行政院國家科學委員會專題研究計畫成果報告

容錯飛控計算機系統效能改進與偵錯診斷系統之建構

Performance Improvement of Fault-Tolerant Flight Control System and Development of Diagnostic System

計畫編號：NSC 88-2623-D-002-012
執行期限：民國87**年**07**月**01**日到**88**年**06**月**30**日**
主持人：郭斯彥　　台灣大學電機工程學系教授

一、中文摘要 (關鍵詞：容錯計算機、三組態備份容錯、微處理器、表決器)

隨著計算機系統的發展，計算機系統已廣泛的應用於各行各業。包括一些具有高危險性或重要的工作，例如核電廠控制系統，手術室的醫療監視系統，以及飛行控制系統。但是計算機本身是由硬體電子零件以及軟體邏輯程序的組合，任何一個部份做出預料之外的反應，都有可能導致整個計算機系統失常或者停頓。對於上述之高危險性與重要的工作，計算機系統的故障會造成鉅大金錢與生命的損失。尤其是用於軍用或是民間飛行控制的電腦，由於在飛行過程中飛行控制系統必須持續監控飛機的狀況，並且即時做出正確反應，如果飛行控制系統的某一部份發生故障，將會造成極大災難。為了確保飛行安全，飛行控制系統必須具備極高的可靠度，一般來說，在系統運作過程中每小時發生錯誤的機率必須小於 10^{-9} **以下。**

近三十年來，關於『如何使計算機系統更可靠』這方面的研究，有了重大的發展。在早期的系統中，增進系統可靠度的方式即是經由不斷的測試、修改而建構出一個接近零缺點的系統。然而，在系統建構的過程中，需要付出相當龐大的人力、資源以及冗長的研發時間。另一方面，工程人員可以藉由各種備份技術，讓計算機系統本身具備容錯之能力。依據設計理念的不同，容錯計算機系統不只能容忍系統部件意外故障所造成的錯誤，更能延伸到容忍系統設計上的缺失，與人員操作不當造成的錯誤。在近二十年中，由於微處理器的發展，計算機系統的體積、重量、耗電量與成本皆大幅縮減，使得硬體備份已確實可行。利用這些技術，容錯計算機系統將可以兼顧功能性與可靠度，同時將成本壓低至實用的階段。本計畫將利用微處理器技術，研究並實際完成一個能應用在飛行控制系統中之容錯計算機雛型(Prototype)**。**

英文摘要 (Fault Tolerant, TMR, Micro-processor, FPGA, Voter)

Computer systems are widely used in variety of tasks. For critical missions such as the medical system in the operation room, the aviation control system, or the nuclear reactor control system, malfunction of the computer system may cause great casualty.

In the past three decades, one can see the significant growth in this issue: how to make the computer system more *dependable*?

One way to achieve higher dependability is to make the computer system fault-free; that is, to build a near perfect physical system, and execute a near perfect procedure. However, it is a dependability-and-resources trade-off. As the computer

systems are getting more and more complex, it is more and more difficult to design, build, and verify the system that can per-form perfect operation at any circumstance throughout the system's life time; and the cost to develop and build a near perfect system is increasing exponentially as the system scale and complexity increase.

Another way to achieve higher depend-ability is to make the system *fault-tolerant*. A fault-tolerant system is a system that will keep operating normally and give the correct responses and answers when one or more of the components fail to perform it's normal task. A more general concept of fault-tolerance also includes *man made fault*, such as the human mistakes committed during hardware or software design and implementation, or *bugs* in short term. Another possible human mistake is the operating error due to the improper design of human computer interface, or the lack of training for operators.

To achieve fault-tolerance, redundant hardware is required. It is a big problem in the early era of computer development while the computers are in the size of a refrigerator, and they are costly. With the advent of microprocessors, the volume, weight, power and cost associated with redundant component decreases dramatically. With microprocessors, fault-tolerant systems can be built highly dependable and affordable.

二、計畫緣由與目的

本計畫的目的是在設計並且實作一用於飛行控制系統之容錯計算機系統雛型。由於此系統需具備高可靠度與即時控制的特性，因此採用以 TMR (Triple Module Redundancy) **為基礎之混合式容錯架構來達成系統需求。此系統由三套相同的微處理器模組以及一組備用模組所構成。所有模組皆在同一時間執行相同的運算，各模組的運算結果送至多數決表決器，經由一多工器，由四組資料中選取三組資料進行比對與多數決表決，並由其中選出一組符合多數的資料作為輸出。當有某一模組送出的運算結果和最後結果不同，該模組即會被標示為故障，而在下次的資料選擇中由備用模組取代。藉由此種阻隔錯誤輸出以及替換故障模組的方式，提供系統容錯的能力。**

應用於飛行操作系統上之容錯計算機系統在軟體上需要能整合硬體上之容錯機構，達成即時控制需求，以及整合任務應用軟體。 本計畫中之軟體研究及設計即以此為基本方向。計畫中將研究軟硬體機構之配合，並建立一可行的容錯軟體雛型為目標。此一飛控容錯計算機軟體與容錯機構的配合方面，主要包括：

(一)容錯管理系統：此管理系統對飛控計算機所產生的不同錯誤，採取相對應的容錯策略。並將發生錯誤的模組之錯誤型態回傳給系統操作者，提供操作者的處理措施之依據。這包括錯誤偵測、錯誤處理，與錯誤顯示。

(二)具容錯功能之作業系統核心：對於應用軟體一依據其所需的反應時間進行工作排程，並提供狀態儲存(Context Saving)**及回覆**(Recovery)**系統功能，配合可行的軟體容錯要求，由應用軟體在適當的地點呼叫，發生錯誤時可退回上一狀態儲存點。**

三、研究方法與成果

透過系統的需求分析，明確定義出本系統的設計目的與規格，並遵循一系統化之設計流程，逐步進行系統的設計與實作。由相關議題之研讀開始，隨著設計流程，系統架構設計、硬體架構設計與細部電路設計逐步完成。其後根據設計實作微處理器模組與表決器。硬體

系統完成後，將進一步與軟體結合，最後進行系統驗證工作。

容錯飛控系統硬體實作的部份，我們製作了四個自行設計之 Intel 486 微處理器計算機系統印刷電路板，包括中央處理器、周邊裝置控制器、記憶體子系統、計時器、以及自訂規格之平行傳輸介面埠。另須四顆 Altera EFM7160LC84 可程式化邏輯元件，用以製作周邊裝置控制器。每個微處理器模組本身即是一完整之微處理器計算機系統；系統中包含了一中央處理器，周邊裝置控制器，記憶體子系統，計時器，以及平行輸出入埠。中央處理器選擇 Intel 80486 及其相容微處理器系列，在線路設計上採用 AMD Am5x86 相容腳位。多數決表決器則是使用 FPGA 可程式化邏輯元件製作。

我們使用的硬體架構採用 32 位元之資料通道，表決器操作頻率和微處理器相同，亦即 25MHz。根據 Altera 所提供之規格，EFM9000 『接腳到接腳』的訊號延遲時間（Pin-to-Pin Delay）為 16ns，所以 25MHz 的速度對於 EFM9000 來說，並不會太慢。表決器讀入資料的動作需要和微處理器模組同步，因此當微處理器結束資料傳輸動作，即代表資料比對開始。根據我們的 AHDL 程式與模擬結果，資料的比對約需花費一個時脈時間，資料輸出動作和故障遮蓋動作採取平行方式處理，因此無論是否有錯誤發生，資料僅需兩個時脈時間即可輸出。將比對時間加入考慮，系統輸出的極速約為每 25ms 輸出 15,625 雙字組，一次輸出大約只佔反應時間的 6.4%。經過模擬測試，我們發現：如果只考慮單純的硬體動作，硬體輸出效率的確可以達成我們預先設定的須求。若再加上偵錯診斷系統負載部份的考量，我們設計在每次輸出之後等待一個回應訊號，CPU 依據這個訊號決定下一步應該做什麼。經過計算可以發現為了達到偵錯診斷的功能我們的系統每次輸出時間增加了 70%。也就是說在 25ms 中能夠輸出 10,246 雙字組。要達到 25ms 輸出 1,000 雙字組的目的，耗費在單純的輸出入上的 CPU 時間就佔了將近 10%。這個數字大致合乎我們之前設定的要求。

容錯操作軟體的部份，飛行控制系統容錯軟體要求快速的反應，通常時間的精確度在 5ms 之內而且此反應時間不但要快速而且要穩定、可預測，即其本身為一即時系統，飛行控制系統容錯軟體需求之一即是一即時系統核心。茲假設飛行控制系統在 5ms 之時段內有 15 % 之餘裕可供容錯處理比較 100 words (1word=16bits), 故處理每一個 word 所容許之時間為 5ms*0.15/100=7.5us, 設硬體處理時間及軟體處理時間各佔一半則處理一個 word 所容許之時間為 3.75us。飛行控制系統軟體除了要求快速的反應之外，針對外界的輸入輸出正確的控制訊號是根本的需求。然而飛行控制系統軟體可能面臨許多潛在的錯誤，依據錯誤的部分、時間、與影響範圍不同，飛行控制系統軟體必須能夠加以偵測，並作出不同處理。處理方式包括局部回復處理（Local Recovery）、整體回復處理（Global Recovery）、預備模組切換（Switch to Spare Module）。

系統結構共分為四層：第零層為硬體層，第一層為 BIOS 層，第二層為作業系統層，第三層為應用程式層。此四層構成即時系統物件，其中第三層僅有擔任核心排班工作的部分包含於系統之內，其餘的是屬於外部的作用者(actor)。

層與層之物件為一關係物件，在第零層與第一層之間的物件為插斷服務程式，其工作在提供透過BIOS的插斷向量表，執行硬體的各項插斷服務。因為此部分程式與硬體直接相關，所以通常用低階組合語言撰寫。第一層與第二層之間的物件為作業系統介面，主要工作在提供作業核心各項任務的執行函式，包

括訊息控制，堆疊控制，內文轉換等各項存取等級只有作業系統核心才能夠執行的動作。第二層與第三層之間的物件為應用程式介面，主要提供使用者(應用程式設計者)簡易的設計函式，使其能透過此介面函式，與作業系統相互溝通，此外，本章之後將會述及的排班演算法的實作函，也是屬於應用程式介面。

錯誤偵測系統之設計與建造部份，由於此系統是用於飛行控制系統，因此對於容錯的控制是相當重要的。我們使用了 Recovery Block 的方法來提升軟體的可靠度。我們使用 Power On Self Test (POST) 來進行系統自我檢測。POST 部分的順序為：微處理器測試，記憶體子系統測試，計時系統測試，表決器測試。系統在各項測試無誤期間，皆會傳出各項測試正確的符號代碼，以辨認過程及順序。

錯誤偵測系統的另外一個功用是提供一系列錯誤偵測、錯誤顯示以及錯誤排除之插斷服務。此錯誤偵測系統可以由軟體來觸發，在系統發生可偵測之錯誤、表決器回報錯誤狀況、或是使用者透過外部監視器發出指令時能夠提供因應的資料傳輸服務。我們首先必須制定一套自定的命令字元組，作為每次資料傳輸後表決器的“反應”。藉此可以瞭解資料傳輸的狀況，同時可以用作進入除錯模式的命令引發動作。

偵錯診斷系統的目的即是提供一系統至外接監視器之間的通道。我們製作了一塊 ISA 介面卡，利用一條排線連接我們的系統和介面卡，然後再把介面卡插在 PC 上，同時撰寫程式來自動監測輸出結果。資料傳輸的方式是利用表決器剩餘接腳產生一個硬體中斷訊號給 PC，每當一次表決結束即產生一次中斷訊號，PC 在接到中斷訊號之後即會引發硬體中斷來讀取資料。利用此種方式，我們可以在 PC 上看到系統輸出資料。不過產生 PC 硬體中斷所要付出的額外時間與將結果列印於螢幕上的時間，導至我們在實際測試時一直發生資料漏失的情況。後來我們降低系統輸出的速度，才勉強能接收資料。ISA 本身為一 16 位元匯流排，理論上我們可以製作兩張 ISA 卡，如此便能傳遞 32 位元的資料。但是在實際的操作中，PC 一次僅能從一個輸出入埠讀入資料，也就是說，就算我們採用兩張 ISA 卡的方式，在 PC 端還是需要執行兩輸入的指令才能讀入一筆 32 位元資料。在這種情況下對於速度的提升完全沒有幫助。

偵錯診斷系統的另一個重點是共享記憶體管理。為方便監測資料的正確性，我們製作了一塊測試用模擬電路板代替微處理器模組的位置。在經由模擬驗證共享記憶體管理元件的正確性以後，我們將系統連接至外接終端機，然後測試各種傳輸模式間的切換。如此便能確定偵錯診斷系統的正常工作。

四、結論與討論

本計畫的目的是在設計並且實作一用於飛行控制系統之容錯計算機系統雛型。由於此系統需具備高可靠度與即時控制的特性，因此採用以 TMR (Triple Module Redundancy) 為基礎之混合式容錯架構來達成系統需求。本計畫所研發的軟硬體系統可以應用在一般需加入容錯能力的系統上，例如

1、需要高可靠度的應用，例如飛行控制
2、具有危險性的措施，例如核能設施
3、具有高使用率的系統，例如銀行交易
4、特殊可靠度需求的電腦系統，例如人造衛星

經由本計畫的執行，我們研發了一個具有容錯能力的飛行控制系統的雛型，藉由這次計畫的執行也使我們得以將所學與實作相結合，當中我們學習到不少實務經驗，也謝謝中山科學院給與我們執行計畫的機會

五、參考文獻

[1] Algirdas Avizienis, “Fault-Tolerance: A Survival Attribute of Digital Systems,” *Proceedings of the IEEE*, Vol. 66, No. 10, October 1978, pp. 1109-1125.

[2] Algirdas Avizineis, and Jean-Claude Laprie, “Dependable Computing: From Concepts to Design Diversity,” *Proceedings of the IEEE,* Vol. 74, No. 5, May 1986, pp. 629-638.

[3] Leslie Lamport, Robert Shostak, Marshall Pease, “The Byzantine Generals Problem, “ *ACM Transactions on Programming Languages and Systems,* Vol. 4, No. 3, July 1982, pp. 382-401.

[4] Janusz Biernat, “Effect of Compensating Fault Models on NMR System Reliability,” *IEEE Transactions on Reliability*, Vol. 43, n 2, Jun 1994, pp. 294-300.

[5] Ma Maode, “Fault-Tolerant Strategy for Real-Time Task Scheduling on Multiprocessor System,” *Proceedings of the International Symposium on Parallel Architectures, Algorithms and Networks,* 1996, pp. 544-546.

[6] M. Karyagina, “Designing for Fault-Tolerance in the Commercial Enviroment,” *Proceedings of the Annual Reliability and Maintainability Symposium*, 1996, pp. 258-262.

[7] Shinichiro Yamaguchi, “Quad-Processor Redundancy for a RISC-Based Fault Tolerant Computer”, *IEICE Transactions on Information and Systems E80-D*, January 1997, pp. 15-20.

[8] Kenneth W. Philip, “Comparative Redundancy, an Alternative to Triple Modular Redundant System Design,” *Microelectronics and Reliability 37,* April 1997, pp. 581-585.

[9] Cecilia Metra, “Compact and Low Power On-Line Self-Testing Voting Scheme,” *IEEE International Workshop on Defect and Fault Tolerance in VLSI Systems*, 1997, pp. 137-145.

[10] K. J. Lee, “Design of a Fault-Tolerant Microprocessor: a Simulation Approach,” *Proceedings of the Pacific Rim International Symposium on Fault Tolerant Systems, PRFTS*, 1997, pp. 161-166.

[11] Robert B. Kerr, “Data Communications Management for The Boeing 777 Airplane,” *IEEE/AIAA 14th DASC Conference,* 1995, pp. 51-56.

[12] Jerald A. Edwards, “High Reliability in Spacecraft Computer Systems,” *IEEE Aerospace Application Conference Proceedings,* Vol.1, 1993, pp. 116-121.

[13] Brian D. Morrison, and Michael N. Robillard, “Flight Test And Certification Plans for Low-Cost Distributed Control-By-Light™ Systems,” *IEEE/AIAA 14th DASC Conference*, 1995, pp. 83-88.

[14] Brian D. Morrison, “Design And Certification of Low-Cost Distributed Control-By-Light™ Aircraft Control Systems for Part 25 Aircraft,” *Proceedings of SPIE – The International Society for Optical Engineering*, 1996, pp. 168-175.

[15] Bob Witwer, “Systems Integration of The 777 Airplane Information Management System (AIMS): A Honeywell Perspective”, *IEEE/AIAA 14th DASC Conference*, 1995, pp. 389-393.

[16] Y. C. (Bob) Yeh, Boeing Commercial Airplane Group, “Triple-Triple Redundant 777 Primary Flight Computer,” *IEEE Aerospace Application Conference Proceedings,* Vol. 1, 1996, pp. 293-307.

[17] Timothy P. Monaghan, “Fault Tolerant System Design in The Concept Exploration Stage of a Mission Critical Computing System,” *IEEE Aerospace Application Conference Proceedings,* Vol. 1, 1996, pp.321-333.

[18] F. Cristian, “Fault-Tolerance in Air Traffic Control Systems”, *ACM Transactions on Computer Systems 14*, 3, August 1996, pp. 265-286.

[19] William R. Dieter, and James E. Lumpp Jr., “Fault Recovery for Distributed Shared Memory Systems”, *IEEE Aerospace Application Conference Proceedings*, Vol.1, 1997, pp. 525-540.

[20] N. Gaitanis, “The Design of Totally Self-Checking TMR Fault-Tolerant Systems,” *IEEE Transactions on Computers,* Vol. 37, No. 11, November 1988, pp. 1450-1454.

[21] Tom Shanley, “*80486 System Architecture*,” 3rd ed. MindShare Inc., April 1995. (ISBN: 0-201-40994-1)

[22] Tom Shanley, “*ISA System Architecture*,” 3rd ed. Mindshare Inc., 1995. (ISBN: 0-201-40996-8)

[23] Ralf Brown, and Jim Kyle, “*PC Interrupts*,” Addison-Wesley, 1991.

[24] Subbarao, “*The 8086/8088 Family of Microprocessors: Software, Hardware, and System Applications*, “ Delmar Publishers Inc., 1992. (ISBN: 0-8273-3800-7)

[25] Barry B. Brey, “*The Advanced Intel Microprocessors: 80286, 80386, 80486*,” Merrill, an imprint of Macmillan Publishing Company, 1993. (ISBN: 0-02-314245-6)

[26] “*MIL-STD-1553 Designer’s Guide*,” 2nd ed., ILC Data Device Corporation, 1982.

[27] “*In-System Programmability Handbook,”* Altera Corporation, February 1998.

[28] AMD 5x86 Referance, PDF file, AMD Corporation, <http://www.amd.com/>.

[29] QuickLogic Corporation data-sheet, <http://www.quicklogic.com/>.

[30] Altera data-sheet reference, PDF files, Altera Corporation, <http://www.altera.com/>.

[31] “MAX-PLUS II Users Manual,” Altera Corporation.