

Ein Werkzeug zur automatischen Konvertierung einer ausführbaren Spezifikation in Steuergerätecode für Festkommaprozessoren

Tool for Automatic Conversion of an Executable Specification in ECU Code for Fixed-point Processors

Dipl.-Ing. Roman Frank Starbek

GIGATRONIK Gesellschaft für Automobilelektronik-Entwicklung mbH

Dipl.-Ing. Uwe Class

TRW Automotive GmbH

Zusammenfassung

Der Einsatz von Codegeneratoren zur automatischen Erzeugung von Produktionscode für Seriensteuergeräte führt zu einer deutlichen Reduktion der Entwicklungszeiten. Der Entwicklungsprozess zur Erzeugung von C-Code für Festkomma-Arithmetik ist allerdings häufig arbeitsintensiv und zeitaufwändig. Zur Vermeidung von Wertebereichüberläufen und unerwünschten Sättigungen von Variablen muss eine Aufbereitung des Gleitkomma-Prototyps mit großer Sorgfalt erfolgen. Durch die Verwendung geeigneter Werkzeuge lässt sich die Entwicklungszeit weiter verkürzen und zuverlässiger Steuergerätecode erzeugen.

In diesem Beitrag wird ein Werkzeug vorgestellt, das den oben beschriebenen Entwicklungsprozess nahezu automatisiert. Am Beispiel eines Auslösealgorithmus für reversible Gurtstraffer wird der optimale Prototyp der Steuerung aufbereitet und in C-Code für Microcontroller zum Einsatz in Steuergeräten für Kraftfahrzeuge überführt. Die Realisierung der Steuerung mit Festkomma-Arithmetik zeigt das gleiche sehr gute Verhalten wie der Gleitkomma-Prototyp.

Summary

The use of code generators to create production code for electronic control units (ECU) results in a significant reduction of development time. However, in most cases the development process for the generation of fixed-point arithmetics is time-consuming and difficult. The preparation of the floating-point prototype must be conducted very carefully to avoid integer overflows and undesired saturations. The use of appropriate tools can reduce the development time even more and generates reliable ECU code.

This paper introduces such a tool for the automatization of the development process as mentioned above. An algorithm for a reversible seat belt pretensioner is used in order to demonstrate how an optimal controller prototype will be prepared and converted into production code for an automotive ECU. The implementation of the controller in fixed-point arithmetics shows the same performance as the floating-point prototype.

1 Einleitung

Aufgrund der steigenden Komplexität von Funktionen in Steuergeräten für Kraftfahrzeuge gewinnt der modellbasierte, werkzeuggestützte Entwicklungsprozess heute zunehmend an Bedeutung. Durch den hohen Wettbewerbsdruck wird eine Reduktion der Entwicklungszeiten für Steuergeräte und entsprechender Software bis zur Serienreife zwingend notwendig. Zusätzlich sind hohe Qualitätsanforderungen an die zu entwickelnden Produkte und damit an den Entwicklungsprozess zu erfüllen. Ein weit verbreiteter Prozess zur zeit- und kosteneffizienten Steuergeräteentwicklung wird durch das V-Modell beschrieben. Mit Hilfe von speziell entwickelten Softwarewerkzeugen hat die GIGATRONIK einen durchgängigen Prozess etabliert, der eine nahezu vollständige Automatisierung zur Entwicklung von Steuergerätfunktionen bis zum Seriencode ermöglicht.

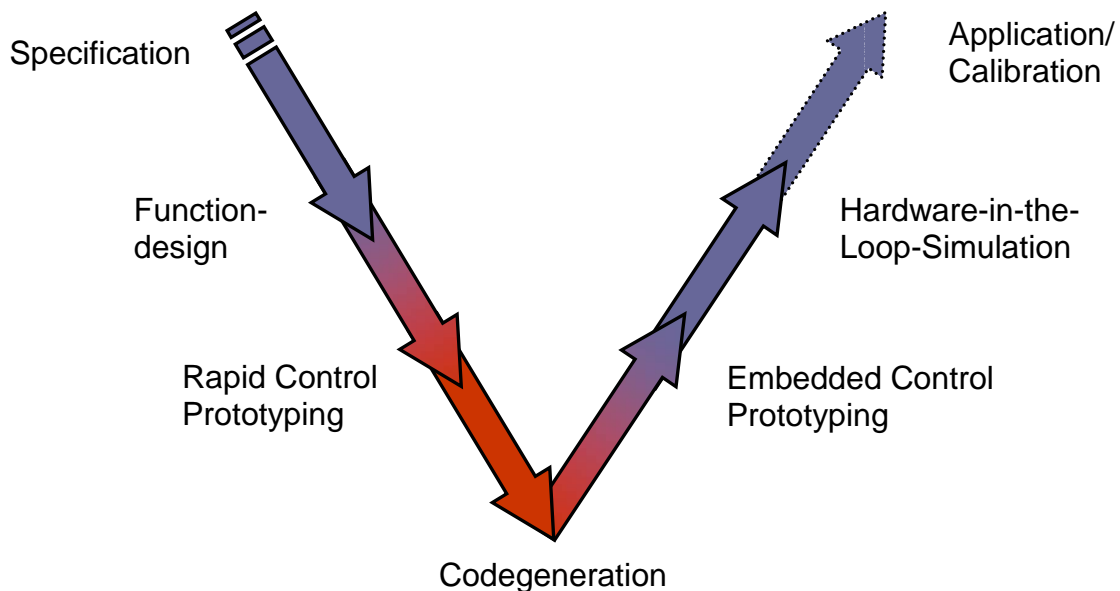


Bild 1: *Entwicklungsprozess nach dem V-Modell
V-Cycle development process*

Bild 1 zeigt die zugehörigen Entwicklungsstufen. Voraussetzung für die Vorgehensweise nach dem V-Modell ist der Funktionsentwurf einer ausführbaren Spezifikation der zu implementierenden Steuerung oder Regelung in Form eines Simulationsmodells. Damit werden Lösungsansätze bereits in der Offline-Simulation und anschließend mit der Methodik des Rapid Control Prototyping (RCP) auf leistungsfähiger Echtzeithardware überprüft [1]. Auf Basis des Simulationsmodells erfolgt dann die automatische Generierung des Seriencodes und dessen Implementierung auf einer universellen, microcontrollerbasierten Prototypen-Steuergeräteplattform mittels Embedded Control Prototyping (ECP) [2]. Die **uniECU**¹ der GIGATRONIK ist solch eine Plattform und ermöglicht den Einsatz verschiedener Prozessoren mit Gleit- oder Festkomma-Arithmetik. Nach der Implementierung der Software auf einem Seriensteuergerät steht dieses dann für die Einbindung in eine

¹ **uni ECU**[®] ist ein eingetragenes Warenzeichen der Gigatronik GmbH

HIL²-Umgebung, für weitere Tests am realen System und anschließend für die Feinabstimmung von Parametern zur Verfügung.

Für die Generierung von Serienelementen werden heute Codegeneratoren eingesetzt. Sie erzeugen den Serienelementen automatisch aus dem Simulationsmodell [3]. Allerdings ist das Simulationsmodell für die Implementierung auf einem Steuergerät mit Festkomma-Arithmetik manuell so aufzubereiten, dass alle Variablen auf dem begrenzten Wertebereich der Festkomma-Hardware dargestellt werden können. Um Quantisierungseffekte klein zu halten und Wertebereichüberläufe zu vermeiden sind die Variablen nach der Wahl entsprechender Datentypen und Wortbreiten geeignet zu skalieren.

Die Skalierung jeder einzelnen Variablen eines komplexen Simulationsmodells ist eine arbeitsintensive und zeitraubende Arbeit. Da maximale Aussteuerbereiche oft nicht bekannt sind, können Wertebereichüberläufe nur durch zusätzliche Sättigungen ausgeschlossen werden. Neben der Skalierung der Variablen müssen außerdem applikationsspezifische Einstellungen, wie z.B. die Vorgabe verschiedener Speicherklassen, manuell durchgeführt werden. Softwareänderungen am Steuergeräteelementen in späten Entwicklungsstadien sind oft nicht mehr oder nur mit erheblichem Zeitaufwand durchführbar.

Die Lösung für eine schnelle und zuverlässige Aufbereitung des Simulationsmodells ist die Verwendung geeigneter Werkzeuge. Damit lässt sich der Prozess von der Weiterentwicklung des Simulationsmodells bis zur automatischen Codegenerierung automatisieren und die Entwicklungszeit deutlich verkürzen.

In diesem Beitrag steht das von der GIGATRONIK entwickelte *Model Conversion Tool* im Brennpunkt. Dieses Werkzeug ermöglicht die automatische Konvertierung eines prototypischen Simulationsmodells in Serienelementen. Als Ziele stehen dabei die Erzeugung von zuverlässigem Code und die Minimierung der Entwicklungszeit im Vordergrund. Damit werden die Kosten verringert und wird die Produktqualität durch die Reproduzierbarkeit des Prozesses zusätzlich gesteigert.

Die Funktionalität des Werkzeuges wird am Beispiel eines prototypischen Auslösealgorithmus für reversible Gurtstraffer dargestellt. Die ausführbare Spezifikation liegt als Simulink³-Modell vor und wird mit dem *Model Conversion Tool* unter Einsatz des Codegenerators TargetLink⁴ automatisch in Festkomma-Code überführt. Dabei steht die optimale Skalierung der Variablen unter Berücksichtigung ihrer maximalen Aussteuerbereiche im Vordergrund (*worst-case*-Skalierung). Wertebereichüberläufe von Variablen auf der Festkomma-Hardware können ausgeschlossen werden, wodurch zusätzlicher Sättigungselementen überflüssig wird. Der Vergleich der Zeitantworten in Gleit- und Festkomma-Arithmetik zeigt praktisch identische Ergebnisse.

Das Beispiel repräsentiert neben der Funktionalität des Werkzeuges auch die Bedeutung einer zugeschnittenen Werkzeugkette im Entwicklungsprozess für Steuergeräteelementensoftware.

² Hardware-in-the-Loop

³ The MathWorks, Inc., Natick, M.A., USA

⁴ dSPACE GmbH, Paderborn, Germany

2 Skalierung von Variablen

Für die Implementierung auf einem Festkomma-Prozessor müssen alle Variablen des Simulationsmodells auf Integer-Größen abgebildet werden. Die Darstellung einer Variablen v , die den Wert einer physikalischen Größe repräsentiert, ist im einfachsten Fall mit Hilfe ihres skalierten Integer-Äquivalents v' durch die Beziehung

$$v = S_v v' \quad (1)$$

gegeben. Dabei entspricht der Skalierungsfaktor S_v dem physikalischen Wert des LSB (*least significant bit*) des Integer-Äquivalents bzw. der Quantisierungsstufe. Der physikalische Aussteuerbereich von v hängt neben dem Skalierungsfaktor S_v auch vom Datentypen (*signed* oder *unsigned*) von v' und von der Wortbreite der Festkomma-Hardware ab.

Zur Vermeidung von Wertebereichüberläufen nach der Implementierung auf der Zielhardware ist der Skalierungsfaktor mit Hilfe der maximal möglichen Aussteuerbereiche der physikalischen Variablen und ihres Integer-Datentyps zu ermitteln. Für den physikalischen Aussteuerbereich $[v_{\min}, v_{\max}]$ der Variablen v erhält man beispielsweise bei Verwendung des Datentyps *signed*-Integer und der Wortbreite b für die Festkomma-Variable v' den Skalierungsfaktor

$$S_v = \frac{\max(|v_{\min}|, |v_{\max}|)}{2^{b-1} - 1}, \quad (2)$$

und bei Verwendung des Datentyps *unsigned*-Integer

$$S_v = \frac{\max(|v_{\min}|, |v_{\max}|)}{2^b - 1}. \quad (3)$$

Für die Skalierung aller Variablen eines Simulationsmodells müssen alle zugehörigen Aussteuerbereiche bestimmt werden. Die Vorgehensweise für die werkzeuggestützte Berechnung der Aussteuerbereiche wird im folgenden Kapitel beleuchtet.

3 Berechnung der Aussteuerbereiche von Variablen in Simulationsmodellen

Voraussetzung für die nachfolgende Betrachtung ist, dass das Simulationsmodell als Simulink-Blockschaltbild realisiert ist.

In Simulink sind alle Blöcke untereinander unidirektional mit Signallinien verbunden. Auf Basis des Signalfusses lassen sich dann die Aussteuerbereiche der Signale von den Modell-Eingängen angefangen blockweise nach bestimmten Rechenvorschriften bis zu den Modell-Ausgängen hin fortpflanzen. Die Rechenvorschrift hängt dabei vom jeweiligen Blocktyp ab, an dessen Ausgang eine neue Signallinie beginnt oder mehrere neue Signallinien beginnen.

Die Fortpflanzung der Aussteuerbereiche über den Signalfluss eines Simulink-Blockschaltbildes wird am Beispiel des folgenden Bildes veranschaulicht. In Bild 2 stehen über den Blöcken jeweils die Aussteuerbereiche der Ausgangssignale.

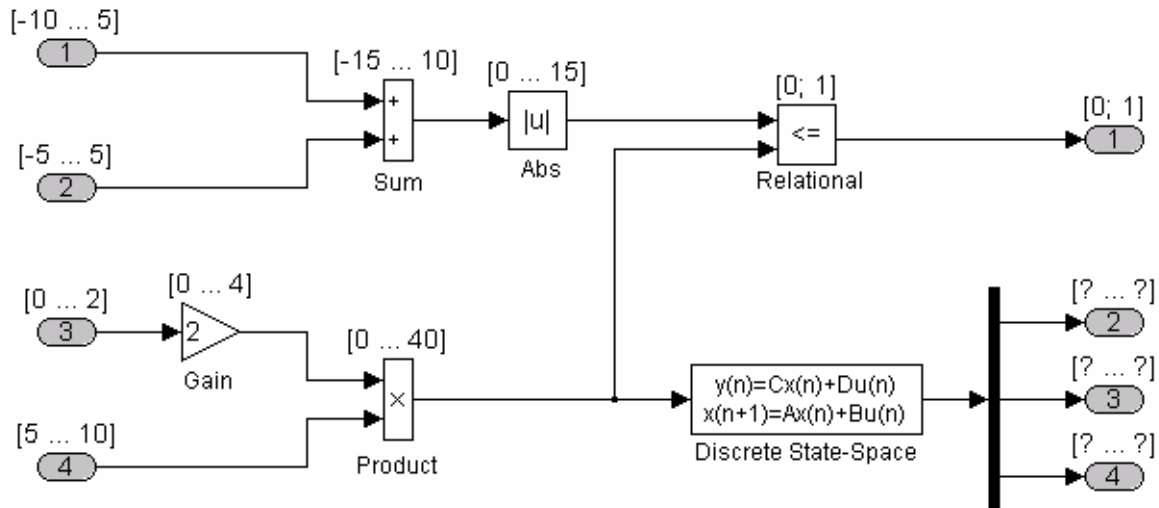


Bild 2: Simulink-Blockschaltbild
Simulink block diagram

Die Ausgangssignale der Eingangsblöcke mit der Nummer 1 und 2 werden in einen Summenblock geführt (Bild 2, links oben). Der Aussteuerbereich des Ausgangssignals des Summenblocks ergibt sich durch die Summe der jeweiligen Minimal- und Maximalwerte der Eingangssignale.

Das Ausgangssignal des Eingangsblocks mit der Nummer 3 wird in einen Verstärkungsblock geführt (Bild 2, links). Der Aussteuerbereich des Ausgangssignals des Verstärkungsblockes lässt sich mit Hilfe des Verstärkungsfaktors berechnen. Im obigen Beispiel wird der Aussteuerbereich mit dem Faktor 2 verdoppelt.

Der Aussteuerbereich des Ausgangssignals des Produktblocks wird berechnet, indem jeweils die Minimal- und Maximalwerte der Aussteuerbereiche der beiden zugehörigen Eingangssignale multipliziert werden (Bild 2, links unten).

Der Betragsblock, der hinter den Summenblock geschaltet ist, verkleinert den Aussteuerbereich des Ausgangssignals, falls sich die Vorzeichen des Minimal- und des Maximalwerts des Eingangssignals, wie im obigen Fall, unterscheiden.

Ein Logikblock weist unabhängig von den Aussteuerbereichen der Eingangssignale immer die Werte 0 oder 1 am Ausgang auf (Bild 2, rechts oben).

Das obige Beispiel zeigt, dass für die Berechnung der maximalen Aussteuerbereiche der Ausgangssignale jeweils einfache Berechnungsvorschriften in Abhängigkeit der Block-Parameter, wie z.B. der Verstärkungsfaktor eines Verstärkungsblocks, festgelegt werden können. Die auf diese Weise ermittelten Aussteuerbereiche gehen schließlich in die Gleichung (2) bzw. (3) aus Kapitel 2 zur Berechnung des Skalierungsfaktors ein.

Die Berechnung der Aussteuerbereiche dynamischer Systeme, wie z.B. diskreter Übertragungsfunktionen, Filter und Zustandssysteme ist jedoch nicht so einfach. Bei begrenzten Eingangsvariablen sind die maximalen Aussteuerbereiche der Zustands-

und Ausgangsvariablen aufgrund der Systemdynamik nicht direkt bekannt (Vgl. Bild 2, rechts unten). Die theoretischen Grundlagen zur Ermittlung der maximalen Aussteuerbereiche von Variablen dynamischer Systeme werden im folgenden Kapitel vorgestellt.

4 Berechnung der Aussteuerbereiche von Variablen dynamischer Systeme

Für ein diskretes SISO⁵-System lässt sich der maximale Aussteuerbereich der Ausgangsgröße berechnen, wenn der Minimal- und Maximalwert der Eingangsgröße bekannt sind. Die Zahlenfolge der Eingangsgröße wird für diese Berechnung nicht benötigt.

Die Ausgangszahlenfolge $\{y_k\}$ des diskreten SISO-Systems kann mit Hilfe der Faltungssumme

$$y_k = \sum_{j=0}^k g_j \cdot u_{k-j} \quad (4)$$

berechnet werden. Dabei wird der Wert der Ausgangszahlenfolge zum Zeitschritt k durch die Faltung der Eingangszahlenfolge $\{u_{k-j}\}$ mit der Impulsantwortzahlenfolge $\{g_j\}$ des Systems bestimmt.

Für stabile Systeme streben die Werte g_j der Impulsantwortzahlenfolge in der Faltungssumme mit steigendem j gegen Null. Für beschränkte Werte $u \in [u_{\min}, u_{\max}]$ der Eingangszahlenfolge sind daher auch die Werte der Ausgangszahlenfolge mit $y \in [y_{\min}, y_{\max}]$ beschränkt. Ein solches System wird als BIBO⁶-stabil bezeichnet und ist Voraussetzung für die Ermittlung des maximalen Aussteuerbereichs der Ausgangsgröße.

Die Berechnung der minimalen bzw. maximalen Werte der Ausgangszahlenfolge erfolgt nun mit der Auswertung der Gleichung (4) für jeden Zeitschritt $k = 0, 1, 2, 3, \dots$, wodurch man für die ersten Schritte

$$\begin{aligned} y_0 &= g_0 \cdot u_0 \\ y_1 &= g_0 \cdot u_1 + g_1 \cdot u_0 \\ y_2 &= g_0 \cdot u_2 + g_1 \cdot u_1 + g_2 \cdot u_0 \\ y_3 &= g_0 \cdot u_3 + g_1 \cdot u_2 + g_2 \cdot u_1 + g_3 \cdot u_0 \end{aligned} \quad (5)$$

...

erhält.

In der Faltungssumme wird dabei für $j=k$ jeweils ein neuer Wert u_0 der Eingangszahlenfolge so bestimmt, dass der Wert y_k der Ausgangszahlenfolge im betrachteten Zeitschritt mit der Faltungssumme ein Minimum bzw. Maximum annimmt.

⁵ Single Input Single Output

⁶ Bounded Input Bounded Output

Die Wahl erfolgt in Abhängigkeit des Vorzeichens des Wertes g_k der Impulsantwortzahlenfolge gemäß Tabelle 1. Durch das Hinzufügen eines neuen Wertes u_0 verschieben sich die bereits berechneten Werte der Eingangszahlenfolge jeweils um einen Zeitschritt.

Vorzeichen von g_k	Minimierung von y_k	Maximierung von y_k
positiv	$u_0 = u_{\min}$	$u_0 = u_{\max}$
negativ	$u_0 = u_{\max}$	$u_0 = u_{\min}$

Tabelle 1: Wahl eines neuen Wertes u_0 der Eingangszahlenfolge im Zeitschritt k
Selection of a new value u_0 of the input sequence at the time step k

Mit dieser Vorgehensweise ergeben sich zwei *worst-case*-Eingangszahlenfolgen, die alle Werte y_k der Ausgangszahlenfolge in ihr Minimum bzw. Maximum treiben und deren Werte sich an den Grenzen u_{\min} und u_{\max} befinden. Bild 3 veranschaulicht diesen Sachverhalt für die Maximierung der Werte der Ausgangszahlenfolge über die Eingangszahlenfolge.

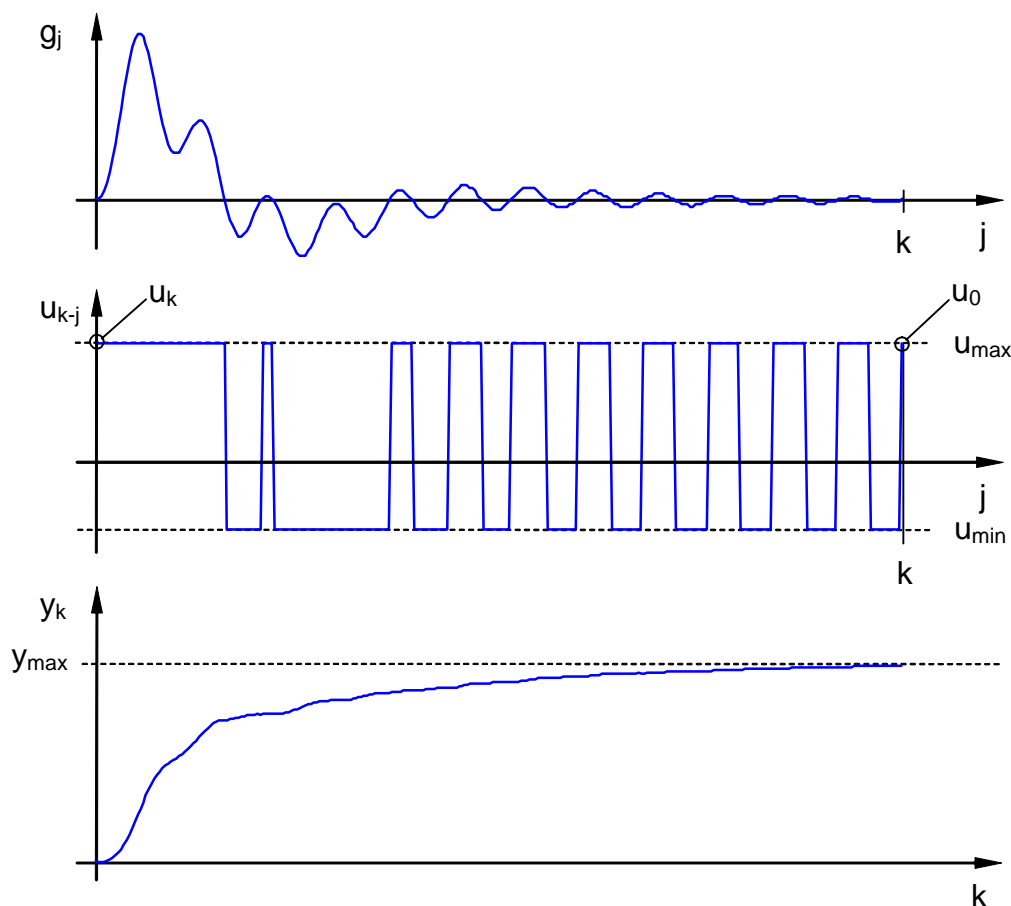


Bild 3: Impulsantwortzahlenfolge $\{g_j\}$ (oben) und Eingangszahlenfolge $\{u_{k-j}\}$ (Mitte) zur Maximierung der Werte der Ausgangszahlenfolge $\{y_k\}$ (unten)
Pulse response sequence $\{g_j\}$ (top) and input sequence $\{u_{k-j}\}$ (middle) for the maximization of the output sequence values $\{y_k\}$ (bottom)

Die Ermittlung der maximalen Aussteuerbereiche von Variablen dynamischer Systeme wurde mit Hilfe von MATLAB M-Funktionen realisiert. Dabei musste für die numerische Auswertung der Gleichung (4) ein Abbruchkriterium eingeführt werden. Dieses führt zu einem Abbruch der Auswertung zu einem Zeitschritt k_{\max} , wenn die Änderung der Ausgangszahlenfolge $\{y_k\}$ kleiner als eine bestimmte Konvergenztoleranz ist [4]. Der dadurch bedingte Abbruchfehler für den maximalen Aussteuerbereich wird durch eine geeignete Restgliedabschätzung korrigiert. Der berechnete Aussteuerbereich geht schließlich in die Gleichung (2) bzw. (3) aus Kapitel 2 zur Berechnung des Skalierungsfaktors S_v ein.

Die oben vorgestellte Vorgehensweise wird auch zur Berechnung der Aussteuerbereiche von Variablen diskreter MIMO⁷-Systeme verwendet. Eine detaillierte Beschreibung zur Ermittlung der maximalen Aussteuerbereiche der Zustands- und Ausgangsgrößen von Übertragungsfunktionen, Filtern und diskreten Zustandsmodellen ist in [5] zu finden.

5 Signalflossanalyse und automatische Skalierung

Die in den vorangegangenen Kapiteln beschriebenen Berechnungen wurden für den Einsatz im *Model Conversion Tool* automatisiert. Voraussetzung für die automatische Berechnung der Aussteuerbereiche aller Variablen nach Kapitel 3 ist allerdings, dass der Signalfloss innerhalb des Modells bekannt ist.

Die Signalflossanalyse von Simulink-Modellen wird mit Hilfe von MATLAB M-Funktionen rekursiv durchgeführt. Das Ergebnis der Analyse ist eine MATLAB-Datenstruktur, die alle Signale des Modells mit allen relevanten Informationen enthält. Dazu zählen z.B. die Abtastzeiten und Datentypen der Signale sowie Eingangs- und Ausgangsinformationen zugehöriger realer (abgetasteter) und virtueller (nicht abgetasteter) Quell- und Zielblöcke.

Auf Basis des Signalflosses bzw. der MATLAB-Datenstruktur folgt bei gegebenen Grenzen der Modell-Eingangssignale die Berechnung der maximalen Aussteuerbereiche aller Variablen nach den Kapiteln 3 und 4 mit Hilfe weiterer M-Funktionen. Die resultierenden Ergebnisse werden anschließend für die automatische Berechnung der optimalen Skalierungsparameter nach den Gleichungen (2) bzw. (3) aus Kapitel 2 herangezogen. Dabei ist die Wortbreite des Festkomma-Prozessors als Standardwert vorzugeben.

Bei unterschiedlichen oder negativen Vorzeichen des Minimal- und Maximalwertes eines Aussteuerbereichs wird automatisch der Datentyp *signed-Integer* verwendet, bei positiven Vorzeichen der Datentyp *unsigned-Integer*.

Nach der automatischen Skalierung werden die Berechnungsergebnisse auf die Datenstruktur zurück geschrieben.

⁷ Multiple Input Multiple Output

6 Model Conversion Tool

Die Signalflussanalyse des Simulationsmodells und die Berechnung der maximalen Aussteuerbereiche nach den Kapiteln 3 und 4 sowie die automatische Skalierung nach Kapitel 5 wurden im so genannten *Model Conversion Tool* zusammengefasst. Der Anwender ist mit Hilfe dieses Werkzeugs in der Lage die notwendigen Schritte zur Überführung der prototypischen Steuergerätesoftware in Serienelemente interaktiv durchzuführen. Bild 4 zeigt die grafische Bedienoberfläche des Werkzeugs, die in der Entwicklungsumgebung MATLAB realisiert wurde.

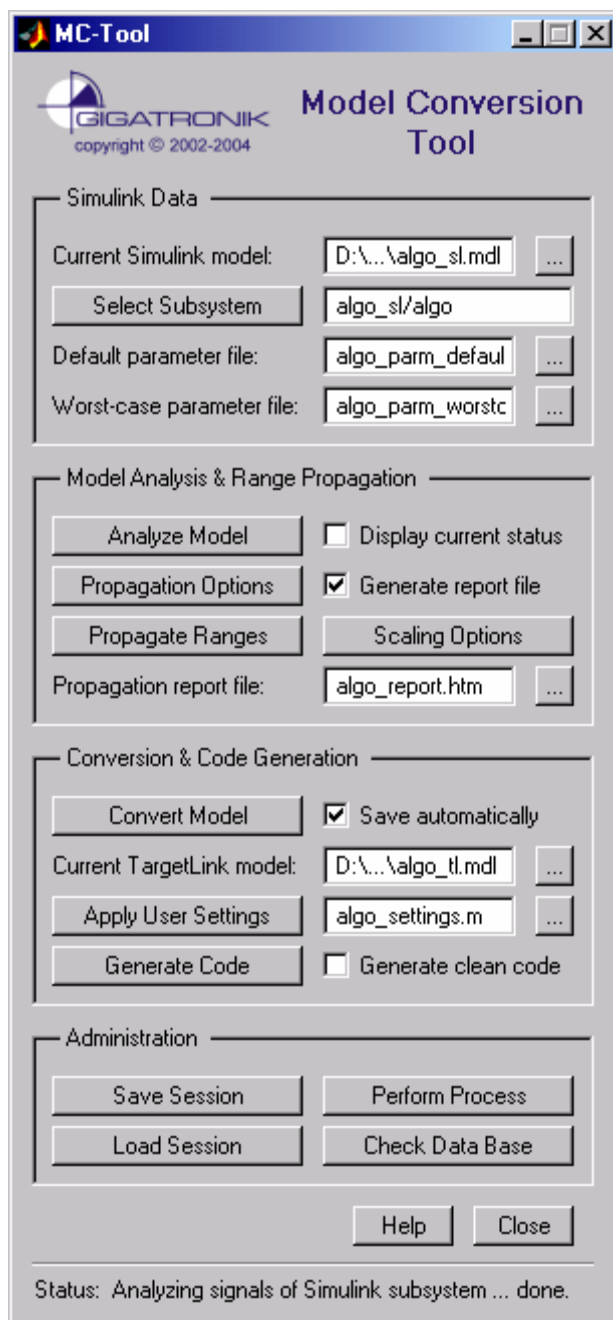


Bild 4: Grafische Bedienoberfläche des Model Conversion Tools
Graphical user interface of the Model Conversion Tool

Die automatisierten Prozessschritte sind die

- Signalflussanalyse des Simulationsmodells,
- Berechnung der maximal möglichen Aussteuerbereiche aller Variablen,
- optimale Skalierung dieser Variablen,
- Übertragung der Skalierungsergebnisse in die Umgebung des Codegenerators,
- Durchführung applikationsspezifischer Einstellungen, und schließlich die
- automatische Codegenerierung.

Zu Beginn des Prozesses sind das Simulink-Modell und das Subsystem, welches die zu konvertierende prototypische Applikation enthält, sowie gegebenenfalls MATLAB M-Dateien für die Initialisierung der Parameter gemäß Bild 4, Bereich *Simulink Data*, auszuwählen. Die Signalflussanalyse des Subsystems erfolgt anschließend im Bereich *Model Analysis & Range Propagation* auf Knopfdruck.

Die Berechnung der maximalen Aussteuerbereiche der Variablen und der zugehörigen Skalierungsparameter ist von den Aussteuerbereichen der Modell-Eingänge abhängig. Daher sind diese zunächst in einer Dialogbox, die in Bild 5 dargestellt ist, vorzugeben.

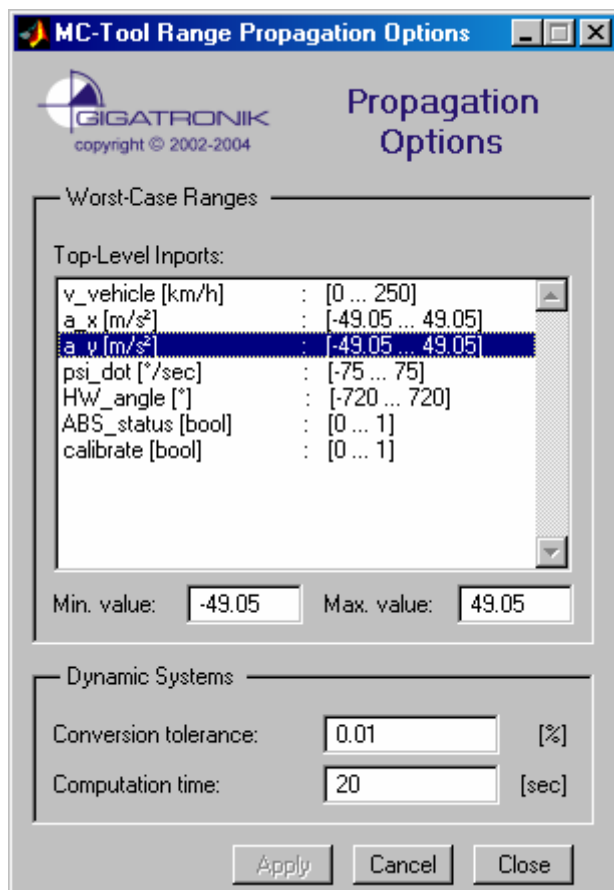


Bild 5: Vorgabe der maximalen Aussteuerbereiche der Modell-Eingänge
Determination of the maximum ranges of model inports

In dieser Dialogbox werden zusätzlich die Abbruchkriterien für die Berechnung der Aussteuerbereiche von Variablen dynamischer Systeme festgelegt. Die Abbruchkriterien sind durch die in Kapitel 4 erwähnte Konvergenztoleranz und durch eine zulässige maximale Rechenzeit gegeben.

Die Berechnung der Aussteuerbereiche und die Skalierung aller Variablen erfolgen dann ebenfalls auf Knopfdruck.

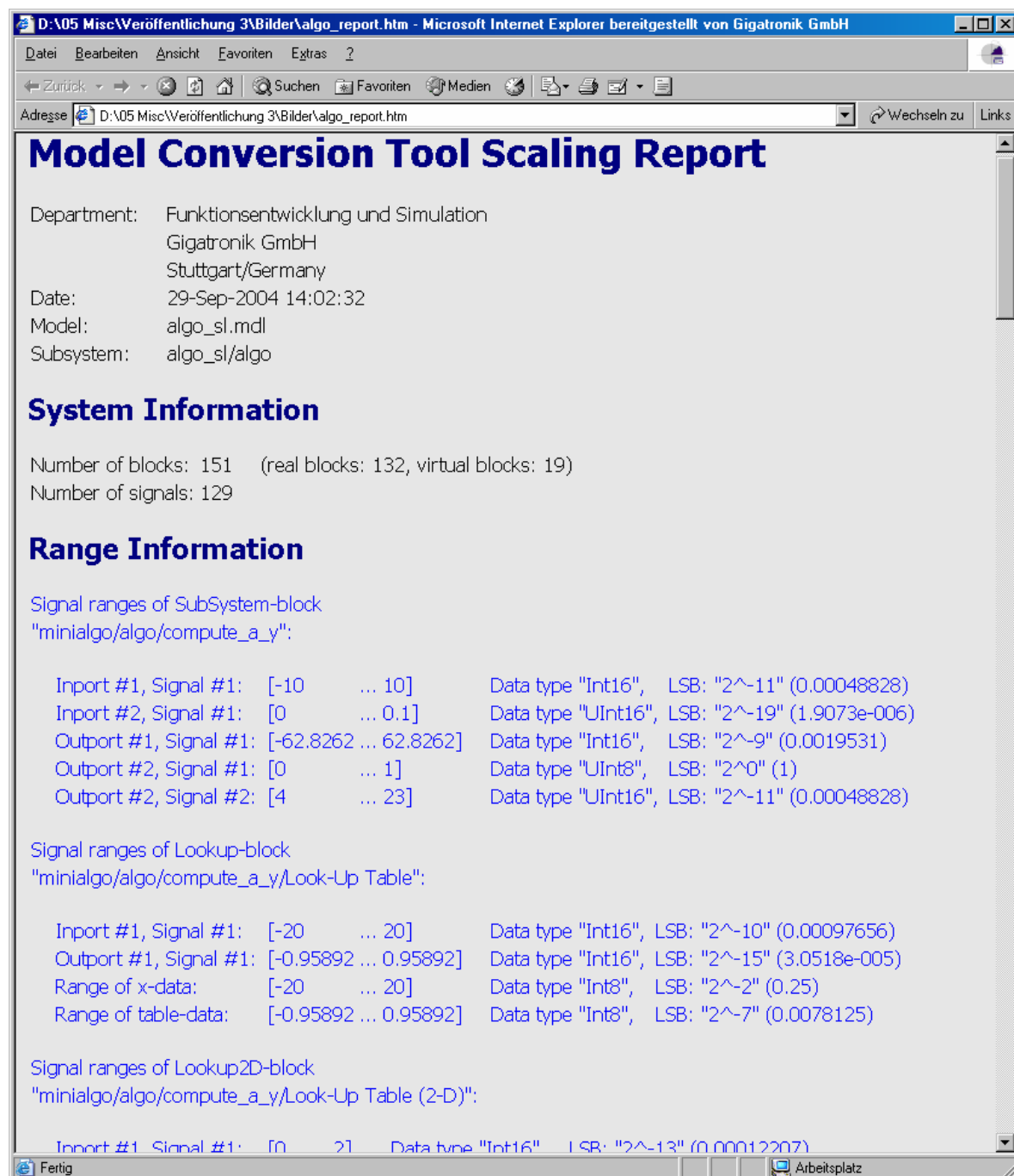


Bild 6: HTML-Datei mit Berechnungsergebnissen
HTML report file with calculation results

Anschließend kann der Anwender die Ergebnisse mit Hilfe einer automatisch generierten HTML-Datei betrachten bzw. kontrollieren. Sie enthält die Aussteuerbereiche der Ein- und Ausgangssignale aller Simulink-Blöcke, die verwendeten Datentypen und zugehörige Skalierungsparameter. Bild 6 zeigt beispielhaft den Inhalt solch einer HTML-Datei.

Werden in einer Applikation Tabellen eingesetzt, so kann separat für die Tabellenwerte eine kleinere Wortbreite vorgegeben werden. Entsprechende Skalierungsergebnisse sind in Bild 6 im unteren Bereich zu sehen. Dadurch wird zwar die Auflösung der Tabellenwerte vermindert, die Effizienz des resultierenden Festkomma-Codes wird jedoch unter Umständen erheblich erhöht.

Für die spätere Erzeugung des Steuergerätescodes ist das Simulationsmodell zunächst in die TargetLink-Umgebung zu importieren. Dies geschieht mit dem *Model Conversion Tool* im Bereich *Conversion & Code Generation*. Unter Verwendung spezieller TargetLink API⁸-Tools in MATLAB M-Sprache wird das Simulink-Modell automatisch in ein TargetLink-Modell konvertiert [4]. Die Skalierung der Variablen des TargetLink-Modells erfolgt anschließend automatisch. Dazu wird die MATLAB-Datenstruktur aus Kapitel 5 herangezogen. Jedes Signal in der Datenstruktur kann einem Ausgangssignal eines TargetLink-Blocks zugeordnet werden. Somit lassen sich die zugehörigen Aussteuerbereiche und die optimalen Skalierungsparameter automatisch auf das TargetLink-Modell übertragen.

Oft sind applikationsspezifische Einstellungen für die Erzeugung des Codes vorzunehmen. Dies betrifft z.B. die Auswahl besonderer Datentypen und/oder Wortbreiten, die Vorgabe bestimmter Speicherklassen, Ausnahmen bezüglich Skalierung für spezielle Variablen, globale Einstellungen für die Codegenerierung (z.B. Optimierungsmodus), und so weiter. Die Durchführung solcher Einstellungen lässt sich unter Verwendung der TargetLink API-Tools automatisieren.

Sind vom Anwender applikationsspezifische Einstellungen in Form einer MATLAB M-Funktion automatisiert worden, so kann diese im Bereich *Conversion & Code Generation* ausgeführt werden, bevor schließlich die Generierung des Seriencodes erfolgt.

7 Anwendungsbeispiel

Die Leistungsfähigkeit des *Model Conversion Tools* wird im Folgenden am Beispiel des Auslösealgorithmus für reversible Gurtstraffer nachgewiesen. Ausgangspunkt ist der Gleitkomma-Prototyp der Steuerung in Form eines Simulink-Modells, der die Referenz für die Entwicklung des Seriencodes für Festkomma-Prozessoren darstellt. Mit dem *Model Conversion Tool* erfolgt die Überführung des Simulink-Modells in Seriennecode automatisch. Zunächst wird der Einsatz reversibler Gurtstraffer in Kraftfahrzeugen vorgestellt und dessen Funktionsweise vereinfacht beschrieben.

Ein reversibler Gurtstraffer hat die Aufgabe, den Sicherheitsgurt in sicherheitskritischen bzw. instabilen Fahrsituationen so zu straffen, dass der Fahrzeuginsasse in seinem Sitz fixiert und damit das Verletzungsrisiko minimiert wird. Dazu wird der Gurt mit Hilfe eines drehmomentgeregelten Gleichstrommotors angezogen bzw. gestrafft. Stellt sich wieder eine stabile Fahrsituation ein, so wird der zuvor gestraffte Gurt gelöst.

Den Befehl zum Straffen bzw. Lösen des Sicherheitsgurtes an die Drehmomentregelung liefert der Auslösealgorithmus. In der Steuerung wird in

⁸ Application Programming Interface: Funktionsschnittstelle für die Verwendung durch weitere Werkzeuge

Abhängigkeit fahrdynamischer Zustände ein so genannter Stabilitätsindex ermittelt, der für die Bewertung des Fahrzustandes herangezogen wird. Der Betrag des Stabilitätsindex ist dabei von der Abweichung sensierter fahrdynamischer Zustände von äquivalenten Größen, die mit Hilfe eines geeigneten Modells berechnet werden, abhängig. Überschreitet der Stabilitätsindex einen definierten oberen Schwellwert, so ist eine instabile Fahrsituation identifiziert und der Gurt ist zu straffen. Unterschreitet der Index einen unteren Schwellwert, dann ist ein stabiler Fahrzeugzustand identifiziert und der Gurt ist wieder zu lösen.

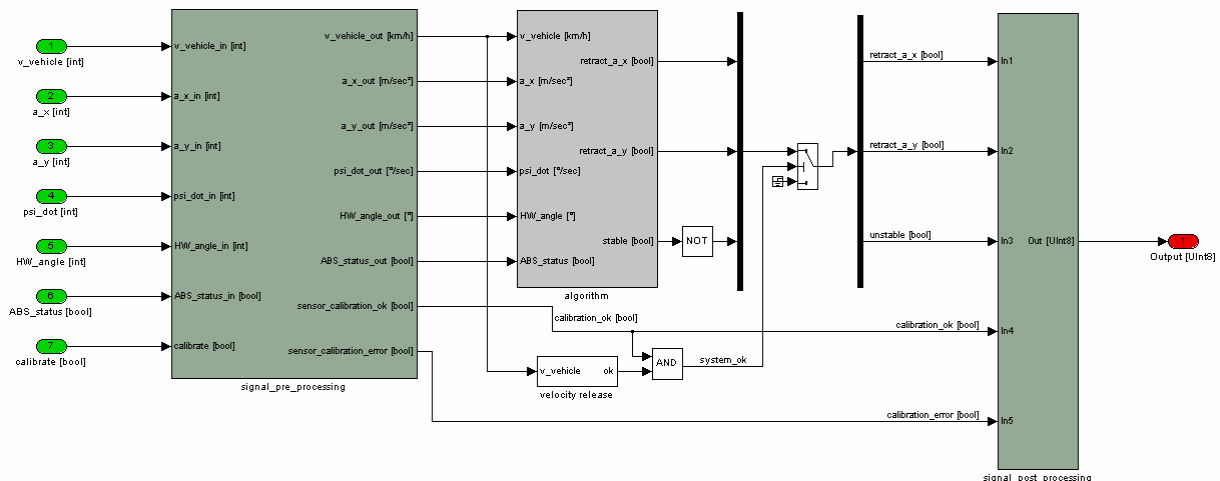


Bild 6: Simulink-Modell des Auslösealgorithmus
 Simulink model of the pretensioner algorithm

Bild 6 zeigt die ausführbare Spezifikation des Auslösealgorithmus als Simulink-Modell. Als Eingangssignale stehen im Wesentlichen die gemessenen Größen Längs- und Quereschleunigung, Gierrate, Lenkradwinkel und Fahrzeuggeschwindigkeit zur Verfügung. Das Ausgangssignal des Auslösealgorithmus wird als so genannter Auslösebefehl bezeichnet. Es wird als Sollgröße zum Straffen bzw. Lösen des Sicherheitsgurts in die Drehmomentregelung geführt.

Der Gleitkomma-Prototyp wurde zunächst in der Offline-Simulation und anschließend unter Echtzeitbedingungen mit leistungsfähiger Echtzeithardware im Experiment in Betrieb genommen.

Für die Überführung des Simulink-Modells in Festkomma-Code mit dem *Model Conversion Tool* sind zusätzliche anwendungsspezifische Einstellungen automatisiert worden. Dazu zählen unter anderem die Vorgabe von Speicherklassen für kalibrierbare Parameter und die Einbindung zusätzlicher Header-Dateien. Ferner betrifft dies auch die Festlegung aufzuzeichnender Zeitantworten bestimmter Signale für die SIL⁹-Simulation in der TargetLink-Umgebung.

Mit Hilfe der SIL-Simulation findet ein Vergleich der jeweils in Gleitkomma- und Festkomma-Arithmetik berechneten Zeitantworten statt [4]. Dabei sind nur geringe Abweichungen infolge von Quantisierungseffekten tolerierbar. Für den Auslösealgorithmus des reversiblen Gurtstraffers ist insbesondere der Stabilitätsindex

⁹ Software-in-the-Loop

näher zu betrachten. Das folgende Bild zeigt die Zeitantwort des Stabilitätsindex jeweils in Gleit- und in Festkomma-Arithmetik für ein Fahrmanöver, bei dem das Lenkrad so mit einem Momentensprung beaufschlagt wird, dass das Fahrzeug kurzzeitig die Stabilitätsgrenze überschreitet.

In Bild 7 ist oben die Zeitantwort des Stabilitätsindex mit dem oberen und unteren Schwellwert für das Straffen bzw. Lösen des Gurts dargestellt. Im Bild unten ist die Zeitantwort des Auslösebefehls zu sehen. Der Wert 1 bzw. 0 steht für den Befehl zum Straffen bzw. Lösen an die Drehmomentregelung.

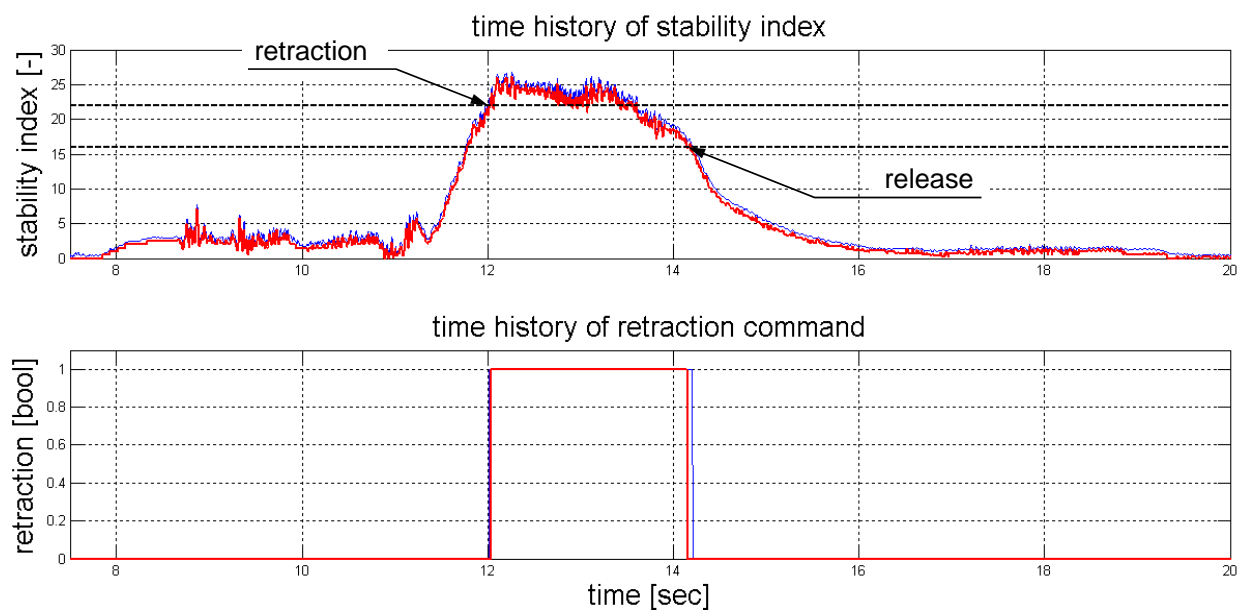


Bild 7: Vergleich von Zeitantworten in Gleit- (dünn) und Festkomma-Arithmetik (fett)

Comparison of floating-point (thin) and fixed-point time histories (bold)

Die Ergebnisse für den Seriencode mit Festkomma-Arithmetik unterscheiden sich kaum von denen für den Gleitkomma-Prototyp der Steuerung.

Die minimale Abweichung der Zeitantwort des Auslösebefehls in Festkomma-Arithmetik von der in Gleitkomma-Arithmetik wurde durch eine Feinabstimmung geeigneter Parameter mittels Embedded Control Prototyping in Fahrversuchen kompensiert.

8 Zusammenfassung und Ausblick

In diesem Beitrag wurden die theoretischen Grundlagen für die Skalierung von Variablen zur Codegenerierung für Festkomma-Prozessoren vorgestellt. Weiterhin wurde die Vorgehensweise für die Berechnung der maximalen Aussteuerbereiche von Variablen komplexer Simulationsmodelle auf Basis des Signalflusses vermittelt. Diese Vorgehensweise wurde in einem Werkzeug realisiert, welches dem Anwender eine schnelle interaktive Überführung der prototypischen Steuergerätesoftware in Festkomma-Code mit *worst-case*-skalierten Variablen ermöglicht.

Durch die optimale Skalierung der Variablen werden Wertebereichüberläufe vermieden, es ist keine zusätzliche Sättigung der Variablen erforderlich. Damit entfällt auch eine ungünstige Beeinflussung einer Regelung oder Steuerung durch Ansprechen dieser Sättigungen.

Die Entwicklungszeit zur Erzeugung des Steuergerätescodes wird durch den Einsatz des vorgestellten Werkzeugs deutlich reduziert. Änderungen von Parametern oder sogar Strukturänderungen des Simulationsmodells können auch in sehr späten Entwicklungsstadien vorgenommen und der Prozess erneut automatisch abgearbeitet werden. Die Reproduzierbarkeit der erzielten Ergebnisse ist sichergestellt und die Zuverlässigkeit des Codes bleibt erhalten.

Die Funktionalität des Werkzeuges wurde am Beispiel eines Auslösealgorithmus für reversible Gurtstraffer nachgewiesen. Es wurde gezeigt, dass mit Hilfe zugeschnittener Werkzeuge die Erzeugung von Festkomma-Code für das Embedded Control Prototyping genauso beherrschbar ist, wie das Rapid Control Prototyping mit leistungsfähiger Gleitkomma-Hardware. Die experimentellen Ergebnisse mit Gleit- und Festkomma-Arithmetik sind praktisch identisch.

Das Werkzeug ist unabhängig von der Umgebung des Codegenerators. Das heißt, die berechneten Skalierungsergebnisse können für den Einsatz verschiedener Codegeneratoren verwendet werden.

Heute bieten einige Codegenerator-Umgebungen die automatische *worst-case*-Skalierung der Variablen für komplexe Simulationsmodelle an. Für die Entwicklung sicherheitskritischer Systeme ist ein Vergleich der Skalierungsergebnisse zweier voneinander unabhängiger Werkzeuge durchaus sinnvoll, um Skalierungsfehler eines Werkzeuges und daraus resultierende Wertebereichüberläufe auf der Festkomma-Hardware ausschließen zu können.

Literaturverzeichnis

- [1] H. Hanselmann: *DSP in Control: The Total Development Environment*. ICSPAT, Intl. Conference On Signal Processing & Technology, Aug. 1995, MA, USA
- [2] A. Farrenkopf, R. Starbek: *FlexRay Expansion Board for Universal ECU Prototyping Platforms*. Sonderheft "FlexRay" der Zeitschrift "automotive electronics & systems" zum FlexRay Product Day, 23. September 2004, Congress Centrum Böblingen.
- [3] U. Kiffmeier, L. Köster, M. Meyer und C. Witte: *Automatische Generierung von Produktionscode für Seriensteuergeräte*. Automatisierungstechnik 47 (1999), Heft 7, S. 295-304.
- [4] dSPACE: *TargetLink Production Code Generation Guide*. dSPACE GmbH, Paderborn, 2004.
- [5] R. Starbek, H. Henrichfreise und U. Kiffmeier: *Autoskalierung dynamischer Systeme zur Codegenerierung für Festkommaprozessoren*. 5. VDI-Mechatroniktagung 2003, 7.-8. Mai 2003, Fulda.