行政院國家科學委員會專題研究計畫 成果報告

多核心系統之跨層級整合設計環境及方法
研究成果報告(精簡版)

計 畫 類 別 ：個別型
計 畫 編 號 ：NSC 98-2218-E-009-022-
執 行 期 間 ：98 年 11 月 01 日至 99 年 10 月 31 日
執 行 單 位 ：國立交通大學電子工程學系及電子研究所

計 畫 主 持 人：賴伯承

計畫參與人員：碩士班研究生-兼任助理人員：江志軒
碩士班研究生-兼任助理人員：陳琬菁
碩士班研究生-兼任助理人員：高智恆
碩士班研究生-兼任助理人員：邱奏翰
碩士班研究生-兼任助理人員：李冠儒
碩士班研究生-兼任助理人員：陳柏諺
大專生-兼任助理人員：顏大剛
博士班研究生-兼任助理人員：郭玹凱

報 告 附 件 ：出席國際會議研究心得報告及發表論文

處 理 方 式 ：本計畫可公開查詢

中 華 民 國 100 年 01 月 29 日

Multi-Level Parallelism Analysis of Face Detection on a Shared Memory Multi-Core System

ABSTRACT

Face detection is one of the fundamental technologies for the future smart objects. However, its computation intensive property thwarts the practice of a real-time application on an embedded device. Parallel processing and many-core architecture have become a mainstream to achieve high performance in the future computing systems. The parallelism of an application needs to be exposed before being exploited by the parallel architecture. This paper performs a comprehensive analysis of the parallelism of a face detection algorithm at different algorithmic levels. This paper has demonstrated that each parallelism level has its own potential to enhance performance, but also imposes different limiting factors to the overall performance. Based on the analysis results and design experience, this paper proposes a multi-staged mixed-level parallelization scheme to retain the performance scalability and avoid the limiting factors. With this scheme, we are able to achieve up to 37.5x performance enhancement on a 64-core system.

1. INTRODUCTION

Face detection is one of the fundamental technologies for the future smart objects. It enables a device to recognize the faces of the main users as well as the people in the surrounding environment. By recognizing faces, the device can thereafter perform many intelligent reactions such as user identification or even the customer background search[16]. However, the high computation requirement prevents the real-time face detection on a mobile or embedded device. It is estimated to take about 3 seconds to process a single image of 512*512 pixels on a 500MHz single-issue ARM v5 architecture [1]. The execution time is 75 times slower than the requirement of a real-time application (processing 25 images in one second). A significant performance improvement is required to achieve the real-time face detection on an embedded device.

Parallel processing and many-core architecture have become a mainstream to achieve high performance in the future computing systems. Embedded processor vendors, such as Tilera[2], ARM[3], MIPS[4], are also moving towards many-core architectures. Even the desktop processor vendors, such as Intel[5] and AMD[6] are proposing many-core products for embedded and mobile applications. The new parallel embedded processors present opportunities to boost the raw computing capability and achieve more energy efficient execution. However, three imperative design aspects have to be concerned before the full advantages of many-core processors can be transformed into superior system performance. First, the algorithmic parallelism of applications needs to be explored and exposed. Second, the characteristics of the highly integrated embedded system need to be analyzed. Third, the possible system bottlenecks need to be identified.

Motivated by the three design aspects, this paper performs a comprehensive analysis on the potential parallelism of the widely used Viola-Jones face detection algorithm [7]. The analysis explores the parallelism in different algorithmic levels. By verifying the results on a multi-threaded cycle-accurate multi-core simulator, this paper demonstrates the significant computation parallelism inherited in the face detection algorithm. However, the superior performance can only be obtained through a careful co-design and optimization crossing four critical design issues, including choosing appropriate parallelism level, balancing workload, reducing synchronization overhead, and memory and interconnect bandwidth. By following the design guidelines concluded from the comprehensive analysis, this paper proposes a multi-staged mixed-level parallelization scheme which achieves 37.5x performance enhancement in a multi-core system with 64 ARM processors.

The rest of the paper is organized as the following. Section 2 introduces related work. The Viola-Jones face detection algorithm is discussed in section 3. Section 4 introduces a reconfigurable multi-threaded cycle-accurate simulator for shared-memory SMP (Symmetric Multi-Processing) systems. Section 5 analyzes the characteristics of the face detection algorithm in different algorithmic levels. Section 6 shows the experimental results on different parallelism levels. Section 7 proposes a multi-staged mixed-level parallelization scheme with better performance scalability. Section 8 draws the conclusion.

2. RELATED WORK

Face detection is extensively studied used in many smart object applications [16]. The Viola and Jones algorithm is one of the most widely used face detection schemes [7]. It provides high accuracy and fast computation. Since the algorithm is so popular, many research efforts have been spent on enhancing the performance of the Viola-Jones algorithm. Wei et al.[8] and Yang et al.[9] realized parallelism of the algorithm by using a specific HW design in a FPGA. Theocharides[10] also proposed a scalable parallel architecture for face detection on FPGA. C. Gao[11] presented a novel approach to use FPGA to accelerate the Haar-classifier based face detection algorithm with highly pipelined micro-architecture and utilizes abundant parallel arithmetic units in an embedded system. Most of the methods focus on using innovative HW architecture or specific HW accelerator to enhance the performance. Our approach concentrates on achieving better performance through exploiting the algorithm parallelism on multi-core systems. It is different from building a specific hardware accelerator to speed up the critical computation in the algorithm. The proposed design can be easily applied to a SMP system without any extra HW implementations.

Chen‘s research [12] is among the first to explore the algorithmic parallelism of face detection algorithm, and is similar to the work done in this paper. The author analyzed the potential parallelism of Ada-boost algorithm and executed on multi-core systems with 4 to 8 processors. A 5.5X performance enhancement was demonstrated by adopting a hybrid scheme of both coarse-grain and fine-grain TLP. Our work differs from [12] in two aspects: (1) this paper not only explores the algorithm parallelism in different levels, but also shows the impact of different design issues; (2) the analysis is extended to a larger scale (64 cores) of multi-processor which demonstrates a significant computation parallelism in the face detection algorithm.

3. THE VIOLA-JONES FACE DETECTOR

The Viola-Jones face detection is widely used in many applications due to its high accuracy and fast runtime. Figure 1(a) illustrates the procedure of the face detection. The image is first loaded and scanned by different sizes of scan windows. The face features within each scan window is calculated based on the

Haar-like scheme [7], which will be performed by the *integral image* block in Figure 1(a). The face features are evaluated by summing the pixel values in the rectangular sub-region (shown in Figure 1(b)). The result will then be sent to the *detection block* which uses cascaded classifiers to scan and find the location of faces.

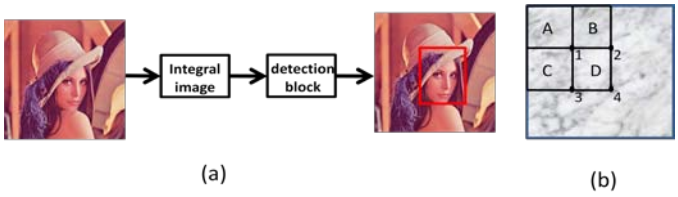

Figure 1 (a) Viola-Jones face detection procedure
(b) Integral image computation.

(1) **Integral image:** The integral image method was firstly introduced to the digital image processing by Crow [13]. In Viola-Jones face detection, it is used for rapid computation of Haar-like features [7], which are defined as the (weighted) intensity difference between two to four rectangles. The integral image is constructed as follow.

$$\mathrm{I}(x,y) = \sum_{x' \le x, y \le y'} i(x',y')$$

I(x,y) is the integral value of all the image pixels in the rectangular region from (0,0) to (x,y). And *i(x',y')* is the value of the pixel at location *(x', y')*. It is fast and efficient to use the integral image to compute the value of the face features in a rectangular sub-region area. As shown in Figure 1(b), for example, the value of point 4 is the sum of the sub-region area *A+B+C+D*. The value of point 3 is the sum of *A+C*. If we want to compute the value of the region D, it can be easily obtained by the value of (I(1) + I(4)) – (I(2) + I(3)), which only requires four value references.

(2) **Cascaded classifier structure with Ada-Boost:** Boost is a method of finding high accuracy by combing "weak" classifiers with moderate accuracy [14]. Ada-boost algorithm is a kind of appearance-based method which have shown superior performance compared to others [9]. There are two types of classifiers, "weak" classifiers and "strong" classifiers. Weak classifiers have arithmetic value thresholds in recording human face features. A number of weak classifiers with similar features are combined together as a "strong" classifier.

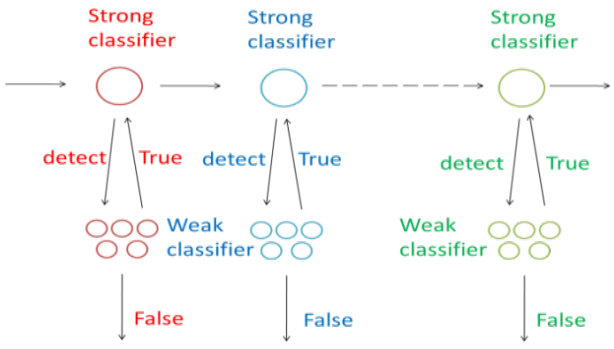

Figure 2 Cascaded structure of strong classifiers

The cascade classifier structure, shown in Figure 2, is a critical component in the Viola-Jones detector. This structure rapidly and efficiently rejects most negative sub-window images while keeping almost all the positive ones. The input sub-window images pass through a series of nodes during detection. Each node represents a strong classifier, which would make a decision on whether the sub-window image should be kept for the next node or rejected. A strong classifier at a later stage contains more weak classifiers to provide better image checks and face detection.

4. Parallelism In Different Levels

The parallelism of the face detection exists in different algorithmic levels. This section discusses the potential parallelism at different levels of the face detection implementation. The face detection implementation is adopted and modified from OpenCV library[17], which applies the idea of Viola-Jones face detection algorithm. As shown Figure 3, the implementation can be divided into three parts. **(1) Resize.** The implementation uses the fixed-size scan window with a well-trained classifier library in the Ada-boost algorithm. Since the scan window size is fixed, an image needs to be resized into different resolutions. **(2) Integral.** This part performs the evaluation of the Haar-like features by using the integral image method. **(3) Detect.** By moving the scan window through the image, the sub-image is sent into the cascade classifier structure to detect the location of a face.

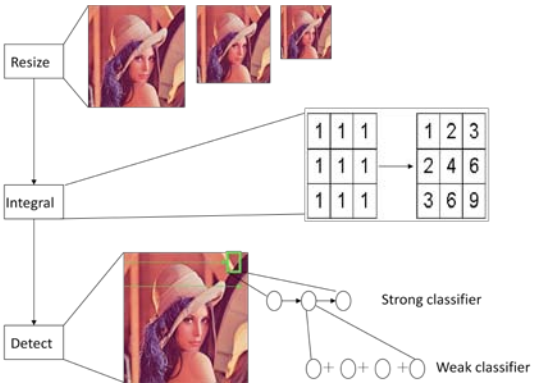

Figure 3 The procedure of face detection implementation

The parallelism of the algorithm can be extracted from different levels. This paper divides the parallelism into four levels (shown in Figure 4). **Top** level parallelism is demonstrated when different sizes of images are processed by different threads concurrently. **Detection** level parallelism can be exposed by performing the "detect" block on different sizes of images concurrently. Other parts (resize and integral) in the algorithm remain in the sequential scheme. **Divided detection** level is similar to the detection level, but the image is further divided into sub-images. The detection tasks of these sub-images are executed simultaneously by different threads. **Weak classifier** level exposes the parallelism by executing different weak classifiers in the cascaded structure concurrently.

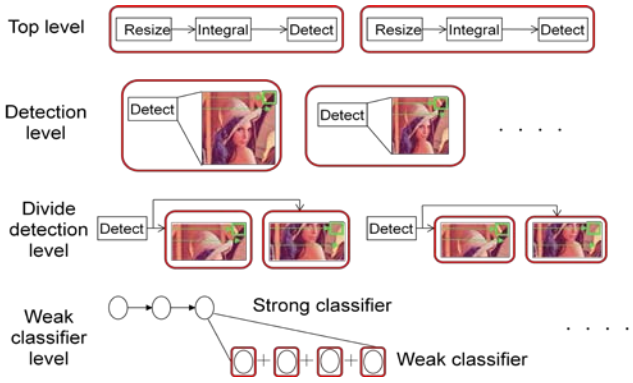

Figure 4 Different algorithmic levels of the face detection implementation

The key design trade-off here is the parallelism granularity and the synchronization overhead. The lower parallelism level gives finer granularity of parallelism which enables better load balance between different concurrent threads. However, having more threads in a system increases the amount of required synchronization among threads and demands higher memory and interconnect bandwidth.

5. Simulation Platform: A Shared-Memory Cycle-Accurate SMP Simulator

This paper uses a multi-threaded cycle-accurate shared-memory SMP simulator [15], which enables HW/SW co-simulation of a SMP multiprocessor system. The simulator contains both the multi-threading library and the SMP hardware model. The reconfigurability of the simulator allows designers to explore not only the multi-threading programs, but also the multi-core architectures. Fig.5 illustrates the organization of the simulator. The top half shows the thread management queue, which is implemented as a FIFO scheme. Processors can create and add new threads to the tail of the thread queue. Idle processors can request tasks from the head the thread queue.

The lower half illustrates the hardware model of a SMP system. The processing core models a single-issue ARM v5 architecture. Each processor has its own data cache and instruction cache where the cache organization is configurable. Processors are connected by a shared-bus. The bus is organized as a multi-master bus and contains a test-and-set lock to support atomic read-modify-write operations. The latencies of bus transaction and memory access are also configurable. The cache coherence is implemented as a simple snooping-based protocol.

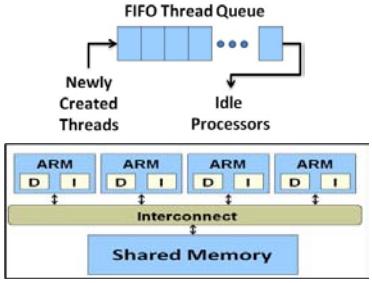

Figure5 A SW/HW organization of a multi-threaded shared-memory SMP architecture

6. Execution Behavior of Different Parallelism Levels

The reference image of the experiment is the picture of Lena, which is widely used as a reference image for digital image processing. The size of the image is 512x512 pixels and is a reasonable size for modern mobile digital devices. The experiment platform is a multi-processor system with various numbers of ARM v5 processors. Each processor is assumed to operate at 500 MHz. Table 1 shows the computation time breakdown of the sequential face detection algorithm. The resize and classifier detection occupy the most significant parts of the execution time. Therefore the parallelization and performance enhancement should be focused on these parts.

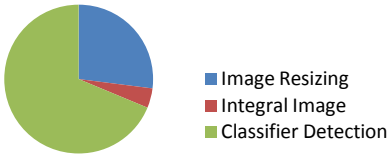

Figure 6 Parts consuming ratio of total execution time when running the algorithm sequentially. (latency is set to 1 cycle)

6.1. Effect of Memory/Bus Latency

Figure 7 compares the execution time of different numbers of processors with various memory and bus latencies. When the latency is more than 20 cycles, the performance cannot keep scaling when there are more than eight processors. More processors also cause the traffic jam on interconnect and memory, which is reflected by the bus activity rate. Memory and bus definitely play an imperative role in the overall performance. However, the focus of this paper is on the parallelism of the face detection application. We would be more interested in the intrinsic computation parallelism which can be exposed through parallelization. To minimize the impact of communication bottleneck, the memory/bus latency is set to 1 cycle. The rest of the paper will use this system scheme to explore the parallelism of the application.

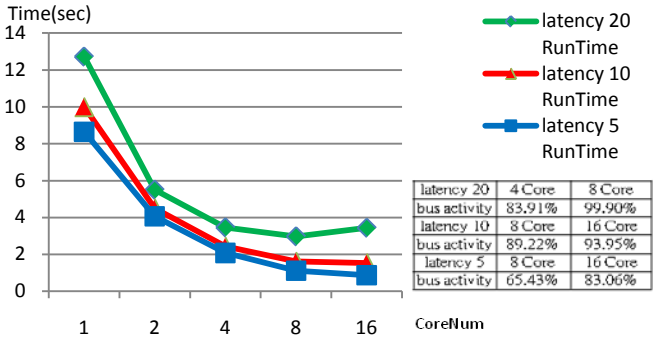

Figure 7 Execution time for different number of processors with various memory/bus latencies. The latency is in the unit of cycles.

6.2. Performance at Different Parallelism Levels

The experiment is conducted in four parallelism levels, including top level, detection level, divided detection level, and weak classifier level. The lower parallelism level gives finer parallelism granularity to achieve better load balance. However, having more threads also increases the synchronization overhead which requires higher memory and interconnect bandwidth to support the performance scalability.

(1) **Top level parallelism.** In this level, the sequential part only occupies a small portion of execution time, which includes loading the image data, classifier library, and creating threads. The most execution time of multiprocessor system is spent on the parallel execution. As shown in Figure8, the execution time is improved as the number of cores increases. However, the performance enhancement slows down when there are more than 8 cores. This is mainly due to the imbalance workloads among different threads. The thread with the longest execution time becomes the performance bottleneck in the 16-processor system. The execution time of the 16-processor system is decided by the execution time of the heaviest thread on a single processor while all the other processors are finished and idled. Better parallelism granularity can help improve the performance.

(2) **Detection level parallelism**. This level only parallelizes the detection block. The other parts, including the thread creation, are still executed sequentially. This strategy makes the speed of creating new threads too slow to sustain the demand of parallel execution. Processors are idled waiting for new threads to be created. Besides, the sequential part of the algorithm becomes the critical part of the program. The performance would not improve when the number processors increases. As shown in Figure8, the execution time does not improve when the system has more than 8 processors.

(3) **Divided detection level parallelism**. This level returns a more satisfied result than the previous strategy. More parallelism of detection level can be extracted. Number of threads created in one time is sufficient and the size of different threads is more balanced. However, the sequential part is still plays the critical role in the program. Therefore the run time is similar to the detection level.

(4) **Weak classifier level parallelism.** This is the lowest level in the algorithm. The number of threads is equal to the number of weak classifiers. Although it exposes the maximum potential parallelism, the overhead becomes the limiting factor of the performance enhancement. Since there could be thousands of weak classifiers, the time required to create these threads is already in average 8.17x of the total execution time of the sequential algorithm on a single processor. Another overhead is the sheer amount of synchronization transactions among the huge number of threads.

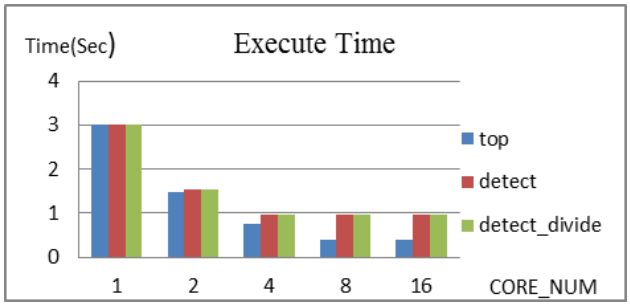

Figure 8 Result of each parallelism level

According to the experimental result, parallelizing the algorithm in higher levels poses the disadvantage of imbalance workload. When parallelizing the algorithm in lower levels, the overhead increases on the aspects of creating new threads and synchronization. The ratio of the sequential part to the algorithm also deteriorates the performance. To achieve the best performance, we should have a scheme to expose the significant parallelism in an appropriate level and avoid the overhead of having too many threads.

7. A Hybrid Parallel Scheme to Achieve High Performance

Based on the experience from the previous section, we propose a multi-stages mixed-level parallelism scheme to achieve the maximum performance enhancement.

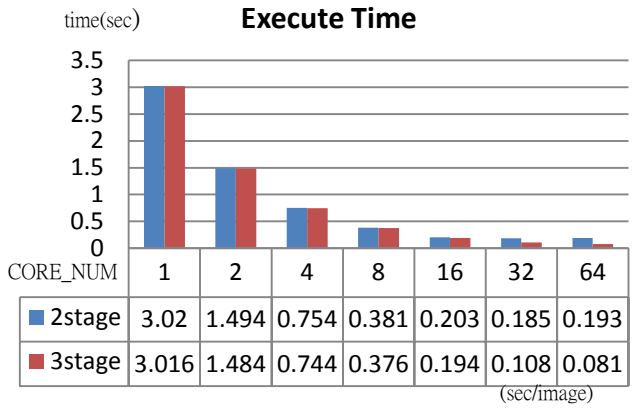

Figure 9 Result of our modified programs

The first implementation is a 2-stage scheme. Stages are executed sequentially. The second stage will start when the first stage is finished. The first stage will initiate multiple threads to perform the resize and integral blocks concurrently. The second stage implements the same scheme as in the "divided detection" level. Since the block "detection" takes most of the execution time in the algorithm and its parallelism can be well scaled in divided detection level. As shown in Figure9, when compared with a single processor, the 2-stage strategy can reach around 16x speed-up on a 16-core system. However, the 2-stage scheme cannot achieve better performance when the processor number is more than 16, due to the imbalanced work load in the first stage.

The 3-stage scheme is implemented to further improve the performance. The first stage now only contains a multi-threaded version of the resize block. The second stage will process the integral part and the third stage will perform the parallel execution of the detection block. This scheme has better balanced tasks, and achieve the superior performance scalability. As shown in Figure9, the 3-stage parallel strategy reaches a 27.8x and 37.5x speed-up on a 32-core and 64-core system respectively.

8. Conclusion

This paper performs a comprehensive analysis of the parallelism of a face detection algorithm at different levels. We have demonstrated that, each parallelism level has its own potential to enhance performance, but also imposes different limiting factors to the overall performance. The imbalanced execution loads among threads adversely impact the performance as well. Based on the analysis results and design experience, this paper proposes a multi-staged hybrid scheme to retain the parallel performance and avoid the limiting factor. With this scheme, we are able to achieve up to 37.5x performance enhancement on a 64-core system.

9. References

[1] FA626 ULTRA HIGH SPEED 32-BIT RISCCPU http://www.faraday-tech.com/techDocument/FA626_ProdBrief_v1.2.pdf

[2] The TILE-Gx™ processor family processor, http://www.tilera.com/

[3] ARM cortex-A9 processor, http://www.arm.com/

[4] MIPS Technologies Announces Symmetric Multiprocessing (SMP) Support for Android™ Platform on MIPS-Based™ SoCs, http://www.mips.com/

[5] Intel multicore technology, http://www.intel.com/

[6] AMD multi-core processing, http://www.amd.com/

[7] C. Zhang and Z. Y. Zhang, "A Survey of Recent Advances in Face Detection ", *Microsoft Research*, June 2010.

[8] Y. Wei, X. Bing, C. Chareonsak, "FPGA Implementation of AdaBoost Algorithm for Detection of Face Biometrics", *In Proc. IEEE International Workshop Biomedical Circuits and Systems*, 2004.

[9] M. Yang, Y. Wu, J. Crenshaw, B. Augustine, R. Mareachen, "Face Detection for Automatic Exposure Control in Handheld Camera", *in Proc. IEEE International Conference Computer Vision Systems*, 2006.

[10] T. Theocharides, N. Vijaykrishnam and M. J. Irwin, "A parallel architecture for hardware face detection", *Symp on Emerging VLSI Technologies and Architectures*, pp. 452-453, 2006.

[11] C. J. Gao and S. L. Lu, "Novel FPGA based Haar classifier face detection algorithm acceleration", *FPL 2008*, Heidelberg, September 2008, pp. 373-378.

[12] Y. K. Chen, W. L. Li and X.F. Tong, "Parallelization of AdaBoost algorithm on multi-core processors", *IEEE SiPS 2008*, Washington DC, 2008, pp.275-280.

[13] F. C. Crow, "Summed-Area Tables for Texture Mapping", *Computer Graphic*, vol. 18, no. 3, pp. 207-212, July 1984.

[14] Y. Freund and R. E. Schapire, "A short introduction to boosting", *Journal of Japanese Society for Artificial Intelligence*, pp. 771-780, vol. 14, no. 5, September 1999.

[15] P. Schaumont, B. C. Lai, W. Qin, I. Verbauwhede, "Cooperative Multithreading on Embedded Multiprocessor Architectures Enables Energy-Scalable Design," *Proceeding 2005 Design Automation Conference (DAC)*, pp. 27-30, June 2005.

[16] SixthSense Project, MIT Media Lab, http://www.pranavmistry.com/projects/sixthsense/

[17] Open Source Computer Vision, http://opencv.willowgarage.com/

國科會補助專題研究計畫項下出席國際學術會議心得報告

日期:100 年 1 月 30 日

計畫編號	NSC 98－ 2218 －E －009 － 022 －		
計畫名稱	多核心系統之跨層級整合設計環境及方法		
出國人員 姓名	賴伯承	服務機構 及職稱	交通大學電子工程系助理教授
會議時間	99 年 6 月 13 日至 99 年 6 月 18 日	會議地點	Anaheim, CA, U.S.A
會議名稱	(中文) 2010 計算機器學會設計自動化國際會議 (英文) ACM Design Automation Conference 2010		
發表論文 題目	(中文)無 (英文)		

一、參加會議經過

此次會議於加州洛杉磯附近的安納罕城市舉辦。總共六天的會議中包含了一百多篇的研究論文報告以及討論。在一樓的展示大廳中也有超過百家的設計自動化大廠參展。整個會議內容與議程的安排非常的豐富。

在論文報告與研討的部分，所有的研究論文被分成超過五十個議題，分別於四天的議程中進行報告與研討。在本年度的設計自動化會議中，先進製程的變異性與電路的可靠性成為許多議題中的討論重點。如何在半導體製程持續微縮化的過程中維持系統的功能性與提升可靠度，仍舊是一個非常難的問題。參與報告的除了世界各地一流的頂尖大學之外，也包含了很多業界的研究成果。這讓議題的討論不會只侷限在研究題目的探討中，而能夠跟業界目前最需要解決的困難相結合，以期待能夠發展出更符合目前需求的相關設計自動化技術。

在一樓的展覽會場中，有超過數百家的廠商參展。參展的廠商除了設計自動化工具的一線大廠(如 Cadence, Synopsys, Mentor 等)之外，也包含了許多具有先進技術的新成立廠商。會場的佈置和各個廠商攤位活動的安排也十分的活潑，在介紹公司的新技術之前都會利用許多不同的表演來吸引人潮。值得一提的是，今年的參展廠商中，台積電的攤位占了非常大區域。台積電今年特別強打所提出的整合性設計環境 (Open Innovation Platform)，希望讓晶圓代工廠不能再單純提供製造服務。台積電將整合本身及第三者的設計工具 EDA 電子設計自動化 (Electronic Design Automation) 及矽智財 (IP)、製程技術及流程服務等，推

出「開放創新平台（Open Innovation Platform)」。

另外一個特殊的攤位是 University Booth，提供了一個攤位讓各個學校可以來展示不同的研究計畫所完成的設計成果。

二、與會心得

此次參加完後，DAC 仍然毫無疑問的是國際間 IC 設計及自動化工具領域中最大且最具代表性的會議。除了論文的審核嚴謹度與研究內容的新穎度之外，這是一個非常好的機會可以與各國的頂尖專家學者齊聚一堂，一起討論未來的發展方向。這個對本身研究的發展，與對國際間設計自動化趨勢的掌握度都有極大的幫助。

除了國內的相關領域研究學者之外，若是能讓學生來參加這種國際一流的會議，會非常大程度的激發學生研究的熱情，並開拓對各領域的了解度，是一個對學生很好的訓練。

三、考察參觀活動(無是項活動者略)

無

四、建議

國內對於學界老師或是研究生出國參加國際研討會的補助通常有滿大的限制，除了在金額的上限外，次數上也有限制。這個雖然有經費預算上的考量，但是缺乏國際的視野和與國際一流研究群接觸的機會，會降低國內研究人員的國際觀與對研究與發展趨勢的掌握度，對整個國家的發展也是一種阻礙。建議在預算的編列上，可以對參與國際一流會議，或是爭取國際會議在台灣舉辦上能夠多加鼓勵，以期能讓台灣學術研究能夠與國際同步，甚至達到領導的地位。

五、攜回資料名稱及內容

1. 大會議程表一份

2. DAC2010 論文集 DVD 光碟一片

六、其他

無

國科會補助計畫衍生研發成果推廣資料表

日期:2011/01/26

國科會補助計畫	計畫名稱: 多核心系統之跨層級整合設計環境及方法
	計畫主持人: 賴伯承
	計畫編號: 98-2218-E-009-022- 學門領域: 積體電路及系統設計

無研發成果推廣資料

98 年度專題研究計畫研究成果彙整表

計畫主持人：賴伯承	計畫編號：98-2218-E-009-022-
計畫名稱：多核心系統之跨層級整合設計環境及方法	

成果項目			量化			單位	備註（質化說明：如數個計畫共同成果、成果列為該期刊之封面故事...等）
			實際已達成數（被接受或已發表）	預期總達成數(含實際已達成數)	本計畫實際貢獻百分比		
國內	論文著作	期刊論文	0	0	100%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	0	0	100%		
		專書	0	0	100%		
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（本國籍）	碩士生	0	0	100%	人次	
		博士生	0	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		
國外	論文著作	期刊論文	0	0	100%	篇	
		研究報告/技術報告	0	0	100%		
		研討會論文	2	2	100%		本計畫研究成果於國際研討會中發表兩篇研究論文.
		專書	0	0	100%	章/本	
	專利	申請中件數	0	0	100%	件	
		已獲得件數	0	0	100%		
	技術移轉	件數	0	0	100%	件	
		權利金	0	0	100%	千元	
	參與計畫人力（外國籍）	碩士生	0	0	100%	人次	
		博士生	0	0	100%		
		博士後研究員	0	0	100%		
		專任助理	0	0	100%		

其他成果 (無法以量化表達之成果如辦理學術活動、獲得獎項、重要國際合作、研究成果國際影響力及其他協助產業技術發展之具體效益事項等，請以文字敘述填列。)	無

	成果項目	量化	名稱或內容性質簡述
科教處計畫加填項目	測驗工具(含質性與量性)	0	
	課程/模組	0	
	電腦及網路系統或工具	0	
	教材	0	
	舉辦之活動/競賽	0	
	研討會/工作坊	0	
	電子報、網站	0	
	計畫成果推廣之參與（閱聽）人數	0	

國科會補助專題研究計畫成果報告自評表

請就研究內容與原計畫相符程度、達成預期目標情況、研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）、是否適合在學術期刊發表或申請專利、主要發現或其他有關價值等，作一綜合評估。

1. 請就研究內容與原計畫相符程度、達成預期目標情況作一綜合評估

■達成目標

□未達成目標（請說明，以 100 字為限）

□實驗失敗

□因故實驗中斷

□其他原因

說明：

2. 研究成果在學術期刊發表或申請專利等情形：

論文：■已發表 □未發表之文稿 □撰寫中 □無

專利：□已獲得 □申請中 ■無

技轉：□已技轉 □洽談中 ■無

其他：(以 100 字為限)

本計畫研究成果已在國際學術會議中發表兩篇研究論文.

3. 請依學術成就、技術創新、社會影響等方面，評估研究成果之學術或應用價值（簡要敘述成果所代表之意義、價值、影響或進一步發展之可能性）(以 500 字為限)

本計畫於 99 年 10 月完成，為一年期國科會新進人員研究計畫，題目為’多核心系統之跨層級整合設計環境及方法(NSC 98-2218-E-009 -022 -)’。此國科會計畫建構完成一套以 ARM 處理器為主的多執行緒多核心系統模擬環境。此多核心模擬器除了包含硬體架構的模型之外，還建構了一套多執行緒的函式庫，可以同時模擬軟體與硬體的執行狀態以及相互間的行為關係。這個平台讓使用者除了可以進行硬體架構的開發與研究外，也可以藉由調整多執行緒的軟體架構來增進整體多核心多執行緒系統的效能。