Performantes Testen von fehlertoleranten Hard- und Software-Systemen in der virtuellen Maschine FAUmachine

Stefan Potyra, Matthias Sand, Volkmar Sieh, Dietmar Fey
Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl für Rechnerarchitektur
Martensstr. 3, 91058 Erlangen
Email: {Stefan.Potyra, Matthias.Sand, Volkmar.Sieh, Dietmar.Fey}@informatik.uni-erlangen.de

***Zusammenfassung*—FAUmachine ist eine virtuelle Maschine, die gebräuchliche PC-Hardware simulieren kann. Es ist außerdem möglich, einen virtuellen Benutzer zu modellieren, welcher den virtuellen PC bedient. Durch eine Experiment-Steuerung besitzt FAUmachine die Fähigkeit, komplette Tests automatisiert durchzuführen. Im Gegensatz zu anderen virtuellen Maschinen werden einzelne Elemente wesentlich stärker an echter Hardware orientiert nachgebildet.**

FAUmachine bietet die Möglichkeit, Fehler in die virtuellen Hardwarekomponenten zu injizieren. Somit kann FAUmachine die Entwicklung von fehlertoleranten Systemen unterstützen.

Ferner erlaubt FAUmachine, die Komponenten der virtuellen Maschine mit einem VHDL-Interface zu koppeln. Beispielsweise kann eine in VHDL modellierte PCI-Karte an den simulierten PCI-Bus angeschlossen werden. Dadurch kann Hardware im Kontext eines komplexen Systems simuliert werden, ohne zunächst einen Prototypen anfertigen zu müssen.

Die Kopplung eines VHDL-Interfaces mit einer virtuellen Maschine für Systemtests ist ein neuer Ansatz, der in diesem Artikel vorgestellt werden soll.

I. Übersicht FAUmachine

FAUmachine[1], [2] ist eine Open-Source Implementierung einer virtuellen Maschine, welche gebräuchliche PC-Hardware simulieren kann. FAUmachine kann dazu genutzt werden, eine Vielzahl unterschiedlicher Betriebssysteme in unmodifizierter Form ablaufen zu lassen. Durch den Einsatz der Laufzeitübersetzung für den CPU-Simulator [3] kann eine hohe Simulationsgeschwindigkeit erreicht werden.

Die Hardwarekomponenten, für die Simulatoren in FAUmachine enthalten sind, umfassen Intel CPUs, IDE- und SCSI-Controller, NE2000 und Intel eepro100 Netzwerkkarten, aber auch Peripheriekomponenten, wie beispielsweise serielle Terminals oder Modems.

Im Gegensatz zu anderen virtuellen Maschinen, wie QEMU [4], VirtualBox [5] oder VMware [6] sind die Simulatoren von FAUmachine stark an der zugrundeliegenden, realen Hardware orientiert. FAUmachine kann auch auf sehr feingranulare Weise konfiguriert werden.

A. Simulationsmodell in FAUmachine

FAUmachine verwendet primär die Methode der diskreten, ereignisgesteurten Simulation. Wird eine Signaländerung (oder auch ein Bustransfer) angestoßen, so wird sofort eine Callback-Funktion in jedem verbundenen Komponenten-Simulator ausgeführt.

Ein echter PC besteht aus unterschiedlichen Komponenten. Signalleitungen und/oder Busse verbinden diese Komponenten. Brücken verbinden wiederum unterschiedliche Busse. Um eine hohe Simulationsperformanz zu gewährleisten, bildet FAUmachine komplette Bustransaktionen als Funktionsaufrufe ab. Diese Methode weißt gewisse Ähnlichkeiten zu *Transaction Level Modelling* auf.

B. Automatisierte Testläufe

FAUmachine bietet einen Experiment-Controller, der an der Schnittstelle des simulierten Rechners mit seiner Umgebung ansetzt. Die Struktur des zu simulierenden Systems wird in VHDL beschrieben. Der Experiment-Controller ermöglicht es, Simulationsläufe skriptgesteuert ablaufen zu lassen. Dadurch werden Benutzerinteraktionen mit dem System simuliert, wie z.B. Eingaben über Tastatur und Maus oder das Anschließen und Abziehen von USB-Geräten. Die Skripte werden in VHDL notiert.

C. Fehlerinjektion

Derzeit bietet FAUmachine die Möglichkeit, permanente und transiente Fehler zu simulieren, wie zum Beispiel Stuck-At-, Coupling- oder Bitflip-Fehler in Speichermodulen. Des Weiteren können Blockfehler von Festplatten und optischen Laufwerken, oder aber Komplettausfälle einer Festplatte injiziert werden. Ferner können Netzwerkfehler nachgeahmt werden.

II. Enge Kopplung hybrider Simulatoren

A. Modellbeschreibung

Zur feingranularen Beschreibung des Modells, welches durch FAUmachine simuliert werden soll, dient die Beschreibungssprache VHDL. Jede Komponente, welche als FAUmachine-Simulator existiert, besitzt eine Beschreibung der Schnittstellen in VHDL in Form einer *Entity*-Definition. Die Struktur des Simulationsmodells sowie die Verhaltensbeschreibung für die Benutzerinteraktion werden in VHDL notiert.

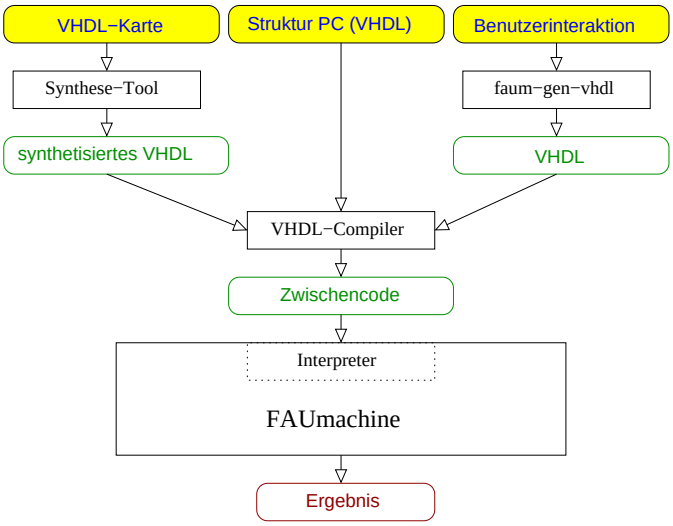

Abbildung 1. Toolchain

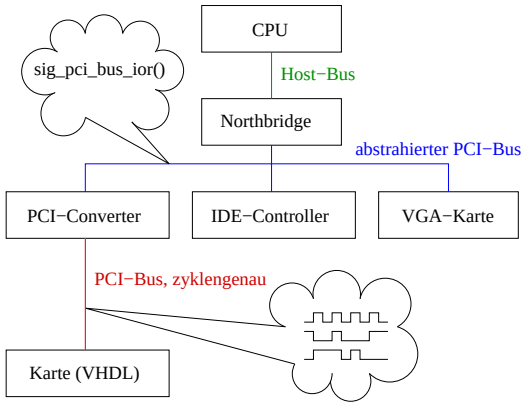

Abbildung 2. PCI-Konverter wandelt abstrahierten PCI-Bus in Signaländerungen

Dadurch, dass das komplette Simulationsmodell in VHDL spezifiziert ist, kann es um beliebigen anderen VHDL-Code erweitert werden: Beispielsweise kann eine in VHDL modellierte Hardware-Komponente mit den FAUmachine-Komponenten interagieren.

B. Toolchain

Abbildung 1 zeigt das Zusammenspiel einzelner Werkzeuge zum Erstellen von automatisierten Systemtests im Rahmen des FAUmachine-Projekts: Um eine VHDL-Beschreibung zu erhalten, welche realer Hardware ähnelt, sollte ein VHDL-Modell zunächst synthetisiert werden. Ein Synthesetool wie beispielsweise *Synopsis* [7] kann dazu verwendet werden, um eine Netzlistenbeschreibung (ebenfalls in VHDL-Form) zu erzeugen. Nur durch den Syntheseschritt können vernünftige Aussagen bei Verwendung von Fehlerinjektion getroffen werden (vgl. [8]). Das FAUmachine-Werkzeug `faum-gen-vhdl` dient dazu, die Beschreibung der Stimuli durch den simulierten Benutzer in eine VHDL-Beschreibung umzuwandeln. Kombiniert mit der strukturellen Beschreibung des Hardware-Systems und der Verhaltensbeschreibung in Form einer VHDL-Netzliste erzeugt der VHDL-Compiler des FAUmachine-Projekts einen vereinfachten und schnell interpretierbaren Zwischencode. Beim Starten des FAUmachine-Simulators wertet ein Interpreter diesen Zwischencode aus, und instanziiert und verbindet die nötigen FAUmachine-Komponenten. Die Simulation der Verhaltensbeschreibung in VHDL durch den Interpreter wird durch den Scheduler mit den anderen Komponentensimulatoren synchronisiert.

C. Abbildung von abstrahierten Bustransaktionen

Da sämtliche Busse innerhalb von FAUmachine lediglich als abstrahierte Bustransaktionen modelliert sind (vgl. Abschnitt I-A), ist ein direkter Zugriff aus VHDL nicht möglich. Diese abstrahierten Buszugriffe müssen zunächst in konkrete Signaländerungen des Busses abgebildet werden. Dies kann beispielsweise als Komponente erfolgen, welche sowohl den abstrahierten Bus, als auch die einzelnen Leitungen des Busses als Verbindungen verwendet, siehe Abbildung 2. Derzeit befindet sich eine solche Komponente für den PCI-Bus in Arbeit.

D. Methode der engen Kopplung

FAUmachine besitzt exakt eine, **globale** Simulationszeit für alle Komponenten-Simulatoren. Ein Scheduler verwaltet sämtliche zeitlich verzögerten Ereignisse. Somit kann der Synchronisationsaufwand, wie er beispielsweise bei paralleler Simulation entstehen würde, vermieden werden.

III. Schlussfolgerung

FAUmachine ist eine quelloffene virtuelle Maschine, welche Betriebssysteme unverändert simulieren kann. Im Kontext dieser virtuellen Maschine ist es möglich, in VHDL modellierte Komponenten zu simulieren. FAUmachine dient somit zur performanten Simulation hybrider Systeme. In die Komponenten von FAUmachine können Fehler injiziert werden. FAUmachine unterstützt daher das Entwickeln von fehlertoleranten Systemen. Die Fähigkeit zur Testautomatisierung bietet ferner die Möglichkeit für Systemtests.

Literatur

[1] FAUmachine Team, “FAUmachine,” URL: http://www.FAUmachine.org/, 2003–2009.
[2] M. Sand, S. Potyra, and V. Sieh, “Deterministic high-speed simulation of complex systems including fault-injection,” in *Proceedings of the 2009 IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2009.
[3] H. Höxer, M. Waitz, and V. Sieh, “Advanced virtualization techniques for FAUmachine,” in *11th International Linux System Technology Conference, Erlangen, Germany, September 7-10, 2004*, R. Spenneberg, Ed., 2004, pp. 1–12.
[4] F. Bellard, “QEMU, a fast and portable dynamic translator,” in *ATEC’05: Proceedings of the USENIX Annual Technical Conference 2005 on USENIX Annual Technical Conference*. Berkeley, CA, USA: USENIX Association, 2005, pp. 41–46.
[5] innotek GmbH, “VirtualBox,” URL: http://www.virtualbox.org/, 2008.
[6] VMware Inc., “VMware,” URL: http://www.vmware.com/, 2001.
[7] Synopsis Inc., “Synopsis,” URL: http://www.synopsys.com/, 2008.
[8] V. Sieh, O. Tschäche, and F. Balbach, “Comparing different fault models using VERIFY,” in *6th Conference on Dependable Computing for Critical Applications*, 1997, pp. 59–76.