# PCI BUS 運動控制模組技術分析

工研院機械所 / 張盈喬

## 壹、前言

隨著電腦科技軟/硬體的快速發展，傳統 ISA 介面擴充卡其 8MHz 的最大傳輸速度及需由使用者指定資源的缺點已無法發揮 CPU 具高速運算及新一代作業系統具即插即用(Plug & Play)能力的優點，亦不符合多媒體裝置快速資料傳輸的需求，因此目前資訊業大部分的擴充卡皆已採用具 33MHz 以上傳輸速度及 Plug & Play 能力的 PCI 介面。在運動控制領域，雖然對資料傳輸速度的要求不比多媒體系統來的高，但爲順應 PC 架構的演進，新一代的 PC Based 運動控制卡也由傳統的 ISA 介面演進至 PCI 乃至 Compact PCI 介面規格。本文將就 PCI 介面規格，分析 PCI 介面運動控制模組開發時所應注意的事項，作爲 PCI 介面運動控制模組使用與開發者的參考。

## 貳、PC Based 運動控制模組介面發展趨勢

電腦可以由許多不同的型式來實現，不同的應用領域對電腦及其介面擴充設備的要求皆不盡相同，表 1 所示[1]是各應用領域電腦擴充設備之介面發展趨勢，其共同趨勢是朝開放性架構、高速、使用方便、高可靠度、低成本等方向發展。

傳統的 PC Based 運動控制模組採用 ISA 介面與 PC 間做資料溝通，其缺點除傳輸速度慢外，使用者於使用前需詳讀使用手冊中有關擴充卡與 PC 溝通所需的 Memory、I/O、中斷等資源之設定，而後經手動調整擴充卡上的 Jumper(跳線座)及由軟體指定的方式設定相關的溝通資源，這樣的方式往往造成使用者的負擔，並且不當的設定容易與 PC 軟體或週邊設備相衝突，造成動作異常甚至引發當機的情形，因此爲提高使用者的便利性及讓視窗作業系統能夠完全掌控系統資源，新一代 PC 擴充設備除傳輸速度的提昇外，另一項訴求是需具有即插即用能力，讓使用者僅需將擴充卡置入 PC 中，開機後系統軟體自動配置該擴充卡所需的資源，並載入相關的軟體，使用者不再需要做資源設定的工作。在 PC99 的規格中[12]，ISA Bus 已被完全屏除於新的架構之外，這意味著市售主機板內 ISA 插槽將很快的被 PCI 插槽所取代，未來的一般電腦將不再支援 ISA 擴充卡，因此從技術開發與 PC Based 零組件供應而言，具 PCI 介面的運動控制卡將是新一代 PC Based 運動控制模組的趨勢。

(表 1) PC 介面發展趨勢

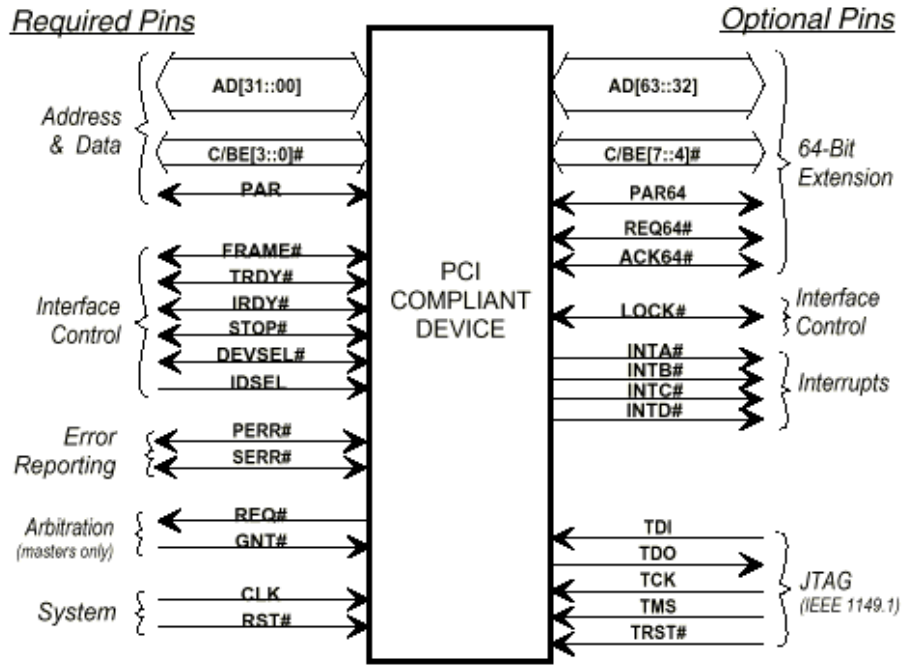
| Application | Old Form Factor | New Form Factor |
|---|---|---|
| Industrial | VME | Compact PCI |
| Desk Top PC | ISA , EISA , MicroChannel | PCI |
| Portable PC | PCMCIA | Small PCI |
| Instrumentation | VXI | PXI |
| Small Form Factor | IP , PC104 | PMC , PC104-Plus |
| Workstation | S-Bus | PCI |
| Consumer | Local Bus / Proprietary | PCI Local Bus |
| Telecom | Proprietary | Compact PCI |

# 參、PCI BUS 運動控制模組開發

PCI 1.0 版是由 Intel 公司於 1992 年公佈，該公司希望 PCI 標準能夠成爲新一代 PC 介面的開放性架構，同時爲了不讓電腦相關業者有 PCI 規格受 Intel 束縛的疑慮，因此讚助成立了 PCI Special Interest Group(PCI SIG) [2]，該協會是一具有公正立場的機構，負責爾後 PCI 介面標準的訂定與維護。目前最新的版本是 PCI Local Bus 2.2，主要規格如下：

- Clock 速度：33MHz(5 volt 或 3.3volt 工作電壓)
  66MHz (3.3volt 工作電壓)
- 位元：32 或 64 bits
- 最大傳輸速度：132 MB/S for 33 MHz , 32-bit PCI Bus
  264 MB/S for 66 MHz , 32-bit PCI Bus
  264 MB/S for 33 MHz , 64-bit PCI Bus
  528 MB/S for 66 MHz , 64-bit PCI Bus
- 接腳數目：124 pins(for 32 bits) / 188 pins (for 64 bits)
- 具自動組態功能(auto configuration)

# 一、PCI Bus 信號




(圖 1) PCI Bus Signal List

PCI Bus 信號如圖 1 所示[3]，主要信號線介紹如下，詳細的信號及時序(timing)定義請參閱 PCI 規格書。

1. PCI Bus 使用分時多工的位址/資料匯流排 AD[31:00]及 PCI Bus 特有的 4 位元指令 C\BE[3:0]# (Bus Command or Byte Enable)來對 PCI Bus 所具有的 Memory , I/O , Configuration Register 等三種組態空間作資料傳輸。
   - 在資料傳輸的位址階段(Address Phase)，AD[31:00]作爲位址匯流排(Address Bus)，傳送起始位址；而 C\BE[3:0]#以 4 位元指令定義欲存取的組態空間(Memory , I/O or Configuration Register)。
   - 在資料傳輸的資料階段(Data Phase)，AD[31:00]作爲資料匯流排(Data Bus)，傳送存取的資料；而 C\BE[3:0]# 作爲位元組致能信號(Byte Enable)。

2. 介面控制信號
   - FRAME# (Cycle Frame)：
     由 PCI Master 驅動，用來指示資料傳輸之開始與結束。
   - IRDY# (Initiator Ready)：
     由 PCI Master 驅動，表示 PCI Master 已 Ready，可做資料傳輸
   - TRDY# (Target Ready)：
     由 PCI Target 驅動，表示 PCI Target 已 Ready，可做資料傳輸
   - DEVSEL# (Device Select)：

由 PCI Target 驅動，當 PCI Target 被 Access 時發出。

- STOP#：

  由 PCI Target 驅動，表示希望 PCI Master 停止目前的資料傳輸。

- IDSEL# (Initialization Device Select)：

  於存取 PCI Configuration Register 期間由 PCI Master 驅動。

3. 每個 PCI device 具有 INTA#, INTB#, INTC#, INTD#4 等 4 條中斷請求輸入線。當 PCI device 是單功能裝置時(Single Function，由下一節介紹的 PCI Configuration Register 中的 Header Type 做設定)，只能允許使用 INTA#；當 PCI device 是多功能裝置時(Multi-Function)可允許使用 INTA# ~ INTD#。

## 二、PCI Configuration Registers

爲方便使用者使用，新一代的介面擴充卡都具有即插即用(Plug & Play)能力，而欲達到此功能，PC 及介面擴充卡在設計上需做下列事項之配合[5]：

- PC 需具介面擴充卡偵測及資源分配之機制

  PC BIOS 與作業系統等系統軟體需具有能夠偵測到目前 PC 上有那些介面擴充卡存在及自動分配系統資源的能力。

- 介面擴充卡上需具有製造商/產品代號及資源需求清單

  爲了讓系統軟體能夠區別插在 PC 上眾多的擴充卡，擴充卡本身需提供自己的製造商及產品代號之資訊。同時擴充卡尚需提供欲向 PC 要求的 Memory、I/O、Interrupt 等資源清單，以備讓 PC 系統軟體讀取並分配資源給該擴充卡。

就硬體設計而言，不同介面形式的擴充卡是否具即插即用的功能關鍵在於介面擴充卡本身是否具備一組規格暫存器(Configuration Register)，以提供上述與 PC 溝通的相關資訊(製造商/產品代號、資源需求清單)。傳統的 ISA Bus 擴充卡並未定義規格暫存器，因此無法讓 PC 自動偵測與分配必要的資源給該 ISA 擴充卡；PCI Bus 在規格上定義了一組規格暫存器(PCI Configuration)，因此只要 PC 端的系統軟體符合 PCI 軟體規格，則可達到即插即用功能。

PCI Configuration Register 基本上是一 64 dword 的組態空間(Configuration Space)，其格式分爲 Header Type 0 , Header Type 1 , Header Type 2 等三種，其中 64 dword 中的前 16 dword 不論是哪一種 Type 皆具相同的定義與使用方式；而後 48 dword 則依不同的 Type 有不同的定義。

大多數的 PCI 元件 configuration Register 皆屬於 Header Type 0，其格式如表 2 所示，其中 Device ID, Vender ID, Status, Command, Revision ID, Class Code, Header Type 是任一 PCI Device 皆必須給定的資料，而其餘部分則依 PCI device 實際的設計做選擇性的規劃。

(表 2) PCI Configuration Space for Header Type 0

| 31 | 16 | 15 | 0 | |
|---|---|---|---|---|
| **Device ID** | | **Vender ID** | | 00h |
| **Status** | | **Command** | | 04h |
| **Class Code** | | | **Revision ID** | 08h |
| **BIST** | **Header Type** | **Latency Timer** | **Cache Line Size** | 0ch |
| **Base Address Registers** | | | | 10h<br>14h<br>18h<br>1Ch<br>20h<br>24h |
| **Cardbus CIS Pointer** | | | | 28h |
| **Subsystem ID** | | **Subsystem Vender ID** | | 2Ch |
| **Expansion ROM Base Address** | | | | 30h |
| **Reserved** | | | **Capability Pointer** | 34h |
| **Reserved** | | | | 38h |
| **Max_Lat** | **Min_Gnt** | **Interrupt Pin** | **Interrupt Line** | 3Ch |

PCI Configuration Header Type 0 主要暫存器的意義如下，詳細的使用方式請參閱 PCI 規格書[3]。

•Vender ID (製造商識別碼)：

由 PCI Special Interest Group(PCI SIG)分配用以識別該 PCI device 製造商，若欲得到一組專用的 Vender ID，需加入 PCISIG 會員。

•Device ID (裝置識別碼)：

由 PCI device 製造商自行指派，用以識別產品名稱。

•Command (指令暫存器)：

PC 與 PCI device 間溝通之設定。例如要求 PC 是否對此 PCI device 作 I/O 或 Memory 資源空間配置等。

•Status(狀態暫存器)：

指示目前的資料傳輸狀態。

•Revision ID (版本識別碼)：

由設備製造商自行指派，用以識別產品的版本。

•Class Code (類別碼)：

設定產品的類別，需規格作設定。

•Header Type：

設定 PCI Configuration 之 Type(Header Type 0,1,or 2)及該 PCI device 是屬於單功能裝置(Single Function device)或多功能裝置(Multi-Function device)。

•Base Address (基底位址)：

在系統自動組態軟體尚未執行前，本暫存器紀錄著該 PCI device 欲向 PC 要求的 Memory 與 I/O 之資源範圍，而後自動組態軟體將依其需求，分配所需的 Memory 與 I/O 資源給該 PCI device，並將所分配資源的 base address 回填入本暫存器。

•Interrupt Pin：

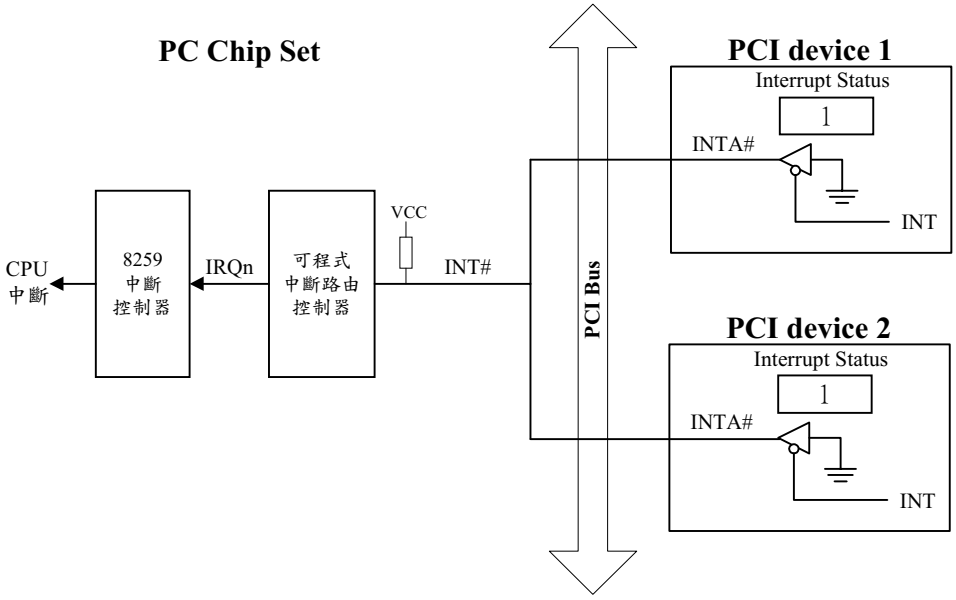
設定 PCI device 使用哪一條中斷請求線，其內含值 01~04 對應到 PCI device 中斷信號的 INTA# ~ INTD#。

•Interrupt Line：

當 PCI device 有要求 PC 作中斷配置時，本暫存器由自動組態軟體填入系統所配置的中斷線號碼，其中 00h ~ 0fh 分別表示使用 PC 中斷控制器上的 IRQ0 ~ IRQ15 中斷線。

# 三、PCI 中斷

PCI 中斷有別於 ISA 中斷，ISA 中斷請求線是正緣觸發(rising edge-triggered)，且不可共用(一個 ISA 中斷線佔用一 PC 硬體中斷)；PCI 中斷是低態動作(active low)、準位觸發(level triggered)且可共用(多個 PCI 中斷線可共用一個 PC 硬體中斷)。[6] [7]




(圖 2 ) PCI 中斷

如圖 2 所示，假設系統中有兩個 PCI 元件，個別發出的 INTA#信號在 PC 端被共同連接至系統晶片組(PC Chip Set)，經可程式中斷路由器規劃爲共同使用一硬體中斷 IRQn，而後經 8259 中斷控制器中斷 CPU。PCI device 上的中斷信號線需設計爲 open collector 形式，以確保多個 PCI device 中斷同時驅動時不會損壞元件，並且需規劃一個中斷狀態暫存器(Interrupt Status)以紀錄目前該 PCI device 的中斷狀態。

1. 在沒有中斷信號發生時，PC 端的 pull high 電阻將使 INT#信號處於高態(high)
2. 當其中一個 PCI device 上之 INT 信號 active 時，其 INTA#信號將被下拉至低態(low)並且其 Interrupt Status 需紀錄爲”1”，此時 PC 端 INT#將爲低態，並且發出 IRQn 信號中斷 CPU。
3. 中斷發生後中斷副程式需分別檢查 PCI device 1 & 2 的 Interrupt Status，以判斷是哪一個 PCI device 發出中斷請求，而後執行中斷程序。此外需清除此 PCI device 上的中斷請求 INT 信號，使 INTA#恢復至高態(high)，讓 Interrupt Status 恢復至”0”之狀態。

# 四、PCI BIOS

PCI device 的 Configuration Register、Interrupt Routing、Special Cycle 在 PC 系統中是屬於 PCI 規格所特有的機制，必須透過 PCI BIOS 才能對它們做控制。PCI BIOS 提供了表 3 所示的函式來對 PCI device 做相關的設定，使用方式是透過 INT 1Ah 軟體中斷指令來呼叫 BIOS 執行所指定的 PCI function，各 PCI function 的輸入及回傳參數請參閱 PCI BIOS Specification。[4]

(表 3 ) PCI Function List

| PCI Function | AH | AL | Class |
|---|---|---|---|
| PCI Function ID | B1h | | |
| Get PCI BIOS Present Status | | 01h | Resource Identification |
| Find a PCI Device | | 02h | Resource Identification |
| Find a PCI Class Code | | 03h | Resource Identification |
| Generate a PCI Special Cycle | | 06h | PCI Support |
| Read a PCI Configuration Byte | | 08h | Configuration Space |
| Read a PCI Configuration Word | | 09h | Configuration Space |
| Read a PCI Configuration Dword | | 0Ah | Configuration Space |
| Write a PCI Configuration Byte | | 0Bh | Configuration Space |
| Write a PCI Configuration Word | | 0Ch | Configuration Space |
| Write a PCI Configuration Dword | | 0Dh | Configuration Space |
| Get PCI Interrupt Routing Options | | 0Eh | PCI Support |
| Set PCI Hardware | | 0Fh | PCI Support |

舉例說明，若要尋找某一 PCI device，可呼叫 *Find a PCI Device* 函式。如表 4 所示，設定 AH 暫存器爲 B1h、AL 暫存器爲 02h，同時將該 PCI device 的 Vender ID, Device ID, 及 Index(板次)分別設定於 CX, DX, SI 暫存器，透過 INT 1A 中斷服務常式，BIOS 將執行 *Find a PCI Device* 函式，並回傳該 PCI device 在系統中的 Bus Number, Device Number , Function Number 等參數，藉由這些回傳參數使用者可進一步呼叫其他 PCI function 做 Configuration Register 存取、Interrupt Routing 或 Special Cycle 設定。

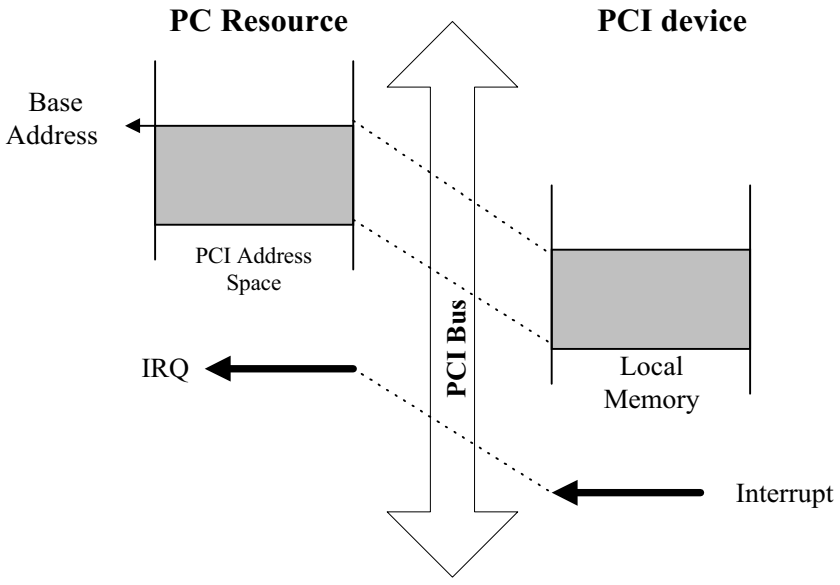
(表 4 ) PCI Function example

| Find a PCI Device | | |
|---|---|---|
| PURPOSE : This function returns the geographical bus, device, and function number of a PCI device that has a specific Device ID and Vender ID. | | |
| ENTRY : | [AH] | 0B1h |
| | [AL] | 02h |
| | [CX] | PCI Device ID |
| | [DX] | PCI Vender ID |
| | [SI] | Index (which instance of the device to search for) |
| EXIT : | [AH] | Return Code :<br>00h = Successful. A PCI device was found.<br>83h = Routine was called with a Bad_Vender ID<br>86h = A PCI Device_Not_Found |
| | [BH] | PCI Bus number (0~255) where device is located. |
| | [BL] | PCI Device/Function Number:<br>Bits Description<br>2:0 Function Number<br>7:3 Device Number |
| | [CF] | Function Completion Status<br>0 = Successful<br>1 = Error |

## 五、**PCI device 資源分配及控制**

當一片 PCI 卡置於 PCI 插槽，開機後系統是如何偵測到它的存在、知道該卡的資源需求、進而分配不與其他裝置相衝突的系統資源給該卡而完成 PC 與 PCI 卡間的溝通呢？[5]

1. 電腦開機後，系統軟體會逐一檢查插槽上 PCI 元件的 Vender ID，當發現某 PCI 插槽上的 Vender ID 不是 0XFFFF 表示該插槽上有 PCI 元件存在。
2. 當系統軟體偵測到 PCI 元件時，將讀取該 PCI 元件的 Configuration Register 以獲得資源需求資訊(Memory range, I/O range, Interrupt number)。
3. 系統自動組態軟體分配沒有衝突的資源給該 PCI 元件，並將所分配的 Memory, I/O, IRQ 等資源訊息填入 Configuration Register。
4. PCI 元件需依 Configuration Register 內的 Memory 或 I/O 資源分配資訊做解碼，讓 PCI 元件上的 Local Memory 對應到 PC 端分配到的系統資源(圖 3)。
5. PC 端應用程式欲控制 PCI 元件時必須先獲知控制該 PCI 元件的 Memory 或 I/O 之 base address 及 IRQ number 等資源分配資訊，而後透過 Memory 或 I/O Port 控制該 PCI 元件，並處理由該 PCI 元件所發出的中斷要求。資源分配資訊讀取的方式在 DOS 環境下是透過 PCI BIOS 讀取該 PCI 元件 Configuration Register 之資源清單；而在 Windows 作業系統下則是透過 Windows95/98/NT 之 Registry 讀取。



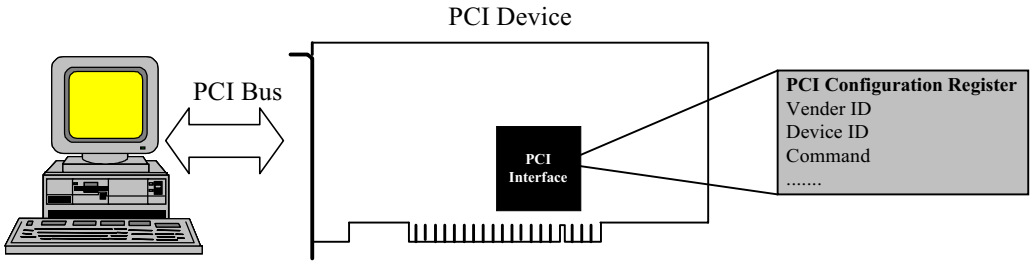

(圖 3 )系統資源 Mapping

# 六、開發實務

面對厚厚的 PCI 規格書，要經由詳讀全部的規格，並在不借助任何其它開發工具的條件下設計出符合 PCI 規格的 PCI device，勢必需要投入相當多的時間與精力，面對競爭激烈且變化快速的市場，這樣的做法往往緩不濟急。以下提出開發的實務以供參考：

■硬體開發

目前有多家 IC 製造公司開發了符合 PCI 規格的 PCI Bus 介面 IC，如 PLX 公司[8]開發的 PCI 9050 – 32 bit bus slave , PCI 9052 –32 bit bus slave (Direct ISA to PCI conversion) , PCI 9080 – 32 bit bus master/slave , PCI 9610 – 64 bit bus master/slave；AMCC 公司的 S5920—32bit bus slave, S5933—32bit bus master/slave 等，另外亦有些 FPGA 製造商提供 PCI IP Code(Intelligent Property Code )。這類介面 IC 與 IP Code 具有符合 PCI Local Bus 規格的 Configuration Register 及信號 timming。使用方式如圖 4 所示，用上述的 PCI 介面 IC 或自行設計含有 PCI IP 的可程式邏輯元件作爲 PCI 裝置透過 PCI Bus 與 PC 溝通的介面，如此可縮短產品硬體的開發時間。




(圖 4) PCI Interface

■軟體開發

由於 Plug & Play，PCI 擴充卡所需的系統資源並非由使用者指定，因此 PC 端 device driver 必須知道該卡所分配到的資源(Memory base address, I/O base address, IRQ number)，並且需有能力透過 Memory 或 I/O port 控制 PCI 擴充卡。在 DOS 作業系統上，使用 Memory 與 I/O 讀取的指令可輕易地控制 PCI Bus 上的擴充卡；而 Windows 作業系統由於掌控了系統資源，因此許多應用軟體沒有能力透過 Memory 或 I/O Port 直接控制 PC 擴充裝置，因此軟體開發人員除須熟悉作業系統架構外，尚需使用 Kernel Programming 或 DDK 等技術做底層驅動程式的開發，這往往造成程式軟體開發人員的一大負擔。目前有些軟體公司提供了 PCI 驅動程式開發軟體，如 Jungo 公司的 WinDriver 驅動程式開發軟體[9]，使用這類的驅動程式開發工具可讓 PCI 擴充卡軟體開發人員無須深入 Windows 底層即可讓應用軟體控制 PCI 擴充卡，可有效地縮短軟體開發時間。

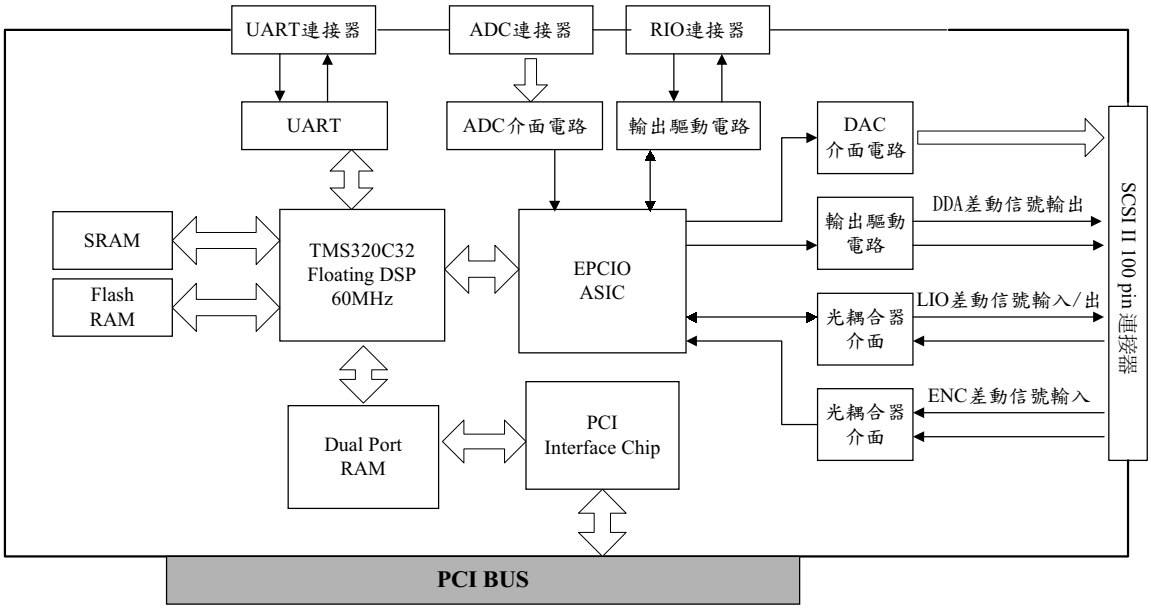
# 肆、PCI Bus 運動控制模組實例

目前國外已有許多廠商開發出 PCI 介面運動控制卡，在國內工研院機械所亦已開發出 PCI 介面 DSP 六軸運動控制卡(PMC32-6000)，PMC32-6000 以 32 位元浮點運算 DSP 晶片爲核心，並整合機械所開發完成之定位與 I/O 控制專用晶片（EPCIO ASIC），構成架構簡潔、功能強大的 PCI 介面六軸運動控制模組。由於 PMC32-6000 內含 DSP 運算晶片，一方面可獨立運作，形成一個嵌入式（Embedded）系統，另一方面，搭配 PC 或 IPC 構成雙 CPU 系統，PC 端負責運動控制命令的下達、資料的蒐集與分析、系統的監控、人機畫面的顯示，而 PMC32-6000 負責控制命令的解譯、軌跡的規劃、伺服控制的執行，如此的系統架構，運作更彈性且功能強大，並可解決現階段以 Microsoft Windows 爲主的作業系統缺乏強即時性(Hard Real-Time)的缺點。另外 PMC32-6000 應用 PCI 介面的優點，除可提高與 IPC 間資料傳輸之速度外，與 IPC 間溝通的初始化將由軟體自動設定，可減少使用者之負擔及因不當手動設定所造成的困擾。PMC32-6000 特色如表 5 所示。

(表 5 )PMC32-6000 特色

| | |
|---|---|
| •PCI介面，資源自動分配 | •OT±、Home等專用Motion IO |
| •6軸閉迴路伺服定位控制 | •內含即時多工核心 |
| •7組編碼器信號(Encoder)輸入 | •運動路徑規劃與補償計算 |
| •6組數位轉類比(DAC)輸出 | •具32位元DSP浮點運算晶片 |
| •8組類比轉數位(ADC)輸入 | •可單機(Stand alone)執行 |
| •128IN / 128OUT遠端串列傳輸 | •具RS232通信介面 |

PMC32-6000 Function Block 及信號流程如圖 5 所示，以 32 位元浮點運算 DSP 晶片(TMS320C32)爲核心，配合 DSP 執行時儲存程式與資料記憶體用的 *SRAM*、關機時儲存系統 firmware 及參數的 *Flash RAM*、與 PC 溝通的 *Dual Port RAM* 及全雙工串列溝通界面 *UART*，並且整合定位與 I/O 控制專用晶片 *EPCIO ASIC*，搭配相關介面電路成爲具 ADC、Remote I/O(RIO)、DAC、DDA、Local I/O(LIO)、Encoder(ENC)等控制單元的運動控制模組。在與 PC 溝通的介面則是透過符合 PCI 介面規格的 PCI Interface Chip 與 Dual Port RAM 作溝通，其中 PC 端與 PMC32-6000 溝通所需的資源由 PC 端資源管理軟體自動分配，具自動組態功能。[10]




(圖 5 ) PMC32-6000 功能方塊圖

PMC32-6000具即插即用功能，使用上當第一次將PMC32-6000置入PCI Bus插槽中並開機後，如圖6所示，Windows作業系統將自動偵測到一新的PCI元件加入系統中，使用者只需依指示之步驟操作即可完成安裝程序。安裝完成後使用者可從以下路徑查看PMC32-6000的資源分配：控制台->系統->裝置管理員。例如在圖7所顯示的例子中，目前系統配置給PMC32-6000的資源是包含中斷信號IRQ11、16K byte的記憶體位址空間(Memory 0xC8000 ~ 0xCBFFF)、128個I/O位址空間(I/O 0xE800 ~0xE87F)，並且這些資源不會與其他的裝置相衝突。

在運動控制系統的使用上，PMC32-6000 提供由 PC 端直接呼叫的運動控制函式庫 PMC32-MCCL (**M**otion **C**ontrol **C** **L**ibrary)，其功能包括絕對與相對座標值設定與讀取、直線/圓/圓弧補間插值、點對點同動/不同動補間插值、連續/單步/脈衝形式 JOG 功能、運動暫停/持續/棄置/延遲功能、運動暫停/持續/棄置/延遲功能、直線/圓弧/圓的進給速度設定等。[11]


(圖6 ) PCI Plug & Play


(圖 7 ) 資源分配實例

# 伍、結語

PC Based 運動控制模組是以電腦作爲系統操作平台，若能夠有效發揮電腦的優勢將有助於提昇競爭力，PC 介面對 PC Based 運動控制模組而言雖不屬於核心技術，但卻足以影響整體的效能。

PCI Bus 彌補了 ISA Bus 速度慢且系統資源無法統合的缺點，目前多數的桌上型電腦擴充設備如網路卡、音效卡、VGA 卡等都已採用 PCI 介面。在運動控制領域由於對 PC 介面傳輸速度的提昇不若多媒體設備那麼來的迫切，加上產業界對產品使用的熟悉性與延續性之需求，因此 ISA 介面運動控制模組並不會很快的退出市場，但須注意的是，在 PC99 的規範中已屏除 ISA 規格，從目前市售主機板具有六個 PCI 插槽而只剩一個 ISA 插槽可知，ISA 擴充槽將很快的從市售的桌上型電腦中功成身退，對慣於採用桌上型電腦作爲運動控制系統平台的使用者而言應有所因應。

# 參考資料


[1] PCI/Compact PCI 技術研討會講義 :工研究院機械所(1999)

[2] PCI Special Interest Group (PCI SIG) (www.pcisig.com)

[3] PCI Local Bus Specification (Rev 2.2) , PCI SIG

[4] PCI BIOS Specification (Rev 2.1) , PCI SIG

[5] Plug and Play System Architecture / Tom Shanley (MindShare, Inc. 1995)

[6] PCI system architecture / Tom Shanley, Don Anderson (Harlow : Addison-Wesley, 1999.)

[7] PCI hardware and software : architecture and design /Edward Solari and George Willse (San Diego, CA : Annabooks, c1998)

[8] PLX co. , www.plxtech.com國內代理商：立治科技

[9] Jungo co. , www.jungo.com ; 國內代理商：興德資訊 www.sinter.com.tw

[10] PMC32-600/6000 使用手冊, 工研院機械所控制器發展部 (www.epcio.com.tw) (www.mirl.itri.org.tw/rd/automation/integration/industrial)

[11] PMC32-MCCL 函式庫使用手冊, 工研院機械所控制器發展部

[12] PC99 System Design Guide Version1.0 , http://developer.intel.com/design/desguide/