

In der Wiederverwendung bewährter Funktionen liegt Sparpotenzial für die Steuergeräteentwicklung. Die Integration neuer Funktionen in bereits existierende Hard- und Software ist dennoch mit großem Aufwand verbunden. Wertvolle Zeit im Entwicklungsprozess lässt sich durch die hardwareunabhängige Modellentwicklung außerhalb eines Steuergerätes einsparen.

HARDWAREUNABHÄNGIGE MODELLENTWICKLUNG FÜR STEUERGERÄTEFUNKTIONEN

Bypassing basierend auf Standards

Die Funktionsentwicklung stellt die elementare Phase in der Entwicklung von Steuergeräten dar. Eine Funktion ist beispielsweise ein Algorithmus, der Fahrzeugdaten auswertet und Ausgangssignale berechnet. Diese Signale lösen Aktionen aus, etwa das Ansteuern von Stellmotoren.

Bei der Entwicklung bzw. Anpassung der Steuergeräte an neue Fahrzeugplattformen wird sehr häufig auf vorhandene Hard- und Software zurückgegriffen. Dennoch verursacht die Integration neuer Funktionen einen hohen Aufwand. Bevor die Algorithmen in Form von Quellcode zur Verfügung stehen, werden sie in aller Regel modelliert und dabei bis zum gewünschten Reifegrad entwickelt. Dies geschieht häufig mit Hilfe von Simulink-Modellen der Firma The Mathworks. Ziel ist ein optimal funktionierender Lösungsalgorithmus im Steuergerät. Zwangsläufig kommt es dabei zu einem Systemwechsel vom virtuellen Modell zur Nutzung des Algorithmus im realen Steuergerät.

Vom Modell zum Steuergerät

Wenn der Algorithmus im Modell den gewünschten Reifegrad erreicht hat, wird er ins Steuergerät integriert. Dazu kann der Entwickler aus dem Modell einen Quellcode generieren, der mit weiteren steuergeräterelevanten Codes kompiliert und danach ins Steuergerät geflasht wird. Für einen ersten Test und zur schnellen Optimierung von Funktionsparametern ist es aber wünschenswert, zunächst eine Kopplung zwischen Steuergerät und Modell zu schaffen. Dies erreicht man über das so genannte Bypassing. In **Bild 1** wird die Funktion $y=f(x)$ nicht im Steuergerät

berechnet, sondern außerhalb im Modell. Dabei muss das Modell ebenfalls auf einer Hardwareplattform laufen. Häufig sind dies spezielle echtzeitfähige Plattformen. Problem hierbei: Der Einsatz von Echtzeithardware für eine große Gruppe von Entwicklern ist sehr kostenintensiv. Stehen nicht genügend Hardwareplattformen zur Verfügung, nimmt die Entwicklungszeit stark zu.

Die Kopplung von Steuergerät und Modell ist oft unterschiedlich gelöst, so dass es in verschiedenen Projekten unterschiedliche Lösungen gibt. Um eine universell nutzbare Lösung zu schaffen, sind Standards notwendig, die

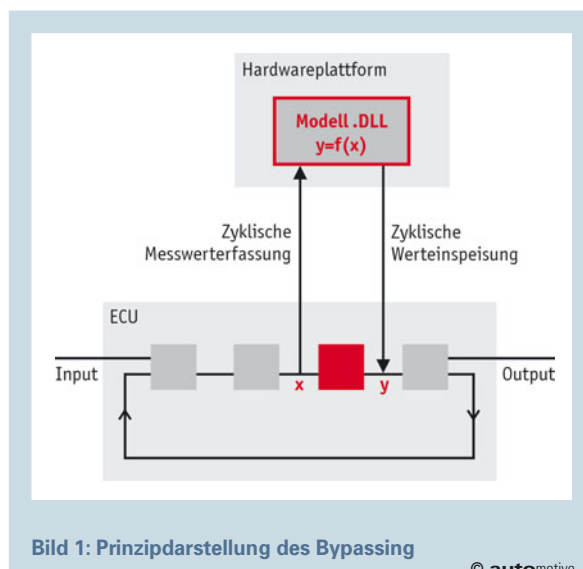


Bild 1: Prinzipdarstellung des Bypassing

das ASAM Interface-Modell liefert.

Zusammengefasst erfüllt eine optimale Lösung folgende Anforderungen:

- Messen und Kalibrieren des mathematischen Modells
- Messen und Kalibrieren des Codes im Steuergerät
- Kostengünstige Hardwareplattform für Modelle
- Wiederverwendbarkeit bei verschiedenen Projekten, unabhängig von der eingesetzten Hardware

Das ASAM Interface-Modell beschreibt drei Interface-Schnittstellen. Interface 1 stellt die Schnittstelle zwischen dem Mess- und Kalibrier-Tool (MC-Tool) und dem Steuergerät dar. Über das Interface 2 werden die A2L-Beschreibungsdateien angebunden, die das Steuergerät mit seinen Parametern und sonstigen Eigenschaften beschreiben. Interface 3 dient der Kommunikation mit anderen Software-Einheiten, beispielsweise einer Prüfstandanbindung.

Eine A2L-Datei beinhaltet Informationen zum Steuergerät, z. B. die Einstellungen des Kommunikationsprotokolls, das verwendete Checksummen-Verfahren sowie Namen und Speicheradressen von Parametern.

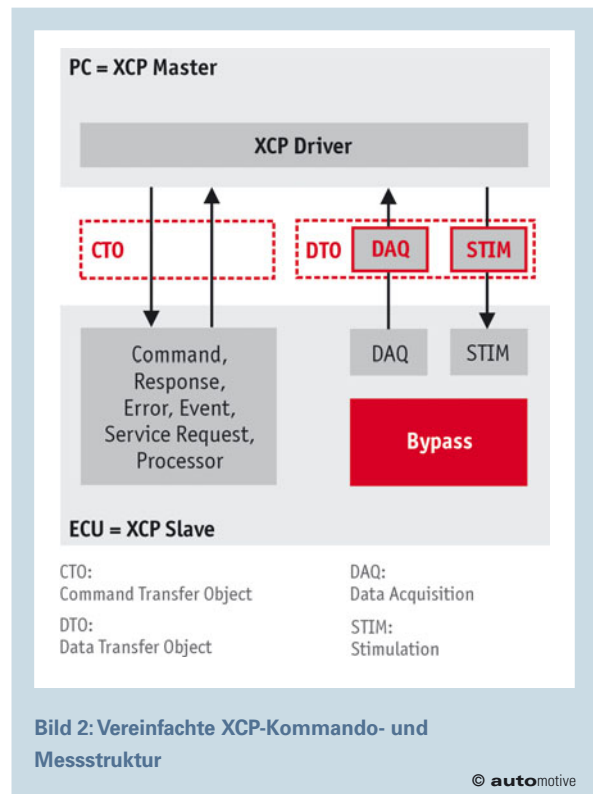
ASAM Interface 1 – Mit XCP Funktionen auslagern

XCP (Universal Measurement and Calibration Protocol) dient als Übertragungsprotokoll zwischen einem Steuergerät und einem MC-Tool. Der größte Vorteil von XCP ist die Trennung von Transport- und Protokollschicht. Über XCP lassen sich online steuergeräte-interne Größen messen und verstellen. Zur Zeit stehen die Standards XCP on SPI/SCI (SxI), XCP on CAN, XCP on USB und XCP on Ethernet (TCP/IP und UDP/IP) zur Verfügung. Anfang 2006 wird der optimierte Draft für XCP on FlexRay in die offizielle ASAM-Spezifikation übergehen.

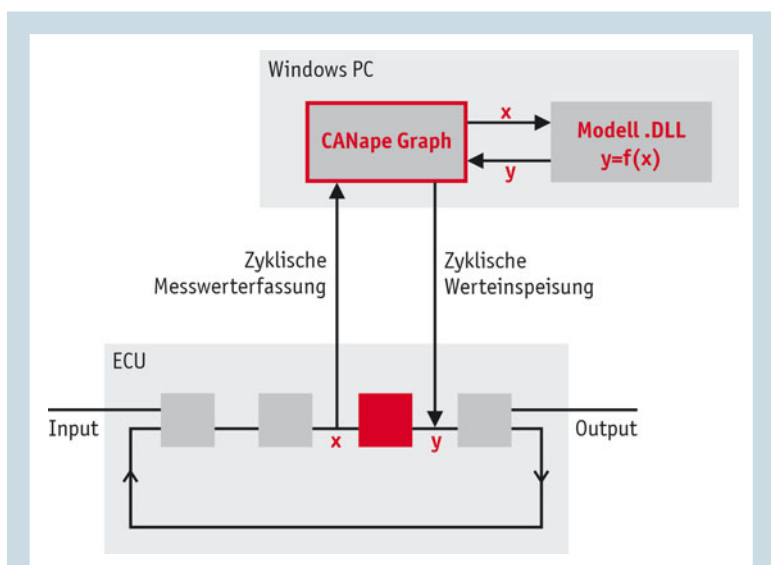
XCP ist als Master/Slave-Struktur realisiert. Der XCP-Slave befindet sich dabei im Steuergerät, der XCP-Master im MC-Tool. Über ein Command Transfer Object (CTO) sendet der Master ein Kommando über den Bus zum Steuergerät. Nach der Ausführung des angeforderten Dienstes quittiert der Slave über den gleichen Weg (Bild 2).

Bei der zyklischen Messung legen DAQ (Data Acquisition)-Listen fest, welche Signale und Parameter aus dem Steuergerät zyklisch zum Master übertragen werden. Das Zeitraster bezieht sich dabei auf so genannte Events, die im Steuergerät ablaufen. Ein Event wird immer dann ausgelöst, wenn ein entsprechendes Zeitintervall abgelaufen ist. Erfolgt die Messung der Signale zum Beispiel in einem 10-Millisekunden-Raster, so werden die aktuellen Werte der zu messenden Signale alle 10 Millisekunden aus den dazugehörigen Speicherstellen herausgelesen, in Botschaften verpackt und über den Bus gesendet.

Der XCP-Master definiert diese Messwertanforderung nur einmal, der Slave



führt sie dann automatisch und dauerhaft durch. Diese Art der zyklischen Messung bietet auch CCP (CAN Calibration Protokoll). Als Besonderheit ermöglicht XCP auch die Wertespeisung vom Master zum Slave. Man spricht dann nicht von Messung, sondern von der Stimulation des Steuergerätes. Das heißt, der Master übermittelt zyklisch Daten an den Slave, der diese dann in die entsprechenden Speicherstellen schreibt. Damit können Daten sowohl zyklisch vom Steuergerät empfangen als auch zyklisch an das Steuergerät gesendet werden.



Technische Umsetzung

Die Hardwareplattform, auf der ein Modell läuft, muss in vorhersagbaren Zeitzyklen eine bestimmte Anzahl an Rechenoperationen durchführen. Dabei darf die Zeitdauer nur gering variieren und eine bestimmte Zyklusdauer nicht überschreiten. Die preiswerteste Hardwareplattform ist sicherlich der PC des Anwenders. Doch genügt ein Standard-PC mit Windows-Betriebssystem den Anforderungen?

Durch die Integration des ASAM-Standards XCP als Kommunikationsprotokoll und einer offenen Schnittstelle zur Nutzung der mathematischen Modelle erfüllt das Mess-, Kalibrier- und Diagnosewerkzeug CANape Graph der Vector Informatik alle Grundvoraussetzungen für das XCP Bypassing.

CANape Graph führt die Größe x aus dem Steuergerät heraus, **Bild 3**. Das Modell berechnet dann den Wert y als Funktion von x und speist diesen Wert wieder in das Steuergerät ein. Das Herausführen der Größe x entspricht dem zyklischen Messen eines Signals. Das Einspeisen des Ergebnisses $y=f(x)$ erfolgt über die zyklische Stimulation. Das definierte und zyklische Auslesen der Signale aus dem Steuergerät mit den benötigten Zeitrastern erfolgt über

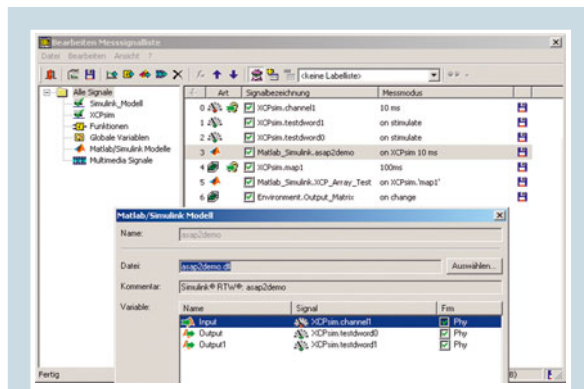


Bild 4: Zuordnung der I/Os zwischen mathematischem Modell und realen Signalen aus dem Steuergerät mit CANape Graph

© automotive

XP, Intel-Pentium-M 1,7 GHz und 512 MByte RAM wurde ein Testaufbau gemäß Bild 3 realisiert. Nur vier von 100 000 Bypass-Zyklen hatten eine Latenzzeit von mehr als 8 ms. Alle anderen Latenzzeiten lagen deutlich unter 3 ms (**Bild 5**). Das Bypassing mit CANape Graph und einem Standard-PC bietet somit Entwicklern von Steuergeräten optimale Bedingungen auch für schnelle Zykluszeiten.

Zusammenfassung

Mit XCP Bypassing lassen sich Funktionen in Modellen außerhalb eines Steuergerätes sehr gut berechnen und optimieren. Dies ermöglicht den Test in einer frühen Phase des Entwicklungsprozesses und ein Reifen des Modells unabhängig von der verwendeten Hardwareplattform. Durch die Integration der mathematischen Modellierung in das Mess-, Kalibrier- und Diagnosewerkzeug CANape Graph stellen PCs eine kostengünstige Basis für den Ausbau solcher Simulationslösungen dar. Die Simulation steht somit allen Anwendern zur Verfügung, die Modelle entwickeln.

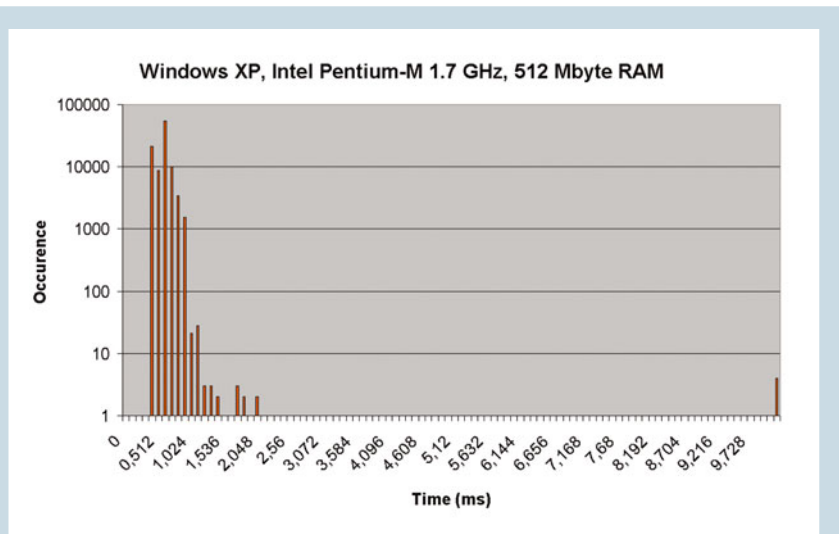


Bild 5: Darstellung der Zeitzyklus-Varianz

© automotive

XCP. CANape Graph übergibt die Daten direkt als Eingangsgrößen an das Modell, das als lauffähige DLL kompiliert wurde. Das Modell berechnet die gewünschte Funktion aus den Eingangsdaten und sendet sie wiederum zyklisch an das Steuergerät zurück.

Die Zuordnung der vom Steuergerät gesendeten Daten x als Eingangsgröße zu dem mathematischen Modell und der Ergebnisse $y=f(x)$ aus dem Modell zum Steuergerät erfolgt einfach über CANape Graph (**Bild 4**). Zur Optimierung der Algorithmen kann sowohl das Steuergerät als auch das Modell über XCP kalibriert werden.

Welche Zykluszeiten erlaubt nun ein Standard-PC? Die Zykluszeit ist die Zeitdauer zwischen Versand der Daten x aus dem Steuergerät und dem Eintrag der Daten y zurück in das Steuergerät. Bei einer PC-Umgebung mit Windows

Weiterführende Links:

- [1] www.asam.de: XCP-protokollspezifische Dokumente
- [2] www.mathworks.com: Informationen zu MATLAB/ Simulink, RealTime Workshop
- [3] www.vector-informatik.de: Demoversion CANape Graph, kostenloser und frei verfügbarer XCP-Treiber (Basisversion), umfassende Informationen zu den Themen Kalibrieren, Testen und Simulieren.



Dipl.-Ing. Andreas Patzer ist Business Development Manager bei Vector Informatik und dort unter anderem für das Mess-, Kalibrier- und Diagnosewerkzeug CANape Graph zuständig.