行政院國家科學委員會專題研究計畫　成果報告

子計劃一：與組織探索階段互動之系統階層驗證技術

計畫類別：整合型計畫
計畫編號：NSC91-2215-E-009-079-
執行期間：91 年 08 月 01 日至 92 年 07 月 31 日
執行單位：國立交通大學電機與控制工程學系

計畫主持人：董蘭榮

計畫參與人員：鄭顯文，江宗錫，張智凱，黃柏涵，林盟淳

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 92 年 10 月 27 日

行政院國家科學委員會補助專題研究計畫 ■ 成 果 報 告 □期中進度報告

對以智財單元為基系統晶片設計之驗證與測試技術開發研究

子計畫一:與組織探索階段互動之系統階層驗證技術

計畫類別：□ 個別型計畫　　■ 整合型計畫
計畫編號：NSC 91－2215－E－009－079－
執行期間：　91 年　8 月　1 日至　92 年　7 月　31 日

計畫主持人：董蘭榮
共同主持人：
計畫參與人員：　鄭顯文
江宗錫
黃柏涵
張智凱
林盟淳
成果報告類型(依經費核定清單規定繳交)：■精簡報告　□完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立交通大學電機與控制工程學系

中　華　民　國　九十二年　十月　二十六日

行政院國家科學委員會專題研究計畫成果報告
對以智財單元為基系統晶片設計之驗證與測試技術開發研究
子計畫一:與組織探索階段互動之系統階層驗證技術
System-Level Verification Interacting with Architecture Exploration

計畫編號：NSC 91－2215－E－009－079－
執行期限：90 年 8 月 1 日至 91 年 7 月 31 日

主持人：董蘭榮　國立交通大學電機與控制工程學系
計畫參與人員：
鄭顯文　國立交通大學電機與控制工程學系
江宗錫　國立交通大學電機與控制工程學系
張智凱　國立交通大學電機與控制工程學系
林盟淳　國立交通大學電機與控制工程學系
黃柏涵　國立交通大學電機與控制工程學系
Email: lennon@cn.nctu.edu.tw

一、中文摘要

系統晶片設計涵蓋很廣的設計空間。設計者通常需要考量許多可能的系統組織包括選擇演算法則、挑選組織元件、建構候選組織。設計如此的複雜系統誠屬不易，而要設計出能完全符合要求、正確無誤的系統更為困難。設計上的失誤必須要儘早排除，否則在後續階段才發現的失誤將造成耗費耗時的再設計周期。因此，設計者必須面對兩項課題，其一是實現設計程序本身、另一是建立正確的設計結果。其中，設計的正確性將為本計劃的主軸。此子計劃第一年之工作為提出組織元素的成本模型、發展成本評估核心公式、定義效能模型資料結構、發展基礎效能模型。

關鍵詞：效能模型、系統階層驗證、系統晶片

Abstract

The System-On-Chip (SOC) design encompasses a large design space. Typically, the designer explores the possible architectures, selecting algorithms, choosing architectural elements, and constructing candidate architectures. Designing such a complex system is hard; designing such a system which will work correctly is even harder. Design errors should be removed as early as possible; otherwise, errors detected at the later stages will result a costly, time-consuming redesign cycles. Thus, the designer should face two distinct tasks in SOC design; carrying out design process itself and establishing the correctness of a design. Design correctness is the main theme of this project. In the first year, the tasks of this project are: proposing cost models of architecture elements, developing cost estimation engine, defining the data structure of performance modeling, developing the fundamental performance models

Keywords: performance modeling, system-level verification, system-on-chip

二、緣由與目的

The goal of this paper is to address the challenges of increasingly complex applications, highly integrated systems and shorter design cycles of system-on-a-chip (SOC) designs. The proposed design methodology is based on efficient and seamless Intellectual Property (IP) reuse. It is divided into two phases: an exploration phase,

and a synthesis phase.

The exploration of architectures (based on a large variety of IP cores) presents several challenges to the designer, such as (1) accuracy of simulation, (2) speed of simulation, (3) hardware/software interfaces [11], (4) efficiency of IP core selection, and (5) lack of a unified environment that can capture algorithm and architecture attributes. We have developed a front-end design tool that achieves efficient and rapid exploration of architectures. Architecture exploration is based on a reusable IP library composed of both hardware and software components. Both hardware and software components are modeled at the same level of abstraction. The front-end tool maps the designer's solution into a system-level simulation model and generates performance results for the early design phase.

Within our proposed exploration tool, we use three axes, {algorithm, attribution, structure}, to describe a target system. Each of these axes can be tuned to compose architecture. Based on the three-dimensional representation, the proposed design tool maps algorithms onto architectures, allowing for hardware/software partitioning and event-driven task scheduling. Using the attributes of components (IP cores), the tool then generates a time-faithful model of the architecture. After simulation, the model will estimate performance measures such as processor utilization, memory size, bus utilization, and interrupt overhead. Based on these measures, designers can find suitable specifications of IP components and make a decision for the selection of architectures, before committing to any IP core or investing in long and costly implementation details.

Intellectual Property (IP) reuse is becoming essential in system-on-a-chip design. It helps designers meet the challenges of increasingly complex applications, highly integrated systems and shorter design cycles. However, effective IP reuse poses several problems, notably in the integration and validation of the foreign IP within the target system. State-of-the-art design methodologies do not address these issues, but rather rely on IP vendors to assist designers with the custom integration of their IP cores. Recently, several EDA companies teamed up with IP vendors to provide some solutions to these problems [1]. The results are interesting, but still rather IP-specific. Our approach is meant to put IP integration technology at the fingertips of the designers themselves, thus reducing the gap between IP vendors and system designers.

Once the architecture and the appropriate IP cores are identified, we use our HW/SW cosimulation engine to generate a system cosimulation test-bed by producing all the required interfaces and wrappings between the different IP cores.

Our cosimulation interfacing approach extends the work described in [2]. It allows for integration and validation of IP cores of different forms (i.e., not limited to C), and involves an optimized synchronization scheme that can run in both event-driven and lock-step modes (based on the different phases of interactions between the IP core and the rest of the system). The API layer adds to the generality and flexibility of the proposed wrapping scheme, and protects the IP core.

三、結果與討論

The cosimulation engine is based on a high-level model describing the interactions of an IP with the rest of the system, and a multi-layered wrapping scheme of an IP core. Using the interaction models and the wrapping scheme, it allows plug-and-play IP integration and validation, and heterogeneous cosimulation.

Several issues must be considered when integrating an IP core. One paramount issue is the intended interaction of the foreign IP with the rest of the system. We describe such knowledge in a high-level model that forms the basis for the synthesis of the IP-specific interfaces to the rest of the system. Consider for example the high-level model of a master/slave interaction depicted in Figure 8.a. Here, both the master and the slave are Intellectual Property cores. In the context of Figure 1, the master is the DSP processor, and the slave is one of the ASICs.

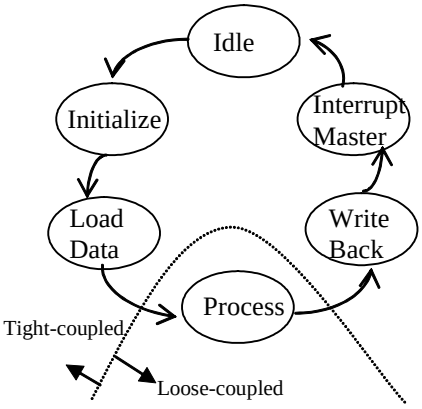

Fig.1 Master/Slave Interaction

An example scenario of a master/slave interaction, as shown in Fig.1, is as follows. Initially, the slave is in an “IDLE” state, waiting for orders from its master. It then moves to an “Initialization” state triggered by the master. During that state, the master is in charge of initializing its slave. Typically, the master will reset the IP core, and write into the register file of the slave to set the necessary information for the execution and completion of the allocated task. For instance, the master can set the pointers to the program and data memories of the slave. It also sets other core-specific information such as the mode of operation. During the following state, the master will initialize the DMA to write the block of data (to be processed) to the local memory of the slave. Finally, when the data is transferred, the master supplies a “go” signal to its slave, triggering it to move to a “Process” state. During that state, the interaction between the master and the slave becomes loose (that is no lock-step simulation is needed). Upon completion of the execution of the allocated task, the slave gets into a “Write Back” state, during which the results are written back to the global memory. The slave then interrupts its master, informing it of the completion of the task. It then moves back to the original “IDLE” state where it waits for new orders from its master.

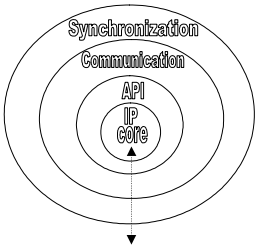

Fig.2 Integration layers of an IP core

Consider the IP model shown in Fig.2. The IP core is wrapped with three layers: the API layer, the communication layer, and the synchronization layer. Such a multi-layered model allows a seamless integration of IP cores of different origins and in different forms. Typically, each IP can be accompanied by its own different set of verification and modeling tools. Such tools include HDL models, Instruction Set Simulators (ISS), C models, and emulator boards. These vary in speed, accuracy, and level of abstraction, and they are often used interchangeably during the design cycle. The goal of our multi-layered wrapping scheme is to allow IP-based system integrators to use IP cores in a plug-and-play fashion, and integrate and reuse different IP models interchangeably.

The IP reuse technology described in this paper is being used successfully in real industrial environments. It is proving to be an efficient and fast way to run heterogeneous cosimulation, and quickly incorporate foreign IP cores into embedded systems. Substantial productivity gains and design-time reductions have resulted from its use. In a high-density central site modem (HDCSM) application, we have been able to wrap several in-house IP cores and set up a heterogeneous cosimulation system platform in less than two weeks. This includes the development of the API functions for the IP cores, and the interaction models with the system. We have developed and tested the heterogeneous cosimulation IP models with little knowledge about the cores themselves. Applying different speedup techniques, we have been able to accelerate the heterogeneous system cosimulation substantially (often by three orders of magnitude). For example, we have been able to speed up the system simulation of the HDCSM application from three instructions per second (RTL) to 1600 instructions per second. Thus far, the IP-based synthesis technology is only partially automated. The communication layer and the client/server interfaces are generated automatically. However, the designer’s assistance is needed to define the interaction model, and the API functions. Figure 3 illustrates analysis environment using the proposed cosimulation

engine.

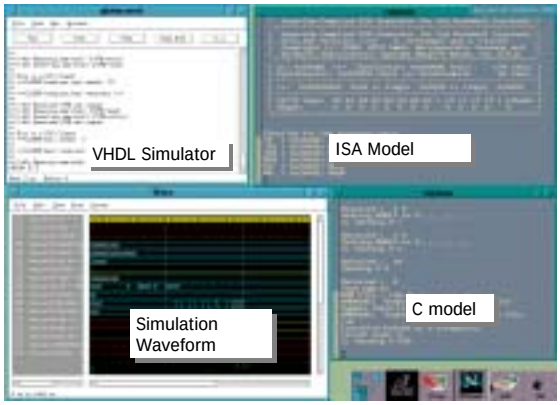

Fig. 3 HW/SW co-simulation environment

四、成果自評

本計畫第三年成功建立軟硬體共模擬環境，可有助於組織探索階段完成軟硬體互動之驗證工作。此技術已應用於各種有線通信及多媒體 SOC 設計上。本計畫之研究成果已發表下列兩篇國際會議論文與一篇國內會議論文：

1. Hsien-Wen Cheng and Lan-Rong Dung, 2002, “EFBLA: A Two-phase matching algorithm for FAST motion estimation,” PCM 2002 .
2. Hsien-Wen Cheng and Lan-Rong Dung, 2003, “A Novel Vario-Power Architecture of Motion Estimation Using a Content-based Subsample Algorithm,” SiPS 2003.
3. Hsien-Wen Cheng and Lan-Rong Dung, 2003, “A Power-Aware Architecture for Motion Estimation,” the 14th VLSI/CAD 2003.

另外，部分研究成果正投稿于 IEEE 期刊。

經由本計畫之執行已培養四名碩士畢業生。該四名碩士畢業生目前服務於系統晶片相關之高科技企業。

五、參考文獻

[1] Steven Vercauteren, Bill Lin, and Hugo De man, “Constructing Application-Specific Heterogeneous Embedded Architectures from Custom HW/SW Applications,” 33rd Design Automation Conference, June, 1996.

[2] C. A. Valderrama, Fran□ois Na□abal, Pierre Paulin, Ahmed Amine Jerraya, “Automatic Generation of Interfaces for Distributed C-VHDL cosimulation of Embedded Systems: an Industrial Experience,” 7th IEEE International Workshop on Rapid Prototyping, pp. 73-77, Thessalonki, Greece, June 1996.

[3] Lan-Rong Dung, Mohamed Ben-Romdhane and Marius Vassiliou, “IP-Based Architecture Exploration,” DesignCon99, February, 1999

[4] Mohamed Ben-Romdhane, Marius Vassiliou and Lan-Rong Dung, “Rapid Prototyping of Multimedia Chip Sets”, ICASSP-99, March,1999

[5] Richard Goering, “New Tools will Force Embedded Designers to Link Hardware/Software Efforts – Codesign turns workplace on its head,” EETimes, Issue:988, January 12, 1998

[6] J.K. Adams and D.E. Thomas, “The Design of Mixed Hardware/Software Systems,” Proc. Of 33rd DAC, 1996, pp.515-520

[7] Lan-Rong Dung and Vijay K. Madisetti, 1996, Fall, “Conceptual Prototyping of Scalable Embedded DSP Systems,” IEEE Design and Test of Computers, pp. 54-65

[8] C. A. Valderrama, Fran□ois Na□abal, Pierre Paulin, Ahmed Amine Jerraya, “Automatic Generation of Interfaces for Distributed C-VHDL cosimulation of Embedded Systems: an Industrial Experience,” 7th IEEE International Workshop on Rapid Prototyping, June 1996.