PC-Based 嵌入式即時多工控制系統

工研院機械所　劉永欽

前言：

PC-Based 控制器由於具有模組化的特性、再加上使用 PC 的週邊介面，以及擁有豐富的軟體資源，不但在應用上可以有相當大的彈性，在成本上也具有絕對的競爭優勢。同時，一般人所質疑 PC 穩定性和可靠度的問題，也在各種努力下，使得 PC 的耐雜訊性、耐振動性、耐高溫性等，皆能達到工業等級的要求。再加上如 DisckOnChip、Compact Flash 等儲存媒體的誕生，使得整個 PC-Based 控制器在惡劣的環境下操作已經是值得信賴的。

儘管在硬體方面，PC-Based 控制器已經有了不錯的進展；然而在軟體方面，尤其是作業系統，並沒有跟隨硬體的腳步達到同樣的水平。相反的，市面上大部分的作業系統（如微軟的視窗作業系統，Windows Operating System），基本是針對個人與企業所設計的，對於工業控制中所需的嵌入式(Embedded Capability)與即時性(Real-Time Capability)等特性，幾乎是完全不支援的。近年來，針對上述的問題，微軟與其協力廠商，還有其他的團隊也提供了一些解決方案，爲 PC-Based 控制器的舞台紮下更穩固的基礎。

本文將針對目前在 PC-Based 工業控制的領域上，最常採用的四類作業系統：DOS、Windows Embedded Family、Windows CE 4.0 以及 Linux，分別說明其在嵌入式、即時性以及多工機能等方面的相關事宜，以作爲讀者在開發 PC-Based 控制器的參考依據。

一、DOS

曾是絕大部份 PC 作業系統的 DOS，應該是透明度最高的作業系統了，原因只是因爲它夠簡單，而且它所運用的 CPU 處理模式爲最簡單且最開放的 x86 真實模式（Real Mode），在此一執行模式之下，所有 PC 的資源包括輸出入埠（I/O Port）、記憶體空間（Memory Space，一般來說 MS-DOS 可直接定址到 1M 以下的位址空間）、中斷（Interrupt）等都可以透過非常容易的方式來存取，同時因爲所需執行空間小於 1 MBytes，使用者可以完全掌握系統的資源而加以充分的運用，這種小而美的特性，不論是在嵌入式與即時性的考量，都算是在眾多作業系統中的佼佼者。即使如此，DOS 也算是快要走入歷史胡同的產品，對大多數的人而言，特別是在學的學生，DOS 對他們而言只是曾經出現在書本上的名詞。然而在工控的領域，DOS 仍然佔著一席之地，尤其在嵌入式與即時性要求很嚴格的場合，它依然活躍著。以下將敍述在 DOS 環境下，開發 PC-Based 控制

器之應用程式所要注意的事項：

1. 記憶體限制

早期的 DOS 有著 640K 記憶體存取限制，這是因爲 CPU 是 16 位元所衍生出來的問題，這對於目前的 PC 動不動就裝配好幾百 Mega Bytes 的 DRAM，簡直是天壤之別。爲了解決這個問題，可以用外掛 DOS Extender 的方式來處理，使得不論是 code 或是 data 節區，都能超越 640K 的限制。值得一提的是，即使 DOS 已經如此歷史悠久，它還是需要付費的。近來在自由軟體基金會（Free Software Foundation，FSF）的推廣下，FreeDOS 逐漸成爲 DOS 開發者的另一個選擇。FreeDOS 支援 32 位元的作業環境，並不會有上述記憶體限制的問題，而且只要遵循 GNU 公眾授權（General Public License，GPL）的規範，就可以免費使用。

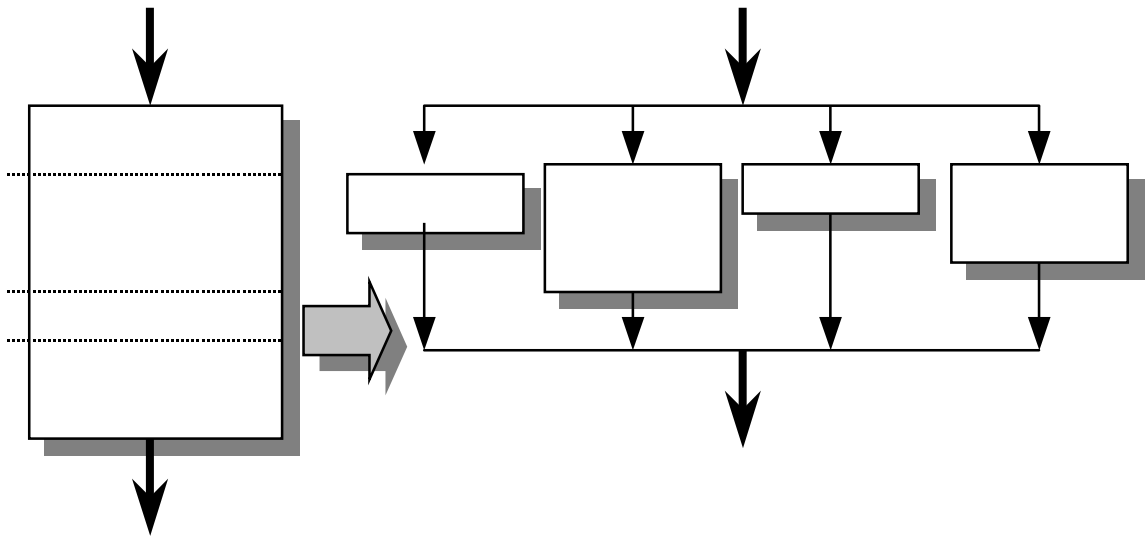
2. 圖形化界面與其他資源

DOS 並不像 Windows 或是 Linux 作業系統，本身已經提供標準的圖形化界面與其他資源（如網路、多媒體等），若是系統整合者有需要使用到這些資源，只能自行開發，而這些處理往往比真正在控制上的設計還要花費更多時間。同樣地 GNU 及其他協力廠商也針對 FreeDOS 與 DOS Extender 提供一系列相關的開發資源環境，例如類視窗的圖形化界面、網路的通訊協定、繪圖的 OpenGL、Game 的 3D Engine 及 PC 週邊的驅動程式等。整體而言，除非是真正的 Embedded 系統，否則在 DOS 下的開發時程會因爲可用資源有限的因素而比其它視窗型作業系統來得緩慢，這點對於 Time-to-Market 要求很高的場合，可能是比較不利的。

3. 多工機制

以機械工業中常見的 CNC 工具機而例，其控制器包含了機床的控制處理、切削路徑計算，以及相關參數的設定等，可說是一套十分複雜的多工系統。在這方面，DOS 本身是屬於單工的作業系統，並不如其它作業系統提供了多工的作業環境，可以同時處理許多工作單元（Task）的排程（Scheduling）與管理。基本上，在 DOS 下可利用許多技巧（如狀態機機制，state machine，如圖一所示）來達到多工的處理。

圖一　狀態機機制



當系統更趨複雜時，不同的工作單元之間同步與協調的工作反而成爲程式設計者的負擔。爲了讓使用者開發程式時更模組化、單純化，多工核心（Multitasking Kernel）便扮演了一個關鍵的角色。在多工核心的環境中，設計者只需獨立包裝其工作單元的功能即可，若需要與其它工作單元進行同步協調的工作，只要呼叫核心服務函式，由核心來處理相關的動作。其開發效率較單工至少提升了一倍以上，並且日後系統維護的工作也因高度的模組化而變的簡單。

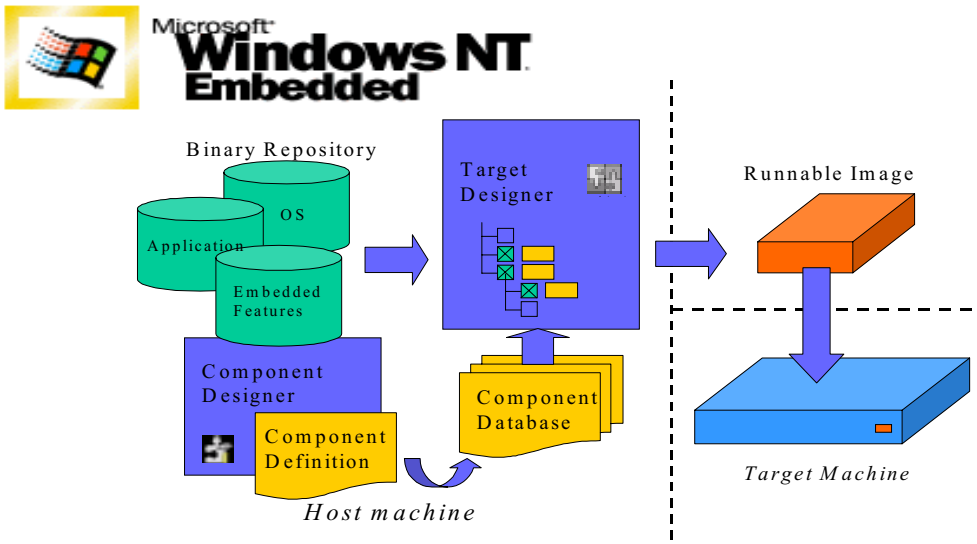
在了解多工核心的功能與重要性之後，是否每個程式設計者都必須自行開發多工核心呢？當然有許多教科書說明了多工核心的設計原理，要自行設計並非不可能。然而值得注意的是，在網路上有許多資源可以運用。比較知名的如『*μC/OS-II*』，許多人將在各式各樣硬體平台上所移植的這套多工核心在網路上分享出來，同時有一個組織在維護與處理已回報的 bug，這又是 Free Software 精神下的產品，對許多程式設計者而言，的確是一大福音。

二、Windows Embedded Family

Windows 雖然是現今 PC 作業系統的主流，幾乎每個人都熟悉其視窗化的操作界面，但是要把它拿來當成工業控制的作業系統，恐怕不是一件簡單的事。第一、在嵌入式方面，不管是 Windows 9x/NT/2000/XP，其所需的磁碟空間都是以幾百 Mega Bytes 來計算。這麼大的空間，目前除非用硬碟來儲存，否則以其它的儲存媒體如 DisckOnChip 或 Compact Flash 都是不合乎經濟效益的。然而硬碟在振動及雜訊干擾的環境下，通常是不太穩定的。第二、在即時性方面，不論是那一個版本的視窗作業系統，其所提供的即時能力都是所謂的『Soft Real-Time』，針對一些需要『Hard Real-Time』的場合，是無法滿足的。Microsoft 的 Windows Embedded Family 與 VenturCom 的 RTX 就是分別爲了解決嵌入式及即時性問題所開發的產品。以下將敍述在 Windows Embedded Family 作業系統下，開發 PC-Based 控制器之應用程式所要注意的技術與相關事項：

1. 嵌入式技術

Windows Embedded Family 包含了 NT Embedded 與 XP Embedded 二個成員，其前身爲 VenturCom 的 Component Integrator（CI），在 1998 年授權給微軟公司，並成爲 NT Embedded 產品。這些平台都提供了將原有的 NT 或 XP 作業系統模組化的工具，再由使用者根據實際的需要，組合成一個全新的 NT/XP Embedded 作業系統，如圖二所示。這種模組化的技術，的確可以將 Windows『瘦身』爲一個嵌入式作業系統，同時也支援不需使用螢幕、鍵盤、滑鼠才能開機的模式（亦即 Headless 模式）。但在商言商，這個專用型的嵌入式作業系統還是需要向微軟付版權費的。

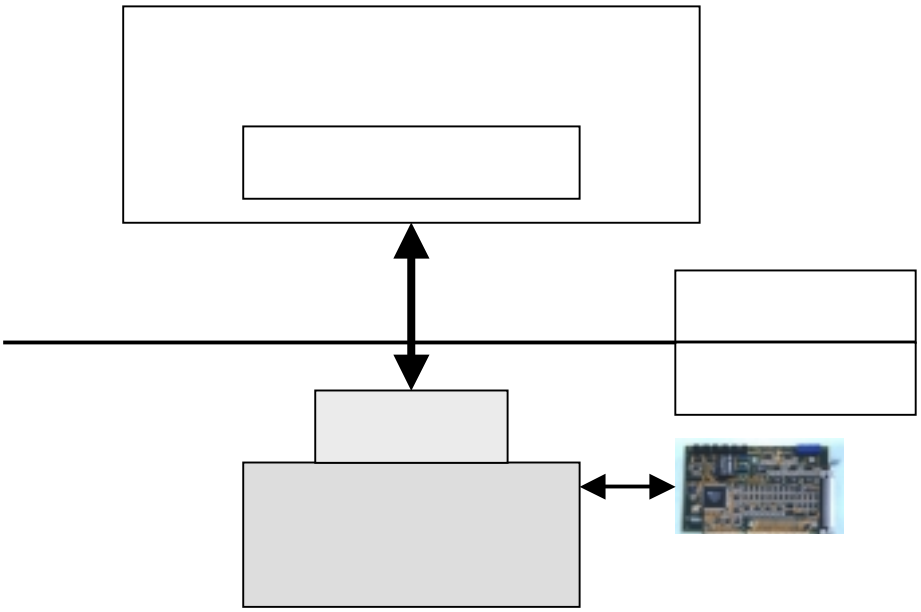


圖二　Microsoft Windows NT Embedded 技術（source : Microsoft）

一般而言，經由重整後新的 Windows Embedded OS，有兩種模式：第一是無法再利用『Install』的方式來增加新裝置的 Compact 模式，其所需磁碟空間較小，甚至只要不到 10M 的空間即可。但是在這種模式下，任何裝置都必須在設計階段就以指定的格式（即所謂的 Kit Definition File，KDF）存在資料庫中。若是想要使用特殊的裝置，如 PC 的 Add-on Card 等，就必須有該項裝置的 KDF 檔，否則就無法在新的 Embedded 作業系統下使用該項裝置。第二是可以利用『Install』的方式來增加新裝置的 Embedded 模式，這種模式允許使用者以在一般 Windows 下安裝新裝置的方式來加入新的硬體，但相對地其所佔的空間也比較大，通常至少需要 60M 以上的空間。

2. 驅動程式開發

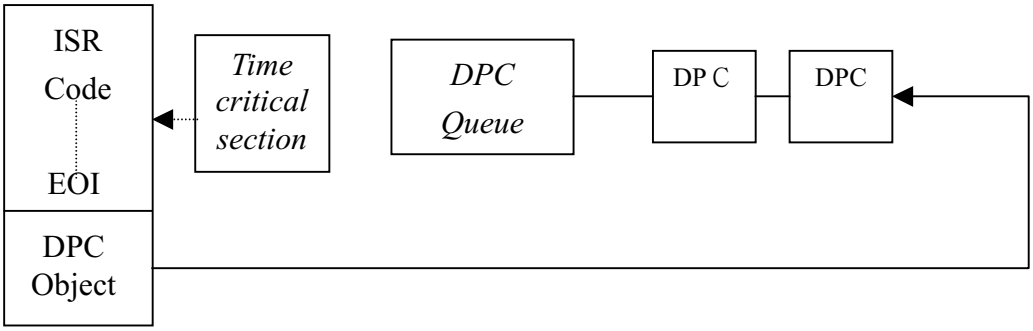
有別於 DOS 作業系統是處於真實模式下的驅動程式架構，Windows 是在保護模式下運作，所以自然無法直接地與 Device 溝通。而且不同版本的 Windows 其本身的 Device Driver Model 也不儘相同，例如 9X 系列的『虛擬裝置（Virtual Machine）』、NT 系列的 Kernel Mode、以及 2000 的 WDM 等。對使用者而言，需要了解各種平台的驅動程式架構，這無非是一種負擔。因此，近年來標榜著只要撰寫一次驅動程式的 code 之後，在其他平台上重新編譯後即可執行的輔助工具（例如 WinDriver），就深受程式開發者的喜愛。這類型產品，其精神在於先幫使用者在各種平台下開發代理（Agent）的驅動程式（如圖三所示），再將其包裝成統一的介面，然後提供使用者呼叫，實際上硬體運作的機制，全完是透過其 Agent 程式來達成。雖然　牲了一些效能，但卻大大提高了平台的相容性，而且在 CPU 的速度愈來愈快的趨勢下，反而突顯了它的優點。



圖三　代理型驅動程式之架構

3. 即時系統技術

前文提及 DOS 是屬於單工作業系統，而視窗本身即爲多工作業系統，採用 Priority-Based 的排程準則，亦即 Priority 較高的工作單元享有較優先的執行權利。只是不論是那一個版本的視窗作業系統，其本身的即時性都只有 10 ms 左右的 Soft Real-Time 能力，這僅是其平均數據而已，通常會有更糟糕的情況產生。不只如此，NT 本身的多工機制對工業控制而言，有些地方設計的並不合理。例如在 NT 底下的驅動程式架構，在中斷服務程式（Interrupt Service Routine，ISR）所包裝出來 Deferred Procedure Calls (DPC，如圖四所示)，基本上會被置於一個佇列（Queue）中，與當初所觸發的中斷的優先權高低並無關連。再加上用來做爲同步機制的 Event、Semaphore、Mutex 等物件，其狀況也是同樣被置於佇列中，對即時系統而言，這樣的安排並非最佳化的方案。



圖四 Windows NT 中的 Deferred Procedure Calls (DPC)示意圖

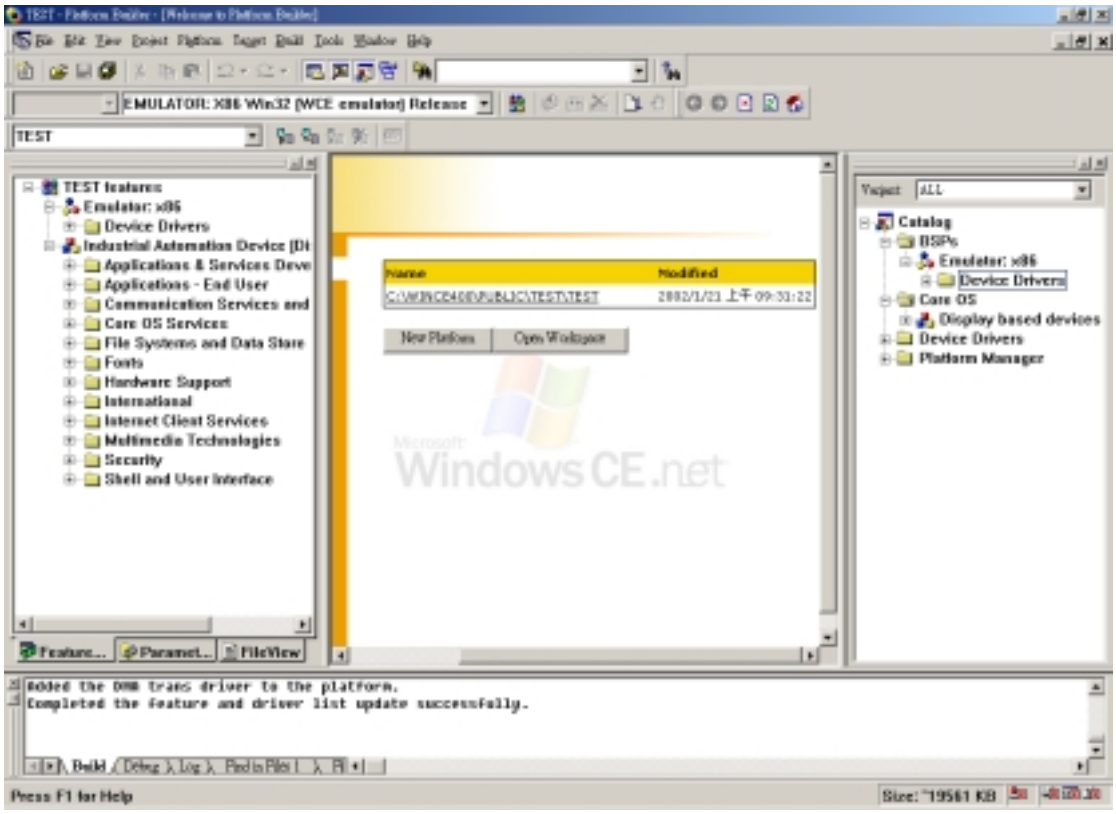
爲了改善 Windows 作業系統的即時能力，許多軟體公司紛紛與 Microsoft 合作，以 third-party 型式開發附加於 Windows NT 環境下的延伸模組，以提升作業系統的即時能力。這類延伸模組其基本出發點都是希望保有原先 Windows 作業系統的優點，同時透過增加的即時子系統（Real-Time SubSystem），使的在此子系統所執行的行程具有至少低於 1 ms 等級的「Hard Real-Time」能力。以 VenturCom 的 RTX 爲例，它提供了一組豐富的 Real-Time API 可以運用於 Win32 子系統與其設計的即時子系統之中，而且在即時子系統之中執行的行程，其即時響應能力可達 50 μs，較原先 Windows NT 所宣稱的數據提升了 100 倍以上。雖然有了這些加強即時能力的軟體工具的輔助，將可在 Windows NT 作業系統之中開發具有高即時性的多工應用程式，但相對地與 DOS 比較起來，在成本方面則會增加許多權利金的支付，對單價較爲便宜的工業控制器而言，通常是無法使用這樣具有不錯的即時性能力之嵌入式視窗作業系統，做爲其 PC-Based 控制器的開發平台。

三、Windows CE

微軟的策略是將 Windows CE 定位爲一種多用途的嵌入式設備作業系統，它是一個 32 位元、先佔式多工、並且支援 x86 以外 CPU 的視窗作業系統，但在早期發展 Windows CE 的許多 Projects 都不是很順利，這個作業系統一路走來可說是備受辛勞。然而隨著一些個人消費性產品（PDA、Hand-held PC 等）的大行其道，Windows CE 才漸露頭角。Windows CE 目前的版本已經到達了 4.0 版（亦即之前所謂的 Talisker），在許多方面，可看得出來微軟對這個作業系統的企圖心，例如支援 IEEE 802.11 無線網路標準、亞洲語系（尤其是繁體中文）、DirectX 技術、IE 5.5 及 Media Player 等。此外，除了嵌入式的特性之外，在即時性方面也做了大幅度的修改，使得 Windows CE 可以稱得上是 RTOS（Real-Time Operating System）。以下將敍述在 Windows CE 作業系統下，開發 PC-Based 控制器之應用程式所要注意的技術與相關事項：

1. 嵌入式技術

要了解 Windows CE 的嵌入式技術，就必須從其開發工具『Platform Builder』著手。Platform Builder 是一個與 NT/XP Embedded 之 Target Designer 相當的 OS 整合開發環境（Integrated Development Environment, IDE），如圖五所示。但不像 Target Designer 僅是模組選擇的介面工具與執行 OS 重建功能之外，它提供了 New Platform Wizard、Export Wizard、Board Support Package (BSP) Wizard 以及基本的組態資料庫來協助新平台的建立；同時也提供了 Kernel Debugger、Application Debugger 及其它遠端診斷工具來進行驅動程式與應用程式的除錯動作。此外，4.0 版的 Platform Builder 還提供了較單純的模擬環境（Emulator），讓初學者能夠更快的上手。對一般的 x86 平台來說，利用其中所提供的 CEPC 組態即可完成 Windows CE 的建立，但對其它的平台而言，例如 Arm、SA11xx、SHx 等系列的 CPU，還要考慮到如 Boot Loader、OEM adaptation layer（OAL）、BSP、驅動程式等方面的開發。整體而言，利用 Platform Builder 所產生的 OS image 比 NT/XP Embedded 系列還要小許多，更適合用於嵌入式系統。



圖五 Platform Builder 4.0

2. 即時系統技術

從 Windows CE 的白皮書中可以發現，微軟宣稱 Windows CE 4.0 爲 RTOS。實際上 Windows CE 4.0 也確實保證了最高優先權的工作單元（Thread）及中斷服務程式（ISR）延遲時間的上限，對 PC-Based 控制器來說，絕對是有正面的價值。以下是 Windows CE 4.0 在做爲 RTOS 的一些努力：

- 支援最多 32 個不同的應用程式
- 支援 256 個等級的優先權系統（Priority System），0 爲最高等級
- 支援優先權錯置（Priority Inversion）的處理
- 支援巢狀式中斷（Nested Interrupts），以保證最高優先權事件先被處理
- 支援 1 ms 的系統核心時脈
- 先進的工作單元排程技術與執行時間控管
- 支援 semaphores 等同步物件處理

其中有關 256 個等級的優先權系統可被分爲四個族群，如表一所示。

表一　Windows CE 4.0 優先權系統之分類

等級	說明
0 ~ 96	保留給真正需要即時能力的驅動程式使用
97 ~ 152	保留給一般 Windows CE 的驅動程式使用
153 ~ 247	保留給即時性需求較低的驅動程式使用
248 ~ 255	保留給完全不需要即時性的驅動程式使用

此外，Windows CE 4.0 也提供了量測即時性能的工具程式：ILTiming.exe 與 OSBench.exe。其中 ILTiming.exe 是用來提供 OEM 量測 ISR 及 IST 的延遲時間；OSBench.exe 則是用來提供 OEM 量測排程的效能。

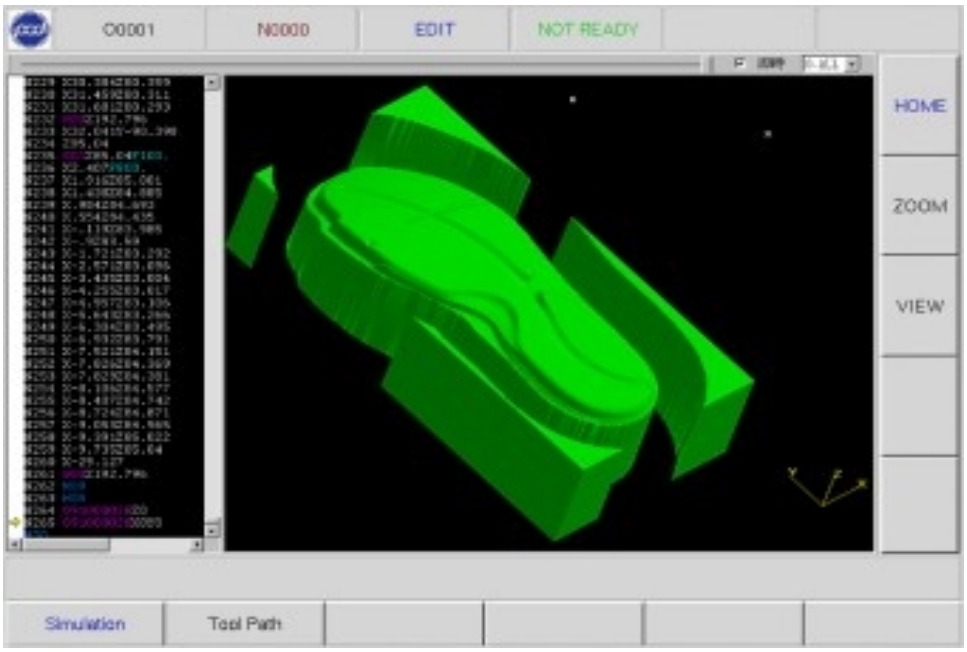
3. 授權模式

Windows CE 的授權方式與傳統的 Desktop PC 作業系統有著完全不同的模式。一般的視窗作業系統只要透過 Vendor 即可買到，而開發 Windows CE 則需向微軟告知所開發 Windows CE 的用途，再依據預測的發行量，向微軟或其它代理商支付授權的權利金。Windows CE 授權的方式（也就是權利金的價位）可分爲三個等級：第一級是完整版（Full Operating System License），這個版本可以使用到 Windows CE 提供的所有資源，包括核心、檔案系統、通訊、連結、事件與訊息處理、繪圖、鍵盤、觸控螢幕、視窗管理及一般的控制元件等。第二級是精簡版（Limited Operating System License），基本是給不會用到很多圖形化界面的產品來使用。第三級是核心版（Kernel Operating System License），則是給完全不

會使用到圖形化界面的產品（Headless）來使用。總體來說，Windows CE 的權利金比 NT/XP Embedded 要來得便宜許多，其市場競爭優勢也相對地提高。

四、Linux

開放式原始碼（Open Source）的主張在最近幾年來獲得不少人的認同，因此也造就了 Linux 成爲微軟的視窗作業系統的第一競爭對手。根據 IDC 的資料，未來的四年，Linux 都會以每年 20%以上的速度成長，這是值得密切觀察的趨勢。仔細思考，開放式原始碼的架構的確對 bug 的處理與量身訂做的模組化提供了最佳的透明資訊。在自由軟體基金會的努力下，不論在嵌入式與即時性方面，Linux 已經提供了相當優異的發展平台。尤其是任何在視窗作業系統上無法完成的核心功能，也都可以藉由開放式原始碼來實現。在資訊家電方面，Embedded Linux 也經有相當不錯的成果，並且在市場也已經佔了一席之地。在工業控制的領域上，也可以找到 Linux 所屬的 PC-Based 控制器（如圖六所示）。以下將敍述在 Linux 作業系統下，開發 PC-Based 控制器之應用程式所要注意的技術與相關事項：



圖六　Linux CNC controller（source： 寶元科技股份有限公司）

1. 嵌入式技術

Linux 的嵌入式技術完全是來自其開放式原始碼的特性，使用者可依據自己的需求來設計專用的 Linux 作業系統。此外，在 Desktop 方面，可以有 Gnome 與 KDE 可供選擇；在人機界面開發方面，也有 C/C++及 Delphi 兩種較爲一般程式設計者熟悉的語言。總而言之，只要能夠了解 Linux 核心的運作，一切的功能都可藉由修改其原始碼來達成。

2. 即時系統技術

嚴格來說，在目前的階段，Linux 本身所能提供的即時性，並不能達工業控制的要求。然而經由許多團隊的努力，在即時性方面也已得到初步的成果，如 RT Linux、KURT、Linux/RK、RED-Linux 等，並且實際應用在 PC-Based 控制器上。以 NMT 的 RT Linux 爲例，其概念爲『架空』Linux Kernel，使其 Real-Time 的 Process 可以優先被執行。另外也有利用系統的時間驅動方式，來達成排程的目的。一般說來，利用系統的時間驅動方式所達到的即時性，1 ms 已經是極限了。架空型的 Linux，則宣稱可以達到 50~100 μs 等級的延遲時間。這樣的數據表現，比上述的三種作業系統都還要來得遜色。在即時性方面，Linux 仍然有很大的改善空間。

3. 授權模式

提到了 Linux，就會讓人連想到免費資源，但是仍然必須注意遵循 GNU 公眾授權（GPL）的規範。GPL 特有的『Copyleft』，是指使用者只要是有修改到核心的部分，就必須將自己所修改的原始碼公開才行。至於與核心無關的模組，則不在此限。其精神在於透過允許他人可以任意修改原始碼的方式，來達到知識交流的目的。儘管人們仍然在討論開放式原始碼本身的價值與商業模式，Linux 的確爲 PC-Based 控制器提供了另一個舞台。

五、Discussion

本文旨在討論目前 PC 上較常用的四類作業系統：DOS、Windows Embedded Family、Windows CE 4.0 以及 Linux，開發 PC-Based 工業控制器之應用程式所要注意的技術與相關事項。其目的並不是要去比較每個作業系統的優劣，只是依據每個作業系統的在嵌入式、即時性方面的特性，分析其成爲工業控制器平台所需注意的技術及商業模式。沒有任何一種作業系統是十全十美的，所謂的優劣，完全是取決於應用的場合。以下是筆者針對上述作業系統在技術方面以外所提出的一些簡單問題與討論，答案就留給讀者自行回答，並以此做爲本文的結束。

1. 微軟在嵌入式系統分別推出了 NT/XP Embedded 及 Windows CE.net 兩項產品，在即時性方面，NT/XP Embedded 有協力廠商提供的解決方案，Windows CE 則爲內建的即時能力，表二爲這些系統的即時性能。要注意的是並不是數字愈小就愈好，Real-Time 的本質應該是夠用就好，太過追求即時能力的等級，反而會造成系統整合方面不必要的負擔。當然微軟在正式場合都會說明這兩種產品的區別：例如在 PDA 就該選擇 Windows CE，在 Desktop 則應選擇 NT/XP Embedded。但是嚴格來說，工業控制正好是介於這兩種產品的灰色地帶，如何選擇，反而成爲一項非技術層次的關鍵問題。

表二　Windows-Based 作業系統的即時性能
（Min/Max in μs on Pentium III 700）

Operation	NT 4.0	CE 3.0	RTX 5.0
SetEvent (no switch)	1.0 / 5000+	1.7 / 8.2+	0.3 / 5.7
Event	1.4 / 5000+	2.8 / 12.2+	0.9 / 8.3
Mutex	1.5 / 5000+	4.0 / 12.0+	1.0 / 8.5
Semaphore	1.4 / 5000+	3.3 / 10.7+	0.9 / 8.3
Yield	1.2 / 5000+	1.5 / 9.5+	0.4 / 6.8
Priority Change	1.3 / 5000+	1.6 / 10.2+	0.7 / 7.3
IST Latency	6.7 / 5000+	6.7 / 26+	4.0 / 24

Source ： VenturCom

2. 關於 FreeDOS 與 Linux 這類 Open Source 的作業系統，大多數人一開始都會被免費資源的招牌所吸引，卻忽略了背後所必須花費的技術問題。開放式原始碼的確帶來一些好處，但相對的必須要有人來將這些免費的資源包裝、甚至是修改成自己的產品。也就是說，儘管已經取得 Red Hat

版本的 Linux Kernel 及 RTAI real-time kernel，接下來還是需要有『人』去將這些『原始碼』做適度的修改，才能變成最終適用的控制系統。既然是以原始碼的方式，基本是就必須由專業人士來處理，而且也不會提供任何 user friendly 的介面工具，一般人要來做這些事情，進入門檻相對地也會提高。此外，Open Source 基本上是一群對軟體的狂熱份子所做的貢獻，但是也沒有義務對後續的維護做任何的承諾。與商業模式的視窗作業系統比較起來，使用 Open Source 做爲開發平台所必須投入的人力可能會比較多。對以營利爲目的的單位來說，選擇用金錢買現成的套裝軟體，還是選擇投入人力的軟體研發，有待決策者的智慧來定論。

六、參考資料

- **http://www.epcio.com.tw/**
- **http://www.microsoft.com**
- **http://www.microsoft.com/windows/embedded/default.asp**
- **http://www.vci.com**
- **http://www.freedos.org/**
- **http://www.controltech.com.tw/**
- **http://www.ucos-ii.com/**
- **http://www.redhat.com/**
- **http://opensource.lineo.com/rtai.html**
- **http://vdkbuilder.sourceforge.net/**
- **James Y. Wilson, Aspi Havewala. 2001. “Building Powerful Platforms with Windows CE®.” Addison-Wesley.**
- **Chris Muench, Randolph Kath. 2001. “The Windows CE® Technology Tutorial.” Addison-Wesley.**
- **Alessandro Rubini. 2000. “Linux Device Drivers.” O’REILLY.**