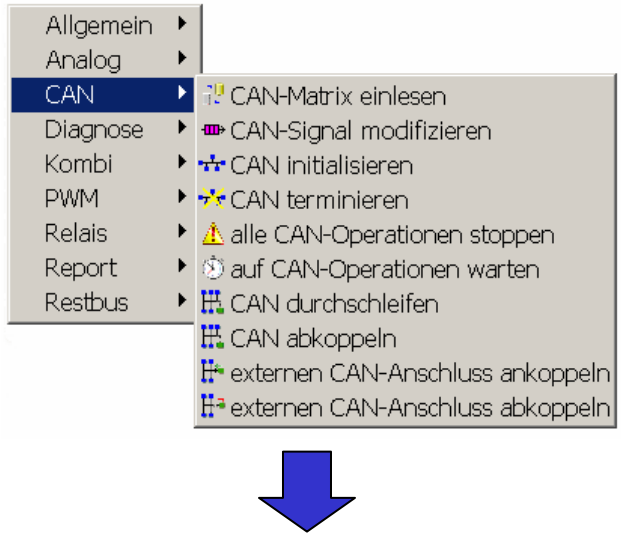
Folie 9
G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

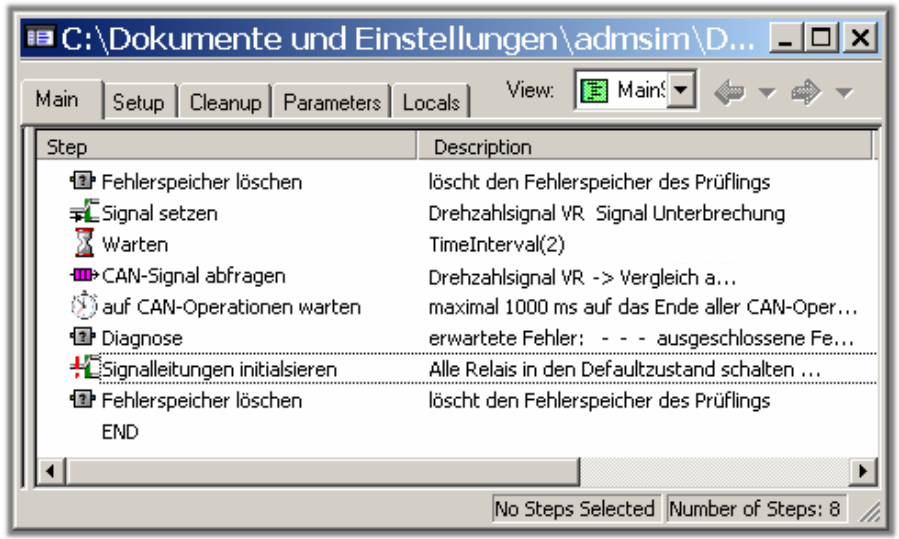
P.A.T.E Software

- Vollständig menügeführte Testerstellung ohne Skriptsprache
- Mehrstufige Testhierarchie

Auswahlmenü für Testschritte



Darstellung eines Testfalls im Editor

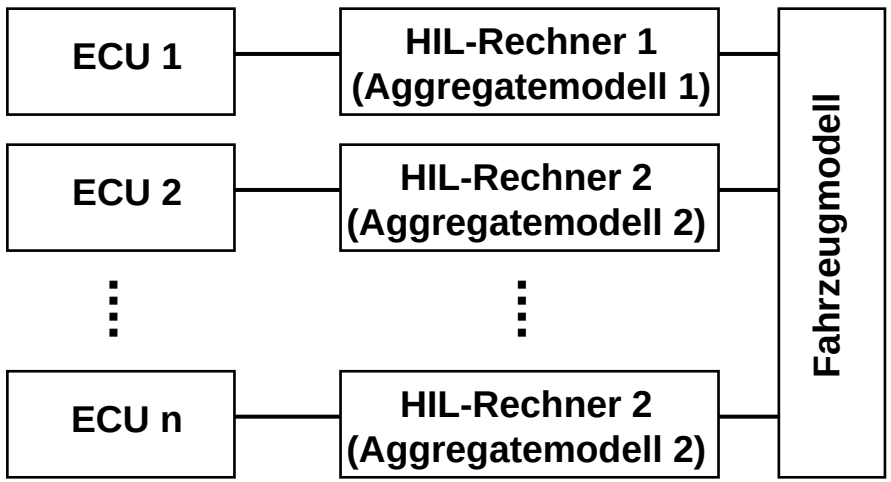


G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Ansätze beim Verbundtest

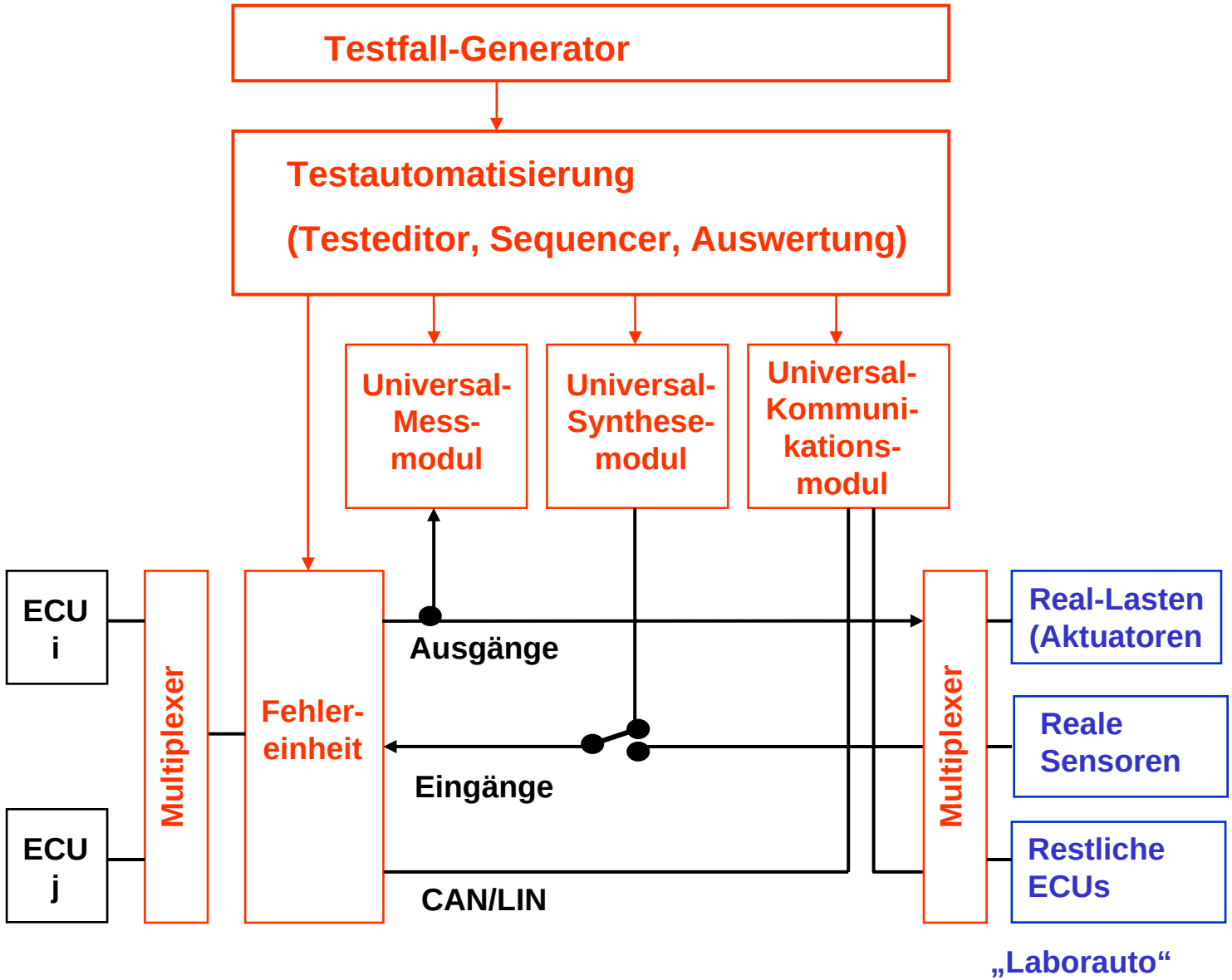
- Heute verbreitet: häufig „massiv paralleler Verbundtest“ durch Parallelbetrieb aller ECUs an einem „Vernetzungs-HIL“, i.d.R. mit vollständiger Substitution der Sensoren
- Vorteil: theoretisch 100 % Nachbildung des realen Systems möglich
- Nachteil: Hoher Investitions- und Pflegeaufwand



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Prinzipieller Aufbau eines Open Loop-Testsystems für einen Steuergeräte-Verbund (vereinfacht)

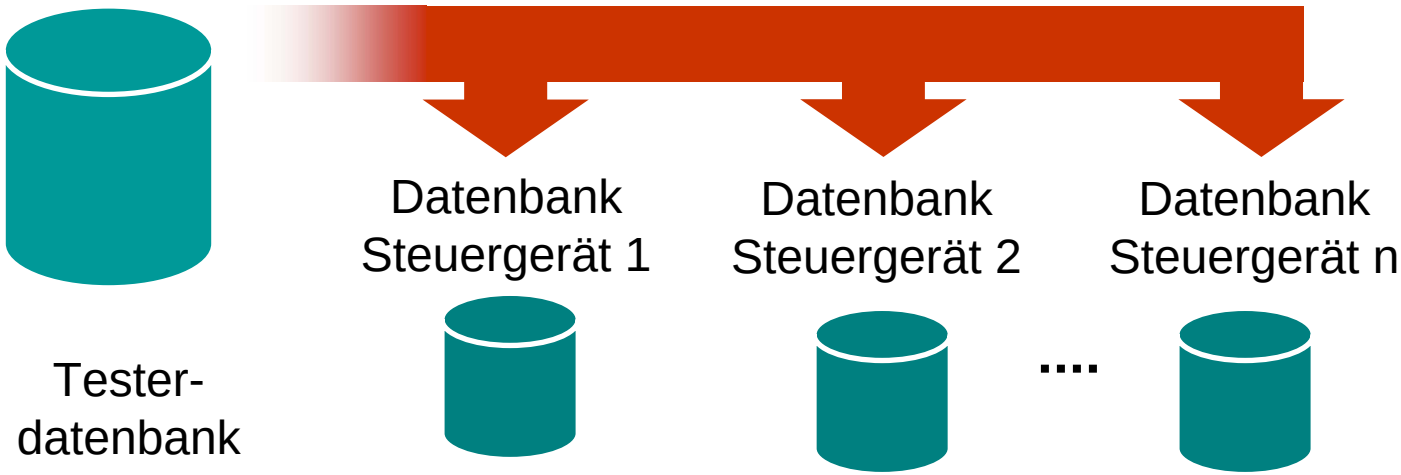


G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Universaltester für Verbundsyteme

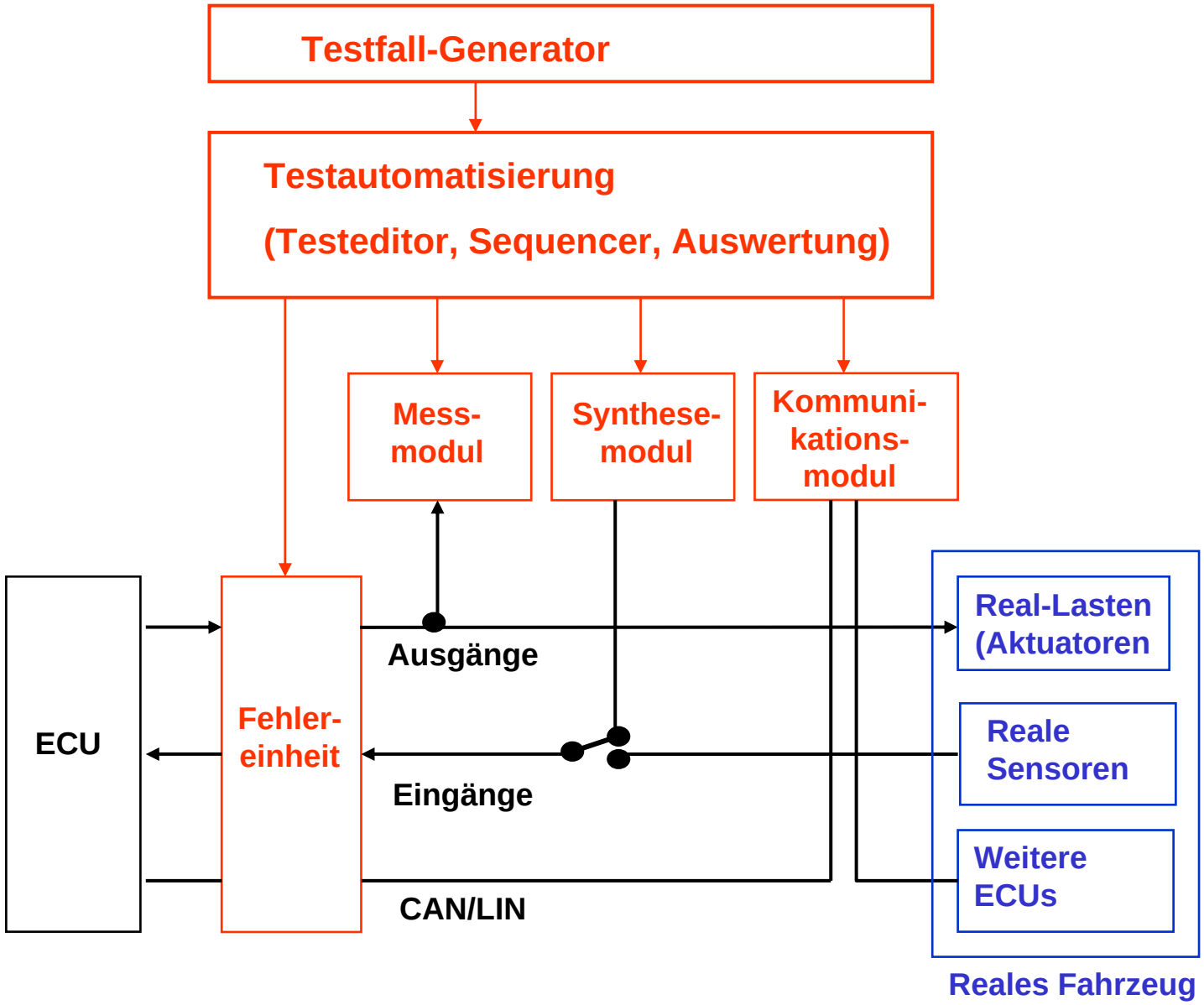
- Vollständig datenbankgestützte Beschreibung des Testaufbaus
- Tester-Datenbank beschreibt das Testsystem, z.B. Anzahl und Typ der verfügbaren CAN-Kanäle, Messkanäle, etc.
- Steuergeräte-Datenbanken beschreibt jeweils ein Steuergerät, z.B. Pinbelegung, verwendete Diagnose-Codes usw.
- Weitere Datenbanken für systemweite Daten (CAN-Kommunikationsmatrix, verwendetes Diagnose-Protokoll, etc.)
- Beschreibung eines konkreten Testaufbaus (z.B. Verbindung eines physikalischen ECU-Pins mit einem physikalischen Tester-Kanal) erfolgt durch Verknüpfung der Datenbanken



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Testsystem-Einsatz im Versuchsfahrzeug



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

P.A.T.E.

mobile

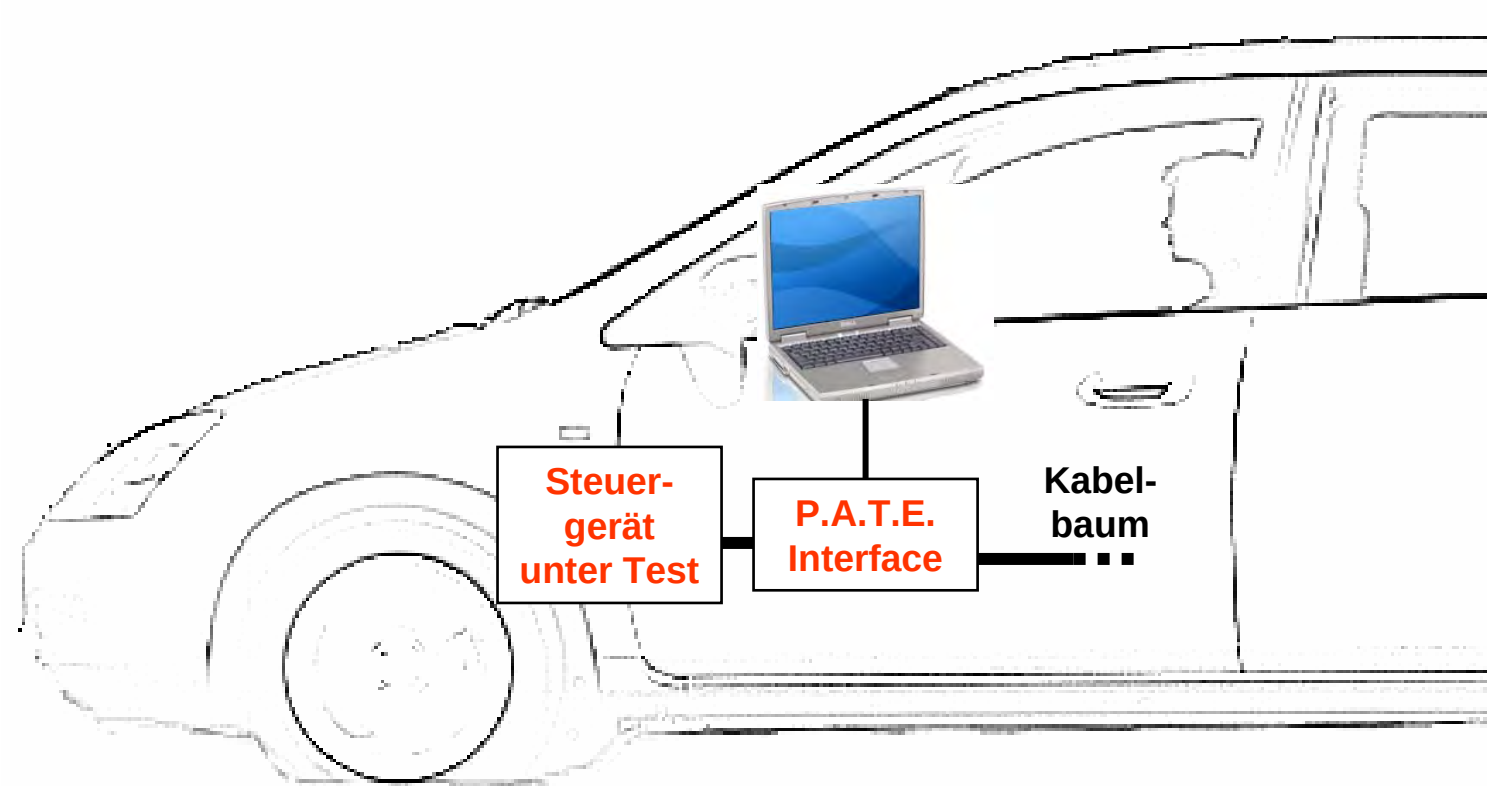
(Prototyp)

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

P.A.T.E. *mobile*

- Kompakte, fahrzeuggerechte Version des P.A.T.E. Interface wird im Fahrzeug montiert und in den Kabelbaum zum Steuergerät geschaltet.
- P.A.T.E. Software wird auf einem Notebook ausgeführt.
- Test- und Analysevorgänge vollautomatisch im Fahrbetrieb



G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Folie 17

G. Baumann
M. Brost
H.-C. Reuss

Forschungsinstitut für Kraftfahrwesen und Fahrzeugmotoren Stuttgart

Zusammenfassung:

- P.A.T.E. System wird seit 2003 bei einem OEM eingesetzt
- Domänen: Fahrwerk, Komfort, Bordnetz (bisher 6 Anlagen)
- Ziele: Serienfreigabe, Regressionstest, Varianten-Management
- Nachweisbare, signifikante Ressourcen-Einsparung

Ausblick:

- FKFS-Forschungsschwerpunkt „Testökonomie“ (Neue Forschungsprojekte mit einem OEM und einem Tier1):
- Automatische Generierung von Testvektoren aus der Funktionsspezfikation
- Vereinheitlichung der Testverfahren für ECU-Software, Hardware und Verbund
- Fachtagung AutoTest am 25./26. Oktober in Stuttgart

G. Baumann
M. Brost
H.-C. Reuss

Testen von Software: Ein Nachweis der korrekten Steuergeräte-Funktionalität ?

ASIM/GI Fachgruppentreffen 2006

München, 21. Februar 2006

HELLA®

Ideen für das Auto der Zukunft

Testen von Software

Inhalt des Vortrages

- Einordnung in Themenumfeld
- Requirements und Entwicklungsprozess
- Testprozessverbesserung
- Vorgehensbeispiele aus Testphasen
- Verbesserungspotenzial
- Resümee

Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Prüfsystem für Funktions-, Leistungs- und Stresstest

Elektrische Nachbildung der Prüflings-umgebung bezogen auf das Fahrzeug:

- Originallasten / -sensoren
- Lastnachbildungen
- Sensor- / Schnittstellennachbildungen
- Restbussimulation
- Nachbildung von digitalen / analogen Signalen
- parallele Überprüfung aller Gerätepins
- Umschalten der Ausgänge auf programmierbare elektronischen Lasten
- zur Prüfung des Überlast- / Open Load-Verhaltens

Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Einordnung der Fehlersimulation

Unterscheidung der Blickwinkel auf gewünschtes Systemverhalten:

- Betrieb mit konsistenten Signalen und Parametern
- Betrieb mit fehlerhaften Signalen und/oder Parametern

Was kann die Simulation leisten:

- Erprobung des Systemverhaltens im Fehlerfall
  - Verständnisgewinn
  - Referenzsignalverläufe
- Die Fehlersimulation ermöglicht die Erweiterung der Spezifikation auch für den Fehlerfall bezüglich Inputsignale oder Parameter

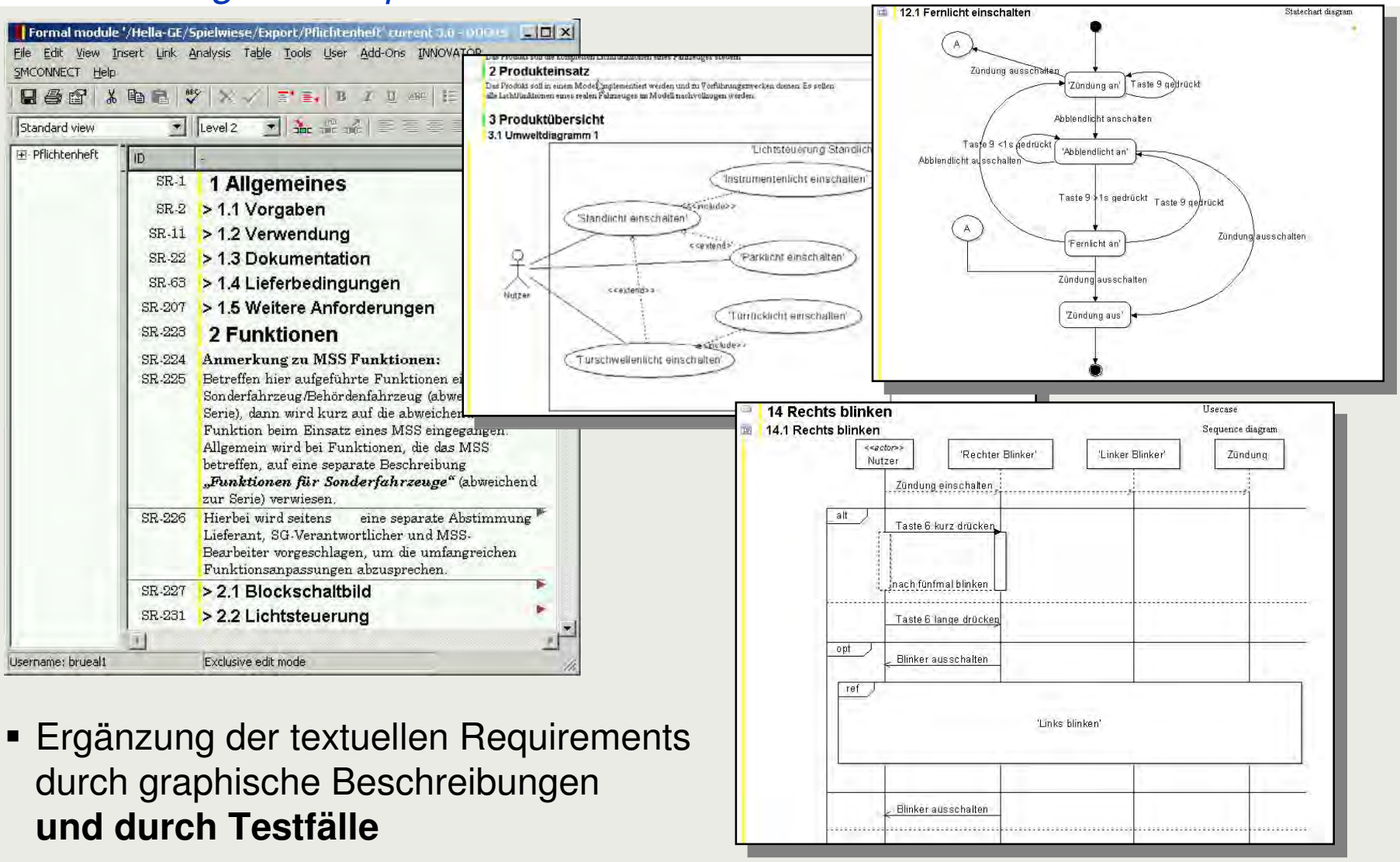
Wo sind die Grenzen:

- Die Simulation liefert Informationen zu explizit dargestellten Szenarien
- Nicht betrachtete / bekannte Szenarien sind weiter nicht evaluiert

Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Präzisierung der Requirements

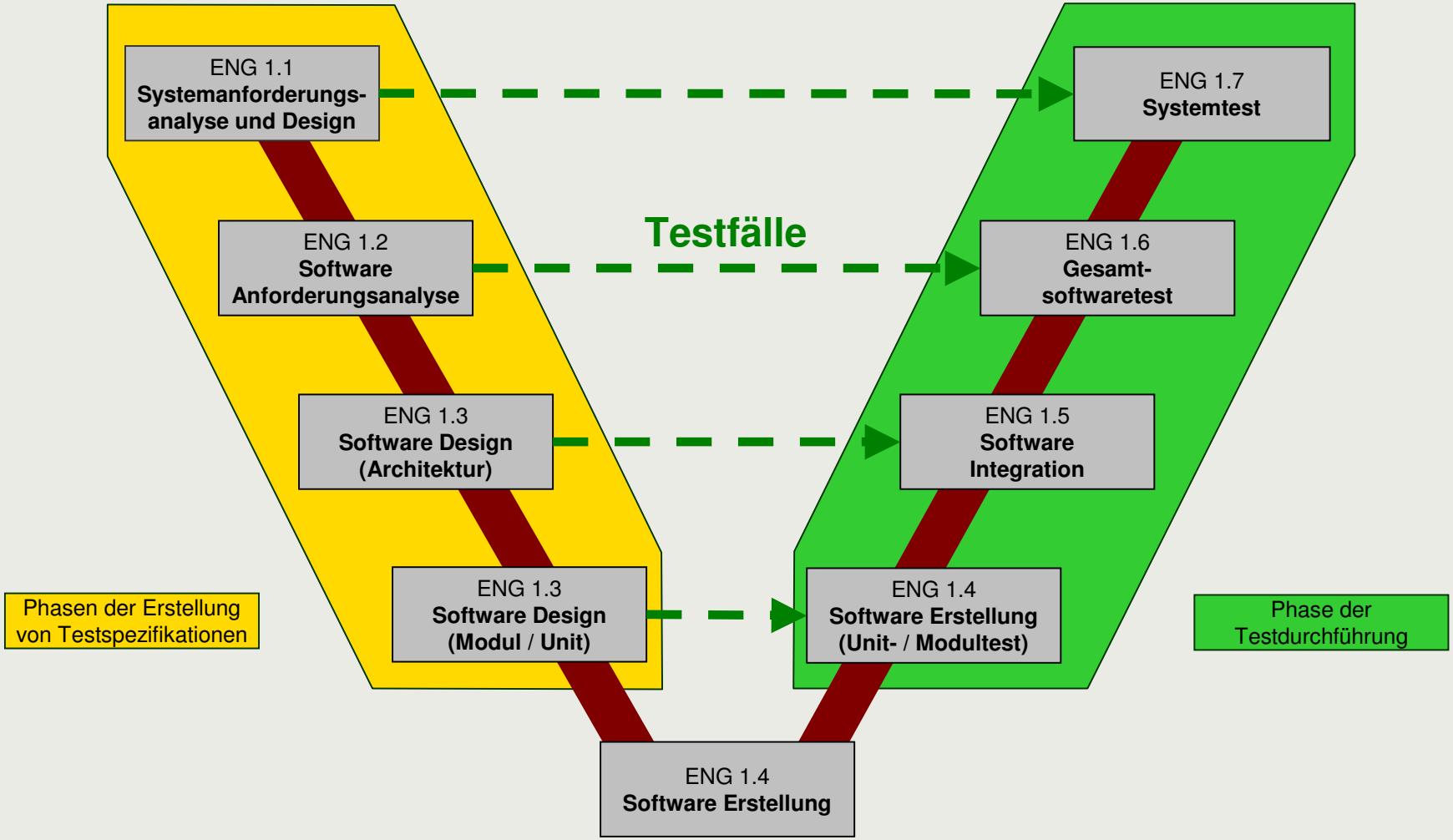


- Ergänzung der textuellen Requirements durch graphische Beschreibungen **und durch Testfälle**

Manfred Melzig/DE/2006-02-21/PBE-1SW2

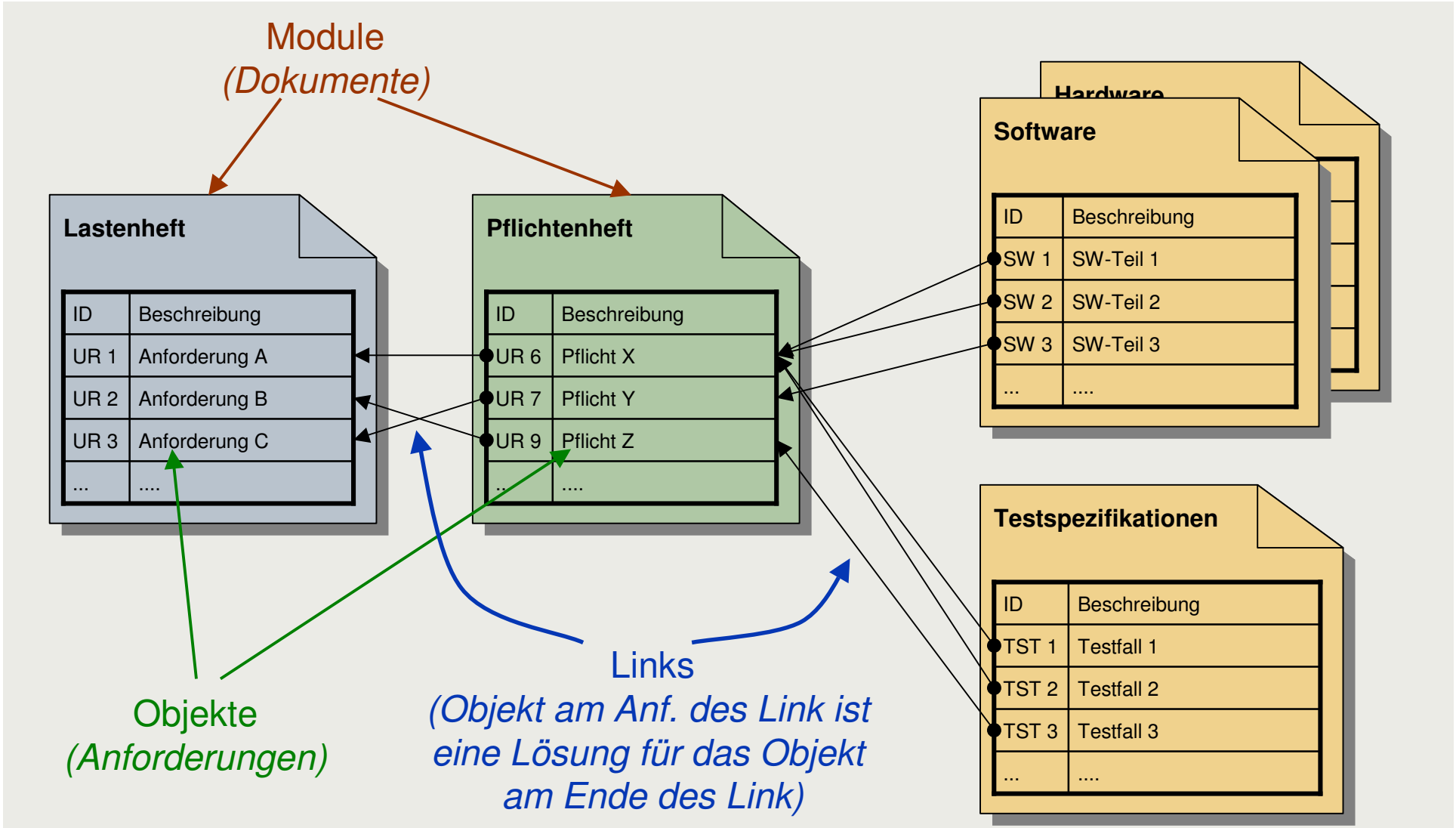
Testen von Software

Testen im SPICE basierten Software-Entwicklungsprozess



Testen von Software

Verbesserung der Testqualität: Darstellung und Verfolgung der Abhängigkeiten



Testen von Software

Testen im Spannungsfeld von Qualität – Kosten – Termine

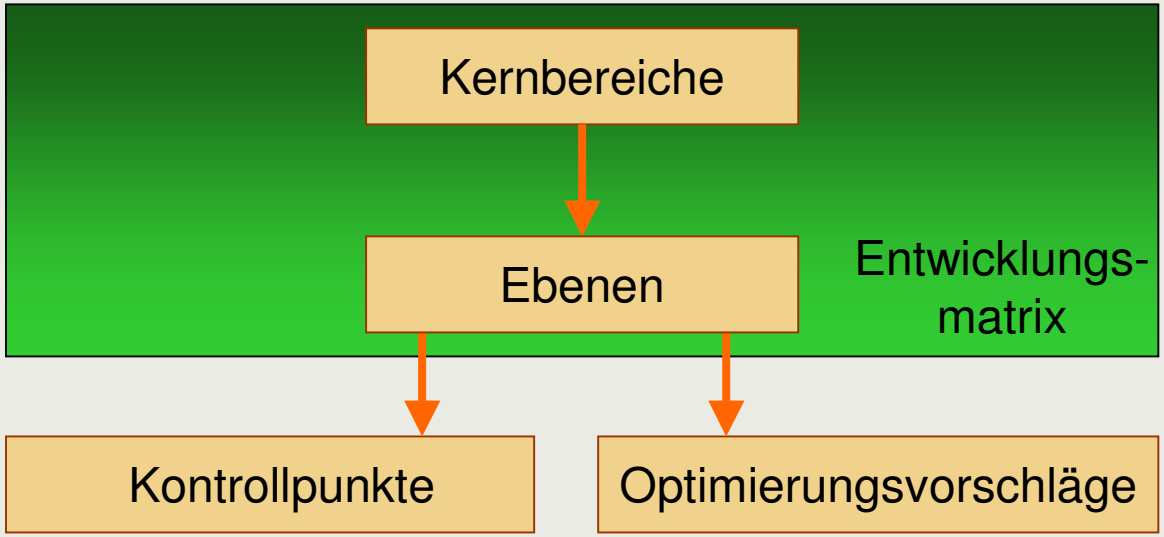
- **These Qualität:**
  Die Erhöhung der Anzahl an Steuergeräten, die zunehmende Funktions-komplexität und die Verkürzung der Entwicklungszeiten steigern das Fehlerrisiko

- **These Kosten:**
  Aufgrund der Steuergeräte Vernetzung und der steigenden Funktions-komplexität der Einzelsysteme erhöht sich der durch das Testen verursachte Kostenanteil eines Steuergerätes erheblich

- **These Termine:**
  Die kurzen Entwicklungszeiten mit den verschiedenen Musterständen setzen einen Zeitrahmen, innerhalb dem die notwendige Testtiefe nur schwer erreicht werden kann und damit Musterlieferungstermine und den SOP gefährden

Testen von Software

TPI® Automotiv - Bewertungsraster zur Optimierung der Testprozesse



Erfasst und bewertet übergreifend den aktuellen Testprozess

- 21 Kernbereiche werden untersucht und anhand verschiedener Ebenen klassifiziert
- Für eine objektive Klassifizierung existieren Kontrollpunkte für die verschiedenen Ebenen
- Durch Verwendung einer Entwicklungsmatrix wird die Performance des Prozesses und die Abhängigkeit unter den Kernbereichen verdeutlicht
- Das Modell gibt detaillierte Verbesserungsvorschläge

Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Beispiel Bewertung nach TPI®

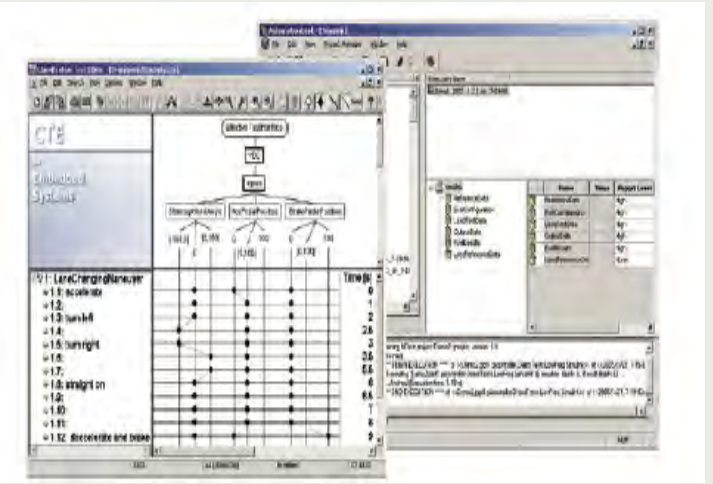
TPI® (Sogeti) Beispiel Bewertung

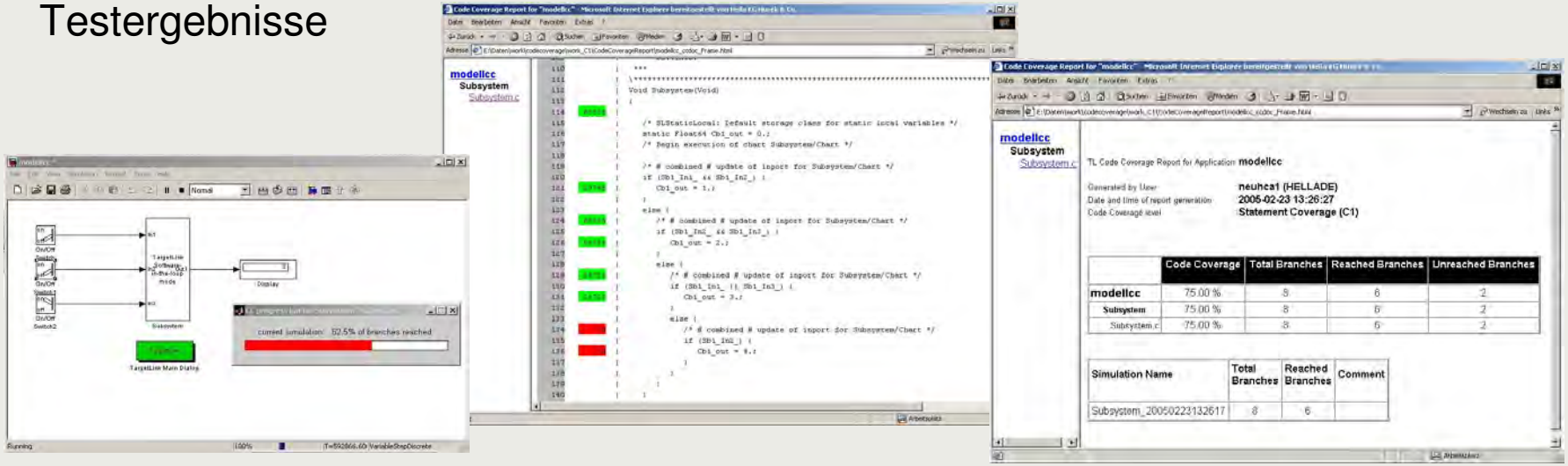
	Kernprozessfeld	kurzfristig	mittelfristig	0	1	2	3	4	5	6	7	8	9	10	11	12	13	Vorschlag für Maßnahmen
					Controlled					Efficient					Optimizing			
1	Teststrategie		B		A					B				C		D		vereinfachte Wiederholungsprüfung einführen
2	Einsatz des Phasenmodells	A	B		A			B										Testware "konservieren"
3	Zeitpunkt der Beteiligung		B			A				B				C		D		Testaktivitäten frühzeitig starten
4	Kostenvoranschlag und Planung		A				A							B				Testaufgaben quantitativ fallbezogen planen
5	Test-Spezifikation	A	B		A		B						C					Wiederverwertung der "Testware"
6	Statische Testtechniken	A	B					A		B								Entwicklung für Test - Testbarkeit prüfen
7	Metriken		A						A			B			C		D	Metriken definieren
8	Testautomatisierung	A	B				A				B			C				Automatisierung einführen (z.B. Build-über-Nacht)
9	Testumgebung	A	B				A				B						C	Testumgebung rechtzeitig aufbauen
10	Testarbeitsplatz		A				A											Testumgebung rechtzeitig aufbauen
11	Engagement und Motivation		B		A				B						C			Vollzeit-Tester und detaillierte Budgetkontrolle
12	Testfunktionen und Ausbildung		B				A			B				C				Aufgaben und Verantwortung definieren
13	Reichweite der Methodik		B					A		B				C			D	Befolgen der einzuführenden Methodik
14	Kommunikation		B			A		B							C			Abweichungen vom Testplan dokumentieren
15	Berichterstattung		B		A			B		C					D			Fehler statistisch auswerten
16	Dokumentation der Abweichungen	A	B		A				B		C							Testfälle auf Basis der Fehler pflegen
17	Testwaremanagement					A			B				C					-
18	Testprozess-Management		B		A		B								C			Aktivitäten quantitativ verfolgen
19	Prüfungen					A				B			C					-
20	Low-Level-Tests							A		B		C						-
21	Integrationstest		B				A			B				C				-

Testen von Software

MTest zum Test von modellbasiert erstellten C-Code

- Testumgebung für Simulink u. Targetlink
- Anbindung des CTE
- Automatische Generierung von Testreports
- Unterstützung der verschiedener TargetLink-Simulationsmodi
  - MIL, SIL, PIL
- Auswertung der Testergebnisse
- Unterstützung von Modell- und Code-Coverage
- Archivierung der Testspezifikation und der Testergebnisse





Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Potenzielle Analyseaspekte des Integrationstests

Unit - Integration

Modul – Integration

- Verklemmungsfreiheit (Dead Locks)
- Handshake / Protokolle
- Schnittstellen / Wertebereiche / Handling

Integration ins Betriebssystem

- Task Synchronisation / Priorisierung / Verdrängung
- Interrupt Verhalten
- Startreihenfolge, Sequenz Check für Hoch- und Runterlauf

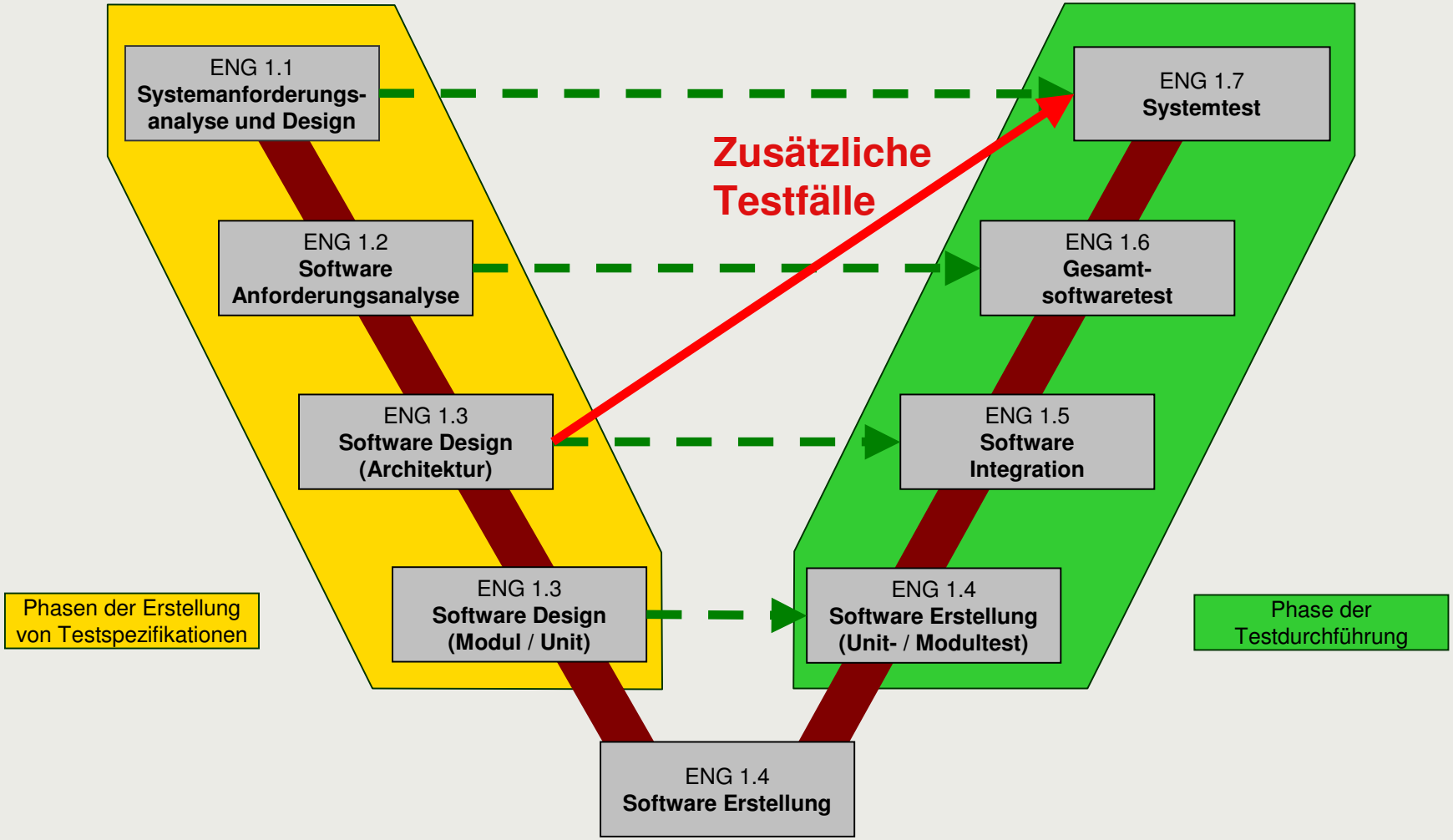
Ressourcen Aussagen

- Maximale Stacktiefe
- ROM / RAM Bedarf
- CPU-Auslastung

Hardware- / Software Integration

Testen von Software

Berücksichtigung von Design-Aspekten im Systemtest



Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Verbesserungspotenziale

Durchgängigkeit der Modellierung

- Vollständige, verifizierbare Beschreibung aller SW-Artifakte

Effizientsteigerung durch Standards

- Eine einheitliche Beschreibungsform für Testfälle würde einen Testphasen übergreifenden Austausch von Testfällen ermöglichen. Potenzielle Kandidaten:
  - Der entstehende IEEE Standard ATML
  - Testaustauschformat „TestML“ aus dem IMMOS-Projekt
  - TTCN-3
  - UML Testing Profile

Automatische Testfallgenerierung / -variation

- Generierung von Testvektoren aus Funktionsmodellen, die eine xx%-tige Pfadabdeckung sicherstellen
- (Automatische) Variation der Reihenfolge von Testfragmenten, bzw. beliebiges paralleles Aufschalten von Testfragmenten, die unterschiedliche Systemfunktionen aktivieren

Formale Verifikation

Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Formale Verifikation

- „vollständiger“ Korrektheitsnachweis für sicherheitskritische Systeme
- Verifikation von: Stateflow-Diagrammen, Subsystems
- Betrachtung **aller** möglichen Systemläufe
- Beweis durch mathematische Algorithmen
- Grundlage: TargetLink Code

EmbeddedValidator

OSC Embedded Systems,Oldenburg

Standard Analyses

- Drive-to State
- Drive-to Configuration
- Range Violation

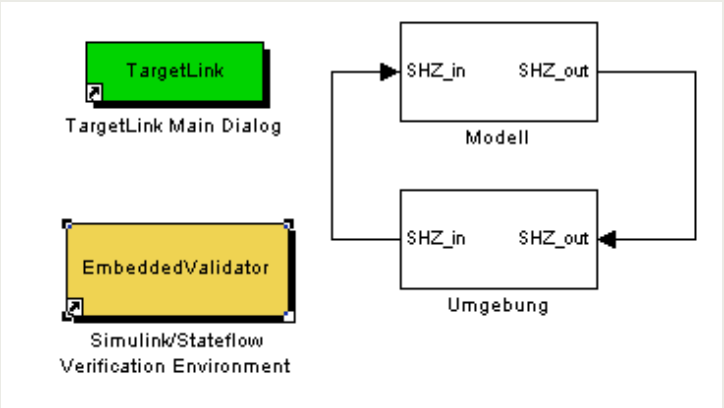
...

Pattern Specification

- „immer P“
- „Q nur nach P“

...

„true“ → Anforderung ist immer erfüllt!



Testen von Software

Resümee

Testen von Software:
Ein Nachweis der korrekten Steuergeräte-Funktionalität ?

Ja: Bezogen auf eine vollständige Abdeckung der Anforderungen durch Testfälle (mit der Einschränkung, das Tests nie vollständig sein können). Ferner, gegen welche Anforderungen?

- Kundenanforderungen (Lastenheft)
- Lieferantenzusagen (Pflichtenheft)

Nein: Wenn sich eine vollständige Abdeckung der Anforderungen aufgrund des Kostendruck (zukünftig) nicht durch halten lässt. Nachweis für „geringer wertige“ Anforderungen werden zurück gefahren.

Nein: Für nicht berücksichtigte Zusammenhänge.

Perspektive:

- Fehlersimulation hilft die Anforderungen weiter zu präzisieren
- Formale Verifikation hilft sicherzustellen, das Anforderungen unter allen Randbedingungen eingehalten werden
- (Zufällige) Variation erweitert die Testtiefe

Manfred Melzig/DE/2006-02-21/PBE-1SW2

Testen von Software

Fehlersimulation mit Spice insbesondere Intusoft‘s Icap Program

"Simulation technischer Systeme"
und
"Grundlagen und Methoden in Modellbildung und Simulation"
München, 20. - 21. Februar 2006

Herbert Müller Thomatronik GmbH

Fehlersimulation mit Spice

- Allgemeines
- Fehlermodellierung von katastrophalen Fehlern
- Parametrische Fehler, Toleranzen und Alterung der Bauteile

Warum Fehlersimulaton?

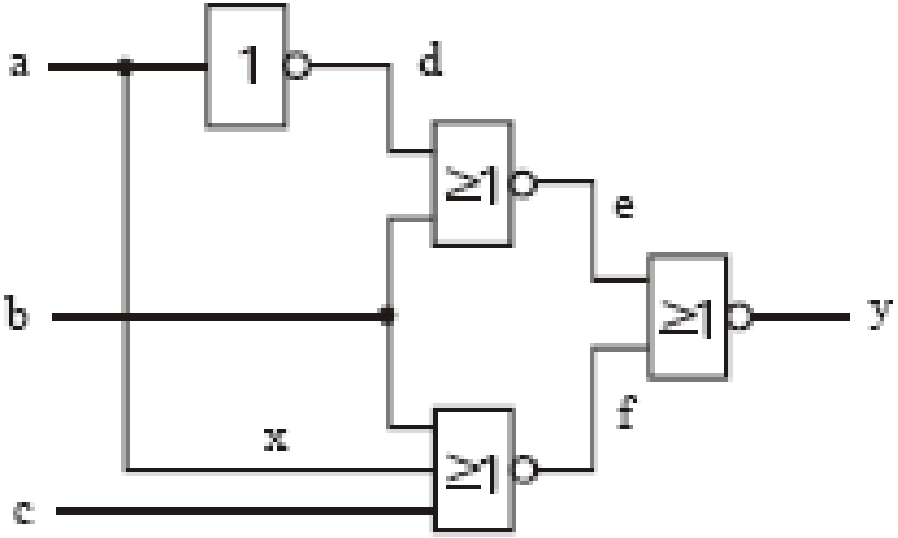
- Elektronische Schaltungen sollen ausfallsicher sein
- Die Funktionsprüfung einer elektronischen Schaltung deckt nicht 100 % Fehlverhalten auf, latente Fehler bleiben unentdeckt.
- Durch Fehlereinbau in die Schaltung können Fehler besser detektiert werden, in dem man die Schaltung auf die Wirkung der Fehler untersucht
- Fehlersimulation für Diagnose, Erstellung von Testprogrammen, eingebauten Selbsttest
- Fehlermöglichkeits- und Ereignisanalyse (FMEA) ist mit Simulation durchführbar.
- Überprüfung der Bauelemente auf Belastungen wie Ströme, Spannungen, Leistung
- Entdeckung von Fehlern beim Entwurf spart Kosten, erhöht die Ausbeute.
- Je weiter Fehler im Lebenszyklus eines Gerätes entdeckt werden desto höher sind die Kosten
- Fehlersimulation entstand ursprünglich in der Luft- und Raumfahrt, wo bei Fehlverhalten Menschenleben in Gefahr sind oder eine Reparatur nicht möglich ist.

Beispiel für latente Fehler

- Bei TTL Schaltungen mit offenem Kollektor ist der Kollektorwiderstand im allgemeinen 1 kOhm
- Einbau eines 2 kOhm Widerstand wird die Funktionsprüfung des Gatters bestehen
- Einbau dieses Gatters in ein Gerät kann durch Rauschen, Temperatur, Anstiegsflanken zu Fehlverhalten des Gerätes führen

Fehlersimulation bei Logikschaltung

- Eingang oder Ausgang liegen permanent auf „0“ oder „1“



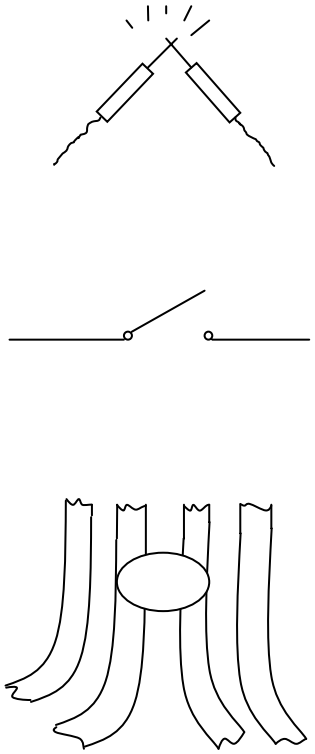
Fehlerabdeckung

$$Fehlerabdeckung = \frac{entdeckte\_Fehler}{mögliche\_Fehler} \times 100(\%)$$

Fehlersimulation mit Spice

- Zahlenbereich beim Rechner begrenzt
- Kurzschlüsse mit kleinem Widerstand modellieren
- Offenen Kontakt mit großem Widerstand modellieren
- Haftfehler (stuck at) mit kleinem Widerstand modellieren
- Parametrische Fehler durch Monte Carlo Analyse, Extremwertanalyse, Sensivitätsanalyse, Worst Case mit Sensitivitätsanalyse, quadratische Mittelwerte

Fehler bei analogen Schaltungen



- Katastrophale Fehler
- Kurzschluss
- offener Kontakt, Unterbrechung
- Überbrückungsfehler z.B. Leiterbahnen bei integrierten/gedruckten Schaltungen
- Haftfehler, Knotenpunkte liegen auf einem bestimmten Potential
- **Parametrische Fehler wie Toleranzen, Alterung**
- Leitungsgebundene Fehler Reflexionen, Übersprechen, Einstrahlungen (EMV)

Fehlerliste nach CASS
USA Navy 34 Fehler

Item	Part	Failure Mode
1	Capacitor	Short Open
2	Circuit Breaker	Mechanical Failure Stuck Open Stuck Closed Failure to open at the „Must open“ point
3	Diode-Rectifying	Short Open
4	Diode-Regulating	Short Open Change in Vz beyond tolerance
5	Diode-Protection	Short Open
6	Diode Switch Two Terminals	Short between leads Open between leads

7	Diode Switch Three or more Terminals (SCRs, Triac, etc.)	Gate Short Gate Open Cathode-Anode Short Cathode-Anode Open
8	Diode – Suppression	Short Open
9	Fuse	Open
10	Inductor	Short Open
11	Logic Devices	Outputs Stuck High Outputs Stuck Low Inputs Stuck High Inputs Stuck Low Inputs Open Power Shorted to Ground
12	Resistor	Open
13	Relay	Shorted Coils Open Coils Stuck Contacts Open Contacts
14	Switch-Rotary	Open Wiper High Resistance Contact
15	Switch-Lever or Push Button	Open normally closed /NC) Contacts Stuck Contacts

16	Transformer	Shorts between Normally Non-Continuous Windings Shorts to Ground Transformer Frame Open Primary Windings Open Secondary Windings
17	Transistor bipolar	Base Emitter Short Base Emitter Open Collector Emitter Short Collector Open
18	Transistor-Field Effect JFET/IGFET/MOSFET	Gate Shorted to any other Terminal Source Drain Shorted Source Drain Open Gate Open
19	Linear Integrated Circuit	Outputs Stuck High or at Positive Vcc Outputs Stuck Low or at Negative Vcc Outputs Open
20	Switch-Thermostatic	Continuity at UUT Operating Temperature Open at UUT Operating Temperature
21	Resistor Network	Open
22	Voltage Regulators	V(in) – V(out) Short V(in) – Ref. Short V(out) – Ref Short Ref. Open
23	Photo Diode	Short Open

24	Photo Transistor	Emitter Collector Short Emitter Collector Open
25	Hybrid Device	Combination of linear and digital failure modes appropiate for internal structure of hybrid device. Failure modes obtained from hybrid designers
26	FET Analog Switch	Control Inputs Stuck High Control Inputs Stuck Low Drain Source Short Drain Source Open
27	Resistor Variable	Open Stuck Wiper Out of Tolerance
28	Delay Line	Short Open
29	Motors – Drive and Servo	Shorted Windings Open Windings Binding
30	Vacuum Tube	Open Filament Open Cathode Open Control Element
31	Cathode Ray Tube	Failure Modes obtained from vacuum tube designer
32	Thermistors	Open
33	BIT Indicators	Open Proper Operation

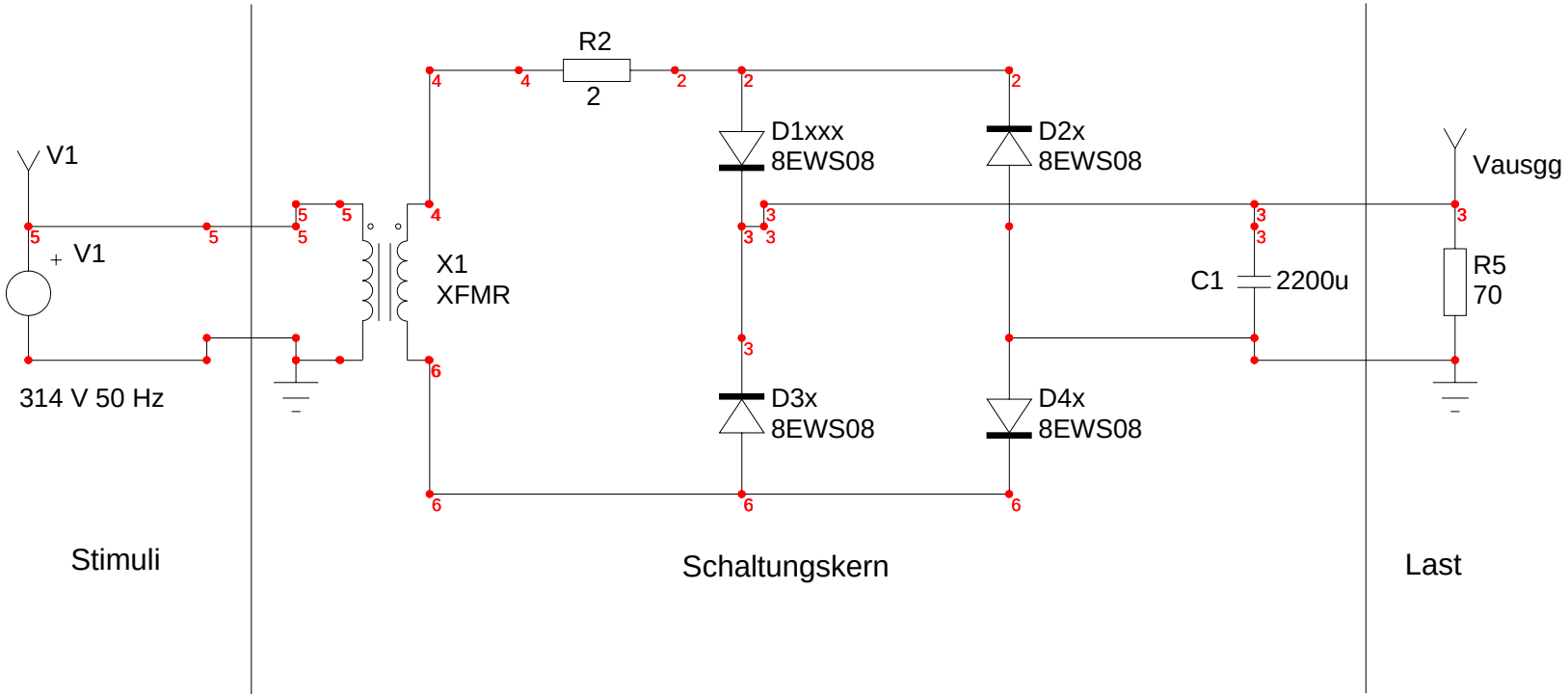
34	Elapsed Time Indicator	Shorted Coil Open Coil Mechanical Jamming
35	Other	Other Failure Modes available from component engineering design data or identified by component manufacturer

Die meisten dieser Fehler sind in Icap integriert und durch den Schaltplaneditor Spicenet wählbar

Fehlersimulation mit Icap

- Fehlereingabe im Schaltplaneditor
- Schaltplaneditor hat mehrere Ebenen
- Aufteilung der Schaltung z.B. Stimuli in eine Ebene, Kernschaltung in eine andere Ebene, Lasten in eine weitere Ebene
- Ebenen für die Simulation zusammenschalten (Konfigurieren) und Analysen (Arbeitspunkt, Zeit-/Frequenzbereich in Setup bestimmen
-

Beispiel für Schaltebenen



Fehlersimulation in Icap

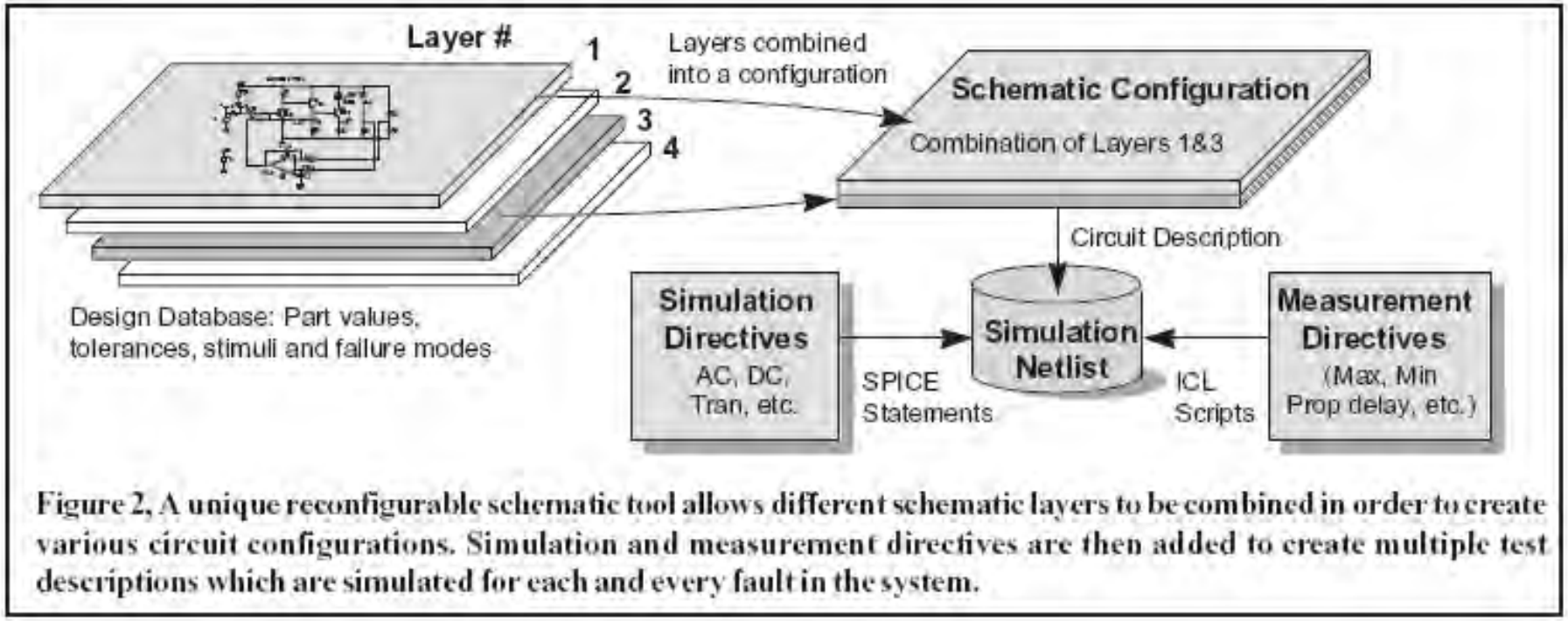
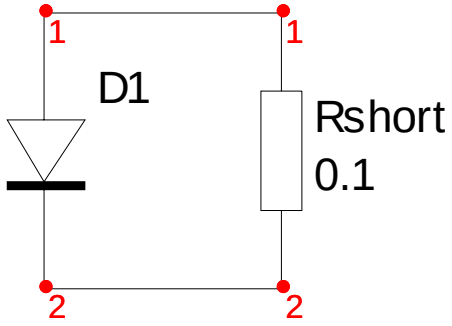


Figure 2, A unique reconfigurable schematic tool allows different schematic layers to be combined in order to create various circuit configurations. Simulation and measurement directives are then added to create multiple test descriptions which are simulated for each and every fault in the system.

- Der Schaltplan wird in mehrere Ebenen aufgeteilt z. B. Ebenen für Stimuli, Kern der Schaltung, Lasten,
- Ebenen werden zur Simulation zusammen geschaltet, konfiguriert
- Analysearten (AC, Tran, OP, Monte Carlo, EVA, RSS WorstCase, WorstCase Sensitivity) definiert
- Simulationsergebnisse max, min, Verzögerungen, Anstiegs-/Abfallflanken etc…. definieren
- Definierte Fehler werden zur Spice Netzwerkliste zusammengeführt

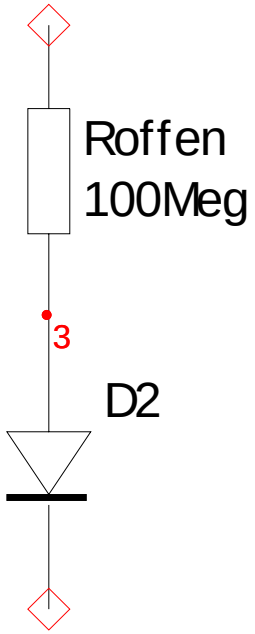
Fehlermodulierung

Kurzschluss Anode
Kathode



Fehler Anode offen

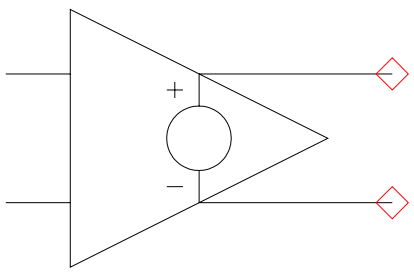
Anode Offen



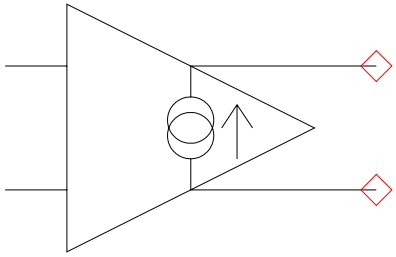
Das Berkeley B Element für die Modellierung von Fehlern

- Nichtlineare gesteuerte Strom-/Spannungsquellen für Haftfehler (Stuck at), Überbrückungsfehler, zeitabhängige Fehler, if then else Konstrukte
- B2=N+ N- V=V(9,8)*V(12)
- BI=N+ N- I=V(5,8)*100*V(10)/(V(8)+V(12)

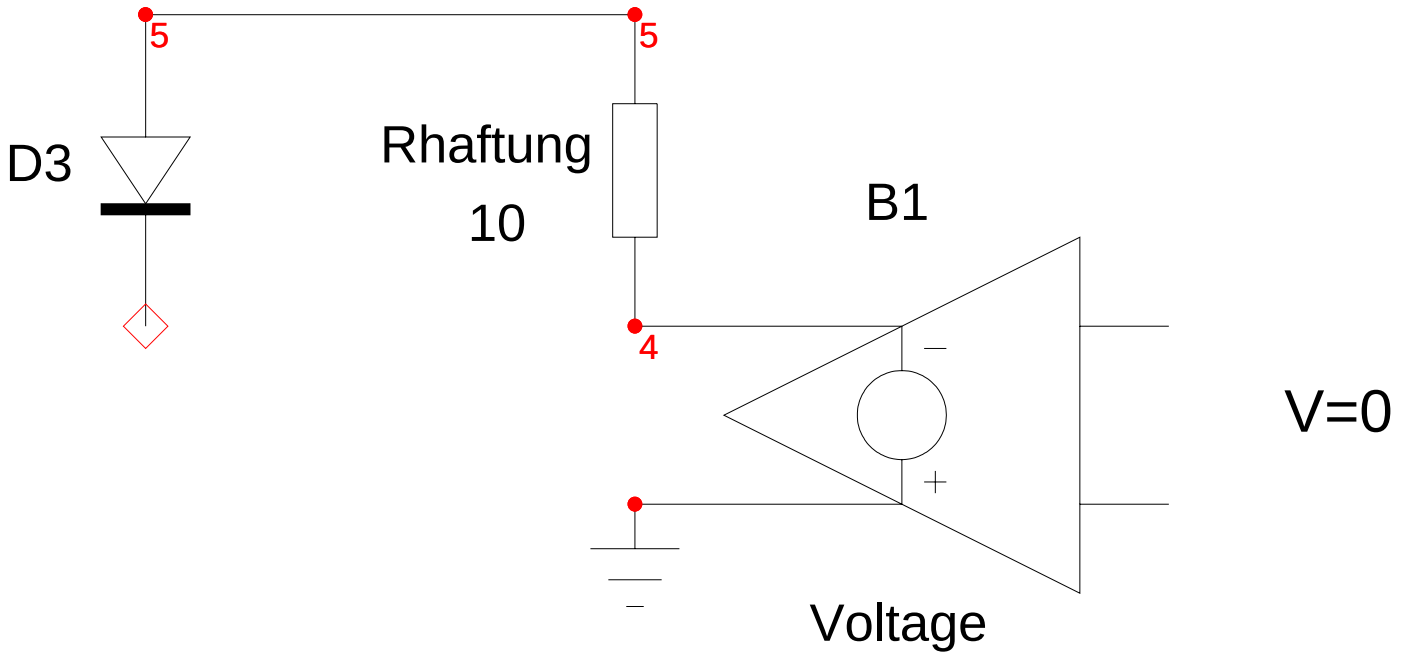
B2



B1



Haftfehler Anode auf Masse

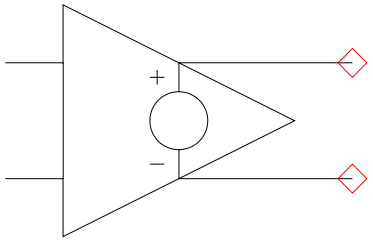


Erweitertes B Modell
Logische Verknüpfungen

- NAND Gatter mit 3 Eingängen
- V=~(V(1) & V(2) & V(3))
- ~ Not Operator, & And Operator

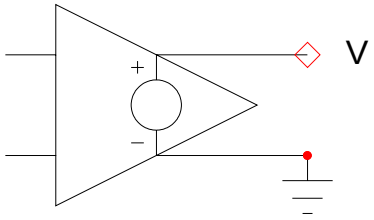
- Die 2 Striche auf der linken Seite sind symbolisch, keine Eingänge

B1

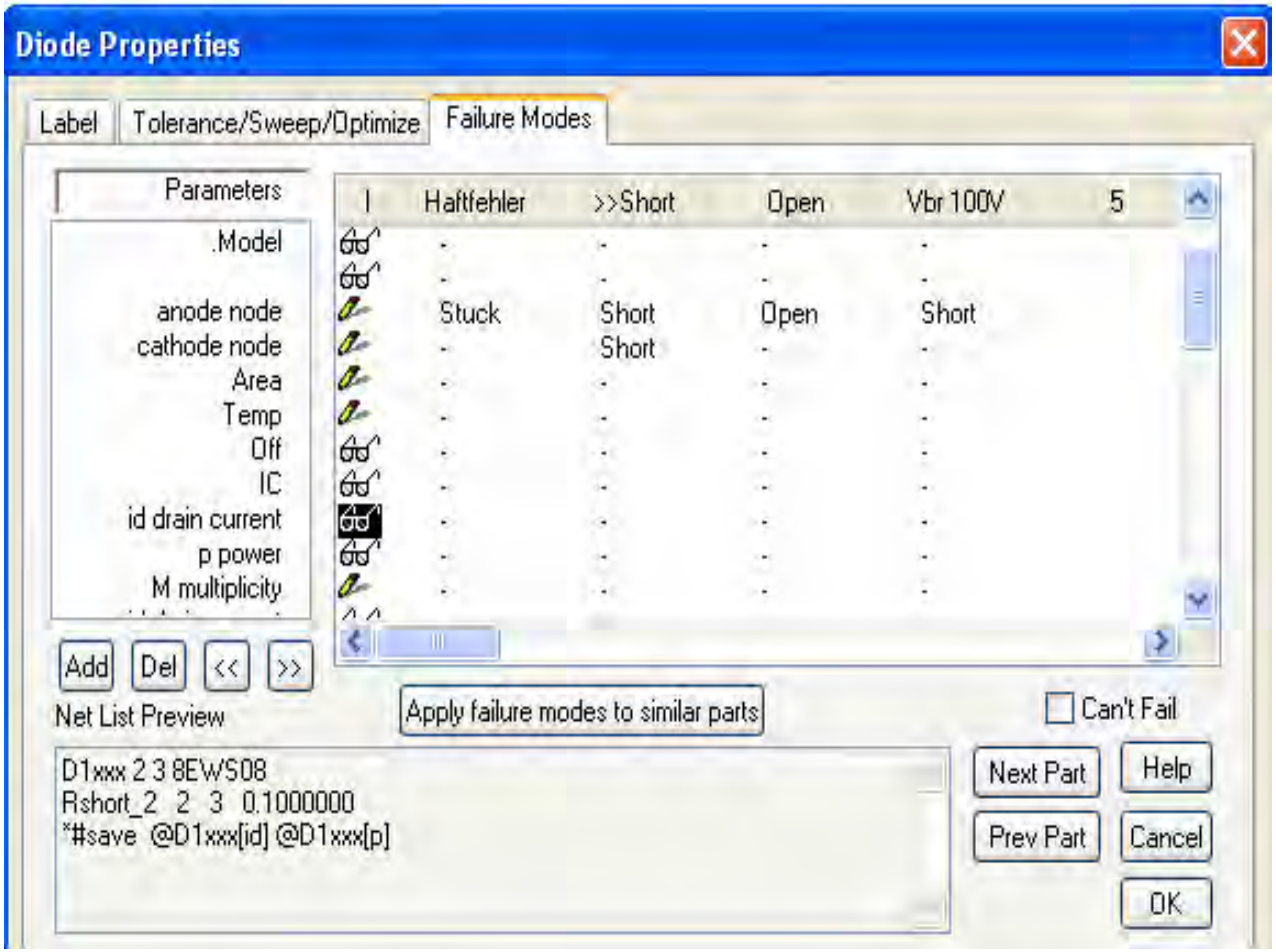


Erweitertes B Model zeitabhängig

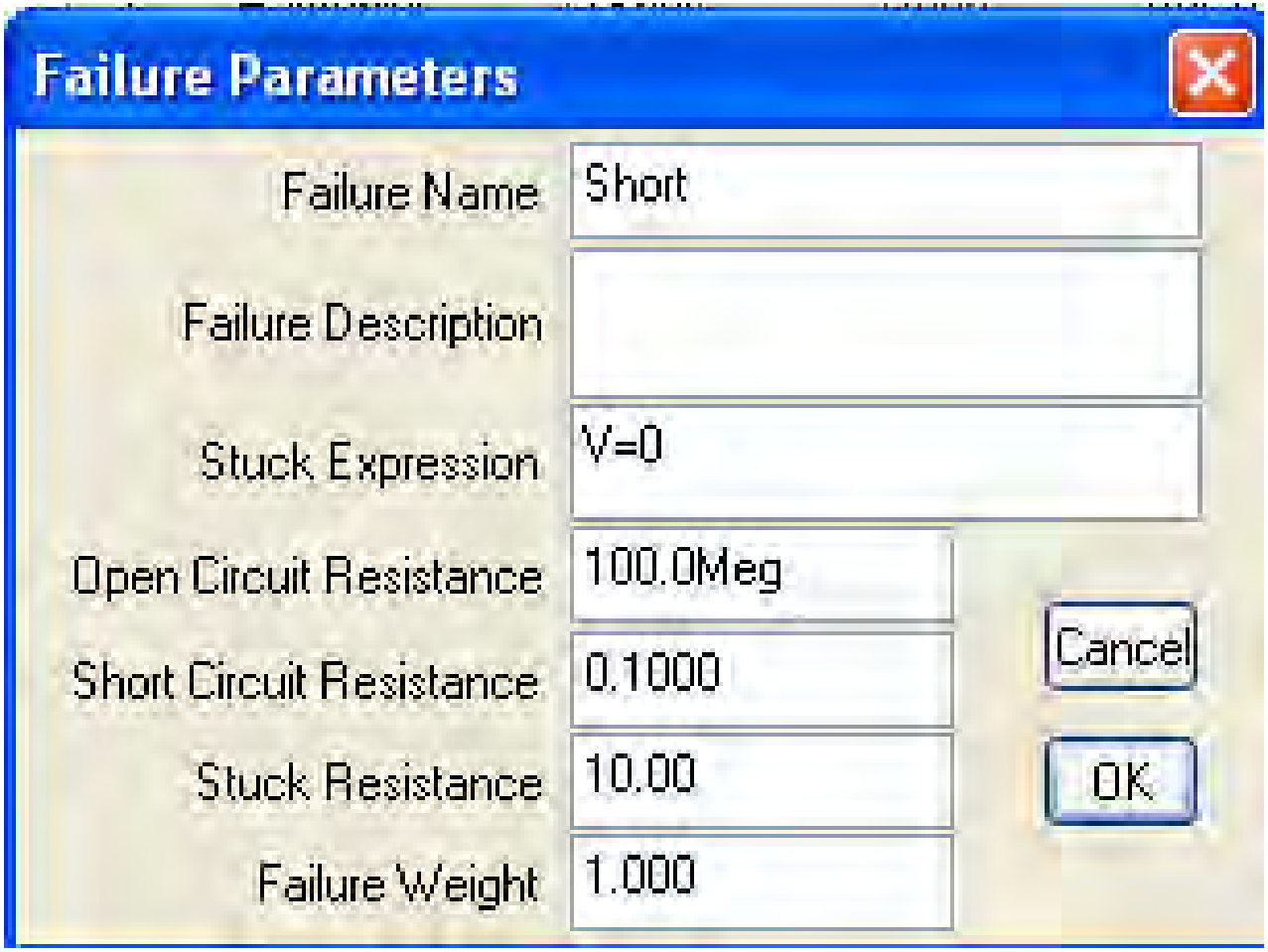
- V=time > 10ns ? V(2) : (V)1
- Nach 10ns ist V= V(2) sonst V(1)



Fehler Anode Kathode Kurzschluss



Fehlermodellierung, B Element (V= oder I=), Gewichtung des Fehlers



Parametrische Fehler
Toleranzen, Alterung

- Monte Carlo Analyse
- Sensitivitätsanalyse
- Extremwerte
- Quadratische Mittelwerte (RSS)

Monte Carlo Analyse 1

- Ein Zufallsgenerator variiert die mit Toleranzen behafteten Bauteile/Parameter
- Anzahl der Simulationen für Lot (korrelierte Toleranzen, z.B. auf einem Wafer) und Case für diskrete Bauteile
- Messwerte definieren, z.B. Ausgangsspannungen, Ströme, Verlustleistungen, Bandbreite bei Filtern, Anstiegs-/Abfall Flanken

Monte Carlo Analyse 2

- Ausgabe: Liste aller Simulationen mit den Toleranzen und definierten Messergebnissen
- Histogramm
- Kumulierte Wahrscheinlichkeitswerte mit Sigmas auf der X-Achse
- Bei Normalverteilug ist die Steigung der Geraden die Standardabweichung

Monte Carlo Analyse 3

- Mit dem Cursor Datenpunkte durchfahren. Durch Klick auf den Report Button erhält man die Parameter und die Sigma Abweichung.
- Für die Trennung von Werten bei bimodaler Verteilung

Sensitivitätsanalyse

- Simulation mit Nennwerten
- Simulation mit 3 Sigma Toleranzwerten
- Die Differenz der beiden Werte, die Sensitivität wird in der *.out Datei abgelegt

Extremwertanalyse (EVA)

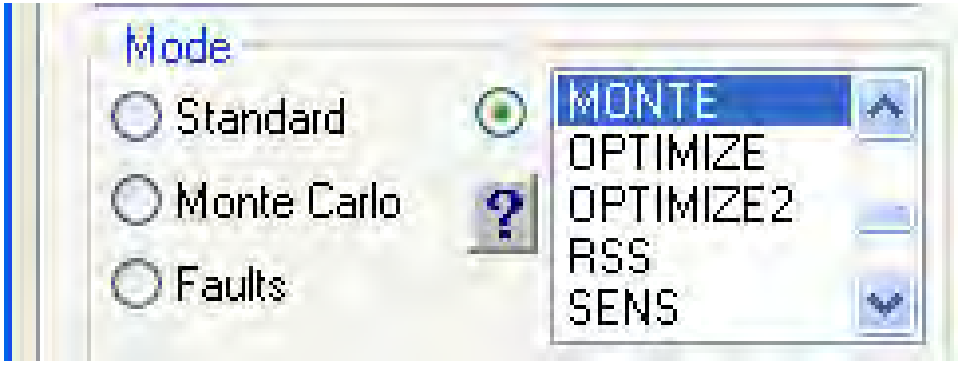
- Erst Simulation mit Nennwerten
- Dann eine Simulation mit extremen Toleranzen mit Berücksichtigung des Vorzeichens einer vorgeschalteten Sensitivitäts Analyse.
- Anschließend Simulation mit diesen Extremwerten und ablegen der Werte in der *.out Datei

Quadratische Mittelwerte (RSS)

- Nach einer Simulation mit Nennwerten erfolgt eine Simulation mit Toleranzen
- Die Differenz wird quadriert und in der *.out Datei abgelegt

Zur Simulation mit Toleranzen in Icap

- Schablonen (Templates) sind für Monte Carlo, Sensitivität, Extremwerte (EVA), Quadratische Mittelwerte (RSS) in Icap integriert



Schablonen (Templates)

- Der Anwender muß die Messungen spezifizieren, z.B. maximale Ströme, Spannungen, Leistungen, Bandbreite bei Filtern oder Anstiegs-/Abfall Flanken

Woher Fehler nehmen?

- Ausfallstatistiken
- Erfahrungen
- Zuverlässigkeitsangaben der Bauelemente Hersteller

arsenal research
Ein Unternehmen der Austrian Research Centers.

Modellierung von elektrischen und mechanischen Fehlern bei elektrischen Antrieben

Dr. Christian Kral

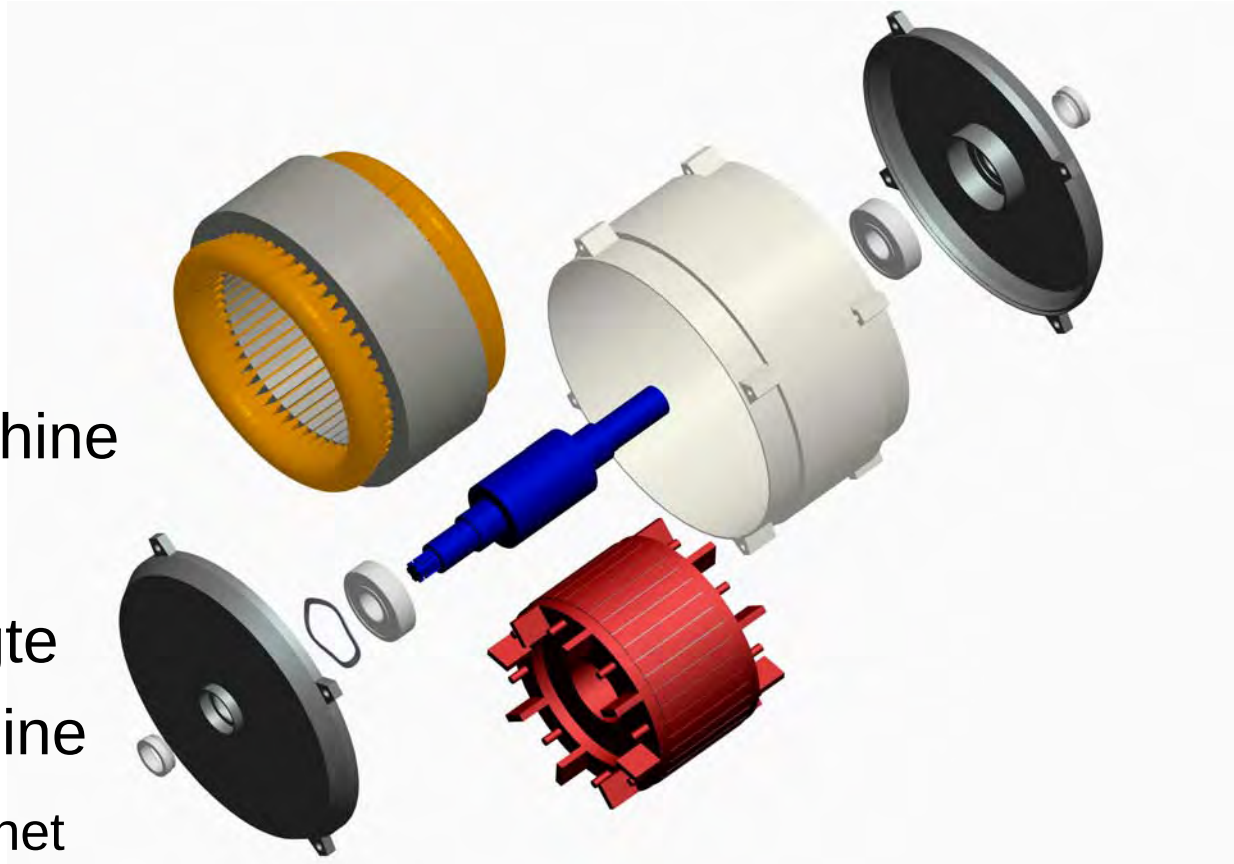
DI Franz Pirker, MSc

Motivation

- Entwicklung von Diagnoseverfahren
  - On-line Überwachung
  - Predictive Maintenance
  - Qualitätssicherung in der Produktion
- Analyse von Fehlerauswirkungen
  - Fehlerindikatoren
  - Fehlertolerante Verfahren
  - Notprogramme

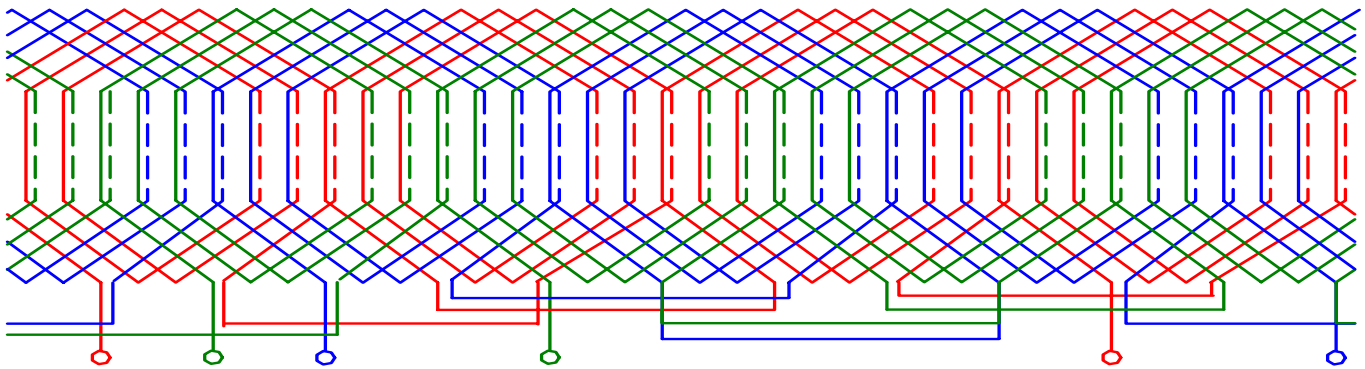
Fehlerarten bei elektrischen Antrieben

- Allgemein
  - Statorwicklung
  - Unwuchten und Exzentrizitäten
- Asynchronmaschine
  - Rotorwicklung
- Permanenterregte Synchronmaschine
  - Permanentmagnet



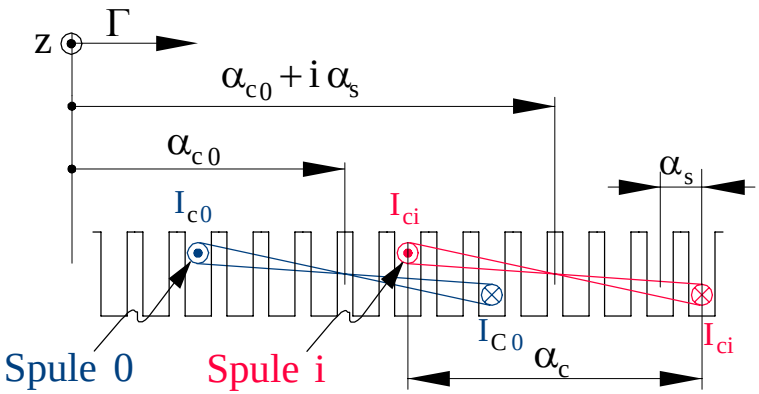
Modellierung der Statorwicklung

- Zusammenschaltung mehrerer Spulen zu einer Wicklung
- Wicklungstopologie von Drehstromantrieben
  - Drei Phasen
  - Serien- und Parallelschaltung
  - Windungszahl



Modellierung der Statorwicklung

- Wicklungsungstopologie ↔ Zonenwicklungsfaktor



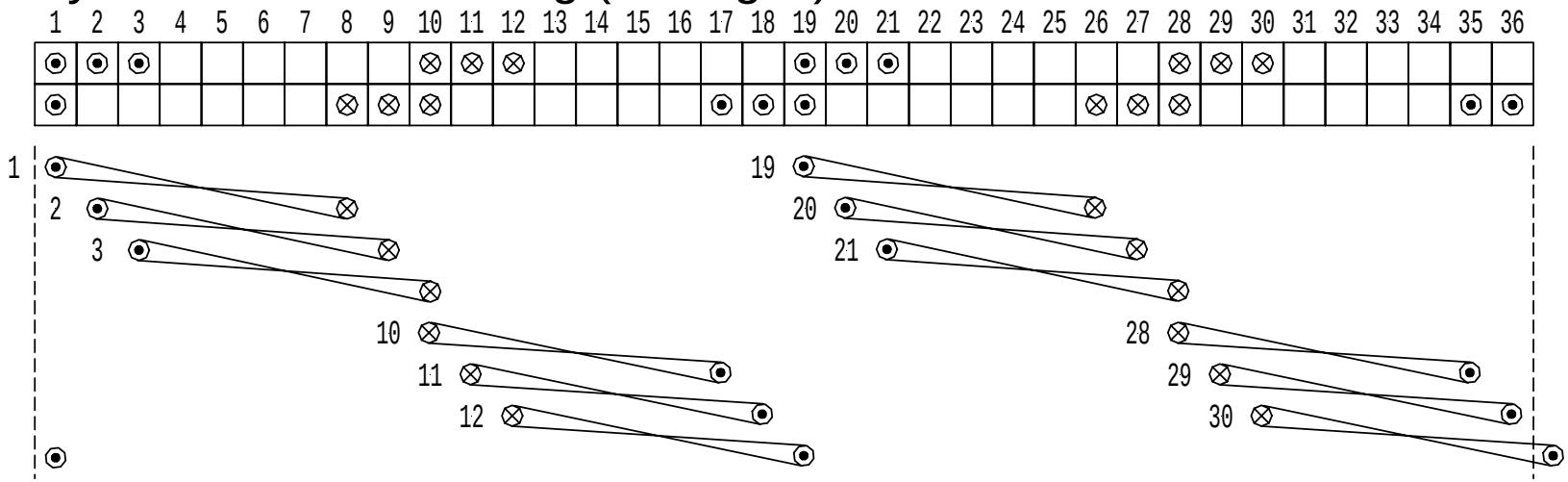
$$\xi_{zi} = \frac{1}{w_s} \sum_{m \in S_i} e_m w_{cm} e^{-jp(\alpha_{c0} + m\alpha_s)}$$

m	Spulenindex
e_m	Richtungsvariable $(+1/-1)$
w_{cm}	Windungszahl der Spulen
w_s	Windungszahl des Stranges
α_s	Zahnabstand [rad]
S_i	Menge der Spulenindizes

- Spulenweite ↔ Sehnungsfaktor

Modellierung der Statorwicklung

- Symmetrische Wicklung (Strang U)



- Definition der Parameter
  - `ys[1,:] = [ 1, 2, 3,19,20,21,10,11,12,28,29,30]`
  - `e[:,1] = [+1,+1,+1,+1,+1,+1,-1,-1,-1,-1,-1,-1]`
  - `Wcs[:,1] = [12,12,12,12,12,12,12,12,12,12,12,12]`

Statorspannungsgleichungen

- Berücksichtigung der Statorstreuung $L_{s\sigma}$
- Strang n

$$U_{sn} = R_s I_{sn} + L_{s\sigma} \frac{dI_{sn}}{dt} + \frac{d}{dt}\left(\sum_{i=0,1,2} L_{sn,si} I_{si} + \sum_{i=0}^{N_r-1} L_{sn,ri} I_{ri} \right)$$

- Gegenseitige Induktivitäten (Grundwelle)

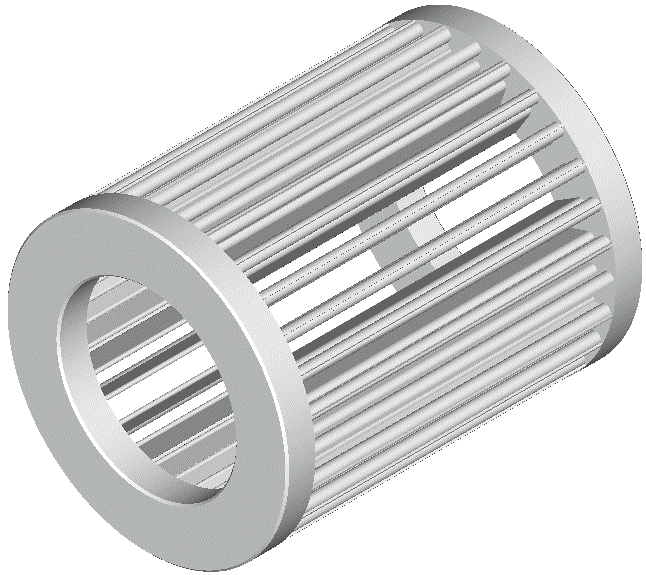
$$L_{sn,ri} = \frac{2\mu_0 D l}{\pi d} w_s \frac{1}{p^2} \xi_{ss} \xi_{sr} \xi_w \,\mathrm{Re}(\xi_{zn} e^{j\,pm\alpha_r} e^{j\,p\Gamma_m})$$

- Spannungsgleichung

$$U_{sn} = R_s I_{sn} + L_{s\sigma} \frac{dI_{sn}}{dt} + \sum_{i=0,1,2} L_{sn,si} \frac{dI_{si}}{dt} + \sum_{i=0}^{N_r-1} L_{sn,ri} \frac{dI_{ri}}{dt} + \Omega_m \sum_{i=0}^{N_r-1} Y_{sn,ri} I_{ri}$$

Modellierung des Rotorkäfigs

- Weite einer Rotormasche $\alpha_r = \frac{2\pi}{N_r}$
- Sehnungsfaktor $\xi_{sr} = \sin(p\,\alpha_r / 2)$
- Schrägungsfaktor $\xi_{w,p} = \frac{\sin(p\,\alpha_w / 2)}{p\,\alpha_w / 2}$



- Rotorinduktivitäten sind unabhängig von der Schrägung
- Spannungsgleichungen analog zu Stator

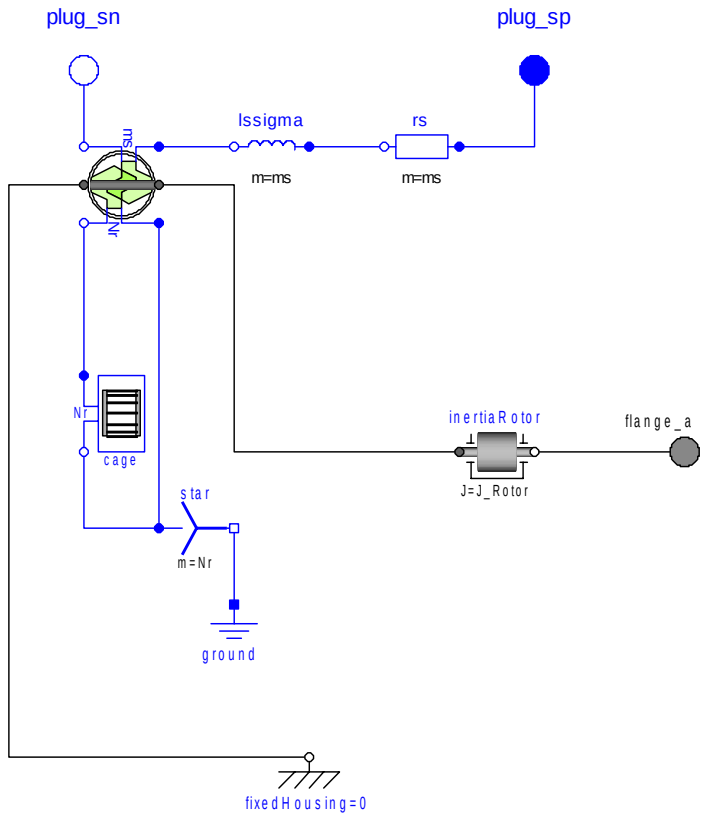
Gleichungssystem

- Statorspannungsgleichungen
- Rotorspannungsgleichungen
- Drehmomentgleichung

$$M_i = \sum_{n=0,1,2} \sum_{i=0}^{N_r-1} I_{sn} I_{ri} Y_{sn,ri}$$

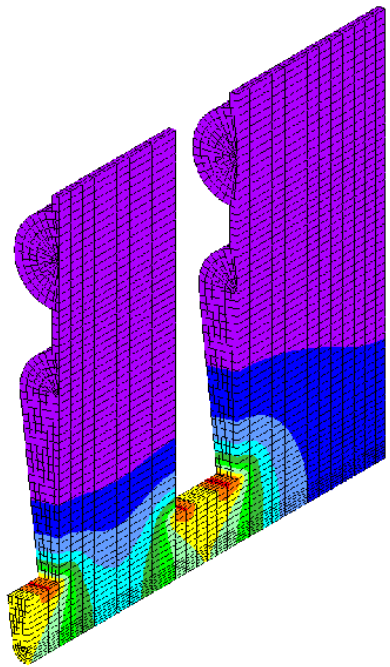
- Drallsatz

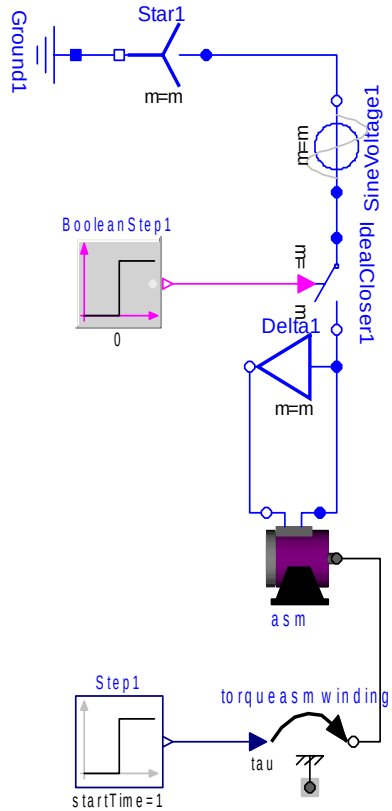
$$\Theta \frac{d\,\Omega_m}{dt} = M_i - M_l$$



Simulation von elektrischen Asymmetrien

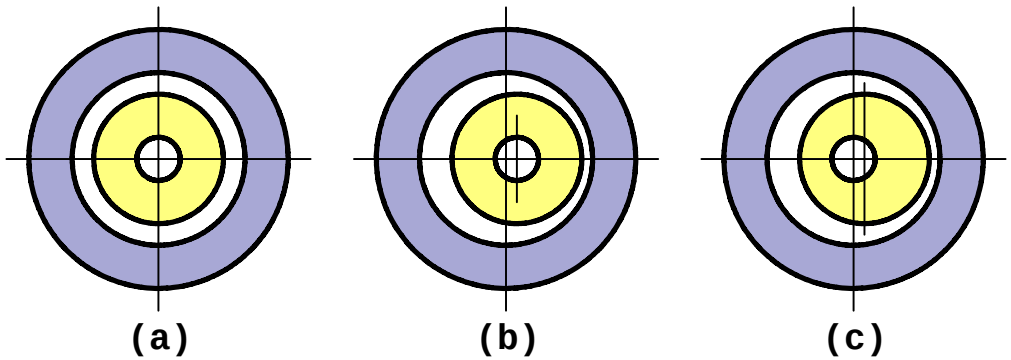
- Asymmetrische Statorwicklung
  - Windungsschluss
  - Phasenschluss
- Asymmetrischer Rotorkäfig
  - Defekter Rotorstab
  - Defekter Endring
- Weiters
  - Schleifringläufer
  - Oberwellen





Modellierung von Exzentrizitäten

- Arten von Exzentrizitäten



a) Zentrischer Rotor

b) Statische Exzentrizität (versetzte Rotationsachse des Rotors)

c) Dynamische Exzentrizität (gleiche Rotationsachsen)

Modellierung von Exzentrizitäten

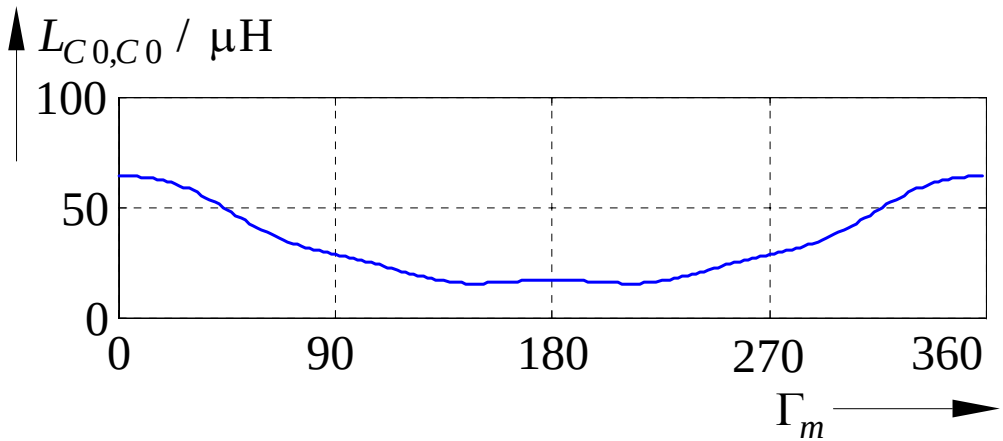
- Oberwellenmodell
- Approximation der inversen Luftspaltfunktion $\lambda = \frac{1}{\delta} = \frac{1}{1 - \varepsilon \cos(\Gamma - \Gamma_e)}$

$$\begin{aligned}\lambda(\Gamma, \Gamma_m) = \frac{1}{2} \sum_{v=0}^{\hat{v}} \sum_{w=0}^{\hat{w}} & ((\lambda_{avw} + \lambda_{evw}) \cos(v\Gamma - w\Gamma_m) \\ & + (\lambda_{avw} - \lambda_{evw}) \cos(v\Gamma + w\Gamma_m) \\ & + (\lambda_{dvw} - \lambda_{bvw}) \sin(v\Gamma - w\Gamma_m) \\ & + (\lambda_{dvw} + \lambda_{bvw}) \sin(v\Gamma + w\Gamma_m))\end{aligned}$$

- Zweidimensionale Approximation der Induktivitäten durch Fourierreihe

Modellierung von Exzentrizitäten

- Pre-Processing
  - Approximationskoeffizienten $\lambda_{avw}, \lambda_{bvw}, \lambda_{dvw}, \lambda_{evw}$
  - Vorab-Berechnung und Speicherung der Induktivitäten als Funktion des Verdrehwinkels (Effizienz)



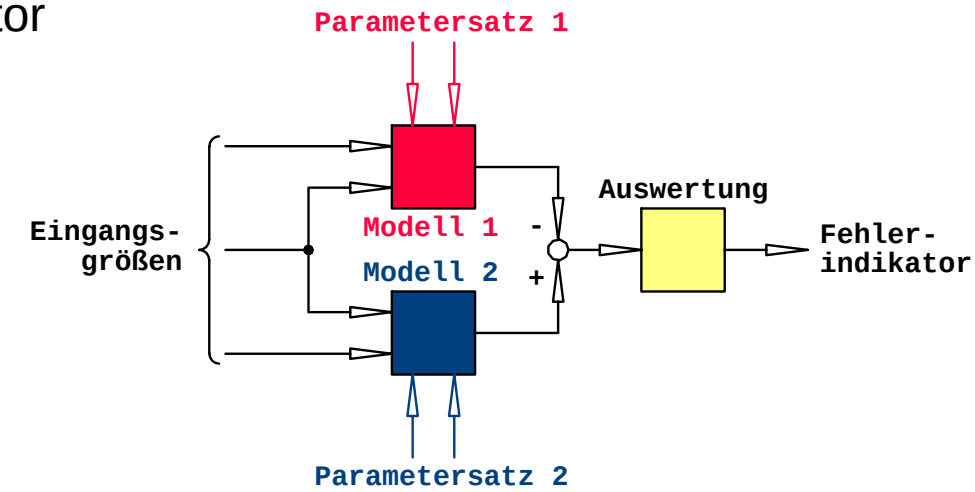
  - Berechnung und Speicherung der Ableitungen der Induktivitäten

Modellierung von Exzentrizitäten

- Pre-Processing
  - MATLAB
- Simulation eines exzentrischen Kurzschlusskäfigläufers
  - DYMOLA ↔ Modelica
  - Einlesen der Induktivitäten und deren Ableitungen
  - Filtern und abtasten der simulierten Ströme und Spannungen
- Post-Processing
  - MATLAB

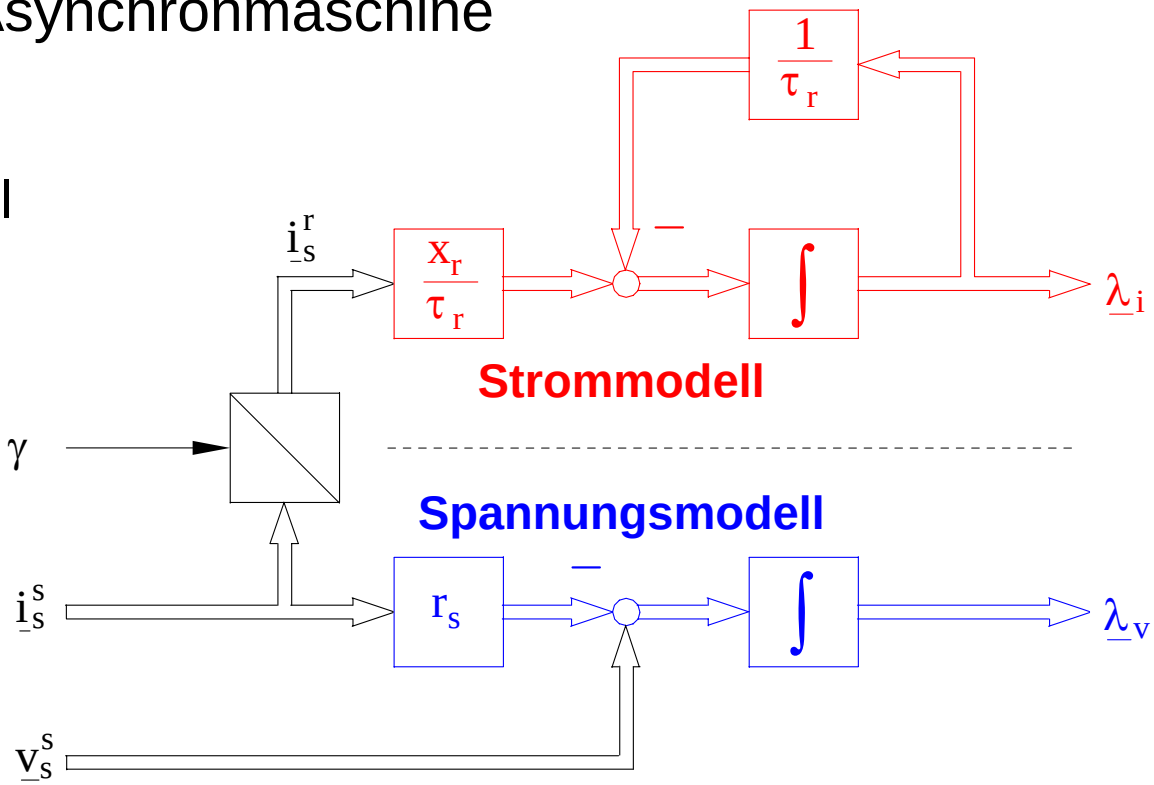
Model Based Monitoring

- Model Based Monitoring
  - Mathematische Modelle zur Berechnung von Zustandsgrößen und Fehlerindikatoren
  - Robuster und zuverlässiger als konventionelle Verfahren
  - Quantifizierbarer Fehlerindikator
- Fehlerarten
  - Exzentrizität / Imbalance
  - Rotorfehler
  - Statorfehler
  - Lagerfehler (in Entwicklung)



Detektion von elektrischen Rotorasymmetrien

- Zwei Modelle der Asynchronmaschine
  - Strommodell
  - Spannungsmodell
- Parameter
- Berechnung der Flussverkettungen
- Berechnung der Drehmomente

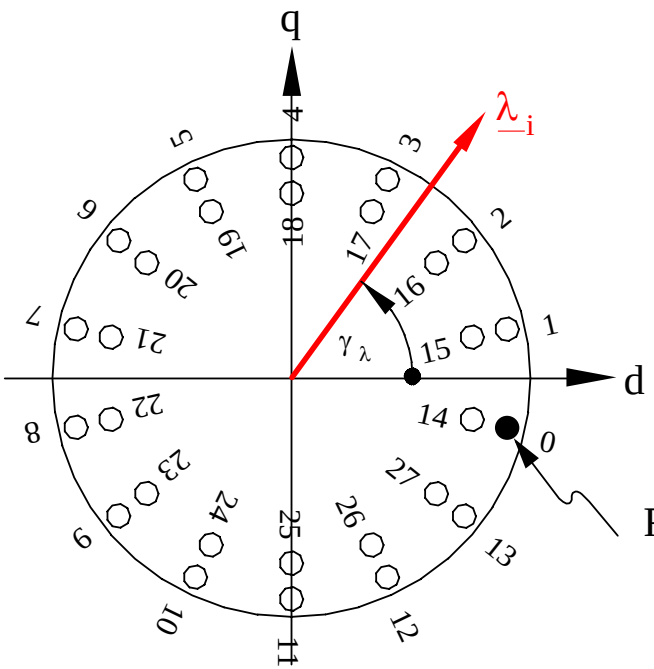


Detektion von elektrischen Rotorasymmetrien

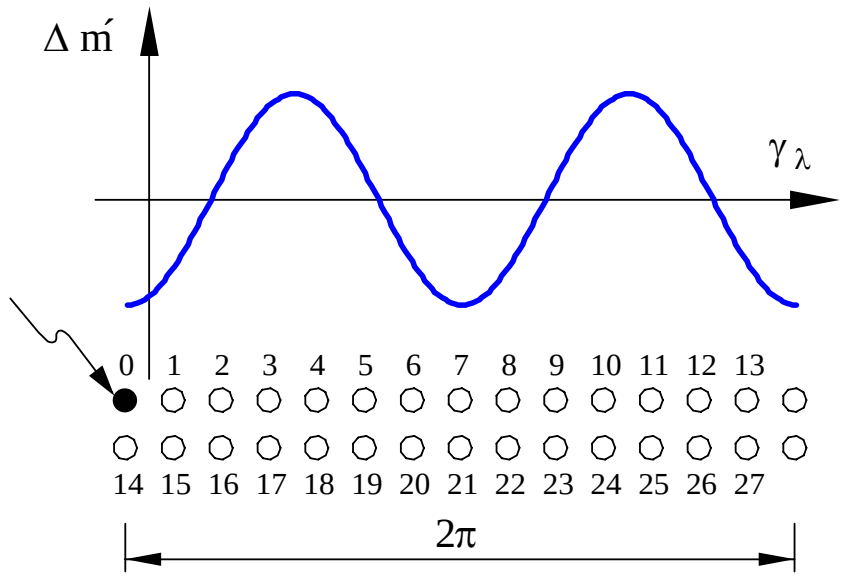
Fehlererkennung:

- Drehmomentdifferenz $\Delta m = m_v - m_i$
- relative Drehmomentdifferenz

$$\Delta m' = \Delta m / m_{Last}$$

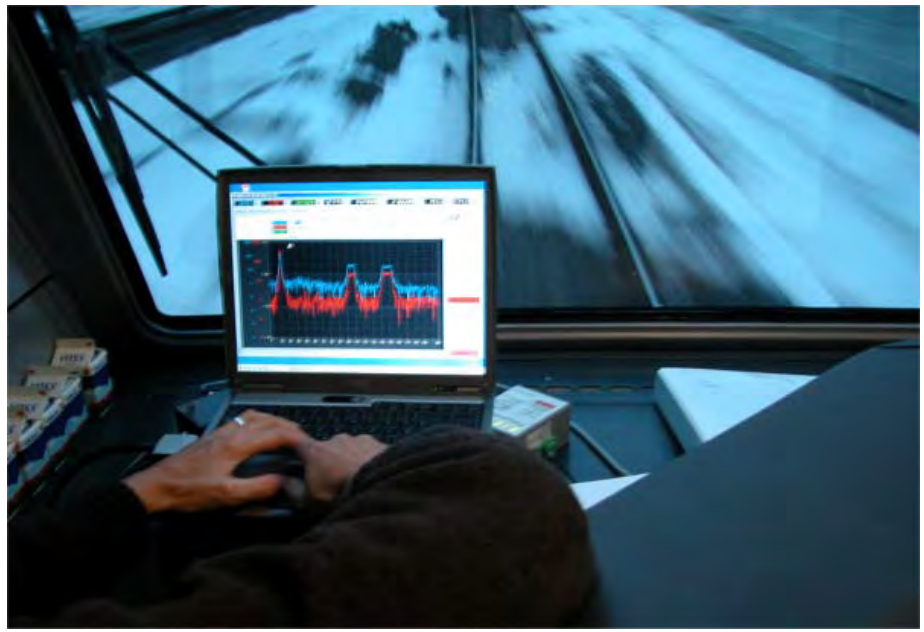


defekter Rotorstab



Anwendungen

- Verstehen der maßgeblichen Effekte und Auswirkungen
- Entwicklung von Fehler-Erkennungs-Algorithmen
- Vergleich mit Messungen
  - MBox
  - MATLAB
- Dienstleistung im Monitoring- und Diagnose-Bereich



arsenal research
Ein Unternehmen der Austrian Research Centers.

Kontakt

arsenal research

Business Unit Monitoring, Energy and Drive Technologies
Dr Christian Kral
Austria, 1210 Vienna, Giefinggasse 2
ph: +43 (0) 50550-6219, f: +43 (0) 50550-6595
mobile: +43 (0) 664-620 78 30

christian.kral@arsenal.ac.at
http://www.arsenal.ac.at

arsenal research
Ein Unternehmen der Austrian Research Centers.

Simulation von elektrischen Antrieben im automotiven Bereich mit der Smart Electric Drives (SED) Library in Dymola

Teil II

Dragan SIMIC

dragan.simic@arsenal.ac.at

arsenal research
Ein Unternehmen der Austrian Research Centers.

Motivation

- Berechnung, Vergleich und Einsparungspotential des Kraftstoffverbrauchs von unterschiedlichen Fahrzeug-Hybridkonzepten
- Reduzierung von Abgasausstoß (CO, NO_X, HC, PM…), Emissionsuntersuchung
- Optimierungspotential, Analyse des Betriebspunktes der VKM und der Nebenaggregate
- Gesamteinsparungspotential des Fahrzeugs

arsenal research
Ein Unternehmen der Austrian Research Centers.

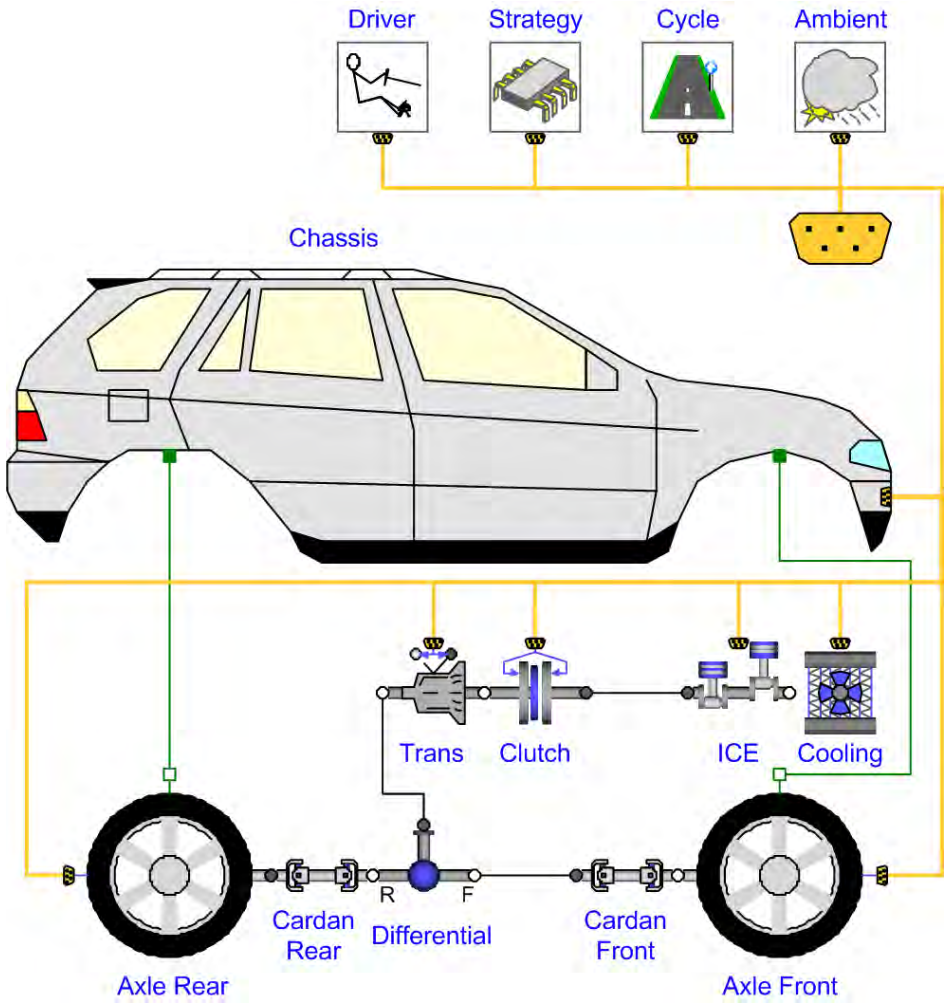
SmartPowerTrains - Library

- Konventionelle Fahrzeuge
- Hybrid Fahrzeuge
- Konventionelle / innovative Nebenaggregate, Antriebe
- Verwendete Modellbibliotheken
  - Modelica
  - Modelica.Mechanics
  - Modelica.Thermal.FluidHeatFlow
  - Modelica.Thermal.HeatTransfer
  - SmartElectricDrive
  - SmartCooling

arsenal research
Ein Unternehmen der Austrian Research Centers.

Konventionell

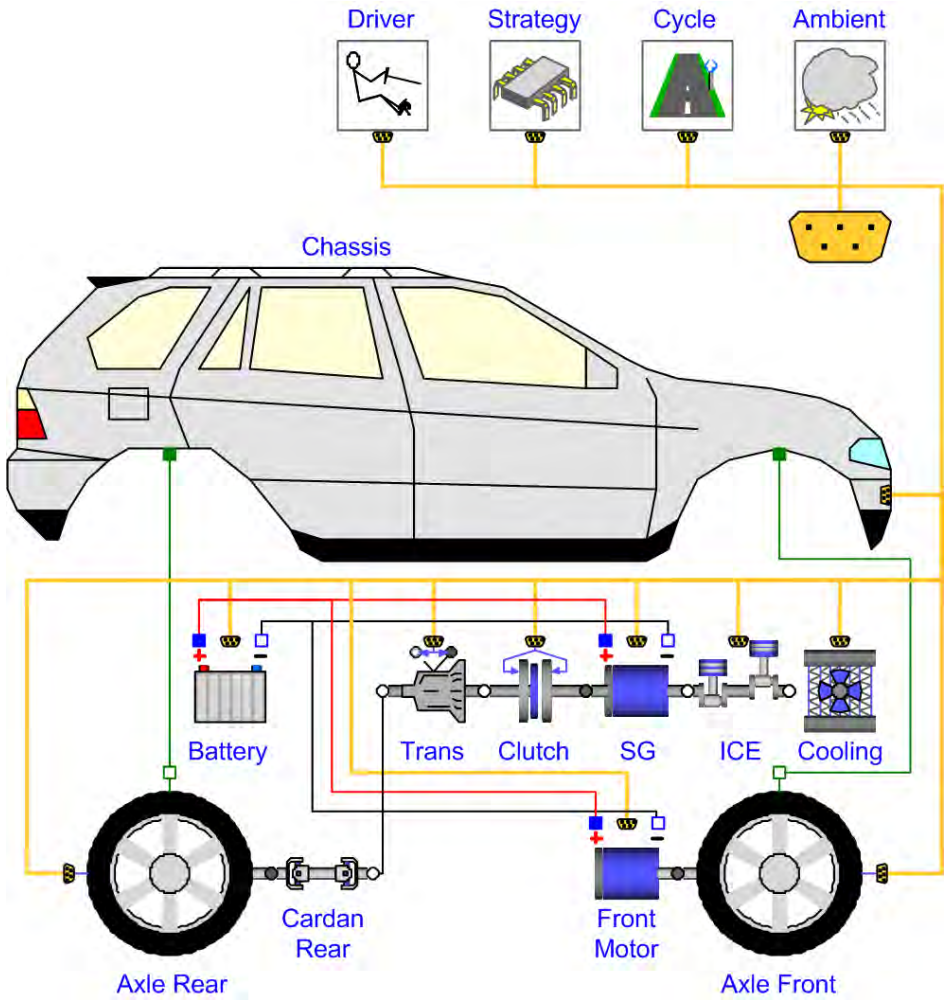
- Konventioneller Antriebsstrang
- Fahrzeugklasse, Allradantrieb
- VKM, Kraftstoff / Emissionskennfeld
- Konventioneller Kühlkreislauf
- Mechanisch angetriebene Nebenaggregate
  - Lüfter
  - Wasserpumpe



arsenal research
Ein Unternehmen der Austrian Research Centers.

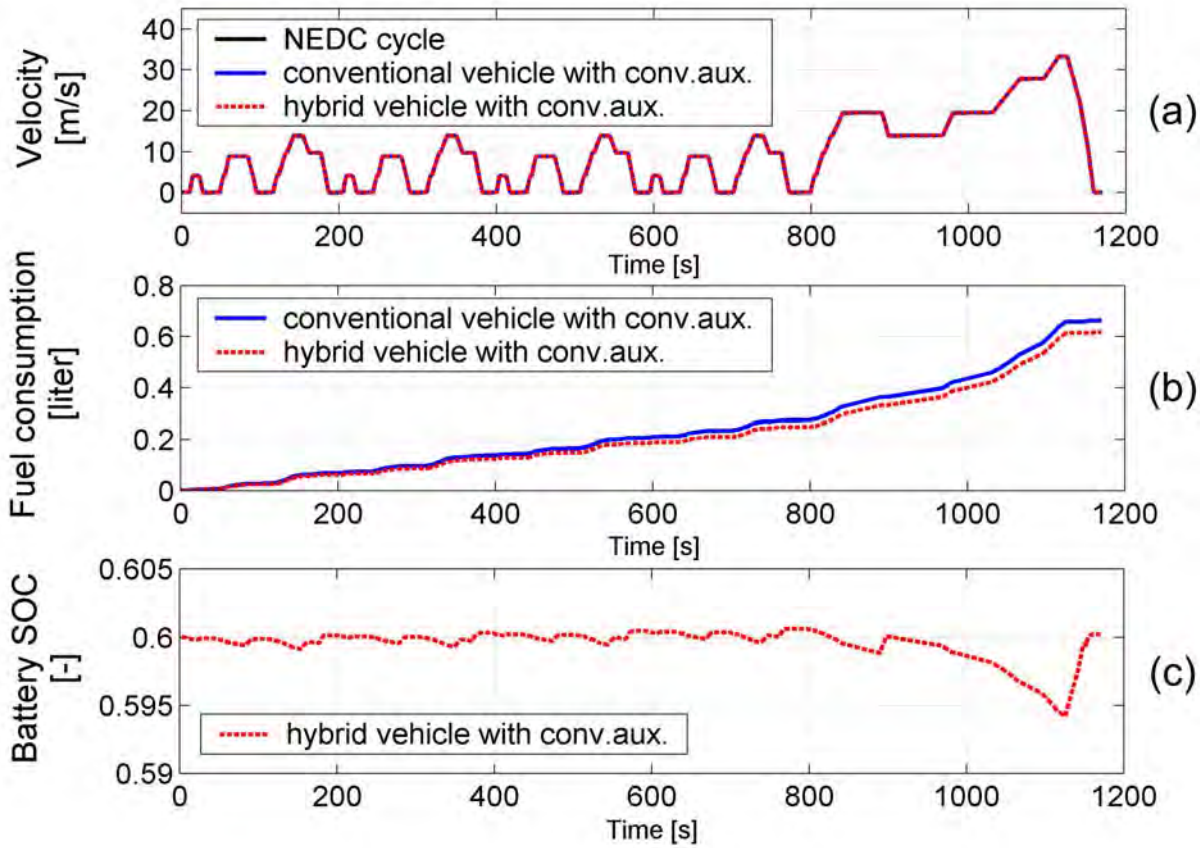
Hybridfahrzeug

- Integrierter Starter/Generator
  - Start von VKM
  - „Boosten“
  - Rückspeisung
- Elektrische Vorderachse
  - Elektrisch Fahren
  - Rückspeisen
- Batterie
- Mechanisch angetriebene Nebenaggregate



arsenal research
Ein Unternehmen der Austrian Research Centers.

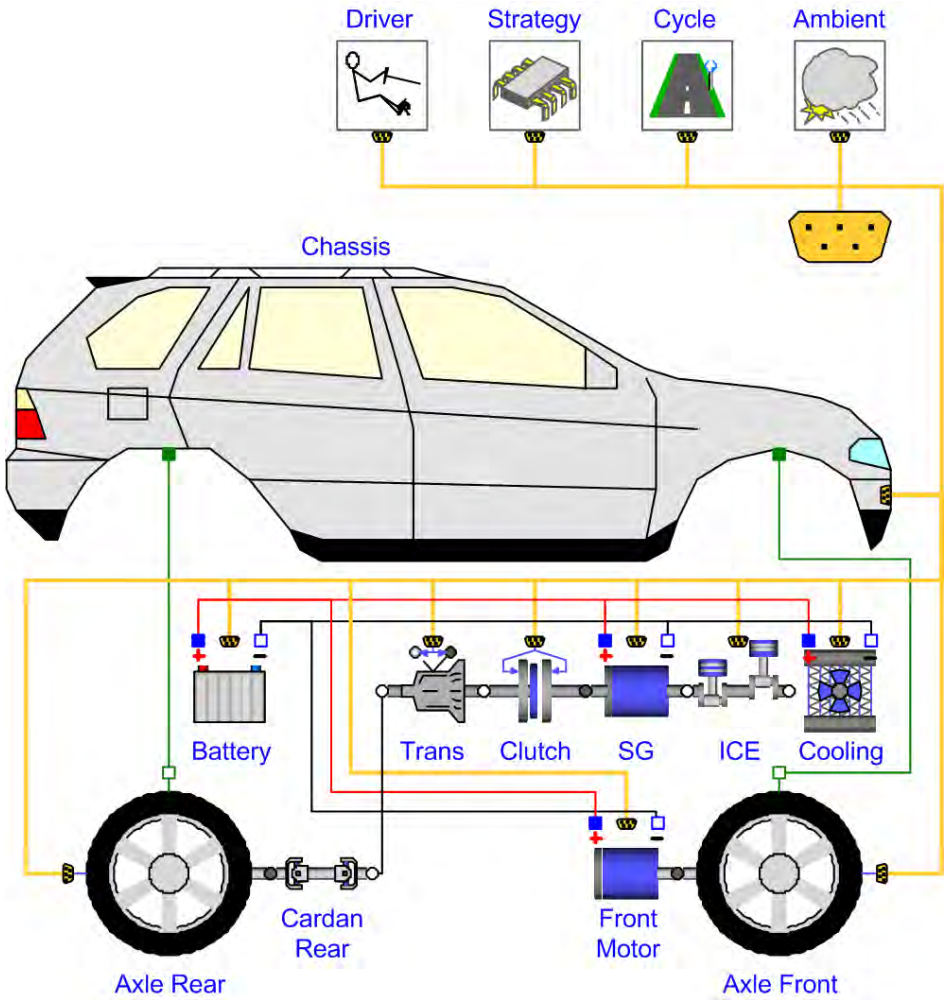
Konventional / Hybrid Vergleich



arsenal research
Ein Unternehmen der Austrian Research Centers.

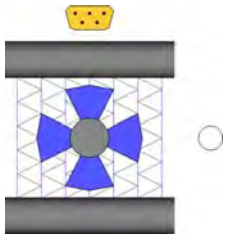
Nebenaggregate

- Integrierter Starter/Generator
- Elektrische Vorderachse
- Batterie
- Elektrisch angetriebene Nebenaggregate
  - Lüfter
  - Wasserpumpe

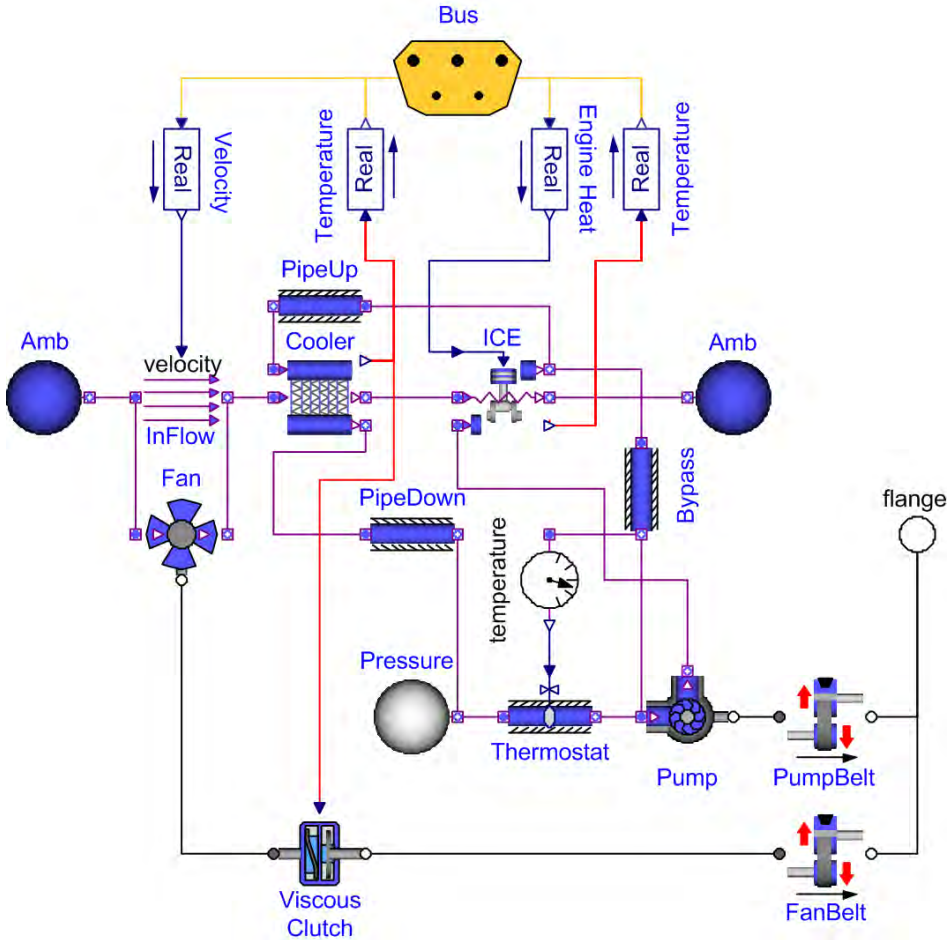


arsenal research
Ein Unternehmen der Austrian Research Centers.

Kühlkreislauf, mechanische Nebenaggregate

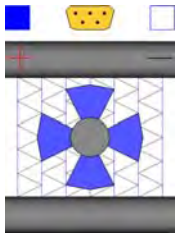


- Riemenantrieb
  - Kaltstart
  - Erwärmungsphase
  - Thermoschock

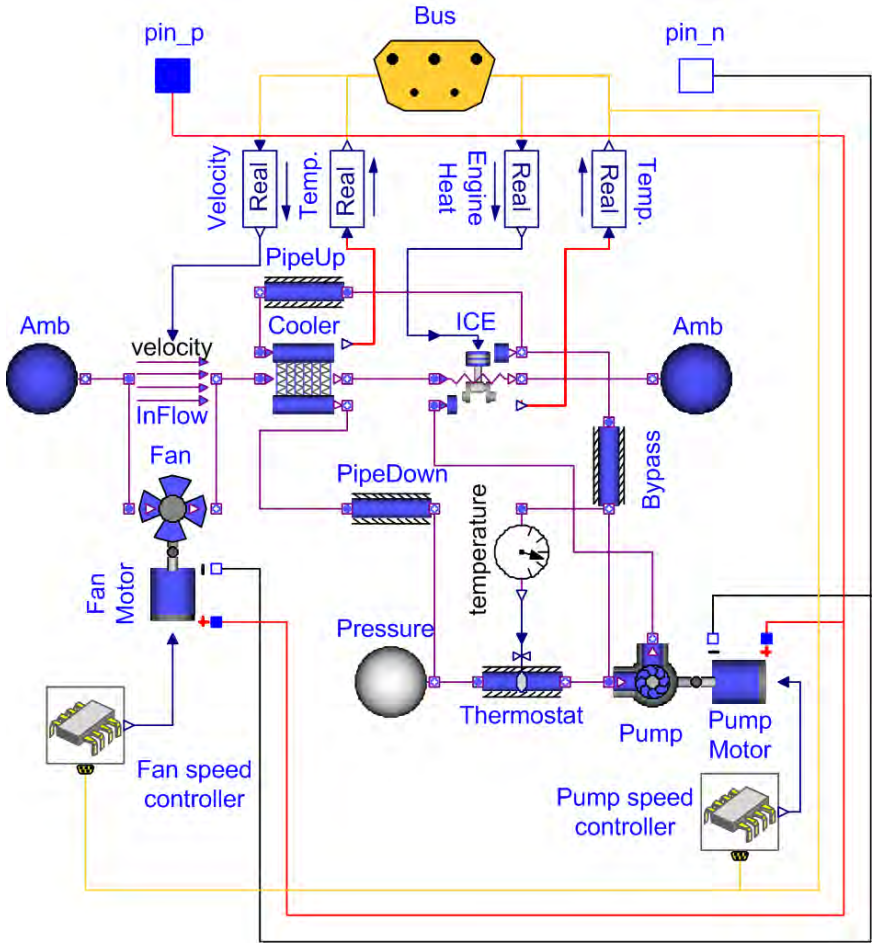


arsenal research
Ein Unternehmen der Austrian Research Centers.

Kühlkreislauf, elektrische Nebenaggregate

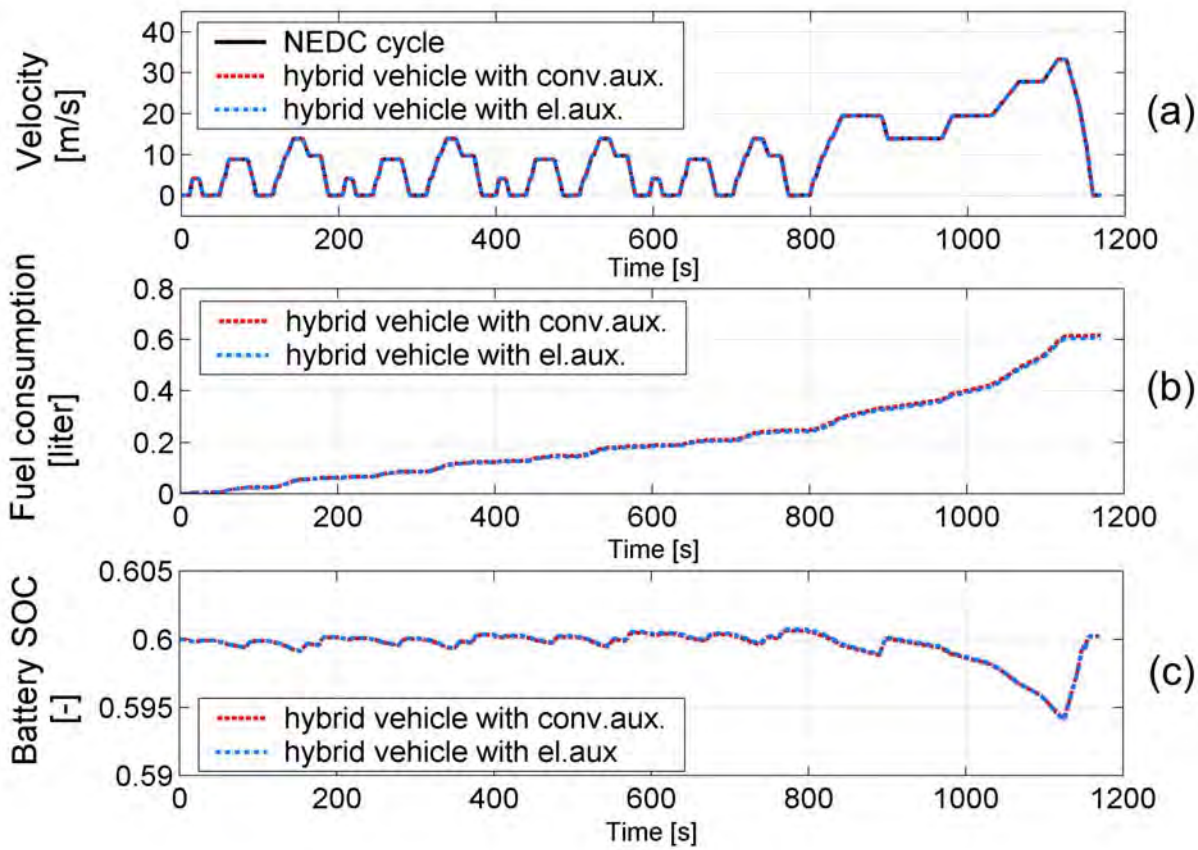


- Elektrisch
  - Drehzahl
  - Luftmassenstrom
  - Kühlflüssigkeitsmassenstrom
  - Stufenlose Regelung
  - Freie Einbaulage



arsenal research
Ein Unternehmen der Austrian Research Centers.

HEV, konventionell und elektrisch angetriebene Nebenaggregate



arsenal research
Ein Unternehmen der Austrian Research Centers.

Einsparungspotential, Emissionen

	Konventionelles Fahrzeug	HEV mit konventionellen Nebenaggregaten	HEV mit elektrischen Nebenaggregaten
Kraftstoff *(lit/100km)*	6.01	5.58	5.50
PM *(g/km)*	0.0499	0.0467	0.0461
CO *(g/km)*	0.630	0.584	0.581
NOx *(g/km)*	0.483	0.403	0.394
HC *(g/km)*	0.0532	0.0446	0.0443
SOC *start*	-	0.600	0.600
SOC *ende*	-	0.600	0.600

arsenal research
Ein Unternehmen der Austrian Research Centers.

Zusammenfassung

- Simulation alternativer Fahrzeugkonzepte
- HEV und elektrische Nebenaggregate
- Untersuchung verschiedener Fahrstrategien
- Ermittlung des Kraftstoffverbrauchs (Einsparpotential)
- Ermittlung der Emissionen

arsenal research
Ein Unternehmen der Austrian Research Centers.

Ich bedanke mich für Ihre Aufmerksamkeit!!

Dragan SIMIC

arsenal research

dragan.simic@arsenal.ac.at

www.arsenal.ac.at