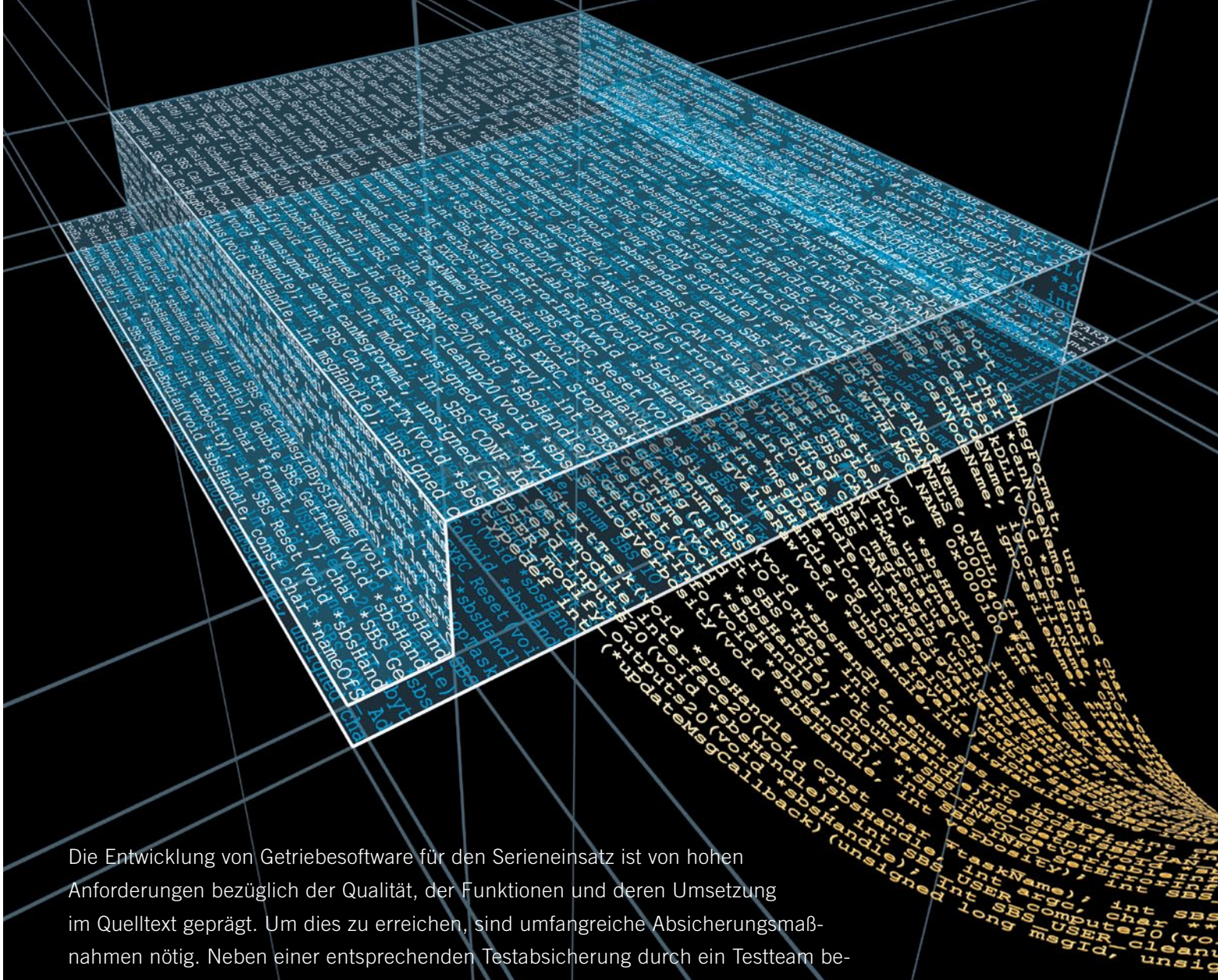
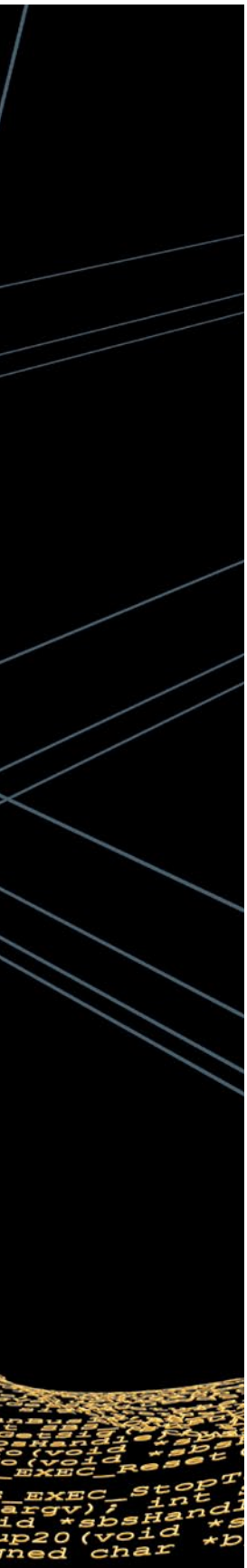
SCHNELLER ZUM VIRTUELLEN STEUERGERÄT



Die Entwicklung von Getriebesoftware für den Serieneinsatz ist von hohen Anforderungen bezüglich der Qualität, der Funktionen und deren Umsetzung im Quelltext geprägt. Um dies zu erreichen, sind umfangreiche Absicherungsmaßnahmen nötig. Neben einer entsprechenden Testabsicherung durch ein Testteam besteht die Notwendigkeit der Funktionsüberprüfung durch die Entwickler. Ergänzend zu HiL-Systemen bedienen sich Entwickler Software-in-the-Loop-Systemen. Eine Software namens Silver ermöglicht derartige Tests auf dem Entwicklerrechner ohne zusätzliche Hardware, bei direktem Zugriff auf den Steuergeräte-Quellcode sogar inklusive Code-Debugging. Die IAV beschreibt, wie sich ein virtuelles Steuergerät mit Silver schnell und kostengünstig aufbauen lässt.



AUTOREN

DR. ANDREAS JUNGHANNS
ist Mitglied der Geschäftsführung der QTronic GmbH in Berlin.

ROLAND SERWAY
ist Abteilungsleiter für Funktion und Software im Bereich Getriebe- und Hybridsysteme bei IAV in Berlin.

DR. THOMAS LIEBEZEIT
ist Funktionsentwickler bei IAV in Berlin.

MIRCO BONIN
ist Funktionsentwickler bei IAV in Gifhorn.

STEUERGERÄTESOFTWARE

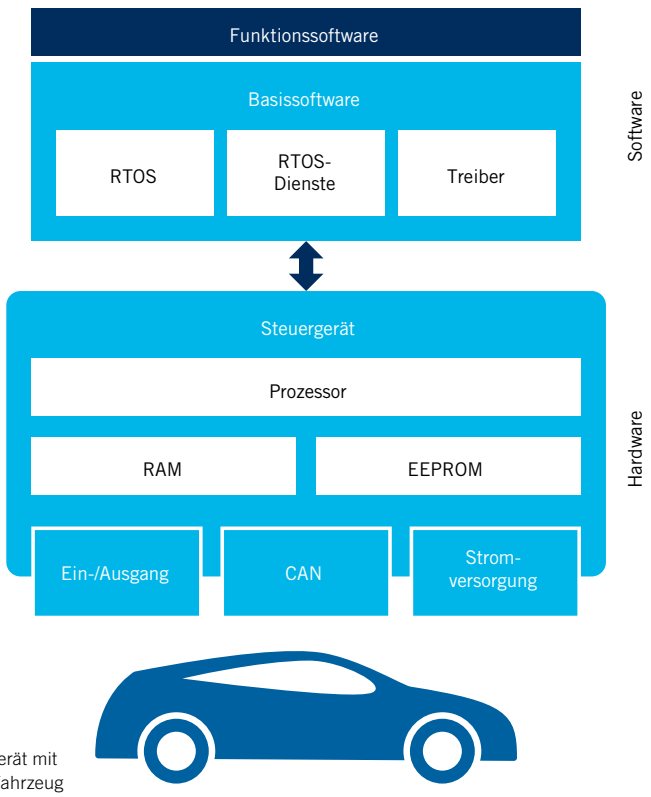
Die Basissoftware in einem Steuergerät dient als Abstraktionsebene zwischen der Funktionssoftware und der Steuergerätehardware, ❶. Die Basissoftware umfasst das Echtzeitbetriebssystem (RTOS) und dessen Dienste sowie die Treiberschicht für den Zugriff auf die Steuergerätehardware und angeschlossene Bussysteme.

Abgesehen von der Bereitstellung von Hardwarezugriffen durch geeignete Treiberschichten, hat ein eingebettetes Betriebssystem vor allem die Aufgabe, Softwarefunktionen, die sogenannten Tasks, entweder nach gegebenem zeitlichen Rhythmus oder als Folge bestimmter Ereignisse aufzurufen. Dabei müssen Deadlocks verhindert werden, die durch ungünstige Verblockung bei Resourcenzugriffen entstehen können. Weitere Dienste, wie zum Beispiel Bootloader, Reset bei Ausnahmen und EEPROM-Zugriff werden auch bereitgestellt.

Trotz intensiver Bemühungen zur Standardisierung von Steuergerätesoftware [1] gibt es auf dem Steuergerätemarkt immer noch ein umfangreiches Angebot an Systemen der verschiedenen Zulieferer im Einsatz bei den OEMs. Ein Trend zur Trennung zwischen Basissoftware-Verantwortung beim Steuergerätezulieferer und Funktionssoftware-Verantwortung beim OEM ist seit Jahren verstärkt zu verzeichnen.

VIRTUELLES STEUERGERÄT UND SILVER-BASIC-SOFTWARE

Eine klare Trennung der Funktionssoftware von der Basissoftware erlaubt es, die Basissoftware auszutauschen und die darun-



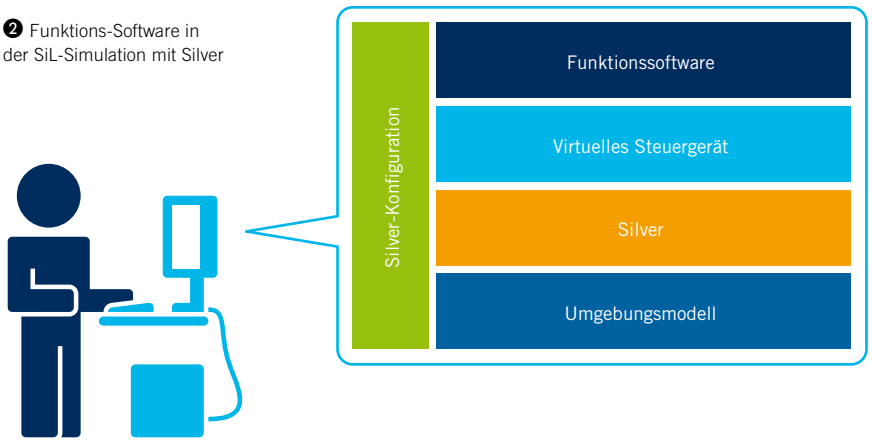
❶ Reales Steuergerät mit Basissoftware im Fahrzeug

ter liegenden Hardware zu simulieren, ❷. Die Grundlage dafür bildet die Virtualisierung des Steuergerätes und damit der Schaffung einer Entwicklungsplattform auf dem Laptop des Funktions- und Softwareentwicklers. Die Verlagerung von Entwicklungs- und Testumfängen in die virtuelle Umgebung erlaubt eine Reihe von methodischen Vorteilen, die Wichtigsten davon:

- die Entkopplung und Parallelisierung großer Entwicklungsteams über Abteilungen oder gar Organisationen hinweg
- die einfache und preiswerte Vervielfältigung eines bestimmten Entwicklungsstandes
- die getrennte Validierung einzelner Veränderungen an der Software ohne die Einflüsse anderer, oft unabhängiger Veränderungen anderer Entwickler
- die Analyse von komplexen Funktionen in Verbindung mit detaillierten Streckenmodellen ohne Echtzeitzwang [1]
- das Aufprägen nahezu beliebiger Hardwarefehler und Timingfehler in das Streckenmodell um Fehlerreaktionen zu überprüfen
- die Analyse auf Basis von Resimulation anhand von Fahrzeugmessungen
- der Einsatz der effizienten Methoden des Desktop-Debuggings für Embedded Software.

Die Virtualisierung von Funktionssoftware ist seit vielen Jahren ein Geschäftsfeld der QTronic GmbH. Dabei entstehen virtuelle Steuergeräte, die auf der Co-Simulationsplattform Silver mit komplexen Streckenmodellen zu Gesamtsystemen gekoppelt werden an denen typische Ingenieursaufgaben ausgeführt werden können. Silver erlaubt, ein solches virtuelles Steuergerät mit Produktionsdatensätzen zu flashen, mit CCP/XCP-basierten MCD-Tools (zum Beispiel Inca, CANape oder CALDesk) zu messen und zu kalibrieren, mit Daten aus dem Feld zu stimulieren und vieles andere mehr [2].

Die wiederholte Virtualisierung von Funktionssoftware hat zu einem großen Satz an wiederverwendbaren Bibliotheken geführt, die heute größtenteils als Silver-Basic-Software (SBS) in aufbereiteter und dokumentierter Form mit Silver ausgeliefert werden und so QTronic-Kunden zur Verfügung stehen.

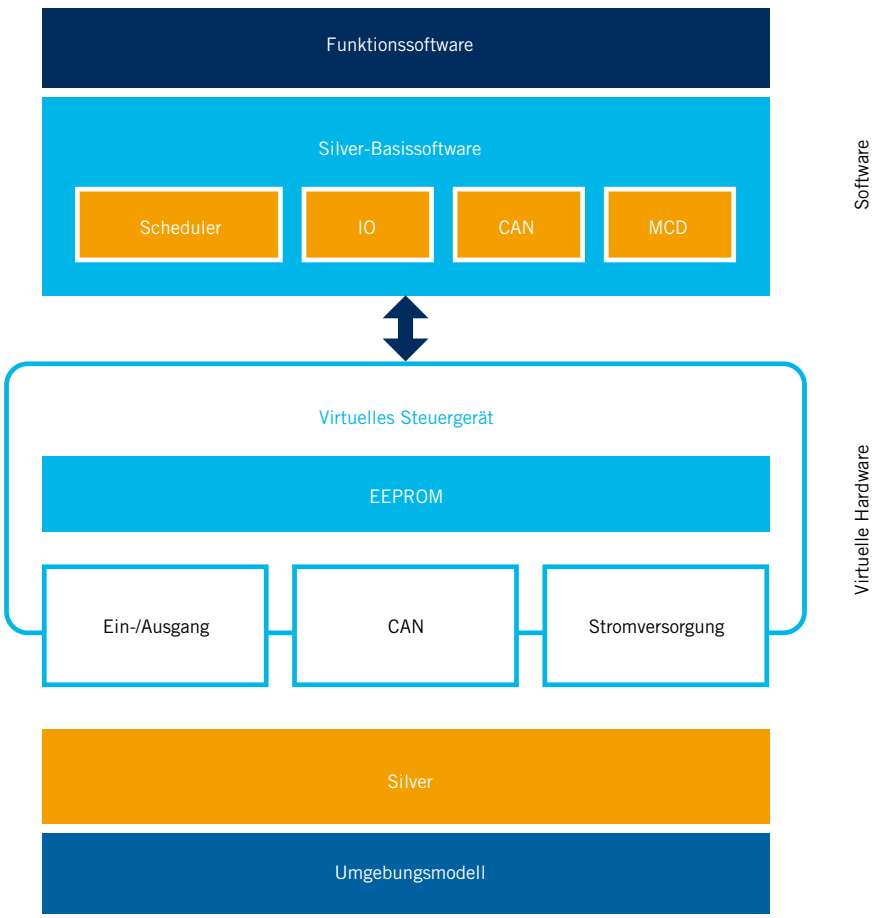


❷ Funktions-Software in der SiL-Simulation mit Silver

IMPLEMENTIERUNG DES VIRTUELLEN STEUERGERÄTS

Diese allgemeinen SBS-Bibliotheken werden dann genutzt, um das Steuergerät zu virtualisieren. Dabei werden die SBS-Funktionen verwendet, um die Funktionsrufe aus der Funktionssoftware in die Original-Basissoftware zu ersetzen oder Hardware-Aspekte des Steuergerätes mit wenigen Zeilen C-Code zu implementieren, ❸.

In der Verwendung von SBS werden zwei Phasen unterschieden: die Konfiguration und die Nutzung. Im Unterschied zu generativen Ansätzen, die die Konfigurationsphase offline legen, dann Quelltext



❸ Virtuelles Steuergerät mit Silver-Basic-Software in der SiL-Simulation

generieren, kompilieren, linken und dann erst ausführen, werden beide Phasen bei SBS zur Laufzeit ausgeführt. Dadurch wird es möglich (jedoch nicht nötig), in die Konfiguration von SBS durch Nutzereingaben am Anfang der Laufzeit aktiv einzugreifen und das Verhalten und sogar die Schnittstellen des virtuellen Steuergeräts von Lauf zu Lauf maßgeblich zu ändern ohne es neu kompilieren zu müssen. Während der Ausführung werden dann Interface-Beschreibungen und andere Konfigurationsdaten interpretiert. Ein solches, laufzeitkonfigurierbares Steuergerät hat eine längere Lebenserwartung als offline generierte und reduziert damit den nötigen Austausch zwischen den Entwicklungspartnern.

Im folgenden wird beispielhaft eine minimale Anbindung von Steuergerätesoftware mittels SBS vorgestellt. Diese basiert auf der Umsetzung einer Software-in-the-Loop-Simulation für eine Getriebe-Funktionssoftware für Doppelkupplungsgetriebe der IAV im Kundenauftrag [3]. Zu Illustrationszwecken wurde das Beispiel stark vereinfacht und spiegelt bei weitem nicht die Komplexität von IAV-Kundensteuergeräten wider. Die getroffenen Aussagen gelten jedoch auch für reale Konfigurationen.

❹ und ❺ verdeutlichen die Implementierung des virtuellen Steuergeräts anhand des vom Anwender erstellten Quelltexts. Dieser untergliedert sich in die SBS-Konfiguration, die Silver-Compute-Funktion, die einmal pro Silver-Macro-Schritt aufgerufen wird, und die Nachbildung der Bios-Funktionen.

Die ersten Zeilen des Beispiels zeigen, wie das SBS-Scheduling konfiguriert wird. Es umfasst hier die Nutzung von zwei Tasks mit jeweils 10 ms Zeitscheiben. Beide Tasks werden durch eine positive Flankenänderung einer Silver-Variable ausgelöst. Die erste Task startet sofort und der andere nach Ablauf einer 5 ms Verzögerung. Mit dieser Methodik können selbst komplexe Scheduling-Algorithmen einfach und schnell konfiguriert werden.

Die Input- und Output-Variablen werden ebenfalls über entsprechende SBS-Funktionen spezifiziert. Dabei werden Silver-Variablen auf globale Variablen der Funktionssoftware verlinkt. Ein solcher Link stellt sicher, dass, je nach Typ der Variable, SBS zur richtigen Zeit die Werte der Variablen von Silver (Input) oder zu Silver (Output) kopiert und gegebenfalls mittels gain und offset skaliert, von Hand oder mittels A2L-Datei spezifiziert. Das heißt: Eine einfache Zeile Quelltext genügt, um eine Input- oder Output-Variable eines Steuergeräts mit Silver zu verbinden.

SBS bildet desweiteren einen Standard-CAN-Stack nach. CAN-Botschaften lassen sich als einzelne Nachrichten oder als ganze DBC definieren. Letzteres stellt Zugriff auf eine Menge von Nachrichten und deren Signalen für den Steuergeräte-Netzwerkknoten zur Verfügung. Damit wird der Konfigurationsaufwand ebenfalls drastisch reduziert, indem man existierende Datenquellen zur Spezifikation des virtuellen Steuergerätes nachnutzt.

Die Koppelung des CAN an die Funktionssoftware findet über eine Nachbildung der entsprechenden Bios-Methoden statt (Bios_RX_CAN). Der Zugriff auf die einzelnen CAN-Signale erfolgt dabei in den Ersatz-Bios-Methoden und nutzt wieder SBS-Funktionalität [5].

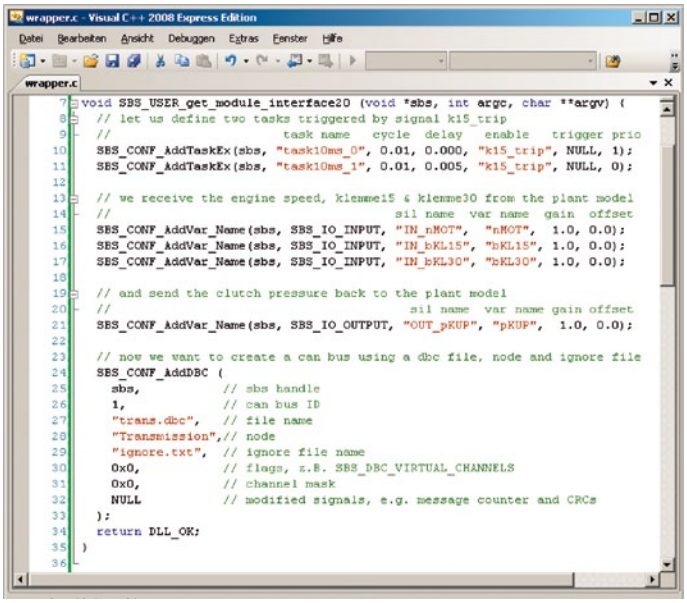
Steuergeräte-Resets werden über die SBS-Funktion SBS_EXEC_Reset emuliert, die sich über eine Auswertung der Klemmen-Signale, die als Silver Variablen zur Verfügung stehen, triggern lässt. Damit wird es mit wenigen Zeilen Quellcode möglich, die Funktionssoftware auf ihre Reset-Sicherheit (beispielweise im Fehlerfall) zu untersuchen.

Der Anwender ist bei der Virtualisierung nie zu einer ausschließlichen Verwendung von SBS-Funktionen gezwungen. Im Anwendungsbeispiel wird das EEPROM projektspezifisch durch eigenen C-Code und über Zugriff auf eine Textdatei realisiert, da SBS dies noch nicht unterstützt. Eigene Implementierungen bedeuten aber immer einen Mehraufwand im Vergleich zur Nutzung von SBS-Funktionen.

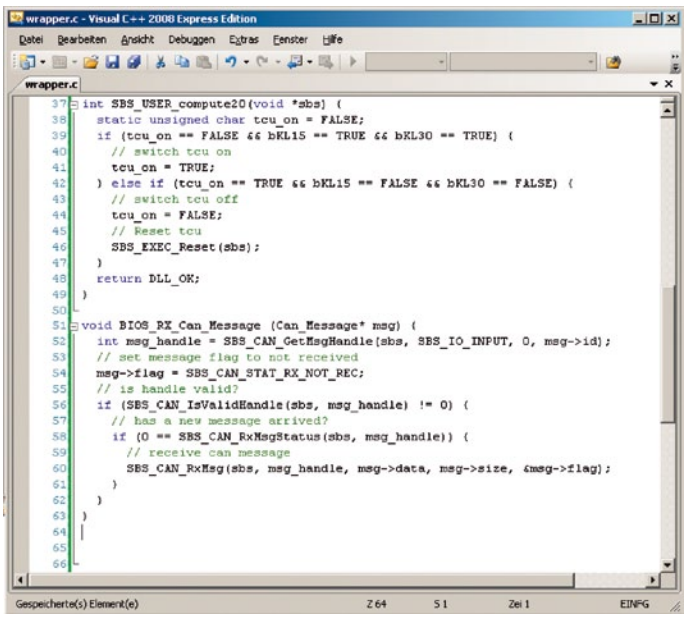
Für die Erzeugung des virtuellen Steuergeräts als Silver-Module-DLL wird Silver mit einem spezialisierten, sehr leicht konfigurierbaren build-Programm (cbuild) ausgeliefert. Damit lassen sich komplexe Verzeichnisstrukturen nach C-Dateien durchsuchen, kompilieren und dann zu einer Silver-Modul-DLL verlinken.

Diese DLL umfasst damit sowohl die Funktionssoftware als auch das virtuelle Steuergerät. Durch Anpassung der Original-A2L-Datei an die Silver-Module-DLL lässt sich das virtualisierte Steuergerät weiterhin mit ASAP/MCD-basierten Werkzeugen kalibrieren und mitmessen, genau wie es im Fahrzeug oder am HiL möglich ist.

Die Virtualisierung des Steuergerätes erfolgt nur einmal für jedes Projekt. Einmal erstellt können Änderungen an der Funktionssoftware mit dem gleichen virtuellen Steuergerät kompiliert werden. Nur bei SiL-relevanten Änderungen an der Schnittstelle zur Basissoftware müssen Anpassungen vorgenommen werden. Das heißt auch, dass die Virtualisierung idealerweise von einem Softwareentwickler verantwortet wird. Funktionsentwickler nutzen dann den entstandenen Build-Prozess für ihre tägliche Arbeit.



❹ C-Code des virtuellen Steuergeräts



❺ C-Code des virtuellen Steuergeräts (Fortsetzung)

FAZIT UND AUSBLICK

SBS stellt Jahre an Entwicklungserfahrungen den Anwendern von Silver zur Verfügung und ermöglicht auf eine effiziente und elegante Art, Funktionssoftware mit einem virtuellen Steuergerät zu verbinden. Durch die Kopplung an Streckenmodelle aus verschiedensten Simulationswerkzeugen werden zudem Systemeigenschaften erlebbar und testbar. Das IAV- Beispiel zeigt, dass sich mit wenig Aufwand Steuergerätequellcode von Anwendern selbst virtualisieren lässt.

SBS ist damit ein wichtiger Beitrag, das Kosten-Nutzen-Verhältnis für SiL-basierte Entwicklungsarbeiten weiter zu verbessern. So wird der verständliche Wunsch zu weiterem Front-Loading von Test- und Validierungsschritten auch für Gesamtsysteme wirtschaftlich realisierbar.

Die QTronic plant weitere Bios-Dienste in SBS zu unterstützen, wie Eeprom-Zugriffe und Fehlercodehandling, um die Virtualisierung noch weiter zu vereinfachen. IAV wird dies zum Vorteil ihrer Kunden für die frühzeitige und effiziente Überprüfung der von ihr entwickelten Steuergeräte-Funktionen verwenden.

LITERATURHINWEISE

[1] Martin Neumann, M., Mario Nass, M., Carsten Paulus, C., Mugur Tatar, M.: Absicherung von Steuerungssoftware für Hybridsysteme, 5. Fachtagung AUTOREG 2011 Steuerung und Regelung von Fahrzeugen und Motoren, 22.-23.11.2011, Baden-Baden, Germany.

[2] http://de.wikipedia.org/wiki/AUTOSAR

[3] E. Chrisofakis, E. et. al.: Simulation-based development of automotive control software with Modelica. 8th International Modelica Conference, 20-22.03.2011, Dresden, Germany.

[4] Liebezeit, Th., Bräuer, J., Serway, R., Junghanns, A.: Virtual ECUs for developing automotive transmission software, 10th CTI Symposium Innovative Fahrzeug-Getriebe Hybrid- und Elektro-Antriebe, 5.-8.12.2011, Berlin, Germany.

[5] Liebezeit, Th., Junghanns, A., Bonin, M., Serway, R.: Software-in-the-Loop using virtual CAN busses: Current solutions and challenges, 5. Tagung Simulation und Test, 10-11.05.2012, Berlin, Germany