Echtzeitverhalten komplexer Systeme – optimal mit beiden Methoden analysieren und beherrschen

Tapio Kramer, Kramer@INCHRON.com
Dr. Ralf Münzenberger, Muenzenberger@INCHRON.com

Die Möglichkeiten und Einschränkungen zweier Methoden zur Analyse des Echtzeitverhaltens von Embedded Systemen werden hier diskutiert: die formale Scheduling-Analyse mittels Worst-Case Betrachtung und die Echtzeitsimulation. Es wird dargelegt, wie die Vorteile beider Methoden zur Analyse des Echtzeitverhaltens in einem modellbasierten Entwicklungsprozess sich ergänzend eingesetzt werden.

Bei der Entwicklung von dynamischen, echtzeitkritischen Embedded Systemen besteht eine wesentliche Herausforderung darin, das Zeitverhalten des Systems bereits mit der Systemarchitektur festzulegen und in der Implementation und Integration die Umsetzung des geplanten Zeitverhaltens sicherzustellen. Das Echtzeitverhalten wird im Wesentlichen durch drei Faktoren bestimmt: die netto Ausführungszeit des Codes auf der Hardware-Plattform; das Scheduling, das den Code zur Ausführung bringt oder unterbricht; die Anregung durch die Außenwelt, die das System zu Reaktionen veranlasst. Alle drei Faktoren werden in einer Analyse des Echtzeitverhaltens berücksichtigt.

Die netto Ausführungszeit eines kleinen Code-Abschnitts auf einer definierten Hardware ist recht einfach zu messen oder statisch zu ermitteln. Je mehr Code-Module aber auf derselben Hardware um die Rechenressourcen konkurrieren und Datenabhängigkeiten die Ausführungspfade verzweigen, desto schwieriger wird die Ermittlung ihrer brutto Ausführungszeiten. Diese resultieren aus dem Konkurrieren um Rechenressourcen mit anderen Code-Modulen und das schließt Blockierungen- sowie Verdrängungen durch andere Betriebssystem-Tasks und Interrupt-Service-Routinen mit ein.
Abhilfe schafft hier ein modellbasiertes Vorgehen zur Scheduling-Analyse mittels leistungsfähiger Methoden um die brutto Ausführungszeiten zu ermitteln und damit die Echtzeitfähigkeit des Systems zu beherrschen.

Analysemethoden zur Ermittlung des Echtzeitverhaltens

Die Analyse des Echtzeitverhaltens eines Embedded Systems kann auf mehrere Weisen erfolgen. Naheliegend, aber enorm aufwändig ist es, die Hardware und Software zu erstellen, integrieren und zu analysieren. Hierfür muss das System in alle möglichen Systemzustände gebracht und ausgemessen werden. Dieser Ansatz ist zeitaufwendig und führt zu der hohen Kosten und Entwicklungszeiten für jede Änderung.
Daher ist es effizienter modellbasiert vorzugehen und mit sich ergänzenden Analysemethoden, möglichst innerhalb einer Werkzeugumgebung, die Systeme frühzeitig auf ihr Echtzeitverhalten zu untersuchen. Zur modellbasierten Echtzeitanalyse werden die netto Ausführungszeiten und relevante Abhängigkeiten aller Code-Module in einem Systemmodell erfasst. Die Aktivierungen und das Scheduling der Prozesse, auf die die Code-Module zuvor gemapped werden, sind darin beschrieben und die Kommunikation des verteilten Systems wird darauf abgebildet.

Formale Analyse

Für die formale Analyse werden als Ausführungszeiten der einzelnen Code-Module die kürzest oder längst möglichen Zeiten verwendet. Die formale Analyse, beispielsweise mit dem Validierungs-Tool chronVAL, hat die Mächtigkeit, aus allen möglichen Anregungen des

verteilten Gesamtsystems die zu ermitteln, die den Best-Case und Worst-Case Fall der Antwortzeit des Systems hervorrufen. [IESS09] Auch der Best-Case ist wichtig zu berücksichtigen, da bei Systemen mit mehr als einer Ressource (Bus, Prozessor, Core) beispielswiese eine Best-Case Ausführung einer Task zu einer Worst-Case Anregung auf einem anderen Prozessor führen kann.
Durch die Reduzierung auf die Best-Case und Worst-Case netto Ausführungszeiten wird jedoch vom konkreten Systemverhalten abstrahiert, so dass es zu einem Verhalten in der Analyse kommen kann, das in der Realität nie auftritt. Dies ist beispielswiese der Fall, wenn in den Modellen bestimmte, komplexe Bedingungen nicht beschreiben sind, die den ermittelten Extremfall in der Realität unmöglich machen. Folglich wird das Systemverhalten hier etwas überschätzt, jedoch mit der Gewähr, dass der Worst-Case und Best-Case Fall berücksichtigt wurden.

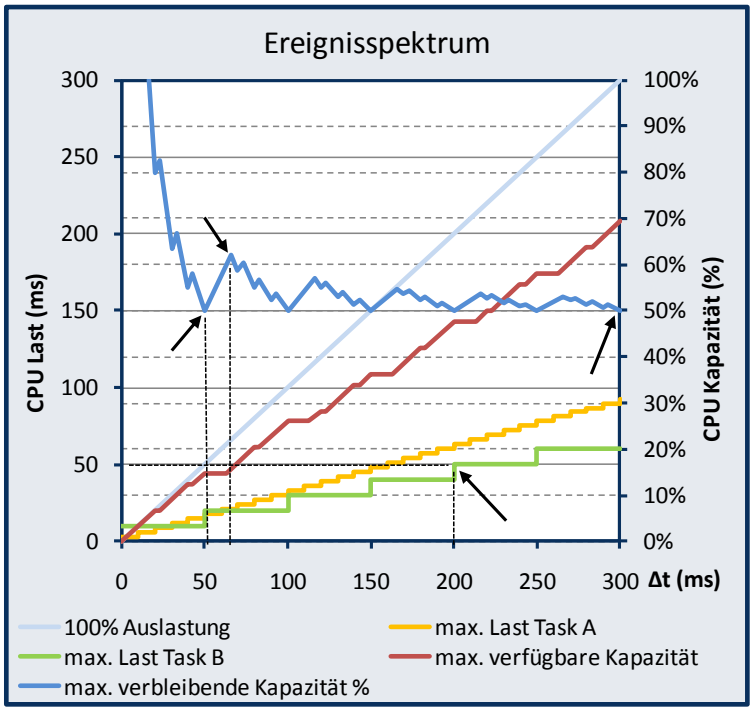

Bild 1: Die Auslastung der Ressource über Zeitintervalle als Ereignisspektrum dargestellt

Als Beispiel zeigt das Bild 1 ein Ergebnis einer formalen Analyse, in dem die Auslastung der Ressource (CPU oder Bus) über Zeitintervalle Δt aufgetragen und als Ereignisspektrum dargestellt ist. Die maximale Last (maximale Rechenzeitanforderung) der Task B mit periodischer Aktivierung alle 50ms ist in einem beliebigen Zeitintervall von 200ms im Worst-Case 50ms. Die maximal verbleibende Kapazität für niederpriore Prozesse ist prozentual dargestellt und zeigt deutlich, dass die über sehr große Intervalle verfügbare Kapazität der Ressource mindestens 50% beträgt. Der Verlauf ist jedoch nicht stetig fallend. Für Tasks mit ca. 60ms Zykluszeit ist die verfügbare Kapazität sogar geringer als bei ca. 50ms Zykluszeit. Ein Systemarchitekt kann hier also schnell ablesen, wo in seinem System noch Kapazitäten für Funktionserweiterungen sicher verfügbar sind. In anderen Darstellungen kann beispielsweise abgelesen werden, was sie maximale Zahl der Mehrfachaktivierungen einer Task ist.

Simulation des Echtzeitverhaltens

Bei der Simulationsmethode wird das gleiche Systemmodell ausgeführt und das Zeitverhalten der Komponenten simuliert. Das Modell wird von externen (IRQ, Buskommunikation) und

internen (Scheduler, Datenfluss, Aktivierungen) Ereignissen angeregt und der zeitliche Verlauf wird aufgezeichnet, dargestellt und ausgewertet.

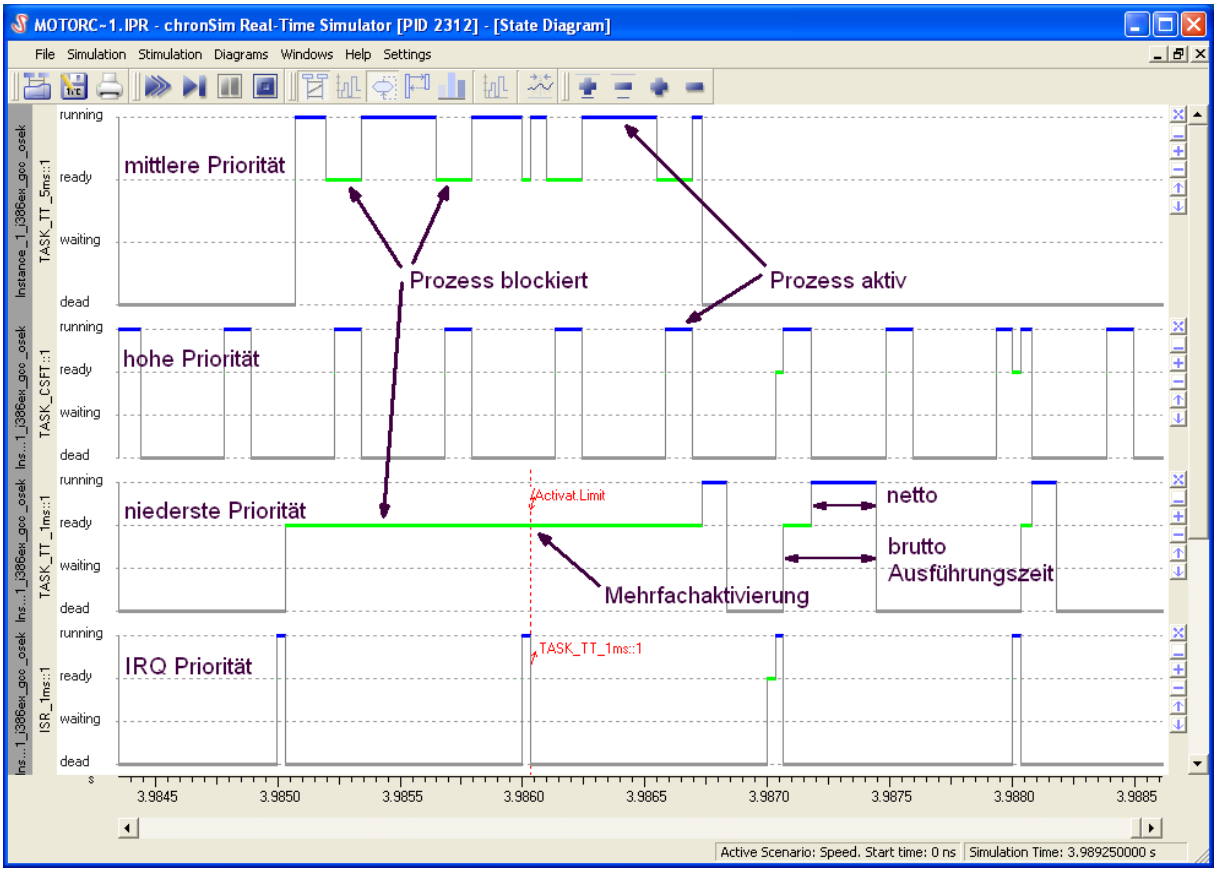

Bild 2: Prozesszustandsdiagramm

Die Simulation eines Systems bietet die Möglichkeit viele verschiedene Systemzustände einfach zu erreichen und zu analysieren. Hierbei gilt für die Simulation das Gleiche wie für Testfahrten, Hardware-in-the-Loop Systemen oder automatisierten Modultests. Je besser der Zustandsraum abgedeckt wird, desto besser ist auch die Testabdeckung. Ob man aber alle möglichen Zustände geprüft hat, also keine kritische Situation übersehen hat, wird durch ausgiebigeres Simulieren und Testen nur weniger wahrscheinlich – nicht sicher ausgeschlossen. Man betrachtet damit das System u.U. optimistisch.

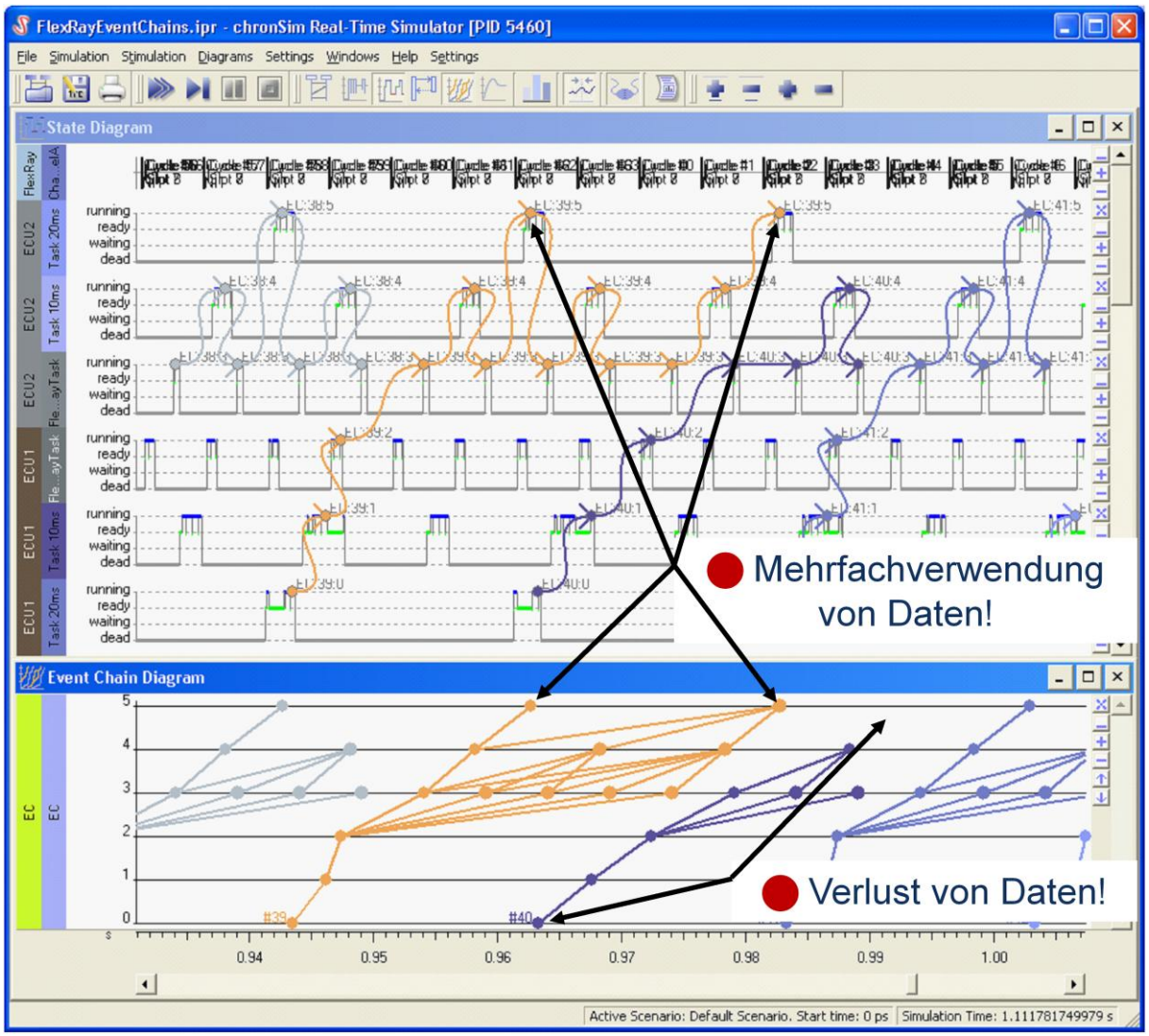

Bild 3: Prozesszustands- und Wirkkettendiagramm mit leicht nachvollziehbarem Datenfluss

In den Prozesszustandsdiagrammen des Echtzeitsimulators chronSIM (in Bild 2 und 3 oben) sind über der Zeit die Prozesszustände dargestellt. Werden im Systemmodell auch die kritischen Datenflüsse als Wirkketten modelliert, kann anhand derer (gebogene, aneinandergereihte Pfeile) verfolgt werden, mit welchen Daten in welcher Iteration einer Tasks gerechnet wird. So können sich durch ungünstige Architektur und Asynchronität zyklischer Prozesse eine Mehrfachverwendung (orangene Wirkkette) und ein Verlust von Daten (dunkelblaue Wirkkette) ergeben. Die hier dargestellte Wirkkette beginnt in der 20ms Applikations-Task auf ECU1, übergibt die Daten asynchron per FlexRay-Kommunikation zu ECU2 und endet in der Applikations-Task mit 20ms Periode auf ECU2. Mit den ermittelten Ausführungszeiten der Tasks und Wirkketten aus der Simulation können Echtzeitkriterien überwacht werden und in statistischen Auswertungen auch die Verteilung der Reaktionszeiten des Systems beobachtet werden.

Komplexe Antwortzeiten ermitteln

Um aus den Systemmodellen mit den netto Ausführungszeiten der Code-Module die brutto Ausführungszeiten der Funktionen und damit wiederum die Antwortzeiten des Systems zu ermitteln, liefern beide Methoden je nach Fragestellung geeignete Resultate.

Bei Systemen mit vielen, asynchronen Anregungen von Außen ist die realistische Stimulation aufwändig, aber entscheidend für die Erfassung realistischer brutto Ausführungszeiten. Ein Fahrerassistenzsystem hat beispielsweise ereignisgesteuerte Signale vom Fahrer aber auch verschiedene zyklische Signale von den Radsensoren, Kamera- und Radarsensoren als Stimuli. Soll dieses System sicher in allen Situationen die geforderten Reaktionen zeigen, muss die Umweltsimulation zur Erzeugung aller möglichen Anregungszustände auch alle Phasenlagen und Frequenzen der asynchronen, zyklischen Signale erzeugen. Zusätzlich müssen auch die ereignisgesteuerten Signalanregungen generiert werden.

Schon durch das systematische Erstellen und Analysieren der verschiedenen Stimulationsszenarien wird das Verständnis des Gesamtsystemverhaltens erhöht. In der anschließenden Simulation werden mit geeigneter Zahl von Stützstellen alle komplexen Anregungsszenarien stimuliert und die simulierten, vielfältigen Systemreaktionen analysiert. Die formale Analyse ermittelt die Reaktionszeiten des Systems in der best- und schlimmstmöglichen Kombination des Auftretens der Stimuli und deckt so sicher die Worst-Case und Best-Case Szenarien ab.

Ähnliches gilt für Systeme mit datenabhängigen Ausführungszeiten der Code-Module. Im selben Fahrerassistenzsystem ist beispielsweise die Ausführungszeit einzelner Algorithmen stark abhängig von den erfassten Daten. Für den Worst-Case Fall wird anhand des Wertebereichs der Eingangsvariablen ermittelt wie oft jede Schleife maximal iteriert und der Ausführungspfad des Algorithmus gewählt, der am längsten ist. Damit errechnet die formale Analyse die Worst-Case Antwortzeiten des Systems. In der Simulation wird die Funktion (bei chronSIM auch als C Code) mit realistischen Daten simuliert, da der Entwickler das Detailverhalten der Funktion kennt und auch solche Daten vorgeben kann. Das erlaubt eine gute Abdeckung der Systemzustände und ermittelt zuverlässig eine realistische Verteilung der Ausführungszeiten des Algorithmus und der Antwortzeiten des Systems.

Häufig gibt es Fragestellungen zu einzelnen, kritischen Funktionalitäten, die innerhalb geforderter Zeitschranken sicher vom System abgearbeitet werden müssen. Sie können mithilfe kritischer Pfade im System beschrieben und im Modell als Wirkketten dargestellt werden (siehe Bild 3 unten). Um die Worst-Case Antwortzeit dieser Wirkketten zuverlässig zu ermitteln, erfolgt die Überprüfung der Wirkketten mathematisch analytisch. Ein Antwortzeitdiagramm (siehe Bild 4 unten) gibt einen schnellen Überblick, ob die gewählte Architektur die kritischen Pfade sicher abarbeiten kann. Die Simulation liefert die verschiedenen Abläufe entlang der Wirkkette und deren Häufigkeitsverteilung (siehe Bild 4 oben).

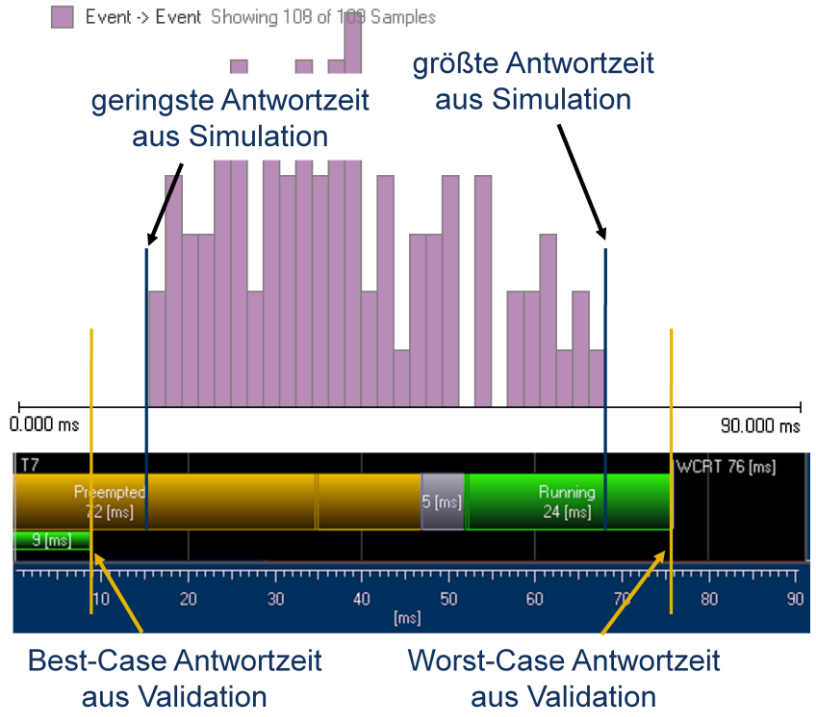

Bild 4: Antwortzeiten als Verteilung aus der Simulation sowie Best-Case und Worst-Case aus formaler Analyse

Best of Both Worlds

Die Analyse des Echtzeitverhaltens komplexer, vernetzter Systeme muss nach verschiedenen Aspekten erfolgen, je nachdem welche Fragestellung zu beantworten ist. Diese ergeben sich ganz unterschiedlich in früher Systemarchitekturphase oder beim Debugging, bei Funktionserweiterung oder Absicherung, bei der Integration oder Optimierung (des Gesamtsystems oder eines Teilaspekts). In einem ‚Best of Both Worlds' Ansatz werden die Vorteile beider Methoden kombiniert und das System für jede Fragestellung nach deren relevanten Aspekten untersucht. [Forum09]
Beispielsweise ist in frühen Entwicklungsphasen das System zunächst grob spezifiziert und kann mit seinen budgetierten Zeitbedarfen und geplanten Verhalten modelliert werden. Das resultierende Systemverhalten wird in der Simulation leicht nachvollzogen und gut verstanden werden. Alternative Systemarchitekturen werden sehr schnell modelliert und bewertet [Kess10]. Parallel dazu wird in der formalen Analyse eine effiziente Designraumexploration unterstützt durch Darstellung der Ereignisspektren (siehe Bild 1) erfolgen.

Die Vorteile des gemeinsamen Einsatzes von Echtzeitsimulation und formaler Analyse, wie es in der INCHRON Tool-Suite möglich ist, zeigen sich auch in frühen Projektphasen. Mittels virtueller Integration von Systemkomponenten werden Entwurfsentscheidungen bereits in der Architekturphase überprüft und ggf. optimiert. Die Integration des Gesamtsystems erfolgt virtuell im Modell und zeigt in der Analyse frühzeitig kritische Situationen im dynamischen Echtzeitverhalten auf – was später zu aufwändigen Fehlersuchen führen würde. Das korrekte Echtzeitverhalten wird so bereits im Design des Gesamtsystems berücksichtigt und nicht nur lokal in den Teilsystemen und Komponenten adressiert. [IAV10]
Sofern die virtuelle Integration in Kollaborationsprojekten mit Software-Anteilen mehrerer Zulieferer eingesetzt wird, erstellen die Beteiligten gemeinsam mit Hilfe der Echtzeitanalyse eine frühzeitige, gute Spezifikation der Echtzeitanforderungen und –eigenschaften. Systemarchitekten, Integratoren und Zulieferer erlangen so ein detailliertes, gemeinsames Systemverständnis [Aug10].

Zusammenfassung

In diesem Artikel wurde gezeigt, dass zu diversen Fragestellungen in unterschiedlichen Entwicklungsphasen mit Hilfe der Echtzeitanalyse durch Echtzeitsimulation und formale Analyse detaillierte Erkenntnisse zur Performance und Echtzeitfähigkeit des Systems gewonnen werden. Dabei liefert die Simulation ein sehr detailliertes Bild des dynamischen Systemverhaltens. Die formale Analyse liefert ein weniger detaillierteres Bild, berücksichtigt aber alle möglichen Abläufe und fokussiert die Extremfälle. Beide Methoden ergänzen sich und decken gemeinsam die verschiedenen Abläufe ab. Ein gemeinsamer Einsatz ermöglicht eine umfassende Analyse des Echtzeitverhaltens von komplexen Embedded Systemen, wie es durch nur eine Methode alleine nicht möglich ist.

Literatur

[Forum09] T. Kramer, Dr. R. Münzenberger; Echtzeitverhalten simulieren und validieren – verstehen und absichern; DESIGN&ELEKTRONIK-Entwicklerforum »Embedded-System-Entwicklung«, Oktober 2009, Ludwigsburg

[IESS09] K. Albers, F. Slomka; An Event Stream Calculus for the Schedulability Analysis of Distributed Embedded Systems; In proceedings of IESS'09, September 2009; www.iess.org; ISBN 978-3-642-04283-6

[Kess10] M. Kessler; Absicherung der Echtzeitanforderungen in einem komplexen Steuergerät; BICC: Innovation Forum Embedded Systems 2010; www.ifes2010.de

[Aug10] B. Augustin; Integration von bisher eigenständigen Steuergeräten und Funktionen – Herausforderungen eines Kollaborationsprojektes; 2. Fachkongress Echtzeitentwicklung 2010; www.echtzeitkongress.de

[IAV10] T. Kramer, Dr. R. Münzenberger; Absicherung des Echtzeitverhaltens mittels virtueller Integration; 4. Tagung - Simulation und Test für die Automobilelektronik, IAV, Mai/Juni 2010, Berlin; ISBN 978-3-8169-3023-5

Autor:

Dipl.-Ing. Tapio Kramer ist bei INCHRON als Marketing und Produkt Manager verantwortlich für die marktgerechte Weiterentwicklung der Software-Tools für echtzeitkritische embedded Systeme. Zuvor bekam er bei Wind River als Key Account Manager Automotive und bei National Instruments in verschiedenen Positionen Einblicke in die vielfältigsten Applikationen aus verschiedenen Industrien.