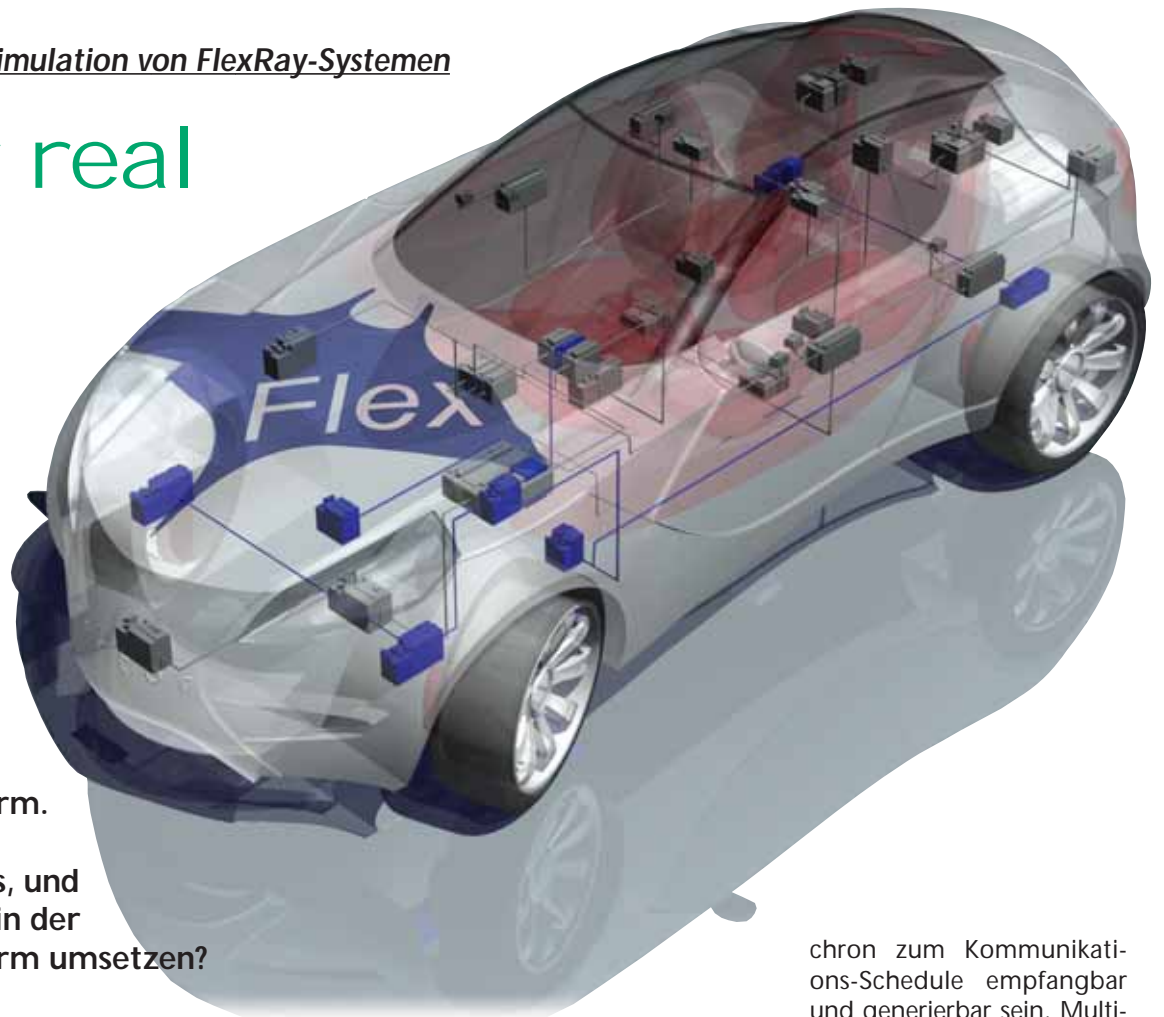


Modellierung und Simulation von FlexRay-Systemen

# Regler real testen

Mit ihrem zeitgesteuerten Kommunikationsansatz stellt die Echtzeitsimulation von Steuergeräten mit FlexRay-Interface spezielle und hohe Anforderungen an die Simulationsplattform. Wie sehen diese Anforderungen aus, und wie lassen sie sich in der Simulationsplattform umsetzen?



Reglersysteme bestehen aus einem geregelten System (z.B. einem Verbrennungsmotor) und einem Regler, der beeinflussbare Stellgrößen vorgibt, um bestimmte Eigenschaften des geregelten Systems (z.B. Drehzahl, Drehmoment und Abgasemissionswerte) mit Soll-Werten in Übereinstimmung zu bringen. Dabei kann ein Regler hierarchisch aus mehreren seriell und/oder parallel geschalteten Subreglern bestehen. Um Regler prüfen zu können, bedarf es spezieller Testumgebungen. Solche Testumgebungen bestehen aus Werkzeugen, die für den zu prüfenden Regler (Sys-

tem Under Test) ein vorhandenes physikalisches Modell (Physical Environment) simulieren.

In der HIL-Entwurfsmethodik (Hardware in the Loop) werden zunächst alle Modelle (Regler und physikalisches Umweltmodell) am Rechner simuliert. Danach werden Schritt für Schritt Regler beziehungsweise Subregler durch reale Komponenten ersetzt. Der Rest wird weiterhin durch eine echtzeitfähige Plattform simuliert. Schließlich und letztendlich kann auch die simulierte Umwelt durch das reale physikalische System ersetzt werden (Bild 1).

Verschiedene Regler werden teilweise auf unterschiedlichen Steuergeräten implementiert, und gerade im Entwurfsprozess sind nicht alle Steuergeräte sofort implementiert. Teilweise müssen andere Steuergeräte si-

muliert werden. Hierbei spricht man von einer so genannten Restbussimulation. Diese wird auch speziell beim Test von einzelnen Steuergeräten eingesetzt. Eine wesentliche Anforderung an die Restbussimulation ist ihre Echtzeitfähigkeit. Das zu überprüfende System geht oft von einem bestimmten Antwortzeitverhalten der anderen Geräte aus. So wird zum Beispiel erwartet, dass Ausgaben der anderen Regler mit bestimmten garantierten Zyklusraten und möglichst geringen Jittern erfolgen.

Als zeitgesteuertes Kommunikationsmedium stellt FlexRay besondere Anforderungen an eine Simulationsplattform. Die Signal/Botschafts-Generierung erfolgt periodisch und mit hohen und unterschiedlichen Zyklusraten. Botschaften und/oder Signale müssen syn-

chron zum Kommunikations-Schedule empfangbar und generierbar sein. Multi-Rate-Systeme mit unterschiedlichen Zyklusperioden müssen unterstützt werden (Cycle Multiplexing). Da das Zeitverhalten von FlexRay deterministisch ist, muss der Jitter bezüglich der Aktivierung und/oder Ausführung von Applikationscode konstant und so klein wie möglich sein. Dies ist eine unabdingbare Voraussetzung, um rechtzeitige Botschafts-Updates garantieren zu können. Die Antwortzeiten sind beim Empfang und Senden kurz. Daher muss der Zeitaufwand des Betriebssystems, des Kommunikations-Stacks und der Laufzeitumgebung so klein wie möglich sein, um schnelle Reaktionen gewährleisten zu können. Insbesondere sollte die Möglichkeit von In-Cycle-Responses bestehen.

Eine weitere Anforderung an die Simulationsplattform besteht darin, die globale Zeitbasis des Kommunikations-

Dr. Carsten Böke  
ist als Senior Software  
Development Engineer  
bei Vector Informatik tätig

mediums in die Applikation zu übertragen. Um zeitgesteuerte Anwendungen korrekt implementieren zu können, muss es möglich sein, die Applikation (Modell) synchron zum FlexRay-Schedule zu betreiben. Außerdem muss die Applikation leicht von der globalen Zeitbasis Gebrauch machen können, um die Implementierung von verteilten Regelungen mit einem gemeinsamen Zeitverständnis zu vereinfachen.

### **FIBEX- Unterstützung nötig**

Schließlich und endlich muss die Simulationsplattform das FIBEX-Datenformat unterstützen. Die Spezifikation des Applikationsverhaltens muss auf einfachem und symbolischem Niveau erfolgen. Die in einer FIBEX-Datenbank spezifizierten symbolischen Signal- und Botschaftsnamen müssen verwendbar sein. Die TDMA-Parameter zur Konfigurierung des »Communication Controllers« müssen aus der FIBEX-Datenbank ausgelesen und automatisch programmiert werden. Das Versenden von Botschaften muss anhand der Schedule-Tabellen für FlexRay automatisch erfolgen können.

Als Bus-Analyse-, Test- und Simulationswerkzeug unterstützt »CANoe.FlexRay« auch die speziellen Bedürfnisse einer Restbussimulation für FlexRay-Systeme. Grundlegende Architektur von CANoe zur Modellausführung ist ein Benachrichtigungskonzept durch so genannte »Notification Handler«. Dies umfasst die Modellaktivierung beim Erhalt von Nachrichten, dem Ablaufen von Timern oder der Erkennung von Fehlerzuständen. Insbesondere wurde dieses Konzept für FlexRay um synchrone Benachrichtigungen zu bestimmten Zeitpunkten im

FlexRay-Zyklus erweitert. So lassen sich Aktionen regelmäßig zum Zyklusanfang oder nach Beendigung eines bestimmten Slots ausführen. Natürlich sind auch Benachrichtigungen beim Empfang oder dem Fehlen von Frames auf dem Bus möglich. Alle diese Benachrichtigungen können auf ein konkretes Cycle-Multiplexing

eingeschränkt werden. Ziel der zeitlich zum FlexRay-Zyklus synchronen Aktivierung von Modellteilen ist es, die Zeitsteuerung des Kommunikationsmediums in die Applikation zu übertragen. Gleichzeitig soll sich die Applikation auf das Schedule synchronisieren lassen. Um die Spezifikation des Modellverhaltens zu verein-

fachen, wurde erstens die Möglichkeit geschaffen, auf die Signal- und Botschaftsdefinitionen in einer FIBEX-Datenbank symbolisch zuzugreifen. Zweitens wurde ein Puffermechanismus (Signal Layer) implementiert, damit zu jedem Zeitpunkt Signalzustände gelesen und geändert werden können. Das Empfangen und Versen-

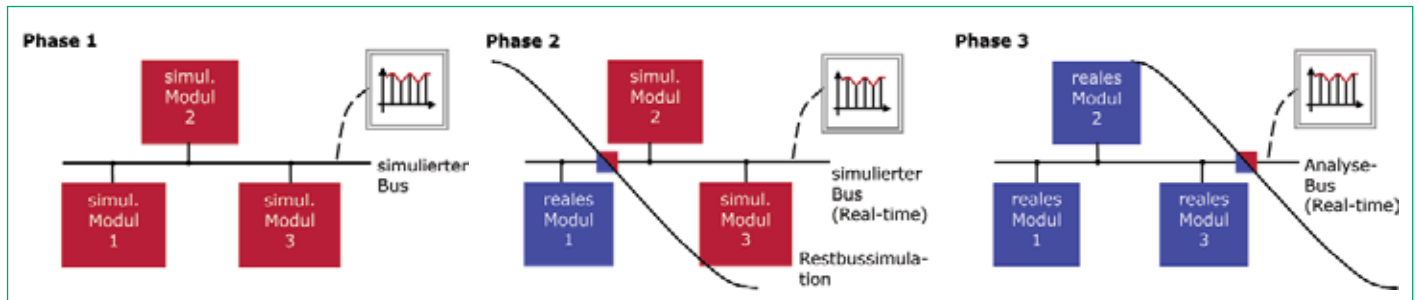


Bild 1 Bei der Hardware-in-the-Loop-Entwicklungsmethodik mit Restbussimulation werden nach und nach simulierte Regler durch reale Komponenten ersetzt. Am Schluss kann auch die simulierte Umwelt durch das reale physikalische System ersetzt werden.

den der zugehörigen FlexRay-Frames erfolgt automatisch im Hintergrund, wobei die Sendezeitpunkte (für statische Frames) durch das Schedule in der Datenbank vorgegeben sind. Außerdem lassen sich so Signale interpretieren und verändern, ohne dabei Frames assemblieren oder de-assemblieren zu müssen. CANoe bietet als Standardlösung die Möglichkeit, das Verhalten von Modellen in Form der Programmiersprache CAPL (Communication Access Programming Language) zu spezifizieren. Bei CAPL handelt es sich um eine C-ähnliche Programmiersprache, die vor ihrer Ausführung kompiliert wird. Die Modellausführung erfolgt dabei nicht interpretiert, sondern nativ auf Maschinenebene. CAPL eignet sich daher besonders für hohe Echtzeitanforderungen und diskrete Modelle.

## Echtzeitfähige Hardware

Eine weitere Möglichkeit der Modellausführung ist die Beschreibung des Modells in einer beliebigen Programmiersprache und die Erzeugung einer so genannten DLL (Dynamic Link Library). Solch eine DLL kann dann über das CANoe-API auf alle Funktionen zur Benachrichtigung, dem Lesen, oder Senden von Botschaften, Signalen oder anderen Datenbankinformationen zugreifen. Mehrere

solche DLLs können in den Simulationsaufbau von CANoe eingebunden und bei Messungsstart aktiviert werden. Basierend auf einer vom »Realtime Workshop« generierten DLL ist es auch möglich, Modelle aus Matlab/Simulink über eine Extension in CANoe einzubinden. Auf diese Weise lassen sich auch besonders komplexe und quasikontinuierliche Modelle in CANoe einbinden. Bei der Ausführung einer Modellsimulation auf einer nicht echtzeitfähigen Hardware und einem nicht deterministischen Betriebssystem kann es zu Unterbrechungen und Verzögerungen bei der Be-

nachrichtigung und Ausführung kommen. Große und nicht deterministische Jitter beim Zugriff auf das Kommunikationsmedium sind die Folge. Verschlimmert wird diese Problematik mit steigender Anzahl und Last der parallel zur Simulationsplattform laufenden Prozesse. Generell lassen sich diese Probleme durch das Ausschalten nicht benötigter Systemdienste (z.B. Graphical User Interface and Human Interaction, Logging, Indexing, Network-Access, Virus-Scanner, Garbage-Collection, Background-Optimizations, etc.) minimieren. Des Weiteren kann man beobachten, dass

sich die Verzögerungen mit zunehmender Last auf dem Rechner und bei der Verwendung einer nicht performanten Hardware vergrößern. Als Lösung für diese Problematik gelten die folgenden Richtlinien:

- Echtzeitsimulationen mit hohen Anforderungen sollten nur auf hoch performanter und speziell dafür zugeschnittener Hardware ausgeführt werden.
- Als Betriebssystem sollte ein dediziertes Echtzeitsystem zum Einsatz kommen.
- Die Echtzeitablaufumgebung (RT-Exec-Path) sollte vom nicht echtzeitfähigen Analyse- (Anlyz-Path) und Bedienteil (GUI) getrennt werden.
- Einzig die Echtzeitablaufumgebung sollte auf der Echtzeithardware ausgeführt werden.

## Zwei getrennte Rechner

CANoe unterstützt diese Anforderungen durch seine Dual-PC-Architektur (Bild 2). Die CANoe-Anwendung spaltet sich dabei in zwei Teile auf: In den »Soft Real-time Client« für die Benutzerinteraktion (GUI), das Abspeichern von Logging-Daten und die Online-Datenanalyse, sowie in den »Real-time Server« für die Ausführung der Modellteile und die Ankopplung an das Kommunikationsmedium oder an weitere I/O-Schnittstellen. Der

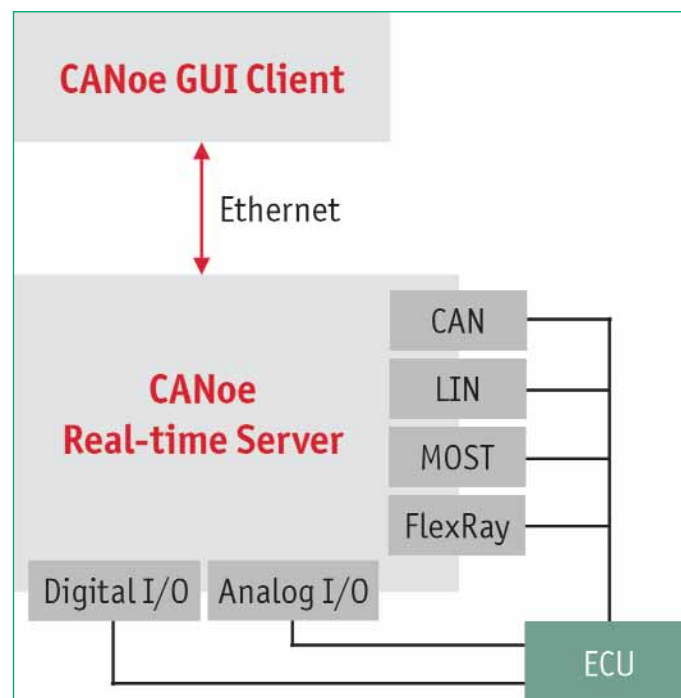


Bild 2 Die Dual-PC-Architektur von CANoe erfüllt die hohen Anforderungen, welche der zeitgesteuerte Kommunikationsansatz von FlexRay stellt

»Real-time Server« kann auf einer speziell für diesen Anwendungsfall zugeschnittenen Hardware ablaufen: einem »Real-time Rack«. Der »Real-time Server« kann optional auf speziellen Betriebssystemen ablaufen. Derzeit werden Windows XP, Windows XP Embedded und zukünftig Windows CE unterstützt. Für höchste Realzeitanforderungen empfiehlt Vector Windows CE. Damit lassen sich Task-Wechselzeiten von unter 10 µs erreichen, und je nach angeschlossener Bus-Interface-Hardware sind Antwortzeiten von unter 100 µs möglich.

Die Dual-PC-Architektur ist für den Anwender völlig transparent, da alle Interaktionen und die Konfigurierung der Messung mit dem Soft Real-time Client erfolgen. Die beiden Rechner werden über Ethernet miteinander verbunden. Die Vorteile einer hoch optimierten Ablaufumgebung sind:

- 100% der Buslast eines FlexRay-Clusters kann potenziell auf dem Real-time Rack verarbeitet werden.
- Sehr hohe Update-Raten und Antwortzeiten sind erreichbar.
- Mehrere Steuergeräte-Knoten können gleichzeitig simuliert werden.
- Die Modelle können exakt synchron zum Kommunikationszyklus abgearbeitet werden.
- »In-Cycle-Responses« sind möglich.
- Verzögerungen (maximale Latenzzeiten) und Jitter werden minimiert und sind konstant.

Vector Informatik bietet hoch performante Schnittstellen für den Buszugriff an. Zurzeit wird die »FlexCard Cyclone II« (in Kooperation mit TZ Mikroelektronik) unterstützt. Diese vereint gute Analyse-möglichkeiten mit einem schnellen Zugriffs-Interface. Die FlexCard Cyclone II ist als CardBus32-Karte für den

PCMCIA-Slot entworfen. Sie kann daher problemlos in allen gängigen Notebooks eingesetzt werden. In naher Zukunft möchte Vector auch ein PCI-Interface anbieten. Hoch optimiert und mit einem eigenen Prozessor ausgestattet, wird dieses die zur Verfügung stehenden Ressourcen noch besser ausnutzen und CANoe von Aufgaben entlasten. Ein

Master-DMA-Interface zwischen PC und Karte garantiert dann einen sehr hohen Datendurchsatz. Ebenso ist für die Zukunft ein FlexRay-Interface geplant, welches über USB angeschlossen werden kann. Dieses ist für die Bus-Analyse ausgelegt. Bedingt durch das verzögernde USB-Interface sind hiermit jedoch nur eingeschränkte Ant-

wortzeiten realisierbar. Alle von Vector angebotenen Hardware-Schnittstellen für FlexRay bieten laut Hersteller minimale Latenzzeiten für das Senden und Empfangen von Frames. (rh)

#### Vector Informatik

Telefon 07 11/80 67 04 92 3  
[www.flexray-solutions.de](http://www.flexray-solutions.de)