Hardware/Software Co-Verifizierungsplattform für eingebettete Multiprozessoren

Aleksandar Simevski* und Milos Krstic*†
* IHP-Leibniz-Institut für innovative Mikroelektronik, Im Technologiepark 25, 15236 Frankfurt (Oder), Deutschland
† Universität Potsdam, August-Bebel-Str. 89, 14482 Potsdam, Deutschland
E-mail: {simevski, krstic}@ihp-microelectronics.com

***Abstrakt*—Es werden eine Strategie und eine Plattform für das Design und die Verifizierung eingebetteter Multiprozessorsysteme vorgestellt. Die Software-Verifizierung kann sehr früh in der Entwurfsphase des Systems beginnen, was nicht nur die Produkteinführungszeit erheblich verkürzt, sondern auch Aufwand und Kosten senkt, da die Plattform die frühzeitige Erkennung von Fehlern, Engpässen und Problemen sowohl bei der Hardware als auch in der Software ermöglicht, bevor Chips hergestellt werden. Es wird eine Fallstudie zum Entwurf und zur Co-Verifizierung eines 8-Kern-Multiprozessors vorgestellt. Bei Verwendung des Co-Verifizierungsansatzes gegenüber dem traditionellen Ansatz ergibt sich eine Verbesserung der Produkteinführungszeit um 30%.**

I. Einleitung

Deep-Submicron-Prozesse ermöglichen die Aufnahme sehr komplexer Systeme. Andererseits wird der Aufwand für die Verifizierung der Funktionalität drastisch erhöht. Die Verifizierungszeit macht 70% bis über 80% der gesamten Entwicklungszeit [1] aus. Ein Vergleich zweier Co-Verifizierungsmethoden eines ARM-Prototyps wird in [2] vorgestellt, in dem die Vor- und Nachteile auf die Geschwindigkeit in Abhängigkeit des Grades der Simulation gegenübergestellt sind. Eine Methode zur Co-Verifizierung der in SystemC beschriebenen 8051-Architektur ist in [3] angegeben. Leistungsfähige kommerzielle Umgebungen [4] zur Beschleunigung der Co-Verifizierung von ASICs basieren auf Hardware-Emulatoren, vorausgesetzt, der Mikroprozessor verfügt über einen ISS (Instruction Set Simulator), ein RTL-Modell (Register Transfer Level) oder eine tatsächliche physische Komponente.

In diesem Paper[1] schlagen wir eine Strategie und eine Plattform für die Co-Verifizierung von eingebetteten Multiprozessoren (MP) durch Hardware/Software (HW/SW) vor, um die Produkteinführungszeit zu verkürzen, indem das SW-Design und die Verifizierung früh im Design begonnen werden. Somit werden HW/SW-Debugging zu gleichzeitigen Prozessen. Während die Software getestet und verifiziert wird, wird der MP auch im Hintergrund funktional verifiziert. Auf diese Weise können Fehler früh in der HW-Entwurfsphase gefunden werden. Es könnten auch SW-Probleme vorgesehen werden, deren Lösung entweder in der Hardware oder in der Software liegen könnte. In einem traditionellen Szenario werden viel Zeit, Geld und Aufwand verschwendet, wenn HW-Fehler nach der Chipherstellung aufgedeckt werden. In einem solchen Fall werden entweder SW-Workarounds angewendet, die in der Regel die Funktionalität und Leistung des Systems beeinträchtigen, oder es wird ein weiterer Zyklus von Debugging, Verifizierung, Produktion und Tests eingeleitet, der die Produkteinführungszeit beeinträchtigt. Die vorgeschlagene Strategie und Plattform soll verhindern, dass solche Szenarien eintreten.

[1] This work has partially received funding from the European Union's Horizon 2020 Research and Innovation programme under the grant agreement No. 870365 (MORAL).

II. Multiprozessor Co-Verifizierungsplattform

Die vorgeschlagene MP-Plattform ist eine Erweiterung unserer vorherigen Single-Kern-Plattform [5]. Abb. 1 zeigt das Konzept für MPs mit N Kernen. Eingabe ist die Anwendungs- oder Testsoftware, entweder in Hochsprache oder in Assembler. Der Compiler/Assembler übersetzt den Eingangscode in den binären Maschinencode des MPs und erzeugt ein binäres Speicher-Image. Hier wird ein Shared-Memory-MP-Speicher angenommen. Die Plattform könnte jedoch leicht für MPs mit verteiltem Speicher erweitert werden mit der Folge, dass mehrere binäre Speicher-Images erzeugt werden müssen, d. h. eines für jeden der separaten Speicherbereiche.

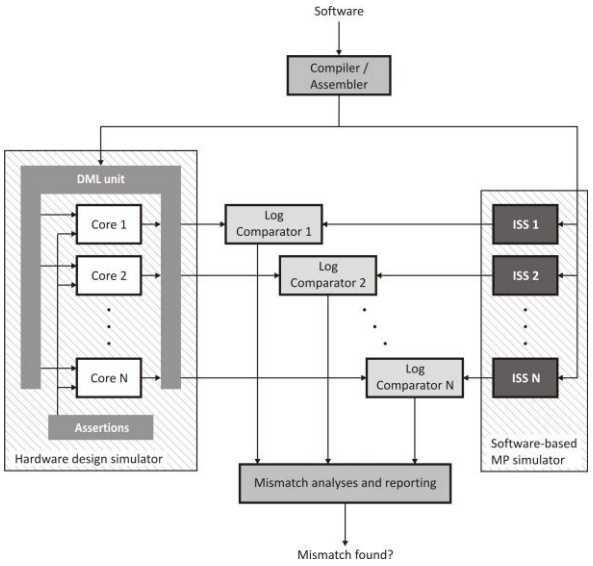

Abb. 1. Multiprozessor Co-Verifizierungsplattform

Das vom Compiler oder Assembler erzeugte binäre Speicher-Image wird in den HDL-Simulator (Hardware Description Language) und in den softwarebasierten MP-Simulator eingegeben. Das RTL-Modell des MP, die Einheit für Drive, Monitor und Log (DML) und die Assertions, die die Funktionalität überprüfen, sind in SystemVerilog geschrieben. Der softwarebasierte MP-Simulator (in unserem Fall in Perl geschrieben) enthält N ISS-Module, die das Verhalten der einzelnen MP-Kerne nachahmen.

Die DML-Einheit steuert die MP-Eingänge an und überwacht und protokolliert die Ausgänge und den MP-Zustand, d. h. den Zustand aller Kerne im MP. Der Core State (CS) ist der Zustand aller General-Purpose-Register, System- und Steuerregister sowie die Ein- und Ausgänge des Kerns. Die DML-Einheit protokolliert den CS-Status nach

jeder ausgeführten Instruktion in separaten Dateien für jeden Kern. Die DML-Einheit enthält auch das Modell des Hauptspeichers (initialisiert mit dem binären Speicher-Image) sowie Debugging-Funktionen wie Disassembly und umfangreiche Berichterstattung. Zusätzlich wird jede ausgeführte Instruktion von mindestens einer Assertion dahingehend überprüft, ob alle durch die Instruktion vorgenommenen Änderungen zulässig sind und Änderungen, die nicht zulässig sind, tatsächlich nicht vorgenommen werden. Hier haben wir unsere bereits entwickelten Assertions in [5] wieder verwendet, indem wir sie einfach auf alle Kerne angewendet haben.

Auf der anderen Seite werden SCS-Logs (Software Core State) vom softwarebasierten MP-Simulator erstellt. Die CS- und SCS-Logs sollten vollständig identisch sein, da für äquivalente MP-Modelle derselbe Code eingegeben wird. Wenn nicht, entsprechen entweder das RTL-Modell oder die ISS (oder beide) nicht der Spezifikation. Auf diese Weise deckt die Plattform heimliche und verborgene Fehler in allen Segmenten auf (HW/SW-Design, Toolkette, DML-Einheit, Assertions usw.) und fördert die Konvergenz mit der Spezifikation, was das Vertrauen in das Design stark erhöht. Die von der DML generierten Berichte weisen früh in der Entwurfsphase des MP auch auf Probleme und Ineffizienzen in der HW/SW, eventuelle Kompromisse, Engpässe und Fehlverhalten hin. Ein wesentliches Merkmal der Plattform ist außerdem, dass der Vorgang vollständig automatisiert ist.

III. Fallstudie: 8-Kern-Multiprozessor

Die in Abschnitt II vorgestellte Plattform wurde zur Implementierung eines 8-Kern-MP verwendet [6]. Die gesamte Entwicklungszeit beträgt 700 Personentage (P.D.), von denen 180 für das reine HW-Design bestimmt sind, d. h. 74% der gesamten Entwicklungszeit werden für die Verifizierung aufgewendet. Um die Verkürzung der Produkteinführung abzuschätzen, nehmen wir an, dass die Teams Hardware (HW), Software (SW) und Verifikation (VER) jeweils eine Person haben, während das Team Tool Chain (TC) zwei Personen hat (insgesamt fünf Personen). Abb. 2 zeigt einen Vergleich des traditionellen Ansatzes, bei dem das SW-Design und die Verifizierung nach der Chipherstellung beginnen, mit dem Co-Verifizierungsansatz, bei dem es zum frühestmöglichen Zeitpunkt beginnt, d. h. nach Abschluss des RTL-Designs. PHY bezeichnet die physikalische Chipherstellung, die bei beiden Ansätzen gleich ist. Andererseits kann die SW-Co-Verifizierung erst beginnen, wenn der Entwurf des Assemblers und einer minimalen Arbeitsversion der ISS (in unserem Fall ca. 60 P.D.) abgeschlossen sind. Abb. 2 zeigt, dass eine Verbesserung der Produkteinführungszeit von ungefähr 30% erreicht werden konnte (360 gegenüber 510 P.D.). Natürlich gehen diese Vorteile mit dem Preis zusätzlicher Arbeit einher, die für die Entwicklung der ISS erforderlich ist, d. h. einem zusätzlichen Mitglied im TC-Team, das im Fall der traditionellen Verifizierung nicht erforderlich ist. Das heißt, der Co-Verifizierungsansatz erfordert insgesamt fünf statt vier Personen.

Tabelle I zeigt die Simulationszeiten sowohl des HD-Simulators als auch des SW-basierten MP-Simulators für eine große Anzahl von Instruktionen. Der SW-basierte Simulator ist fast zehnmal schneller als der HDL-Simulator und bestimmt praktisch die Simulationszeit. Aus Tabelle I geht hervor, dass die Verifizierung von zehn Millionen Instruktionen des 8-Kern-MP fast 26 Stunden dauert.

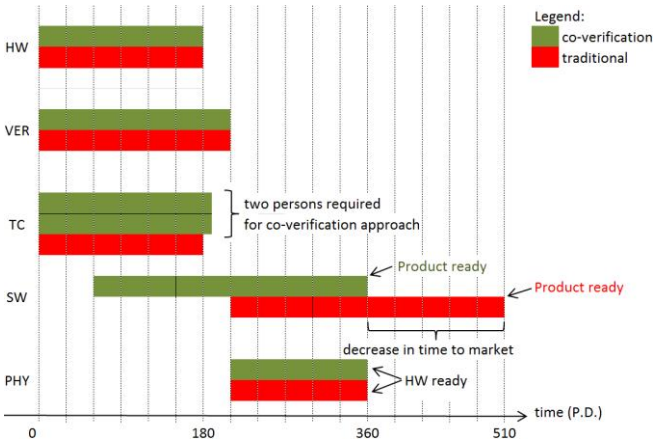

Abb. 2. Multiprozessor Co-Verifizierungsplattform

TABELLE I
HDL und SW-basierte Simulationszeiten in Sekunden

Instr. Anzahl (Million)	HDL Simul.	SW-basierte Simul.
1	9.534	930
2	18.363	1.981
3	28.177	2.798
5	47.391	4.608
10	92.732	8.946

IV. Zusammenfassung

Es wird eine Plattform für die Co-Verifizierung von Multiprozessoren vorgestellt, die das Debuggen und Auffinden von Problemen in der Hardware während des SW-Designs und der Verifizierung ermöglicht. Neben der Verkürzung der Produkteinführung durch Parallelisierung der Prozesse von HW- und SW-Design und -Verifizierung wird das Potenzial zur Behebung von Fehlern und Problemen vor der eigentlichen Chipherstellung drastisch erhöht, was viel Aufwand, Geld und Zeit spart und zusätzlich Produktverzögerungen eliminiert. Der dritte Vorteil ist die Möglichkeit, umfangreiche Berichte zum HW- und SW-Verhalten zu erstellen, die auf mögliche Ineffizienzen, Engpässe oder andere Probleme bei HW und SW hinweisen.

Literatur

[1] H. Foster. Does design size influence first silicon success? Verification Horizons, a publication of Mentor Graphics Corporation, 11(1):4–7, Feb. 2015.

[2] G. Wang, Q. Shi, Z. Yu, and Z. Yu. The study of HW/SW coverification on ARM-prototype system. In Solid-State and Integrated Circuit Technology (ICSICT), pp. 1847–1850, Oct. 2008.

[3] L. Semeria and A. Ghosh. Methodology for HW/SW coverification in C/C++. In Asia and South Pacific Design Automation Conference (ASP-DAC), Proceedings of, pp. 405–408, Jun. 2000.

[4] Cadence Design Systems. Cadence Palladium XP II verification computing platform. Technical Brief, 2013.

[5] A. Simevski, R. Kraemer, and M. Krstic. Platform for automated HW/SW co-verification, testing and simulation of microprocessors. In 13th IEEE Latin American Test Workshop (LATW), Apr. 2012.

[6] A. Simevski. Architectural framework for dynamically adaptable multiprocessors regarding aging, fault tolerance, performance and power consumption. PhD thesis, BTU Cottbus-Senftenberg, Jan.2015.