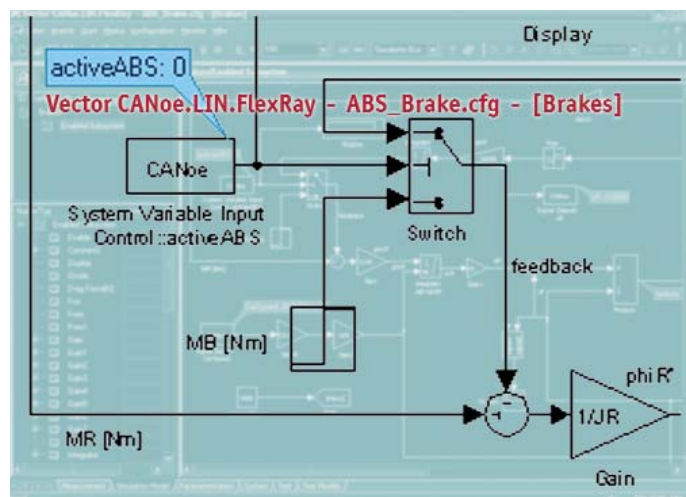


Steuergeräte modellbasiert entwickeln

Software-Simulation mit Matlab/Simulink und CANoe

Matlab/Simulink ist ein in vielen ingenieurwissenschaftlichen Disziplinen verbreitetes Werkzeug. Es wird im automobilen Umfeld beispielsweise zum Modellieren dynamischer Systeme sowie für das Beschreiben von Zuständen und deren Übergängen verwendet. Da diese Algorithmen auf Steuergeräten ablaufen, die untereinander kommunizieren, ist es im Verlauf des Entwicklungsprozesses notwendig, den Zugang zum Fahrzeugnetzwerk bereitzustellen.

Von Jochen Neuffer und Dr. Carsten Böke



Der Zugang erfolgt mittels Simulink-Blöcken, die mit der Bus-Hardware kommunizieren, direkt aus dem Modell. Daraus resultieren jedoch verschiedene Nachteile:

- Das Modell ist immer netzwerkspezifisch. Zwar können durch eine intelligente Strukturierung mit Subsystemen der Netzwerk- und der Applikationsanteil getrennt werden. Jedoch ist es vorteilhafter, ein generisches Modell zu haben, das sich sehr einfach auf unterschiedliche Netzwerke anpassen lässt.

- Zusätzliche Protokollschichten wie Netzwerk-Management (NM), Interaction Layer (IL) oder Transport-Protokolle (TP) sowie eine Restbussimulation sind nur sehr aufwendig in einem Simulink-Modell realisierbar.

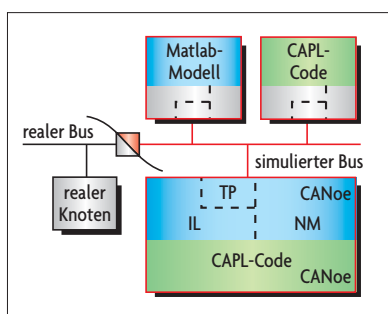


Bild 1. CANoe-Restbussimulation mit simulierten Knoten.

Ein einfacherer Zugang zu einem Netzwerk ist beispielsweise mit der Simulations- und Test-Software CANoe von Vector gegeben. Mit dieser Software wird typischerweise das Steuergerätenetzwerk mit den erforderlichen Knoten simuliert (Restbussimulation). Die Applikationsschicht eines Knotens, also jener Teil, der durch ein Matlab/Simulink-Modell beschrieben ist, setzt dabei auf einer reinen Signalschnittstelle auf (Bild 1). Es besteht kein Bezug mehr zum konkreten Netzwerk. Das Modell schreibt lediglich Signalwerte an seinem Ausgang beziehungsweise liest Signalwerte an seinem Eingang. Durch diese Abstraktion auf Signalebene kann der Anwender seine vorhandenen Modelle nahezu unverändert weiterverwenden.

Aber nicht nur Signale lassen sich zwischen CANoe und dem Simulink-Modell austauschen, sondern auch Systemvariablen. Dies sind CANoe-interne Variablen, die entweder der Benutzer explizit anlegt oder die automatisch erzeugt werden. Nach der Installation des CANoe-Matlab-Integrationspaketes von Vector steht ein Simulink-Blockset zur Verfügung, das entsprechende Blöcke für den Datenaustausch mit CANoe bereitstellt. Es besteht im Wesentlichen aus Funktionen zum Schreiben und Lesen von

Bus-Signalen, Systemvariablen und Umgebungsvariablen.

Offline Mode für Rapid Prototyping

Dieser Modus ist für sehr frühe Entwicklungsphasen zu wählen, wenn am Modell noch häufig Änderungen vorgenommen werden und das real vorhandene Netzwerk noch keine Rolle

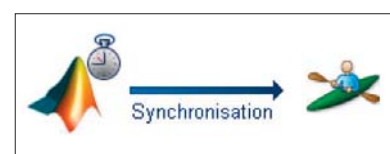


Bild 2. Offline Mode – Zeitbasis von Simulink.

spielt (Bild 2). Der Applikationsentwickler kann dennoch das Modellverhalten unter Einbeziehung des vollständig simulierten Netzwerks testen. Ein in dieser frühen Phase entwickeltes Modell kann auch in allen späteren Phasen ohne weitere Netzwerkadaptungen weiterverwendet werden.

Die Simulation selbst wird im Offline Mode in der Simulink-Umgebung gestartet. CANoe arbeitet im Offline Mode, und Simulink ist der Master für die Simulationszeit. Die Simulation wird entsprechend der jeweiligen

Rechnergeschwindigkeit durchgeführt. Alle Simulink-Debug-Funktionen können hierbei genutzt werden, eine reale Hardware mit CANoe-Anbindung zum Bus ist nicht notwendig.

■ Synchronized Mode für frühe Entwicklungsphasen

Auch dieser Modus ist für frühe Entwicklungsphasen konzipiert, bei denen das Modell noch nicht seinen endgültigen Stand erreicht hat (**Bild 3**). Zusätzlich zum Offline Mode bietet der Synchronized Mode allerdings die Möglichkeit, reale Hardware einzubinden. Auch hier wird die Simulation in Simulink gestartet. Im Gegensatz zum Offline Mode dient jedoch die Zeit aus CANoe als Basis für die Simulationszeit, und die Simulation wird in Echtzeit gerechnet. Als Einschränkung ist hier zu erwähnen, dass das Modell nur



Bild 3. Synchronized Mode – Zeitbasis von CANoe.

so komplex sein darf, dass es schneller als Echtzeit gerechnet werden kann. Denn durch das Anpassen der Simulink-Zeit an die CANoe-Zeit wird die Ausführungsgeschwindigkeit in Simulink verlangsamt und an die Echtzeit-Basis von CANoe angeglichen. Wenn das Modell jedoch in der Berechnung zu aufwendig wird, kann diese Ver-

langsamung nicht mehr stattfinden. Erfüllt das Modell jedoch die oben genannte Bedingung, ist in dieser Betriebsart der Zugang zu realer Hardware in vollem Umfang gewährleistet. Auch die Debug-Möglichkeiten von Simulink sind nutzbar.

■ Online Mode oder Hardware-in-the-Loop Mode

Hier wird die Berechnung des Modells komplett in der CANoe-Umgebung ausgeführt. Dazu wird aus dem Simulink-Modell eine DLL erzeugt, die in CANoe geladen und ausgeführt werden kann. Der Real-Time Workshop von Matlab/Simulink steuert hierbei die Code-Generierung. Um Code für CANoe zu erzeugen, wurde für die Real-Time-Workshop-Umgebung ein spezielles Target Makefile entwickelt, das den Generierungsprozess steuert. Mithilfe des Microsoft-Visual-Studio-Compilers wird dann der generierte Code compiliert und gelinkt. Das Resultat ist eine Nodelayer-DLL, die eine CANoe-interne API implementiert und als Komponente an einen Knoten im Simulationsaufbau hinzugefügt werden kann. Andere Beispiele solcher Komponenten sind die OEM-spezifischen Interaction Layer, die Komponenten für Netzwerk-Management oder für Transportprotokolle. Diese Komponenten werden von CANoe beim Start einer Messung geladen und nach Abschluss der Messung wieder freigegeben.

Die Modellausführung findet somit komplett in der Echtzeit-Umgebung

von CANoe statt. Alle Ereignisse, wie Aktionen am Netzwerk, Timer oder Benutzereingaben, werden zyklusgenau berechnet. Das Modell ist Teil der CANoe-Konfiguration und kann auf einfache Weise weitergegeben werden.

■ Modellberechnung in CANoe

Um die Modellberechnung besser zu verstehen, ist das Verständnis des Ablaufmodells in CANoe von Bedeutung. CANoe rechnet grundsätzlich ereignisgesteuert. Ereignisse sind hierbei eintreffende Netzwerkbotschaften oder Timer. Bei jedem eintreffenden Ereignis wird die Simulationszeit in CANoe auf den Zeitstempel des jeweiligen Ereignisses gesetzt. Dabei wird die Zeitbasis abgeleitet von hochauflösenden Nicht-Windows-Timern auf den Netzwerkschnittstellen. Somit wird eine hohe Genauigkeit der Zeitstempel erreicht, obwohl CANoe unter Windows läuft. Wird nun der Konfiguration ein generiertes Matlab/Simulink-Modell in Form einer DLL hinzugefügt, muss dieses zyklisch gerechnet werden. Wie oben erwähnt, sind auch Timer Ereignisse, welche die Simulationszeit von CANoe setzen können. In dem Modell wird also ein Timer mit genau der Zeit gesetzt, die in den Solver-Einstellungen des Simulink-Modells spezifiziert ist.

■ Parametrieren des Modells

Eine häufige Anforderung der Anwender ist es, das Modell nachträglich zu

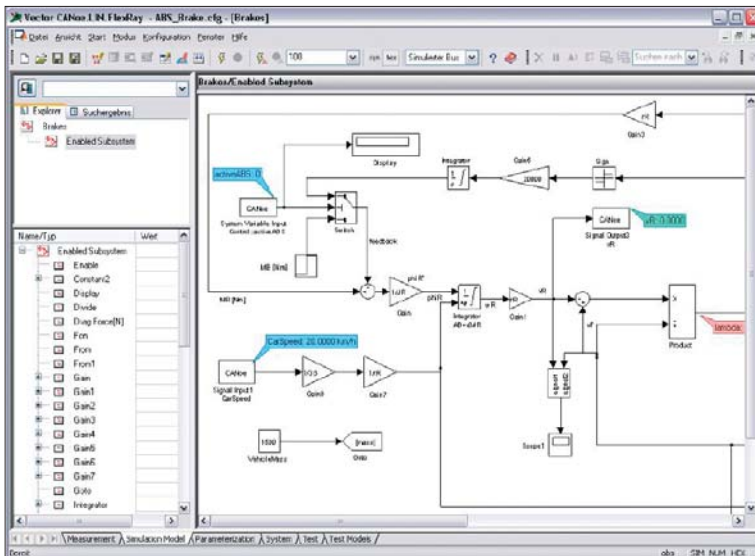


Bild 4. Model Viewer in CANoe.

verändern, ohne es erneut compilieren zu müssen. Die Anwendungsfälle lassen sich folgendermaßen einteilen:

- ▶ Die Modelle unterscheiden sich in den einzelnen Steuergeräte- oder Fahrzeugvarianten lediglich durch interne Parameter. Das Ändern der Parameter ist, z.B. im Rahmen von automatischen Tests, auch zur Laufzeit möglich. So können dieselben Modelle für verschiedene Fahrzeugvarianten verwendet werden.
- ▶ Im Modell sind zu Simulationsbeginn bestimmte Anfangsbedingungen festzulegen, welche die eigentliche Modell-Logik nicht verändern dürfen, aber in verschiedenen Testszenarien durchaus verschieden sein können. Beispiel: Es sollen die Verstärkungsfaktoren bei Regelschleifen verstellt werden, um so unterschiedliche Regelcharakteristiken zu testen.
- ▶ Der Real-Time Workshop und/oder ein Microsoft-Compiler stehen nicht zur Verfügung, die Zeit für häufiges Neu-Compilieren ist zu lang, oder die

den Knoten im Simulationsaufbau wäre die Folge. Um diesen Anwendungsfällen gerecht zu werden, lassen sich die für CANoe erzeugten Komponenten auch nach dem Compilieren noch parametrieren. Bei der Code-Generierung werden hierbei für jeden Parameter im Modell Systemvariablen generiert, die CANoe anschließend liest und schreibt. Modellparameter sind typischerweise die Eigenschaften der einzelnen Simulink-Blöcke. Das kann bei einem Integratorblock sein Anfangszustand sein oder bei einem Sinus-Block dessen Frequenz, Phase, Offset oder Amplitude. Eine Systemvariable, die beispielsweise den Verstärkungsfaktor eines Simulink-Gain-Blocks repräsentiert, lässt sich dann in den Analysefenstern (Grafik, Daten, Trace) sowie Panels, CAPL-Programmen und Tests verwenden. Ein weiterer Vorteil ist die Möglichkeit zur Feinabstimmung des Modells, da interne Parameter iterativ leicht zu variieren sind.

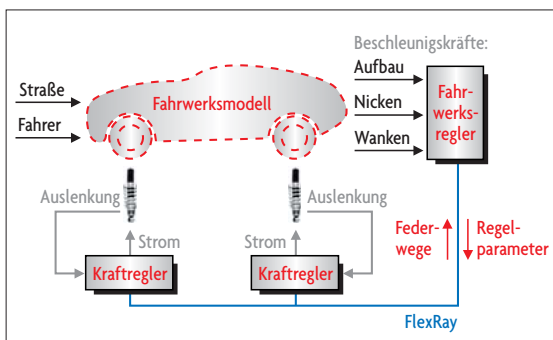


Bild 5. Modell einer aktiven Fahrwerksregelung.

Modelldatei selbst steht nicht zur Verfügung. So können auch Nutzer ohne Matlab/Simulink-Installation die Modelle noch zur Laufzeit ändern.

In solchen Fällen ist es sinnvoll, das Parametrieren nicht über einzeln generierte DLLs zu realisieren – ständiges, zeitaufwendiges Umkonfigurieren der Komponenten an

den internen Parameter per Drag & Drop aus dem Modellanzeigefenster in die CANoe-Analysefenster verschoben werden. Für diese Anzeige ist keine Matlab/Simulink-Lizenz nötig.

Simulation einer aktiven Fahrwerksregelung mit einem FlexRay-Bus

Der Entwurf einer aktiven Fahrwerksregelung inklusive eines Fahrzeugmodells dient als anspruchsvolles Beispiel. Das Fahrzeugmodell soll eine ausreichend realistische Fahrdynamik nachbilden können und mit aktiven Dämpfern als Plattform für die Fahrwerksregelung dienen. Mittels der Regelung soll ein möglichst komfortables Fahrverhalten erzielt werden. Hieraus ergibt sich die Aufteilung des Gesamtmodells in zwei Hauptblöcke:

- ▶ Umgebungsmodell (Umgebungssimulation für Regler) einschließlich eines Mehrkörpermodells des Fahrzeugs (mechanisches Fahrzeugmodell), eines Modells des Antriebsstrangs (vereinfachtes Motormodell), eines Modells der Bremsen, vier Radmodellen und eines Fahrbahnmodells.
- ▶ Fahrwerksregelung einschließlich Fahrzeugbeobachter, eines übergeordneten Fahrwerksreglers und vier untergeordneten Kraftreglern für die aktiven Dämpfer.

Das Umgebungsmodell unterteilt sich in das Fahrzeugmodell und das Fahrbahnmodell. Das Fahrzeugmodell wird als Mehrkörpermodell mit 15 Freiheitsgraden entworfen. Dabei ist das Mehrkörpermodell symbolisch mit Hilfe eines Computer-Algebrasystems definiert, und daraus sind die Bewegungsgleichungen abgeleitet (ca. 4400 Additionen und ca. 20 800 Multiplikationen). Das Fahrzeugmodell bietet weitere Eingänge zur Stimulierung: Fahr- und Bremspedalstellung, eingelegter Gang, Lenkwinkel, Handbremsenstellung und Regler-Zielwertvorgabe (komfortabel oder sportlich). Die Übermittlung dieser Größen ist in CANoe mittels Systemvariablen realisiert. Dabei können die Fahrervorgaben durch ein Bedien-Panel mit entsprechenden Kontrollelementen interaktiv vorgegeben werden. Durch ent-

Betrachten eines Simulink-Modells in CANoe

Matlab/Simulink-Modelle können in CANoe betrachtet werden, sofern sie in Form einer generierten Komponente vorliegen. Nach entsprechender Konfiguration am jeweiligen Knoten steht hierzu ein separates Fenster in CANoe zur Verfügung (Bild 4). Um das Modell nachträglich zu verändern, können alle generierten, mo-

sprechende Makroaufzeichnungen lassen sich diese auch als Fahrprofil automatisiert abspielen. Das Fahrbahnmodell wird durch eine Look-up-Tabelle modelliert, die für jede Position die Höhe der Fahrbahnoberfläche und die Oberflächennormale enthält. Ebenso wird die Oberflächenbeschaffenheit durch ihren Reibwert an jeder Position beschrieben.

Die Fahrwerksregelung besteht aus einem LQ-Regler mit Beobachter. Der Regler berechnet die Sollkräfte für die aktiven Kraftelemente in den Radaufhängungen. Dabei werden im Wesentlichen zwei Regelungsziele verfolgt:

- ▶ Reduzierung der Aufbaubeschleunigungen. Ergebnis für den Pkw-Fahrer: erhöhter Komfort.

- ▶ Radaufstandsfläche konstant halten (Minimierung der Variation der Radaufstandskräfte). Ergebnis für den Pkw-Fahrer: erhöhte Sicherheit und verbesserte Fahrdynamik

Der Beobachter rekonstruiert die nicht messbaren Fahrzeugzustände und nutzt dafür ein vereinfachtes lineares Fahrzeugmodell (sieben Freiheitsgrade). Der Fahrwerksregler benötigt die Wank- und Nickwinkelbeschleunigungen des Aufbaus aus zwei Gyro-Sensoren sowie die Hubbeschleunigung des Aufbaus aus einem Beschleunigungssensor als Messgrößen. Diese Größen werden durch die Fahrzeugsimulation zur Verfügung gestellt und in CANoe ebenfalls über Systemvariablen an den Fahrwerksregler übermittelt. Der Kraftregler regelt die Kraft der aktiven Kraftelemente auf ihren

von der Fahrwerksregelung vorgegebenen Sollwert. Dieser ist beispielsweise als einfacher PI-Regler für eine Stromregelung ausgelegt.

In CANoe werden sechs Simulationsknoten definiert (**Bild 5**): Dies sind der Knoten Umgebungssimulation (Fahrzeugmodell mit Fahrwerksmodell), der Knoten Fahrwerksregler und vier Kraftregler-Knoten. Die sechs Knoten tauschen die vier Sollkräfte für die Kraftregler und die vier Auslenkungen der Radaufhängungen über einen FlexRay-Bus aus. Das zyklische Übertragen der Signale über einen Bus stellt im geschlossenen Regelkreis eine diskrete Abtastung mit einer Totzeit dar. Diese wirkt sich oft – aufgrund ihrer Unvorherbestimmbarkeit und ihrer Größe – negativ auf die Qualität der Regelung aus. Durch das Verwenden eines FlexRay-Busses werden diese Signale sehr zuverlässig und ohne Jitter übertragen. Zudem ist es möglich, diese Signale mit einer niedrigen Zykluszeit (2,5 bis 5 ms) zu übertragen. Dadurch ist die Qualität der Gesamtregelung gegenüber einem CAN-Bus wesentlich verbessert.

Die CANoe/Matlab-Integration ermöglicht die gleichzeitige Nutzung von Simulink zur Modellierung von komplexen Applikationsverhalten sowie die Integration des konkreten Fahrzeugnetzwerkes innerhalb des selben Entwicklungsprozesses. Der Anwender kann während der Entwicklung in seiner gewohnten Matlab/Simulink-Umgebung arbeiten und muss sich nicht um netzwerkspezifische Details kümmern.

.sj



Dipl.-Ing. Jochen Neuffer

studierte Nachrichtentechnik an der Fachhochschule Esslingen. Seit 2002 arbeitet er bei Vector Informatik und ist dort als Product Management Engineer im Bereich Tools for Network and Distributed Systems tätig.

jochen.neuffer@vector.com



Dr. rer. nat. Carsten Böke

studierte Informatik an der Universität Paderborn. Von 1995 bis 2004 war er wissenschaftlicher Mitarbeiter im Heinz Nixdorf Institut, Fachgebiet Entwurf paralleler Systeme. Seit 2004 ist er Senior Software Development Engineer bei der Vector Informatik GmbH und entwickelt dort Werkzeuge für Bus-Analyse und Bus-Simulation von FlexRay-Systemen.

carsten.boeke@vector.com