行政院國家科學委員會專題研究計畫成果報告

計畫名稱：**硬體-軟體同步合成方法論**
Hardware-Software Cosynthesis Methodology
計畫編號： NSC 89-2215-E002-045
執行期限： 89 **年** 8 **月** 1 **日 至** 90 **年** 7 **月** 31 **日**

主持人：**陳少傑　臺灣大學電機工程研究所　教授**
計畫參與人員：陳忠政，謝永強，楊昇龍，鍾道文
臺灣大學電機工程研究所

一、 摘要

硬體-軟體相輔合成一分散式崁入系統，如系統分割、合成、模擬及驗證，都必須考慮到分散式的各個副系統所在之實際狀況及特性。因為分散式崁入系統中常包含有一些相似功能之物件，為了減少設計的複雜度，因此設計之重複使用(Design Reuse)**技術常被使用到分散式崁入系統的設計中，所以在我們提出的分散式崁入系統相輔合成設計** (Distributed Embedded System Co-Synthesis**，簡稱** DESC)**方法論就是使用物件導向的相輔設計技術，因為物件導向技術考慮到物件設計之重複使用性及系統之實際的限制問題。**

Abstract

The hardware-software co-synthesis design of a distributed system is more complex than that of a centralized one, because each phase of codesign, including copartitioning, cosynthesis, cosimulation, and coverification, must consider the inherent physical restrictions imposed by the distributed characteristics of such a system. To reduce design complexity, design reuse techniques have to be applied because distributed systems often contain several parts that are similar in function. Hence, an object-oriented (OO) codesign approach, which considers physical restriction and object design reuse, is adopted in our newly purposed *Distributed Embedded System Co-Synthesis* (DESC) methodology.

二、 計畫緣由與目的

近幾年來，由於積體電路製程技術以及 VLSI **設計技術之進步，使得電子零組件之功能愈形豐富，性能也不斷提升，複雜度也愈來愈高。一個系統晶片上可包括處理器或** ASIP**、**ASIC**、記憶體、介面及聯結系統等。一個最佳化的應用系統必須把運算工作適當地分配到處理器或**ASIP **上的軟體及**ASIC **內的硬體去執行。硬體－軟體的同步設計對系統設計師是一大挑戰。硬體與軟體元件的使用與再使用能比傳統的積體電路設計方法花更少的時間而產生更高品質（即成本／效能、適應性等）的產品。**

早期沒有自動設計的工具可用，在系統設計上之硬體的設計及軟體的設計兩大部分，需經由個別的初步設計，及多次的溝通與協調才能完成一完整的設計。然而在訂定規格中何者應使用硬體或軟體僅能借助經驗，但是在整個設計過程仍然存在著許多的變數，所以在設計的中途中可能會發現當初的判斷是錯誤的而無法滿足系統的規格要求，因此做了許多的重複性之工作以致浪費許多的時間。尤其當系統到達硬體及軟體整合的階段時往往產生許多困難，為了解決整合常需重新設計而產生額外的費用。所以為了解決以上的問題，因此有人提出了「軟硬體相輔合成設計」的方法，這種方法的提出將更有效的提昇整個系統的效能、可靠度以及降低系統所需之費用。當然應用相輔設計的技術於崁入系統時其所涉及的技術就更複雜了，所以近期許多研究者於這個領域上投入相當多的努力。

我們針對此一問題提出一套完整的相輔設計方法論稱之為「分散式崁入系統相輔設計」（Distributed Embedded System Co-synthesis，簡稱 DESC）。DESC 方法論中使用物件導向的方式來作軟硬體合成，所以能提供設計之重複使用（Reuse）；同時其所用之「多層次分割」（Multi-Level Partitioning，簡稱 MLP）技術中因考慮分散式系統中實際存在的限制而得到較好的軟硬體分割結果；再者經由系統原型（Prototyping）之製作，可以很快驗證合成結果之系統功能的正確性。

三、 研究方法

DESC 的方法論中使用三種模塑（Model）；第一種是「物件模塑技術」（Object Modeling Technique，簡稱 OMT），這種模型提供作為設計系統的描述與輸入。第二種是「線性混合自動機」（Linear Hybrid Automata，簡稱 LHA），這種模型提供作為設計系統時之內部模塑以及作為驗證之用。第三種是「SES/workbench 模擬模型」，這種模型提供作為設計系統時之系統效能評估之用。

OMT 物件模塑技術是使用物件模型（Object Model）、動態模型（Dynamic Model）與函數模型（Functional Model）來描述一個系統，其中物件模型是用來描述各個物件存在系統的樣式及相互的關連。動態模型是用來描述一系統中各個物件間其時間上的相互關係。函數模型是用來描述一系統資料的轉換及流向。在我們提出的分散式崁入系統相輔設計方法論中，我們並不假設任何特殊的內部連接架構，而此一任意連接架構可以由設計者經由在建立物件模型中副系統之間的連結關係而決定。

LHA 線性混合自動機模型是由一組有限變數及包含標籤的多重圖形所組成，而 LHA 模型是一種正規化的表示方法，它可以直接將系統的描述後經由使用 HyTech 的工具去執行以驗證系統的設計是否滿足系統時間的要求。

SES 模擬模型以物件導向為基礎是一種非常受歡迎的系統效能評估之模擬工具，因此我們的方法論中利用 SES 模擬模型做為系統軟體分割完後之系統效能評估，其中執行 SES 模擬模型的工具是 SES/workbench。

除此之外我們提出一個新的「多層次分割」（MLP）演算法，主要是提供

了在設計分散式崁入系統時能考慮到實際限制之軟硬體分割的方法。MLP 演算法包含有三個層次的迴圈，利用系統軟硬體的費用及效能為參數再加上考慮分散式系統的特性，經由演算法的找尋可以找到啟發式的最佳解（Heuristically Optimal）。MLP 演算法也考慮分散式元件的分享技術，因此大大地降低了系統的費用。

MLP 演算法主要分成三個迴圈，最外圈是設計空間的探索（Codesign Space Exploration，簡稱 CSE），主要目的是探索在已知的系統費用之下可以使用的處理器（Process Element，簡稱 PE）個數 中間迴圈為系統結構分割（System Structural Partitioning，簡稱 SSP），主要目的是如何把已有的處理器安排至不同分散式崁入系統中。最內圈是二元搜索相輔分割（Binary Search Copartitioning，簡稱 BSC），主要目的是利用二元搜索方式再配合我們提出之參數來做軟硬體之細部分割以及元件分享可達到降低系統成本之功能。

在系統軟硬體分割之後，緊接著是系統合成的工作，在 DESC 方法論中軟體合成是採用工作排程方式來完成，硬體合成是採用系統層級的物件導向技術來完成。在軟硬體合成之後，可依據此結果建立一快速原型（Rapid Prototyping）用於確認系統功能之正確性。使用 LHA 模型輸入至 Hytech 驗證工具做系統時間驗證以觀察設計結果是否符和系統所要求之時間限制。

四、結果與討論

我們提出一個完整相輔設計方法論稱之為分散式崁入系統相輔合成設計 DESC，它是擴充傳統集中式控制系統系統相輔設計方法和加強了現有的分散式系統相輔設計方法，其特點是使用物件導向設計之重複使用功能、使用多層次軟硬體分割方法以及考慮分散式系統實際限制如副系統間之距離等。本研究主要的貢獻有下列幾項：

1. 提出一完整之分散式崁入系統相輔設計的方法論簡稱為 DESC。
2. DESC 考慮了分散式崁入系統中實際的一些限制，而這些限制以往的研究並沒有提及。
3. 三種系統模型即 OMT、LHA 及 SES 模型已被有效的整合且應用在 DESC 方法論中。
4. 提出一個新的分散式崁入系統之軟硬體分割演算法，其特色是簡單且有效率以及為多層次式，本演算法考慮了分散式副系統間之物件的分享、硬體的叢集（Clustering）及軟體的重整（Grouping）。
5. 軟體合成方面，提出一狀態圖（State-Diagram）的排程方法；硬體合成方面，提出物件導向方法來做系統硬體的合成。
6. 為了保證設計的結果都能滿足系統需求的規格，我們使用三種不同之技術去做驗證，即效能的模擬、原型硬體功能的模擬及時間的驗證等。

五、計畫成果自評

本計畫進行相當順利，原計畫書中所預計完成之各工作項目均已達成，並有多篇論文發表 [10-13]。

六、 參考文獻

[1] R. Ernst, J. Henkel, and T. Benner, “Hardware-software co-synthesis for micro-controllers,” IEEE Design and Test of Computers, pp. 64-75, Dec. 1993.

[2] G. De Micheli and R. Gupta, “Haredware/Software Cosynthesis for Digital Systems,” IEEE Design and Test of Computers, pp. 29-41, Sept. 1993.

[3] R. Gupta, C. Coelho, and G. De Micheli, “Program implementation schemes for hardware-software systems,” IEEE Computer, pp. 48-55, Jan. 1994.

[4] A. Jantasch, P. Ellervee, J. Oberg, A. Henani, and H. Tenhunen, “Hardware/software partitioning and minimizing memory interface traffic,” in Proc. EURODAC, 1994, pp.226-231.

[5] P. Chou, E. Walkup, and G. Borriello, “Scheduling strategies in the co-synthesis of reactive real-time systems,” IEEE Micro, vol. 14, no. 4, pp. 37-47, Aug. 1994.

[6] D. Thomas, J, Adams, and H. Schmitt,”A model and methodology for hardware-software co-design,” IEEE Design and Test, vol. 10, no. 3, pp. 6-15, Sept. 1993.

[7] N. Binh, M. Imai, A. Shiomi, and N. Hickichi, “A HW/SW partioning algorithm for designing pipelined ASIP’s with least gate counts,” in Proc. DAC, 1996, pp. 527-532.

[8] J. Cong and Y. Ding, “Combinational logic synthesis for LUT based field programmable gate arrays,” TODAES, ACM trans. Design Automat. Electron. Syst., vol. 1, no. 2, pp. 145-204, Apr. 1996.

[9] R. Niemann and P. Marwedel, “Hardware/software partitioning using integer programming,” in Proc. EDTC, 1996, pp. 473-479.

[10] T.-Y. Lee, P.-A. Hsuing, and S.-J. Chen, “A Case Study in Hardware-Software Codesign of Distributed Systems–Vehicle Parking Management System,” in Proc. PDPTA, vol. 6, June, 1999, pp. 2982-2987.

[11] J.-M. Fu and S.-J. Chen, “Hardware-Software Coverification of Distributed Embedded Systems,” in Proc. PDPTA, vol. 6, June, 1999, pp. 2995-3001.

[12] T. Y. Lee, P. A. Hsiung, and S. J. Chen, “Hardware-Software Multi-Level Partitioning for Distributed Embedded Multiprocessor Systems,” *IEICE Trans. Fundamentals of Electronics, Communications, and Computer Sciences,* Vol. E84-A, No. 2, pp. 614-626, Feb. 2001.

[13] T. Y. Lee, P. A. Hsiung, and S. J. Chen, “DESC: A Hardware-Software Codesign Methodology for Distributed Embedded Systems,” *IEICE Trans. Information & Systems,* Vol. E84-D, No. 3, pp. 326-339, March 2001.