Vom Modell zum Seriencode

In der Automobilelektronik wird die Funktionsentwicklung für Steuergeräte (ECUs) heute immer mehr von Entwurfsmethoden bestimmt, die auf mathematischen Modellen der betreffenden Steuer- und Regelkreise basieren. Diese Tendenz beruht vor allem darauf, dass im Gegensatz zu formalen Beschreibungen von Funktionen nichtformale Beschreibungen wie zum Beispiel reine Textdarstellungen häufig zu Qualitätsmängeln im gesamten Entwicklungsprozess führen können.

Modellbasierter Reglerentwurf

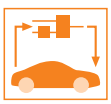
Beim modellbasierten Entwurf werden die Regler grafisch mit Hilfe von Block- und Zustandsdiagrammen dargestellt. Am weitesten verbreitet zur Beschreibung der Modelle sind dabei die Software-Werkzeuge MATLAB®/Simulink®/Stateflow® von The MathWorks. Durch die Integration von Simulink (für Blockdiagramme) und Stateflow (für Zustandsdiagramme) können die unterschiedlichsten Arten von komplexen Steuerungen und Regelungen in einem einzigen Modell miteinander kombiniert werden.

Durch Branchenexperten der Automobilelektronik wurde in den letzten Jahren herausgefunden, dass zwischen 15 und 40 Prozent aller Software-Fehler, die bei der Serienüberführung von Steuergeräte-Funktionen gefunden wurden, ihre Ursache in unvollständigen und mehrdeutigen Spezifikationen hatten. Zwischen 40 und 60 Prozent aller Probleme entstanden wiederum durch Programmierfehler während der Software-Implementierung. Der niedrigere Wert von 40 Prozent wurde hier vor 2 Jahren von einem Lkw-Hersteller berichtet. Im Bereich Pkws haben zwei verschiedene Fahrzeughersteller (OEM) – der eine im Jahr 1999, der andere im Jahr 2002 – unabhängig voneinander eine Fehlerquote von 60 Prozent bestätigt. Bei der Implementierung und Programmierung von Steuergeräte-Software kann offensichtlich noch einiges verbessert werden.

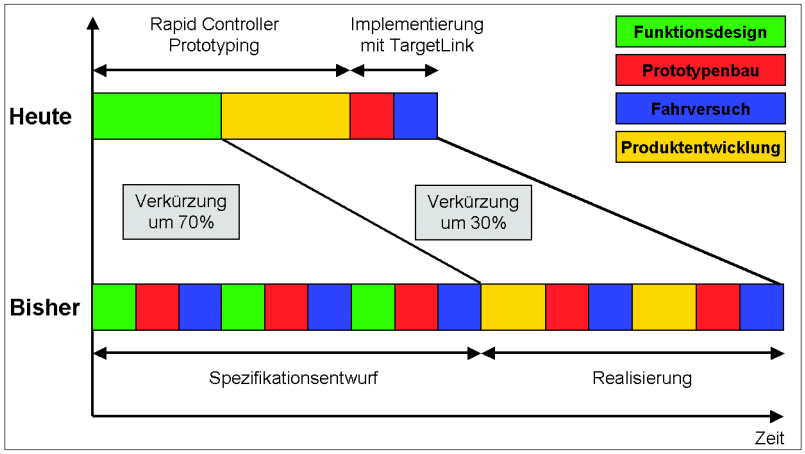
Mit der Verwendung mathematischer Modelle – die letztlich ausführbare (d. h. simulierbare) Spezifikationen sind – lässt sich ein Großteil der durch mehrdeutige und unvollständige Spezifikationen hervorgerufenen Probleme ausräumen. Verständlichkeit und Transparenz innerhalb des Entwicklungsprozesses steigen für alle Beteiligten erheblich. Die Modelle sind gleichzeitig Grundlage für Echtzeit-Anwendungen im Rapid Control Prototyping (RCP). Sehr komplexe Regel- und Steuerkreise bleiben darüber hinaus mit Hilfe mathematischer Modelle selbst dort noch kontrollierbar, wo mit nichtformalen Beschreibungen schon jeglicher Überblick verloren wäre. Vor allem in den Automobil-Entwicklungsbereichen Antriebsstrang und Fahrwerk besteht seit etwa 10 Jahren ein deutlicher Trend hin zum modellbasierten Entwurf. Jedoch werden in der Industrie immer noch verbreitet nichtformale Beschreibungen für Spezifikationen verwendet – mit einem großen Aufwand für Nachbesserungen innerhalb des gesamten Entwicklungsprozesses als Folge.

Die mathematischen Modelle dienen neben der reinen Beschreibung der Steuer- und Regelkreise einerseits als Grundlage für menschliche Programmierer. Genauso können sie aber auch von automatischen Code-Generatoren in qualitativ hochwertigen Steuergerätecode umgesetzt werden. Gute Code-Generatoren reduzieren nicht nur die Entwicklungszeit, sondern bringen auch eine erhöhte Software-Qualität hervor und verhindern menschliche Programmierfehler. Zur Zeit werden automatische Code-Generatoren primär für neue funktionale Software-Bestandteile in elektronischen Steuergeräten eingesetzt, d. h. für die Steuerungs- und Regelungsfunktionen. Die hardwarenahe Software und die Software-Infrastruktur werden manuell erstellt. Da der Funktionscode in Steuergeräten einen Anteil von etwa 70 Prozent einnimmt, hat die Verwendung automatischer Code-Generatoren in der Automobilelektronik eine große Bedeutung gewonnen, sowohl bei der Erhöhung der Code-Effizienz und -qualität in Steuergeräten als auch für die Verbesserung der Sicherheit elektronischer Steuergeräte.

Code für das Rapid Control Prototyping (RCP)



Wenn möglich sollten Regler- und Steuerungsentwürfe immer mit einem RCP-System erprobt werden. Dafür wird der Regler auf der experimentellen Echtzeit-Hardware implementiert, entweder als Ersatz des geplanten Steuergerätes (Fullpass-



Effizienz-Zuwachs durch RCP und TargetLink bei einem japanischer Fahrzeughersteller (OEM)

Prototyping) oder als separate, ausgelagerte Steuergeräte-Funktion (Bypass-Prototyping). Für die Implementierung wird eine automatische Code-Generierung benötigt. Jedoch ist dieser automatisch erzeugte Code für das RCP im Vergleich zur automatischen Seriencode-Generierung für möglichst viele Freiheiten beim Testen ausgelegt und ist somit nicht für die bei Seriensteuergeräten limitierten Ressourcen wie Speicher oder Ein- und Ausgänge optimiert. Dafür ist RCP-Hardware besonders leistungsstark und flexibel. Die MicroAutoBox – ein RCP-System von dSPACE – kann darüber hinaus durch die kompakten Abmessungen und die Robustheit direkt im Fahrzeug eingesetzt werden. Obwohl es Überdeckungen zwischen Code-Generatoren für RCP und solchen für Seriencode gibt, sind kommerzielle Code-Generatoren in der Regel auf eine der beiden Anwendungen spezialisiert.

Die Bedeutung des RCP liegt vor allem in der Verifizierung des Reglerentwurfs und der Erzeugung einer tatsächlich ausführbaren Spezifikation. Mögliche Probleme durch fehlerhafte Spezifikationen werden so ausgeschaltet. Der gesamte Entwicklungsprozess wird zudem effizienter, denn Ideen können schnell und unkompliziert mit der RCP-Hardware ausprobiert und Erfolg versprechende Lösungswege zügig verifiziert werden. Ein japanischer OEM-Hersteller berichtete, dass durch RCP die Entwurfsphase um 70 Prozent verkürzt werden konnte. Durch den Einsatz eines Seriencode-Generators konnten 30 Prozent der Implementierungszeit eingespart werden, obwohl der Seriencode-Generator erstmals eingesetzt wurde, noch eine Lernkurve zu durchlaufen war und der Prozessor des Steuergerätes nur Festkomma-Arithmetik verarbeiten konnte.

Anforderungen an Seriencode-Generatoren

Seriencode-Generatoren müssen eine ganze Reihe von Anforderungen erfüllen, die beim RCP nicht so wichtig sind. Im Gegensatz zu handprogrammiertem Code müssen Seriencode-Generatoren automatisch Code mit äußerst hoher Effizienz erzeugen. Darüber hinaus erwartet man völlige Fehlerfreiheit, Reproduzierbarkeit und eine gute Dokumentation des erzeugten Codes. Die hohe Code-Qualität muss auch über Versionen des Code-Generators hinweg eingehalten werden. Dies ist eine absolut nichttriviale Forderung, solange sich die Modellierungswerkzeuge noch erheblich weiterentwickeln müssen und auch der Seriencode-Generator schon allein deshalb permanenter Weiterentwicklung unterliegt. Oft kommt es zudem vor, dass schon längst Code für bestimmte Modellteile existiert. Dieser Code muss dann in den automatisch erzeugten Code integriert werden können (und auch umgekehrt). Für maximale Flexibilität bei der Code-Erzeugung ist es für den Entwickler erforderlich, Variablenklassen selber definieren zu können und Variablen frei nach den Projekterfordernissen und vorhandenen Konventionen benennen zu können. Außerdem möchte man bei der Code-Partitionierung so wenig wie möglich eingeschränkt sein, um Funktionen wiederverwenden und Multitasking durchführen zu können. Es sei auch noch die oft geforderte Anbindung an Applikationssysteme erwähnt, wofür ASAP2-Kompatibilität notwendig ist. Alles in allem ist automatische Seriencode-Generierung angesichts der hier geschilderten Anforderungen eine sehr anspruchsvolle Aufgabe für das entsprechende Software-Werkzeug.

Effizienz von automatisch erzeugtem Seriencode

Lange Zeit mochten Software-Spezialisten nicht glauben, dass Seriencode-Generatoren einmal so gut wie menschliche Programmierer sein oder diese sogar übertreffen würden. Schlechte Erfahrungen mit frühen Code-Generatoren haben diese Skepsis gefördert. Heute können Seriencode-Generatoren sehr wohl die Effizienz von handprogrammiertem Code für die funktionalen Bestandteile von Steuergeräten erreichen. TargetLink, der Seriencode-Generator von dSPACE, hat bereits viele Vergleichstests hinter sich, und Benutzer in aller Welt haben berichtet, dass die Effizienz des mit TargetLink erzielten Codes sich hinsichtlich Tempo und Speicherbedarf ohne weiteres mit handprogrammiertem

Effizienz von TargetLink-Code: Vergleichstest von Delphi

ROM	RAM	Stack	Geschwindigkeit
0,96 – 1,1	0,97 – 1,2	1,2 – 1,25	0,75 – 1,2

Generierter Code besser als handprogrammierter Code

Vergleich der Effizienz von handprogrammiertem und generiertem Code (Antriebsstrang / Chassis)

Code messen kann. Oft wurden sogar die Ergebnisse der handcodierten Software übertroffen. Denn während der Seriencode-Generator zu jeder Zeit den Code optimiert, haben menschliche Programmierer oft nicht die Zeit, um nach wiederholten Modelländerungen immer und immer wieder den Code zu optimieren. Als Besonderheit ist zu erwähnen, dass TargetLink für verschiedene Compiler-Mikrocontroller-Kombinatio-

nen jeweils automatisch das optimale Codierungsmuster auswählen kann, und zwar ohne vom ANSI C Standard abzuweichen. Bei einer Chassis-Anwendung auf einem Infineon C167-Prozessor konnten durch die automatische Wahl des optimalen Codierungsmusters 27 Prozent Geschwindigkeitsvorteil erreicht und der Speicherbedarf um 10 Prozent reduziert werden.

TargetLink ist außerdem in der Lage – in diesen Fällen abweichend vom ANSI C Standard –, sich auf verschiedenste Compiler-Erweiterungen (spezielle Datentypen, spezielle Funktionen) einzustellen oder auch vereinzelte und gut begründete Einfügungen in Assembler-Sprache vorzunehmen, wenn die Effizienz des Codes dadurch erheblich gesteigert werden kann.

Vergleichstest: Addition zweier Festkommazahlen mit Sättigung

Methode		Lösung 1	Lösung 2	Lösung 3	Lösung 4
Motorola M68332	Ausführungszeit	2.2 µs	2.3 µs	1.8 µs	n/a
	Codegröße	34 Bytes	28 Bytes	20 Bytes	n/a
Infineon C167	Ausführungszeit	1.8 µs	4.0 µs	1.0 µs	n/a
	Codegröße	36 Bytes	60 Bytes	20 Bytes	n/a
Infineon TriCore	Ausführungszeit	1.2 µs	1.0 µs	n/a	<0.05 µs
	Codegröße	48 Bytes	48 Bytes	n/a	10 Bytes

Takt: 20.0 MHz, kein Wartestatus

Das beste Codierungsmuster wird ausgewählt

Lösung 1: ANSI C, Plausibilitätsprüfung
Lösung 2: ANSI C, Vergrößerung der Wordbreite
Lösung 3: Inline Assembly Code, Prüfen des Überlauf-Flags
Lösung 4: Spracherweiterung, Sättigungsdatentyp

Auswahl des optimalen Codierungsmusters in TargetLink (Vergleichstest)

Die hier beschriebenen Optimierungen („Target Optimization“) sind spezielle Features von TargetLink und sollten nicht mit sogenannter Target Integration Software verwechselt werden, wie sie von manchen Code-Generatoren angeboten wird. Target Integration Software umfasst meist nur Make-Files und einige Hilfswerkzeuge. Hingegen bedeutet „Target Optimization“ einen erheblichen Entwicklungsaufwand beim Hersteller des Code-Generators. Durch „Target Optimization“ werden besonders für kleinere Mikrocontroller mit beschränkten Ressourcen sowie für Prozessoren mit spezieller Hardware (z. B. integrierten Features für digitale Signalverarbeitung) hervorragende Optimierungsergebnisse erzielt.

Da hinsichtlich der Portabilität das Ziel darin bestehen sollte, Modelle zu portieren und nicht den Code selbst, spielen Abweichungen vom ANSI C Standard für die automatische Code-Generierung eigentlich keine große Rolle. Solche Abweichungen entsprechen zwar nicht den im Allgemeinen vorzufindenden bisherigen Programmierrichtlinien, denn diese wurden zur Disziplinierung der menschlichen Programmierer eingeführt. Der Code kann aber andererseits zu jeder Zeit passend für einen neuen Mikrocontroller oder eine neue Steuergeräte-Umgebung wieder generiert werden („Just-in-Time-Generierung“). Es kann dann sogar Code für einen Festkomma-Controller generiert werden, wenn vorher für einen Fließkomma-Controller entwickelt wurde und umgekehrt. Ein japanischer Automobilelektronik-Zulieferer hat eine Evaluierung von TargetLink durchgeführt mit Blick auf die Portierung von Reglerfunktionen zwischen verschiedenen Mikrocontrollern. Für die mit Simulink/Stateflow beschriebenen Funktionen eines elektronischen Stabilitätsprogramms (ESP) wurde mit TargetLink Code für vier verschiedene Mikrocontroller erzeugt. Selbst ohne spezielle Optimierung („Target Optimization“, s. o.) konnten hinsichtlich der Effizienz des Codes exzellente bis akzeptable Ergebnisse erreicht werden. Der Aufwand für die Portierung reduzierte sich um 75 Prozent im Vergleich zu vorher, als noch keine automatische Code-Generierung eingesetzt wurde.

Es gibt Situationen, in denen grafisch dargestellte, mathematische Modelle die Code-Generatoren vor besondere Herausforderungen stellen. Software-Werkzeuge wie Simulink und Stateflow lassen den Anwendern große Freiheiten darin, auf welche Art und Weise eine bestimmte Funktionalität beschrieben wird. Dadurch können auf Code-Ebene sehr einfache Operationen wie if-then-else-Konstrukte im grafischen Modell komplexer aussehen und auch auf verschiedene Weise modelliert werden (z. B. Logik-Blöcke und Schalter oder Zustandsautomaten, Art der Zusammenschaltung der Blöcke). Trotzdem erwartet man vom Code-Generator effizienten Code. TargetLink ist speziell für solche Situationen gerüstet und kann die Bedeutung von Modellkonstrukten über die Grenzen zwischen Zustandsdiagrammen und Flussdiagrammen hinweg erfassen.

Mit dem Processor-in-the-Loop-Modus bietet TargetLink den Benutzern die Möglichkeit, den generierten Code genau zu analysieren und mögliche Auswirkungen des eigenen Modellierungsstils zu betrachten. Im Processor-in-the-Loop-Modus wird der generierte Code auf einem realen Mikrocontroller ausgeführt, mit demselben Kern wie dem Seriensteuergerät. Mit Hilfe der automatischen Instrumentierung können dann die Ausführungszeiten pro Schritt, der Stack-Speicherbedarf und der RAM-Speicherbedarf aufgezeichnet, grafisch oder numerisch analysiert und nachträglich verändert werden.

Qualitäts- und Sicherheitsaspekte

In Unternehmen, die noch keine praktischen Erfahrungen mit der automatischen Seriencode-Generierung sammeln konnten, bestehen häufig Vorbehalte hinsichtlich der Sicherheit und der Qualität des automatisch erzeugten Codes. Dort, wo TargetLink bereits benutzt

Größe der Anwendungen

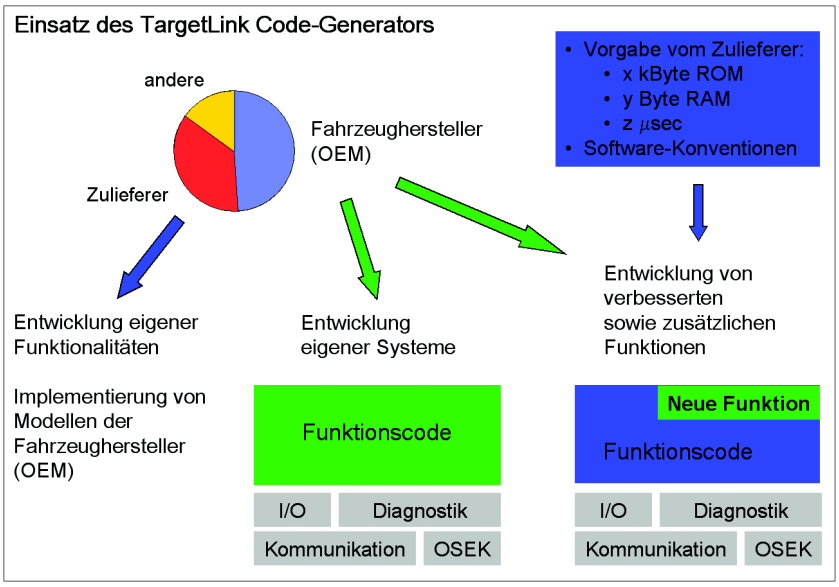
Anwendung	Größe der generierten Anwendung	Gesamter generierter Steuergeräte-Code in Prozent	µProzessor
MAN: Diesel-Rußpartikel-Filter-Steuerung (CRT)	82 KB ROM	80 % inkl. Diagnose, restliche 20 % Treiber und OS	Infineon C167 16 Bit
Magneti Marelli: GDI-Motorsteuerung	70 000 Code-Zeilen	70 % Funktions- und Diagnose-Code	SH 7055 – Festkomma
Magna-Steyr: Allradantrieb	3 258 Blöcke, 57 Statecharts, 160 Zustände, 45 KB Code, 1 528 KB RAM	alle, ausgenommen Low-Level-Treiber usw.	Motorola HCS12

Neue Anwendungen:
tendenzieller Anstieg des Prozentanteils an automatisch generiertem Code

wird, hat sich jedoch die Zuverlässigkeit des auf diese Weise erhaltenen Codes bewiesen. Angesichts der unendlichen Kombinationsmöglichkeiten von Modellkonstrukten und Parametern kann es natürlich keine Garantie dafür geben, dass niemals ein Fehler in automatisch erzeugtem Code auftreten wird, trotz intensiver Tests und Qualitätskontrollen. Jedoch ist die Wahrscheinlichkeit eines Code-Fehlers bei Code-Generatoren im Vergleich zu menschlichen Programmierern sehr gering. Denn der Mensch hat gute und weniger gute Tage, lernt und vergisst wieder, und Arbeitsplätze von Programmierern werden auch mal neu besetzt. Der Code-Generator hingegen lernt mit jeder Verbesserung, jeder Fehlerbehebung und jeder neuen Situation dazu und „vergisst" das Gelernte niemals. Bei einem Fahrzeughersteller (OEM) bestätigten die dortigen TargetLink-Benutzer: „Beim Überprüfen des automatisch generierten Codes für unsere sicherheitskritischen Anwendungen haben wir regelmäßig Modellierungs- und Entwurfsfehler gefunden aber keine Fehler im Code."

Zur Vermeidung mangelnder Code-Sicherheit und fehlender Portabilität bei der Handprogrammierung sicherheitsrelevanter Systeme für Autos hat sich der MISRA-Standard (**M**otor **I**ndustry **S**oftware **R**eliability **A**ssociation) als Programmierleitfaden durchgesetzt. Der MISRA-Standard wird von TargetLink eingehalten, wo dies sinnvoll ist. Es gibt allerdings begründete Ausnahmen, wie sie vom Standard auch durchaus zugelassen sind. Denn automatische Code-Generatoren machen nicht die typischen Fehler menschlicher Programmierer. Als Beispiel seien Bit-Shift-Operationen genannt, die laut MISRA nicht erlaubt sind und durch zeitintensivere Divisionen ersetzt werden müssen. Für Programmierer ist es normalerweise ein guter Rat, sich an die entsprechende MISRA-Regel zu halten. Der automatische Code-Generator ist aber in der Lage, eine Situation komplett zu analysieren und jedes Mal korrekten Code zu erzeugen. Deshalb kann – wenn vor allem Effizienz gewünscht ist – eine MISRA-Regel vom Code-Generator guten Gewissens außer Acht gelassen werden.

Um die systematische Entwicklung von TargetLink bei dSPACE sicherzustellen, zu verbessern und nach einer anerkannten Qualitätsnorm auszurichten, arbeitet das TargetLink-Entwicklungsteam zur Zeit auf den Assessment-Level 3 der Norm ISO/IEC 15504 (SPICE; **S**oftware **P**rocess **I**mprovement **C**apability d**E**termination) zu. SPICE verbindet Konzepte von ISO9001 und CMM, wobei die sogenannten „Audits" von unabhängigen Organisationen durchgeführt werden. Bei der Einhaltung des Entwicklungsprozesses nach SPICE wird ein sehr hoher Software-Qualitätsstandard – auch nach revisionsbedingten Änderungen – erreicht. Dies ist angesichts der notwendigen Weiterentwicklungen äußerst wichtig. Eine irgendwie geartete Zertifizierung nur eines Codes in einer bestimmten Anwendung oder eines Code-Generators einer ganz bestimmten Versionsnummer unter ganz bestimmten Bedingungen hat demgegenüber nur begrenzten Wert. Aus diesem Grund wurde SPICE von einer deutschen Automobil-OEM-Initiative zum Auditing von Steuergeräte-Software-Anbietern und konsequenterweise auch von Werkzeug-Anbietern ausgewählt. Mitglieder der „Herstellerinitiative Software" sind Volkswagen, DaimlerChrysler, BMW, Porsche und Audi.



Erfolgreicher Einsatz von automatisch generiertem Code für Seriensteuergeräte

Auch in der Luftfahrt konnte mit TargetLink erzeugter Code bereits seine hohe Qualität beweisen. Bei einem Hersteller von Kabinendruck-Kontrollsystemen wurde für TargetLink-Code der höchste Zertifizierungsgrad in der Zivilluftfahrt innerhalb des DO178B-Standards erreicht.

Trotz des systematischen Entwicklungsprozesses muss ein Seriencode-Generator sorgfältig getestet werden. Auf der ersten Testebene werden elementare Code-Konstrukte (ANSI C, ANSI C mit Erweiterungen, ANSI C mit Erweiterungen und ausgewählten Einfügungen in Assembler-Sprache) für alle Mikrocontroller und alle unterstützten Compiler getestet. Das Testprogramm umfasst zur Zeit eine halbe Million Testmuster. Alle Testmuster werden mit unterschiedlichen Parametern getestet, was zu 8 Millionen Testfällen führt. Ein einziger Test pro Compiler-Mikrocontroller-Kombination braucht automatisiert eine volle Woche rund um die Uhr. Selbst Compiler-Fehler können in dieser Phase aufgedeckt werden. Auf der zweiten Testebene werden speziell konstruierte sowie „echte“ Simulink/Stateflow-Modelle herangezogen. Simulationen des Idealverhaltens und des Verhaltens des tatsächlich erzeugten Codes werden automatisch miteinander verglichen. Aktuell sind 1 500 Modelle und 100 000 Testmuster im Einsatz, wobei ein voller Testdurchlauf zehn oder mehr Tage beansprucht. Die Zahl der verwendeten Testfälle wächst mit jedem gefundenen Fehler, so dass sich die Sicherheit des Code-Generators immer weiter erhöht.

Automatisch erzeugter Code – bereit für die Serie?

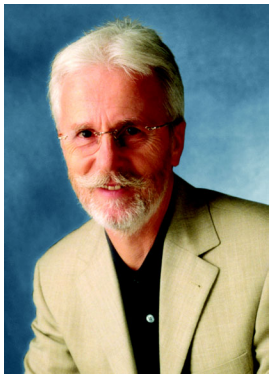
Da mit automatischer Seriencode-Generierung die Entwicklungszeit verkürzt und gleichzeitig die Qualität verbessert werden kann, ist das Interesse der Industrie an Code-Generatoren sehr groß. Eine Reihe von Fahrzeugen ist mit generiertem Code schon in Serie gegangen.

Etwa 50 Prozent aller TargetLink-Lizenzen sind bisher an Fahrzeughersteller (OEM) ausgeliefert worden und etwa 35 Prozent an Zulieferer von elektronischen Systemen und mechatronischen Komponenten. Außerdem zählen Ingenieurdienstleister und Forschungseinrichtungen zu den TargetLink-Benutzern.

Die ersten Serienanwendungen für automatisch erzeugten Code lagen im Bereich der Add-on-Funktionen und nicht-sicherheitskritischen Systeme. Dazu zählt zum Beispiel eine mit TargetLink für den Nissan Sentra umgesetzte Motormanagement-Anwendung nach dem kalifornischen SULEV-Standard, mit dem die Autohersteller sich verpflichten, die Schadstoffemissionen bei Kraftfahrzeugen auf ein Minimum zu reduzieren. Das Fahrzeug war bereits 6 Monate nach der TargetLink-Einführung, im Frühjahr 2000, fertig zur Auslieferung. Mittlerweile werden komplette Motormanagementsysteme von Serienfahrzeugen mit TargetLink realisiert. Für solche Anwendungen ist es wichtig, dass sich die Code-Generierung nahtlos in die Software-Architektur der Steuergeräte integriert. Somit wird auch die vollständig integrierte Unterstützung von Echtzeit-Betriebssystemen nach dem OSEK-Standard wichtig. Bisher wurde TargetLink-Funktionscode nach manueller Vorbereitung in die Software-Infrastruktur des Steuergerätes integriert. Die nächste Version ermöglicht die Code-Generierung für OSEK/VDX-konforme Echtzeit-Betriebssysteme direkt aus MATLAB/Simulink. Auch modellübergreifendes Datenmanagement wird mit Hilfe des neuen dSPACE Data Dictionary möglich sein.

Schlussfolgerung

Die automatische Code-Generierung für Seriensteuergeräte ist im Bereich der Automobilelektronik bereits sehr erfolgreich im Einsatz. Die Bereitschaft zur Einführung und der Einführungszeitpunkt unterscheiden sich je nach Fahrzeughersteller, jedoch ist die Technologie selbst als ebenso unausweichlich anzusehen wie seinerzeit die Einführung der Programmiersprache C. Negative Befürchtungen hinsichtlich der Effizienz des automatisch erzeugten Codes haben sich nicht bestätigt. Ebenso erfüllt der automatisch erzeugte Code die geforderten hohen Qualitätsansprüche und im Zuge der zunehmenden Erfahrungen sowie mit Blick auf die vielen abgeschlossenen Projekte wird sich diese Erkenntnis auch durchsetzen.



Der Autor

Dr.-Ing. Herbert Hanselmann (54) ist geschäftsführender Gesellschafter (President & CEO) der dSPACE GmbH, Paderborn. Er studierte Elektrotechnik an der Universität Karlsruhe und promovierte dort zum Thema Regelungssysteme. Danach forschte und lehrte er am heutigen Mechatronik-Labor der Universität Paderborn. 1988 gründete er mit drei weiteren Ingenieuren die dSPACE GmbH.