非同步傳輸網路卡的設計與製作

指導老師：侯廷昭

參賽隊員：邱孟希　楊順欽　張大偉　詹嘉隆

國立中正大學電機工程學系

摘要：

在以碼格(Cell)為基礎的 ATM (Asynchronous Transfer Mode)網路環境中，欲傳送或接收的資料必須快速的處理以符合高速需求。因此無法用軟體作 ATM 層協定的處理。我們發展出一個用於 PC 上，具有 TAXI 介面的 ATM 網路卡 (Network Interface Card)，在這個網路卡上，我們設計兩顆 Actel FPGA (Field Programmable Gate Array) 以實現 ATM/AAL5 的功能，並利用並行 CRC (Parallel Cyclic Redundancy Code) 的觀念來設計 HEC (Header Error Control) 及 CRC-32，以利用較低的 FPGA 系統時脈支援 100Mbps 的傳輸需求。

我們還設計第三顆 FPGA 作為 PCI Bus Controller，以充份利用 PCI Bus 的高速頻寬。這個網路卡的驅動程式是參考 Packet Driver 修改而成。因此所有 Packet Driver 支援的應用程式均可使用此網路卡。我們並在 Microsoft Windows95 環境上作了視訊點播的應用以証明其泛用性，而利用此 ATM 卡可建立日後其他有關 ATM 研究的測試平台。

關鍵詞: 非同步傳輸模式、網路卡、FPGA、PCI、Packet Driver、並行 CRC。

1. 簡介

1.1 研究動機

近幾年來高速網路的技術方興未艾，而 ATM 技術由於使用固定長度的碼格(Cell)為處理單位，並且採用交換 (Switching) 的技術，因此成為適合應用於廣域網路與區域網路的高速網路技術。高速 ATM 網路的實現不僅需要所有相關元件及子系統都能符合高速運作的要求 (100Mbps 以上)，其較高的頻寬延遲乘積 (Bandwidth-Delay Product)的特徵更使得傳統的流量控制 (Flow Control) 及阻塞控制 (Congestion Control) 的機制不再能有效的解決網路塞車的問題。因此這幾年來，有關 ATM 的研究，不論是速度或容量的提昇，還是有效的控制機制等，都是世界各地研究人員的努力目標。

我們中正大學電機系的老師學生們在多年前即已開始從事 ATM 網路的研究。有感於論文分析的研究不能滿足我們對於 ATM 高速網路的認知，因此決定投入 ATM 實作。我們的目標是要能對系統軟硬體技術有足夠的掌握，以實驗不同的控制機制，並尋求軟硬體整合後的系統速度的極限。

要架構一個基本的 ATM 網路系統，除了 ATM 交換機 (Switch) 之外，各個 Host (如 PC，工作站) 需要配備 ATM 網路卡。我們購買過高階的工作站 ATM 網路卡，及較便宜的適合 PC 使用的 ATM 網路卡。前者一般在使用上沒有問題，但價格較貴，後者則多半無法正常的運作。但無論如何，兩者皆提供有限的控制功能，且無法讓使用者作 System Level 的修改，任何修改都經由 API (Application Programming Interface)進入。

由於以上種種因素，為了能建立一個具有各種實驗能力的 ATM 測試平台，我們決定自行製作 ATM 網路卡，內容包含晶片和電路板的設計，驅動程式的撰寫，以及最後的系統整合。

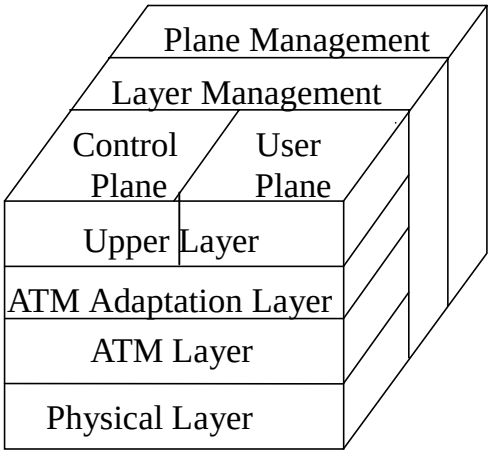
1.2 背景資訊

在 ATM 網路環境中，基本的傳送單位是 53 位元組的碼格 （Cell），與 Ethernet 及 FDDI 以訊框 (Frame)為單位不同，以 Cell 為單位的好處可大大簡化 Switch 的設計，並使端到端 (End to End)的延遲時間大幅降低。

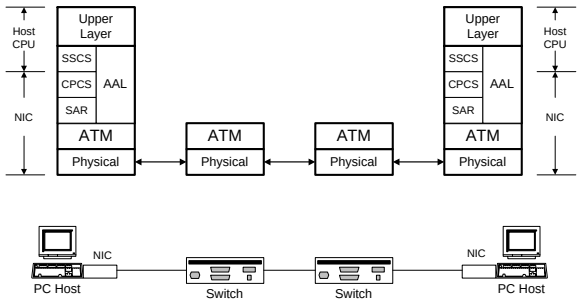
ATM 的相關協定最先於寬頻整合服務數位網路（Broadband Integrated Service Digital Network）的規格內訂定。類似於 ISO 的七層協定參考模型（Protocol Reference Model），ITU-T 也制定了寬頻整合服務數位網路的參考模型，規範了在 B-ISDN/ATM 網路上相關協定及功能的關係位置。如圖一所示，B-ISDN 的協定參考模型分為三個 Plane：User Plane、Control Plane 以及 Management Plane。User Plane 提供了使用者應用資料的傳送，

Control Plane 主要在處理連線的建立和解除，Management Plane 提供管理的功能，包含 Layer Management 以及 Plane Management。Layer Management 負責 Layer 有關的管理而 Plane Management 負責整個系統的管理及協調工作。Control Plane 與 User Plane 共用同樣的實体層（Physical Layer）和 ATM 層（ATM Layer），以及部份的 AAL 層（ATM Adaptation Layer）。實体層主要的功能是 Bit Stream 的傳送和接收，包含 Bit Timing， Cell Rate Decoupling，Cell Delineation 等功能。ATM 層的功能包括 Header 的產生及取出，VPI/VCI 的轉換，Cell Multiplex/Demultiplex 及 Generic Flow Control。AAL 層可再區分為 CS（Convergence Sublayer）以及 SAR （Segmentation And Reassembly Sublayer） 兩個子層。CS 子層是上層（Upper layer）與 ATM 層的轉換介面，CS 子層又可再區分為 SSCS （Service Specific Convergence Sublayer）及 CPCS （Common Part Convergence Sublayer）兩個次子層。SAR 子層專門處理切割上層的資料成為 ATM cell 或把 ATM cell 重組成上層的資料單位。在這個網路卡我們著眼在 User Plane 及部份 Control Plane 的實現。（即使沒有 Management Plane 的功能，這個網路卡仍舊可以建立連線並傳送資料。）

在我們的設計中，為了避免 Host CPU 因做了太多協定處理而成為高速網路傳輸中之瓶頸，將一些較固定、簡單，且 Load 重的協定處理（如 AAL，ATM Layer Cell-based 的協定處理）交由硬體來處理，所以我們在網路卡上實現了 AAL CPCS，SAR，ATM 和實体層的功能， AAL SSCS 及上層的功能交由 Host CPU 去處理。這樣的設計增加硬体的難度，但簡化 Host 軟体（如 Device Driver）的負擔和複雜度。



圖一 B-ISDN Protocol Reference Model



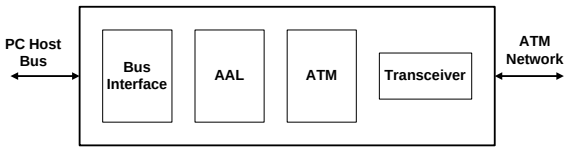
圖二 ATM 網路元件及其相關協定

圖二所示，網路卡上執行實体層，ATM 層和 AAL/SAR CPCS 的功能，PC CPU 執行 AAL SSCS 及其上層的軟体，ATM 交換機則執行實体層和 ATM 層功能。如此一個基本的 ATM 網路便可運作。

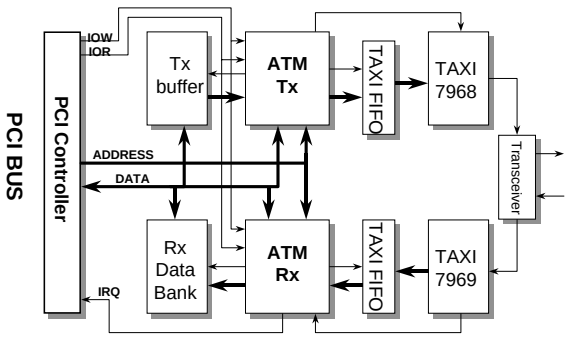
1.3 網路卡架構

依據前一節的介紹，一個 ATM 網路卡上應包含下列幾個模組（如圖三所示）：一個 Transceiver 執行實体層功能，一個 ATM 模組執行 ATM 層功能，一個 AAL 模組執行 SAR 及 CPCS 的功能，及一個 Bus Interface 模組作為與 PC 主機相連的介面。

圖三 ATM 網路卡模組



Transceiver 由於技術層次較高，我們採用現成的元件。ATM 和 AAL 我們自行設計，以 FPGA 實現。但由於我們所使用的 FPGA 晶片有 Gate Count（4K）的限制，我們將 Transmit 部份的 ATM/AAL 作成一顆晶片，Receive 的 ATM/AAL 放在另一顆晶片，這樣兩顆晶片的使用率較為均衡。最後再以第三顆晶片實現 PC Bus Interface。圖四為這塊網路卡的硬体架構圖。藉由 PCI Controller，此網路卡經由 PCI Bus 與 PC 主機溝通。ATM_Tx 為執行傳送端 ATM/AAL 功能的 FPGA 晶片，ATM_Rx 為執行接收端 ATM/AAL 功能的 FPGA 晶片。



圖四　ATM 網路卡硬体架構

Tx_Buffer 採用 IDT7200*2 是用來調整 PCI Bus 和網路卡內部運作速度的差異。Rx_DataBank 採用 IDT7203*6，共分為三個 Bank，用來暫存重組中的 CPCS-PDU （Protocol Data Unit）。Transceiver 是採用 100Mbps TAXI 的光電元件，需配合兩顆 TAXI 晶片（AM7968/AM7969）來運作。在 ATM/AAL FPGA 與 TAXI 晶片組之間有 TAXI-FIFO 來調整網路卡內部速度與 TAXI 速度的差異。在 ATM/AAL FPGA 內亦包含 TAXI Controller 的電路以控制 TAXI-FIFO 與 TAXI 晶片組（AM7968/AMT7969）之間的 Timing。

2. 設計考量與問題解決方案

圖四的 ATM 網路卡架構是我們考慮下列幾個設計因素後決定的。

- ATM cell 如何經由 TAXI 晶片組傳送與接收?

  根據 ATM Forum UNI 規格，每一個 Cell 在 100Mbps TAXI 傳輸介質上傳送是把 Cell 的 53 個 Symbol pairs 跟在一個 Start_of_cell Symbol（TT）之後，而且他們必須是連續傳送，不能有空隔。為了滿足這項要求，我們放了 TAIX-FIFO 在 ATM/AAL 晶片之間，以調整兩者間速度的差異。這兩者之間的同步是靠著 TAXI-Controller 來達成。

- 我們是否需要在網路卡上擺一個微處理器?

  微處理器的角色主要在提供 Control 和 Management 的功能。是否需要微處理機取決於成本與功能的孰輕孰重。因為我們現階段所著重的是 User Plane 的功能，而 ATM/AAL 的 FPGA 晶片已能充份達成我們所要求的功能，所以我們不採用微處理器以降低成本及設計複雜度。

- 我們應採用何種緩衝記憶体?

  在我們的設計裏，傳送端的緩衝器（Tx_Buffer）與接收端的資料暫存器（Rx_DataBank）是分開的，以提昇系統效能。在傳送端，Tx_Buffer 的容量不需很大，我們採用 FIFO 的緩衝記憶体。這樣的選擇允許新資料在搬進網路卡 Tx_Buffer 的同時，在 Tx_Buffer 的舊資料也能搬至 ATM_Tx 做處理。

  在接收端，重組中的 CS PDU 暫時存放在 Rx_Data Bank 直至完全重組完畢。因此 Rx_DataBank 的容量需求較大，以容許多個 CS-PDU 同時被重組。為了簡化設計，我們採用半動態式（Semi-dynamic）的 Bank 結構來設計 Rx_DataBank。在 ATM_Rx 內，我們設計了一個 Content Addressable Memory（CAM）來管理 Rx_Data Bank。

- 我們如何產生 ATM cell 的 Header?

  傳送端 ATM Header（不含 HEC 及 PT 內的 *more* Bit）的產生，是在 CS PDU 開始傳送時，由驅動程式先將相關資料寫入 ATM_Tx 之內，之後每一個 Cell 的 Header 可由 ATM_Tx 自動產生。由於 ATM_Tx 內沒有類似 ATM_Rx 內的 CAM，因此 ATM_Tx 在處理完一個 CS PDU 後才能處理下一個 CS PDU，不能做 Cell Interleaving。

- AAL 的功能如何實現?

  ITU-T 推薦了５種型態的 AAL 協定，AAL1，AAL2，AAL3/4，以及 AAL5。AAL1 及 AAL2 針對需要嚴格時間限制的應用，AAL3/4 用來支援變化速率及無時間限制的應用，但 AAL3/4 的 Overhead 相當可觀，而為了簡化 AAL3/4 的功能，ATM Forum 及 ITU 提出了簡單而有效的 AAL5 協定。因此我們選擇 ATM/AAL5 協定來實現高速的通訊界面。AAL5 的協定中，0-47 位元組的填充（Padding）以及 CRC-32 的計算是最重要的動作。在傳送端，當上層要求 AAL5 的服務時，網路卡需計算 PAD 長度以組裝一個 CS-PDU 並將其切割為多個 48 位元組的 SAR-PDU。在我們的網路卡，PAD 的計算是瞬時產生的（Calculated On-the-fly）。在接收端，因為不同的 CS-PDU 的 cell 交錯進入網路卡，CRC-32 的計算需不斷的在不同的 CS-PDU 間交錯進行。利用一組 CRC-32 的電路，我們可以分時重組不同的 CS-PDU，但任何未完成的 CRC-32 的計算必須先暫存在 CAM 裏。

3.設計原理

我們所設計的三顆FPGA均是依照相關規格來設計。

PCI Controller 晶片的設計遵循 PCI Bus 的規格：

• PCI Special Interest Group, PCI Local Bus Specification.

ATM_TX 和 ATM_Rx 的設計遵循下列規格文件：

• ITU-T I.361 B-ISDN ATM Layer Specification
• ITU-T I.363 B-ISDN ATM Adaptation Layer Specification
• ATM Forum, ATM User-Network Interface Specification.

TAXI chipset的使用則參考

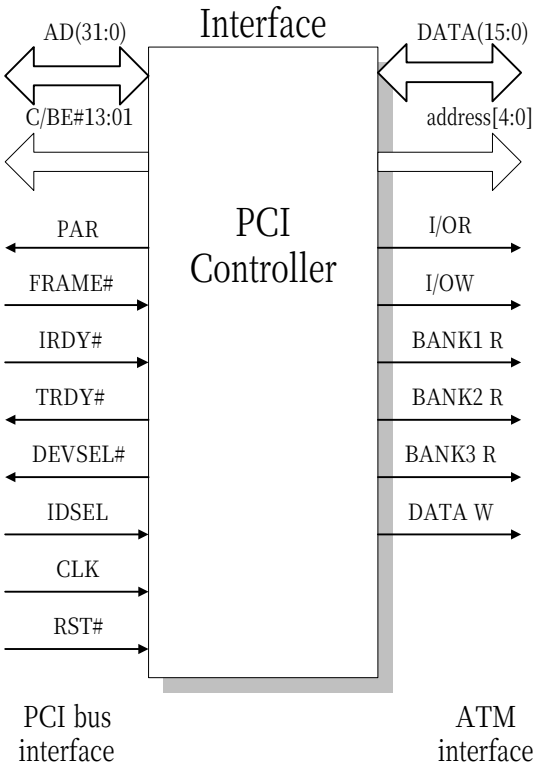
• Advanced Micro Devices, Inc. AM7968/AM7969 125 TAXI chip Integrated Circuits Data Sheet and Technical Manual.

在 AAL5 的格式中, 定義了 CRC-32 來偵測 AAL5 的CPCS-PDU是否完全正確，而每一個碼格的 Header 也定義了一個位元組具一個位元改正能力的 HEC (Header Error Control)，想要以串列方式作 Checksum Generation 及Checking 必須在 FPGA 上以 100Mbps 以上的時脈運作才行，為了解決這個問題，原本已經有人設計了並行 CRC 的演算法，但是有其限制，我們採用本系陳景章教授及博士班學生張慶龍共同研究成功的並行 CRC 演算法，此演算法業已獲得中華民國及美國專利，用這些方法的好處是若以 8 個位元的並行處理的話， FPGA 的時脈可降至原來的 1/8，這使得本網路卡的時脈能低至 12.5MHz 得以正常運作。

4. 軟硬体說明

4.1 PCI Controller FPGA

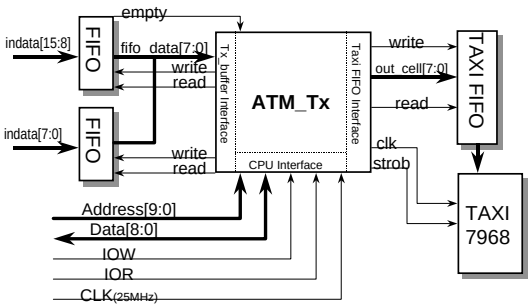
PCI Bus 是在 1992 年由 Intel 公司提出的一種 Local Bus，因工作時脈達 33MHz，Bus 寬度為 32 位元，因此為速度快，且具與 CPU 無關之跨平台特性，一推出就廣為業界採用，但 PCI Controller 比 ISA Controller 複雜許多，也因此，必須再以一顆 FPGA 充當 PCI Bus 及 ATM Card 的界面，如圖五為 PCI Controller 之 Interface，區分為與 PCI Bus 連接之 PCI Bus Interface 及與 ATM Card 上之元件相連接之 ATM Interface， PCI Controller 需要將 PCI Bus 之 Timing 轉換成 ATM 界面的 Timing，在設計時需注意因 33MHz 之高頻造成的誤動作。



圖五 PCI Controller Interface

4.2 ATM_Tx FPGA

此顆 FPGA 設計重點在於滿足 1TU-T I.361 與I.363 所建議之功能，採用 Actel 2為設計工具，其所提供之介面如圖六所示，圖七為其硬體架構。



圖六 ATM_Tx Interface

介面說明

1.Tx_Buffer 介面：因為傳送端之緩衝器為 FIFO，所以此 Tx_Buffer 介面即滿足 FIFO 介面。
2.CPU 介面：提供 Host CPU 來設定與讀取此 FPGA 之內部暫存器，且為了減少整個網路卡所

需之 TTL 元件，整個板子的解碼電路亦由此顆 IC 所提供。

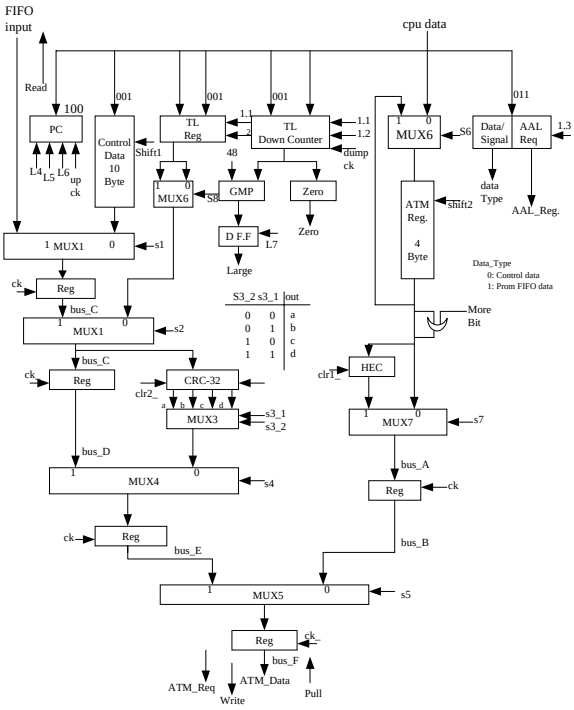
3.Physical介面：因採用 FDDI 之 TAXI 為實體層，其與 ATM_Tx 間乃以 FIFO 來緩衝，所以此 Physical 介面即為滿足 FIFO 的介面, 且此介面亦提供 TAXI-FIFO 與 TAXI7968 間的控制信號。

提供之功能

1.AAL5 部份滿足 ITU-T 之建議，其功能包括：
- －Non-assured Data Transfer
- －Pipeline Operation
- －0-47 Octets Padding
- －32bit CRC Generation
- －Adding 8 Octets CPCS-PDU Trailer
- －SAR Sublayer Functions

2.提供 ATM 層的功能有
- －8bit HEC 的產生
- －*more* Bit 的設定
- －組成 53 Octets 的 ATM Cell

3.提供 TAXI 的 Controller



圖七 ATM _Tx 硬體架構

測試條件

1.在動作過程中 Tx_Buffer 有 Empty 狀況時，此表示上層資料傳送速度比 ATM_Tx 處理速度慢時，則此顆 ATM_Tx 必需 Halt 住，直到 Tx_Buffer 變成Nonempty。

2.Empty Signal Random Time 的由High 變Low。

3.CPCS-PDU Length=88 bytes:此為 CS-PDU 無 Padding 的情況。

4.CPCS-PDU Length=89 Bytes :此為 Padding 分佈在 2 個 Cell 內。

5.CPCS-PDU Length=97 Bytes:此為 Padding 僅在最後一個 Cell 內。

6.CPCS-PDU Length=2 Bytes。

7.CPCS-PDU Length=256 Bytes。

8.CPCS-PDU Length=1024 Bytes。

9.CPCS-PDU Length=2048 Bytes。

10.CPCS-PDU Length=4096 Bytes。

ATM_Tx Chip使用方法

此 Chip 提供下列 Register 供 Host 設定。
- －ATM-Header Register
- －CPCS-PDU_length_low_byte Register
- －CPCS-PDU_length_high_byte Register
- －ATM_Tx_start Register

當 Host 有 Data 要透過網路卡傳送時，需對 ATM_Tx 做下列設定:
- －此 CPCS-PDU 的相對應 VPI/VC1: 將 ATM Header 前 4 個 Byte (不包含 *more* Bit)連續寫入 ATM_Header Register 之 Address。
- －此 CPCS-PDU 的 Length 需為偶數: 將 CPCS-PDU Length 的 Low Byte 寫入 CPCS-PDU_length_low_byte Register 之 Address，High Byte 寫入 CPCS-PDU_length_high_byte Register 之 Address。
- －ATM_Tx 的啟動: 當設定完畢後, 則可透過寫入 ATM_Tx_start Register 之Address值為 0x01，而啟動 ATM_Tx 開始運作。

當上述設定完畢後，即可將欲傳送之 CPCS-PDU 搬到 Tx_Buffer 內。

4.3 ATM_Rx FPGA 之設計

此顆 FPGA 設計，採用 Actel 2 為設計工具，其所提供之介面如圖八所示，圖九為其硬體架構。以下就對此 FPGA 的介面，及提供之功能做一介紹。

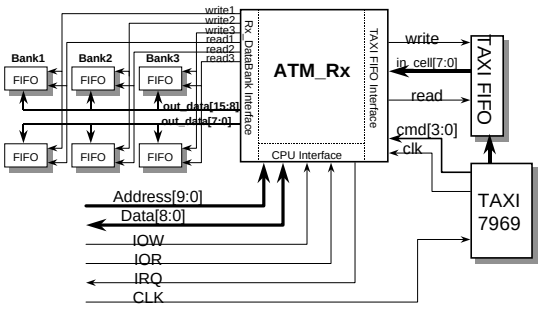
介面說明

1.TAXI介面：當 CMD<3:0> 值為 2 時，表示 TAXI 開始將收到的 Cell 寫入 TAXI_FIFO，當整個 cell 完全寫入時, 將發出 Indication 的訊號通知 ATM_Rx將 Cell 讀走。

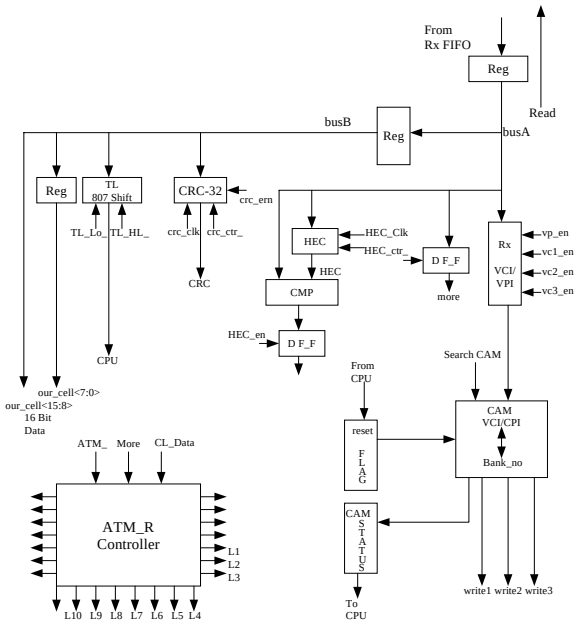
2.CPU 介面：當 ATM_Rx 將整個 CS-PDU 重組結束時，將送出 Interrupt 的訊號通知 Host CPU 將整個 CS-PDU自 Rx_DataBank 搬離，附帶的 ATM_Rx 也提供目前的 Status (包括 Bank

Number 與 CRC Check 結果) 及 CS-PDU Length 告知 CPU。

3.Rx_dataBank 介面：ATM_Rx 將重組的 Cell 寫入Rx_DtaBank，所以此介面也提供 reset， bank number 及 write line。



圖八 ATM_Rx Interface



圖九 ATM_Rx 硬體架構

提供之功能

1.CPCS Functions
- －32 Bits CRC 檢查
- －從 CPCS-PDU中抽出AAL-SDU

2.SAR Sublayer Function

3.ATM Layer Functions
- －8-bit HEC檢查
- －從 ATM Cell中抽出 SAR-PDU

4.TAXI 介面

5.CPU 介面
- －AAL-SDU Length
- －Status Report
- －Reassembly Buffer Bank Number
- －Reassembly Buffer Bank Release

4.4 TAXI 晶片組

TAXI 是此介面卡之 Physical Layer，做為介面卡間之資料傳送工作。而TAXI到 ATM_Tx 及 ATM_Rx 間均有個 FIFO，而 FIFO 和 TAXI 間之資料傳送須有個 Controller 做為其間的資料流程控制，使資料的傳送能夠運作正常。在最後 TAXI 利用一個光電元件，用光纖連接至 ATM Switch。

以下我們就，TAXI 晶片及 TAXI 與 FIFO 間之 Controller，作更進一步介紹。

TAXI Chip Set

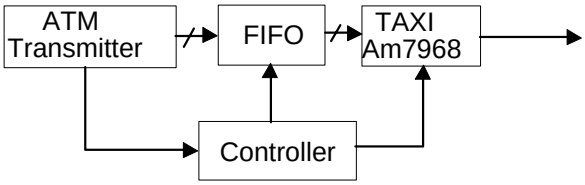
TAXI 共有一對晶片，為 AMD 所製作之 Am7968 及 Am7969 ，Am7968 其主要功能是將並列之資料輸入轉為串列之資料輸出，而 Am7969 是將 Am7968 所輸出資料接收進來，並將之再轉換為並列之資料輸出。

這對晶片有下列的幾點特性：

1. 其 DATA 及 COMMAND 資料可以有三種組態方式
   - ●八條 DATA 線和 四條 COMMAND 線
   - ●九條 DATA 線和 三條 COMMAND 線
   - ●十條 DATA 線和 二條 COMMAND 線
2. 自動多工／解多工 DATA 及 COMMAND
3. 利用 STROB 及 ACK 這兩條線可以做資料的非同步輸入
4. 可以直接驅動 Coaxial Cable 或 Twisted Pair，其並提供一個簡易的介面去接上光電元件。

TAXI <-> FIFO Controller

Transmitter 方的 Controller

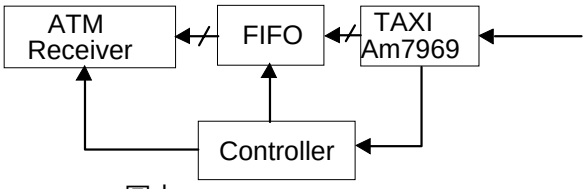


圖十 TAXI Transmitter Controller

當 ATM Transmitter 將一個完整之 ATM Cell 送入 FIFO 後，它便會送個訊號給

Controller ，Controller 收到此一訊號之後便開始其工作，也就是送 *read* 訊號給 FIFO 叫 FIFO 送 DATA 出來給 TAXI 並送 STROB 訊號給 TAXI，叫 TAXI 將由 FIFO 過來的資料轉換送出。

Receiver方的 Controller:



圖十一 TAXI Receiver Controller

當 TAXI 收到資料後，送 STROB 給 Controller，然後由 Controller 送 *write* 訊號給 FIFO，將由 TAXI 收入之資料寫入 FIFO，當 Controller 計數53後，即已收入一完整之ATM Cell後，便由 Controller 送出一個訊號去通知 ATM Receiver，促使其去 FIFO 讀取資料做處理。

4.5 網路卡軟體驅動程式

由 FTP 軟體公司所制定的 Packet Driver 介面是在 Data Link Layer，它提供了一個標準的介面，使得網路應用程式可以與實際的網路獨立，達到Hardware-Independent，使應用程式更具可攜性。

當 Physical Layer 更動時，上層 API (Application Programming Interface) 根本無須更動，我們只須更換一個新的 Packet Driver。 Packet Driver 提供的 Function 依其種類可分為：

1. 基本型：如 driver_info ()，send_pkt ()，terminate () 等。
2. 延伸型：其增加了一些較不常用的介面如，set-rcv-mode( )，set-multicast-list( )等。
3. 高性能型：如 as_send_pkt ()，drop_pkt () 等。

Packet Driver利用三個整數來定義一個網路介面：<class,type,number>

Class: IEEE 802.3 Ethernet, Pro NET...

Type: 3C503,NE2000，....

Number: 0: 1st Network Interface

1: 2nd Network Interface

Packet Driver提供下列的Function 給上層的API來使用。

driver_info()
access_type()
release_type()
send_pkt()
terminate()
get_address()
reset_interface()
+get_parameters()
+as_send_pkt()
+drop_pkt()
*set_rcv_mode()
*get_rcv_mode()
*set_multicast_list()
*get_multicast_list()
*get_statistics()
*set_address()
*send_raw_bytes()
*flush_raw_bytes()
*fefch_raw_bytes()
*signal()
*get_structure()

*表示 Extended Packet Driver Functions

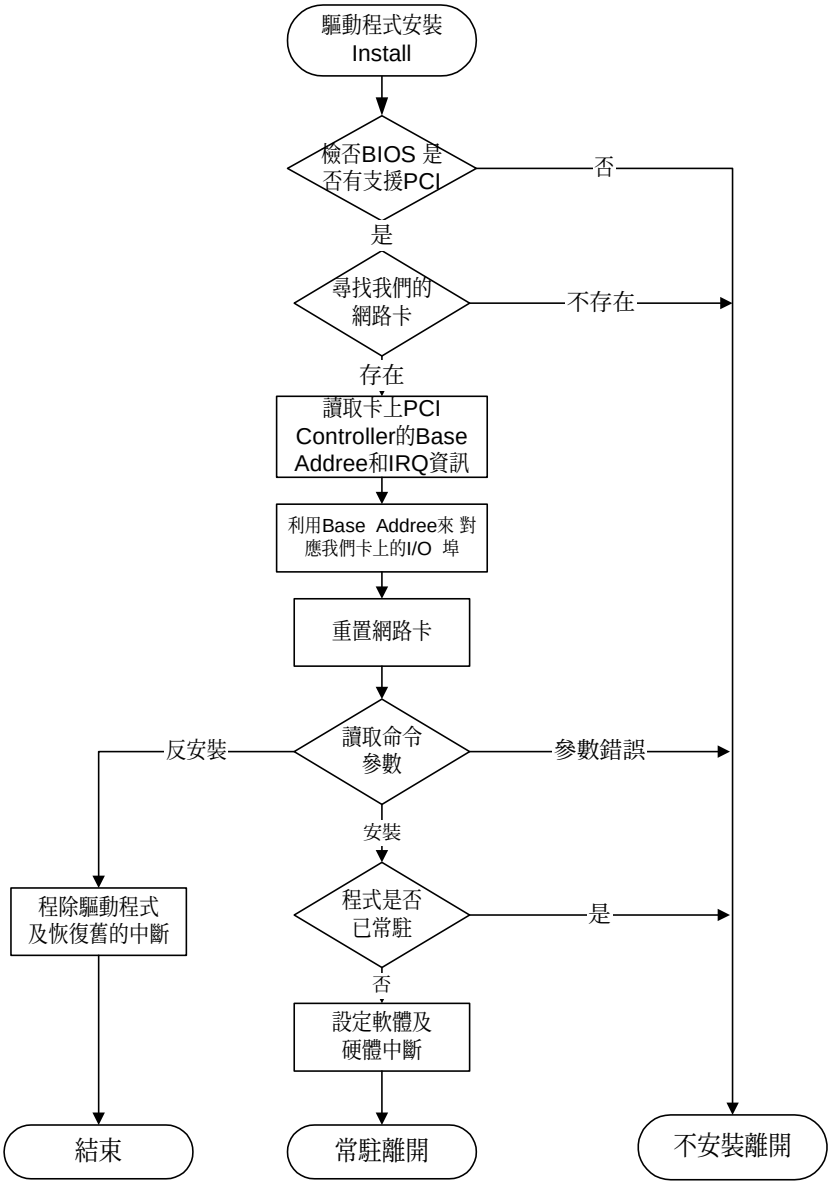
+ 表示 High-performance Packet Driver Functions

為了使ATM 網路卡能使用上層的 API，我們只須將 Packet Driver 修改使其符合 ATM card，同時提供與 FTP software 所規定出的標準介面。目前我們在 DOS， Microsoft Windows3.1 及 Windows95 的環境上已作了一些應用以証明其通用性，而利用此 ATM 卡更可當作日後其他有關研究的測試平台。

Driver流程圖及原始程式磁片說明

以下為 PCI ATM NIC 的 Device Driver 流程圖說明，包括了程式主檔，主要做 PCI Controller Initialization 以及軟體中斷向量的設定（原始程式 pciatm.asm），Transmission Routine，用來做傳送的控制程式(原始程式 pcitx.asm），以及Receive Routine，用來做為接收的控制程式（原始程式 pcirx.asm）。

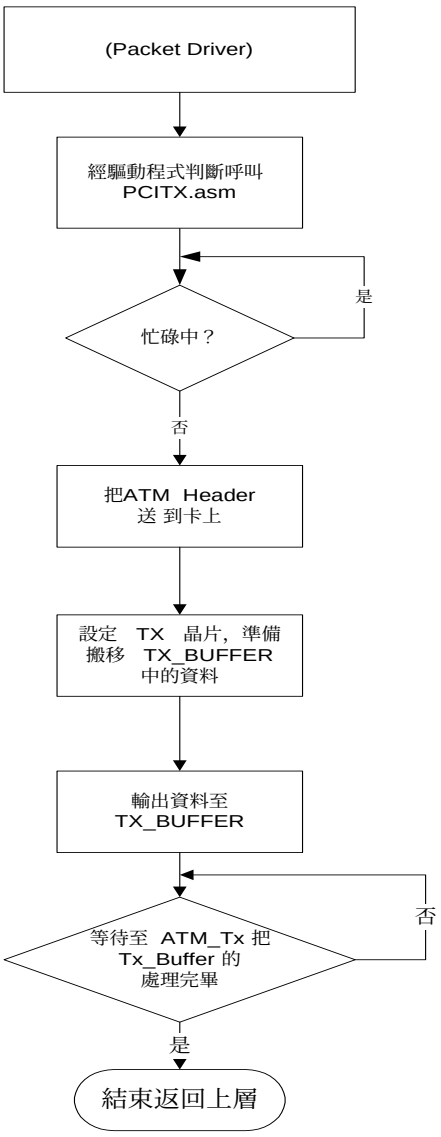
圖十二是我們 Driver 在 Install 時的初始化流程，在使用者下安裝指令之後，首先，會先判斷這個主機板之 BIOS 是否有支援 PCI Device 功能，我們才可利用 BIOS 中斷服務來尋找我們的網路卡是否有存在，如果這個 BIOS 有支援 PCI 的話，在開機時，就會自動分配各個 PCI Device 資源，再寫入各 Device 之 PCI Controller，所以在找到網路卡之後，讀取 BIOS 分配的 I/O Base Address 和 IRQ，由 Base Address 可知道我們所要求 I/O Port 的位址，即可對卡上的 I/O Port 進行讀寫。在重置網路卡之後，我們再檢查所下命令的參數，如果是安裝時，會先看是否本身已經常駐，如果尚未常駐則在設定軟體和硬體中斷服務程式後常駐離開；如果是反安裝的選項，則將驅動程式從記憶體移除，並恢復舊的中斷服務程式。



圖十二 PCI ATM NIC Driver 初始化流程

圖十三是我們網路卡在傳送時的控制流程，上層的應用程式利用 軟體中斷(Software Interrupt)，來呼叫 Packet Driver 的中斷服務程式，來傳送資料，驅動程式利用 AH 暫存器來判斷其為傳送要求，再呼叫副函式 pcitx。在開始傳送之前，先檢查網路卡是否在忙碌，有尚未送完之資料，直到卡送完前一筆資料，我們才可繼續傳送資料。在傳送一筆資料之前，我們要把這一資料所用ATM Header 送到卡上暫存器，然後設定 ATM_Tx 晶片，此時 TX_Buffer 一端由 Driver 搬進資料，另一端由 ATM_Tx 把資料搬出，並且把資料加 ATM Header，切成 53 Bytes 的Cell，送至TAXI界面，再由TAXI送至網路，等到資料完全由 ATM_Tx 送出之後，這個處理程序才算完成，再返回上層應用程式。

圖十四為在網路卡資料接收時的流程，在網路卡上，我們有三個Bank可組裝三筆資料，當網路卡接收到資料時，在一筆資料完全接收完成之後，會發出硬體中斷 (Hardware Interrupt)，呼叫Driver硬體中斷服務程式來接收資料，要接收資料，我們需要先知道是那一個Bank的資料組裝完成，在知道了那一個Bank之後，再讀取其資料長度，並向上層應用程式要求Buffer，把資料搬到Buffer，再看是否有其他Bank有組裝完成的資料，重覆前幾個步驟，直到把所有Bank的資料都接收完成，再返回上層，由上層處理這些資料。



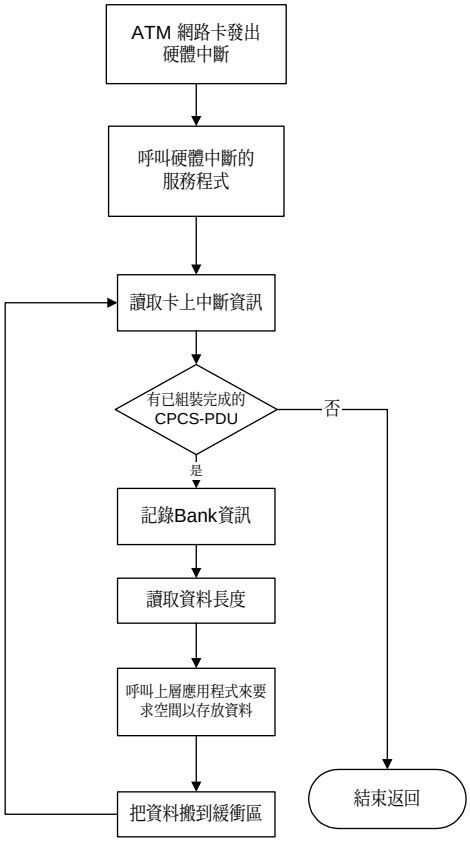
圖十三 PCI ATM NIC 傳送控制流程

4.6 網路卡內部動作流程

此 PCI ATM NIC 的細部 Operation Flow 詳述如下:

A.傳送端的資料與控制流程

當 PC 平台有資料要透過網路上傳送時，首先 PC 需先對 ATM_Tx 作設定，其包括此 CPCS-PDU 之 Length，相對應之 VPI/VCI，而後啟動 ATM_Tx。當 PC 啟動 ATM_Tx 後才將欲傳送之 CPCS-PDU 搬到 Transmission Buffer (Tx_Buffer)。當 ATM_Tx 發現啟動 Register 被設定後，則 ATM_Tx 根據被設定的 VPI/VCI，CPCS-PDU Length 來做 AAL5/ATM Layer Transmission 的協定處理，其處理方式，符合 ITU 之 I.363 之建議。



圖十四 PCI ATM NIC 接收流程

ATM_Tx 採用 Pipeline Operation 設計 (Host 一邊將資料搬到 Tx_Buffer，ATM_Tx 一邊將 Tx_Buffer 內的資料 Segment 出去），當 Tx_Buffer Empty 的時候，ATM_Tx 會發現 Tx_Buffer Empty，則 ATM_Tx 會 Halt 住直到 Tx_Buffer 再有資料才會繼續往下做，而此 Tx_Buffer 由 Empty 變成 Nonempty 與由 Nonempty 變成 Empty 會在不定時間發生。

ATM_Tx Segment 的 ATM Cell 乃先放入 Tx_TAXI_FIFO 內，當 Tx_TAXI_Controller 偵測到 Tx_TAXI_FIFO 內已有一個 Cell 時，則以 100Mbps 的 Rate 將 Tx_TAXI_FIFO 內的 Cell 送給 TAXI Chip。其中 Tx_TAXI_Controller 主要負責ATM_Tx，Tx_TAXI_FIFO 與 Tx_TAXI Chip 的 Timing 與協定處理。

B.接收端的資料與控制流程

當Rx_TAXI Chip 收到一個 Cell 時，則將此 Cell 放入 Rx_TAXI_FIFO。當 Rx_TAXI_Controller 確定 Rx_TAXI_FIFO 內已有一完整的 ATM Cell 時則通知 ATM_Rx 去接收一 ATM Cell。

當 ATM_Rx 開始接收一 ATM Cell 時，先計算此 Cell 之 HEC 是否正確，若錯誤且內部的 CAM 亦無記錄，則 Discard 此 Cell。此 ATM_Rx 內有一 CAM，其有三個 Entry，每個 Entry 管理一個 Receiver Bank，主要記錄有:此 Bank的狀態，Intermediate CRC_32值，接收完整的 CPCS-PDU Length。現階段的 Reassembly Buffer 的管理乃是以 Semi-dynamic Bank 方式。若確定要接收此 Cell 時，則經由其 VCI/VP1 來 Search CAM 看此 Connection 的 Status，若為 New Cell則看是否有 Free Bank 可接收，若無則 Discard，若有則將此 Cell 放入此 Free Bank，且修改此 Free Bank 的 Status。若 Search CAM 的結果為已在 Reassembly 之 Connection 的 Cell，則將此 Cell 放入其相對應之 Bank。

若此 Cell為此 CPCS-PDU 的最後一個 Cell (*more* Bit=1)，則表示此 CPCS-PDU 組合完畢，ATM_Rx 即將此組好之 CPCS-PDU 的 Status 寫入 CAM 的相對應 Entry，其 Status 有 CPCS-PDU Length，CRC-32 是否正確，此 Completed Reassembly 之 Bank Number。當寫好 Status 後則發出一個 Interrupt，通知 Host 說已有一 CPCS-PDU 組合完畢。

當 Host的 ATM Driver收到 ATM_Rx 的 Interrupt 時，藉由讀取 ATM_Rx 相對應的 Status 得知相對應的 Information，以決定是否接收此 CPCS-PDU 或 Purge 此 CPCS-PDU。當此 CPCS-PDU Data 已由 ATM NIC 搬到 Host，則 Host 必需設定 ATM_Rx 之 Release Register，以通知 ATM_Rx 哪一個Bank 的 Data 已搬到 Host，可繼續使用此 Bank。

此 Host 亦提供 ATM_Rx Reassembly Timer Management。當 Host Detect到有一個 Bank的 Reassembly 已經 Timeout，則可通知 ATM_Rx Reset 此 Bank，以免此 Bank 無法再使用。

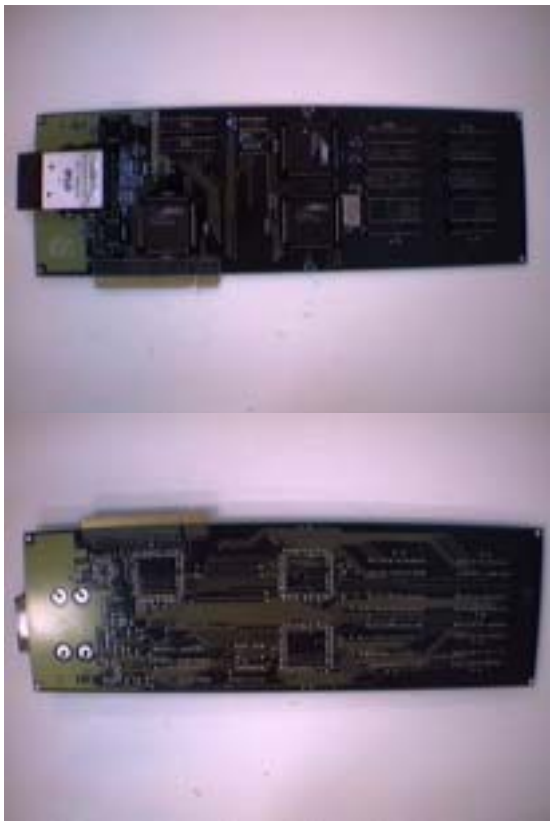
4.7 網路卡實體圖

圖十五為此網路卡零件面及銅箔面之實體圖。

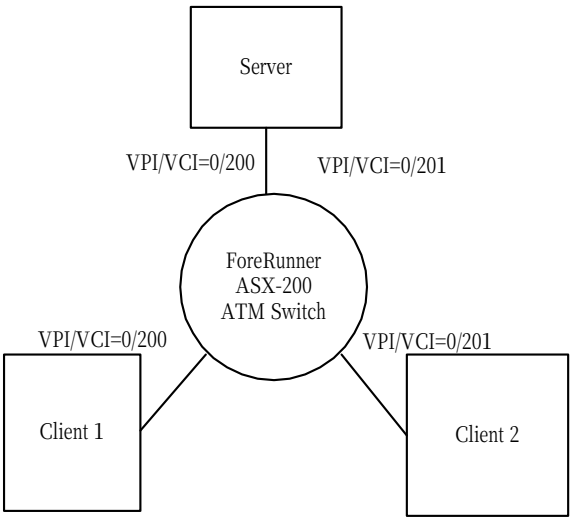
5. 應用實例說明

5.1 Video On Demand Based on ForeRunner ASX-200 ATM Switch

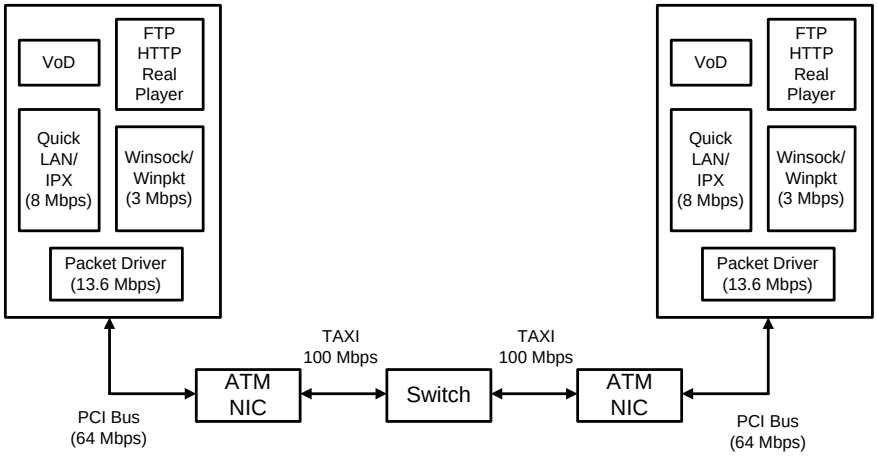
圖十六為利用 ForeRunner ASX-200 ATM Switch 連接三台 ATM PC 的實驗平台，ASX-200 設好兩條雙向的 PVC，一條連接 Server與 Client 1 (VPI/VCI=0/200)，另一條連接 Server 與 Client 2 (VPI/VCI=0/201)，在三台 PC 各自掛上了自己的 Packet Driver 及 QuickLAN 的程式後，以網路磁碟機的方式，Server可將自己的硬碟機或是光碟機開放給兩台 Client 使用，Client 因此可執行本身的程式來運用 Server 端的資料。目前我們展示 Client 端以 Xing MPEG Player可順暢地播放 Server 端的 MPEG 檔。另外一台 Client 也可隨意點選需要的檔案，在區域網路的 VOD 應用上，這樣的方式可說最為簡單有效的方式。



圖十五 PCI ATM 網路卡零件面及銅箔面的實體圖



圖十六 利用FORE Switch 實驗簡易VOD的架構



圖十七 從應用程式至ATM網路間各區段的傳輸效能

5.2 Windows95 TCP/IP Application Based on Trumpet Winsock

由於目前尚未發展好此 PCI ATM NIC 的 NDIS Driver for Windows95 and WindowsNT，在上網看了一些文章提到了 Windows95 仍可以在 Packet Driver 上架上 Trumpet Winsock 程式成為另一種可以支援 Windows95 的諸多 Winsock 應用程式的方法，本組組員於是抓了此程式來測試，終於也成功地可在無 NDIS Driver 的情況下得以執行如 WWW Server，Netscape，WS_FTP 等應用程式，驗證了本 ATM 網路卡的通用性。

6. 效能分析

圖十七為量測效能的相關軟硬体關係圖，就協定階層及傳輸路徑來說明，由網路端至應用程式端可區分為 NIC (Network Interface Card)，界接 PC 及網路卡的 PCI Bus ，Packet Driver驅動程式，以及如 IPX/QuickLAN，或 Winpkt/Trumpet Winsock， Ws_Ftp之應用層，我們分別量測以下列出之 Throughput，作為區辨整個協定階層傳輸路徑的瓶頸，以作為將來設計改善效能之用。

如圖十七所示在 Network Interface Card Layer 由於完全利用 Hardware 處理 AAL5 的 CPCS-SDU (Service Data Unit)，因此可說具備大約 100Mbps的 Throughput，而 PCI Bus 經實際量測結果約為64Mbps(圖十八顯示24Bytes Data在PCI Bus上傳送所需之時間為2.982μs)。作為作業系統跟網路卡之軟體界面的 Packet Driver 由於在接收時無法以 DMA 方式將 Cell直接送到 PC 上，必須在整個 CS-SDU 收入 Rx_DataBank，才能夠以中斷的方式由 CPU 讀到 PC 上，因此 Packet Driver 只有大約13.6Mbps的效能。

QuickLAN 是以 Novell IPX 為基礎的區域網路的網路磁碟機的應用程式，Server 可以把自己的磁碟機或光碟機開放給 Client 成為 Client 自己的磁碟機，因此任何在 Client 的應用程式均可透過區域網路使用 Server 端的影音檔達到即時播放的目的，也就是簡易的隨選視訊系統。

QuickLAN 配合本專題成品經實際量測可達8Mbps以上的效能，以目前 DVD MPEG2 的典型速率4~6Mbps而言，只要 Client 具 DVD 的播放能力，均可即時放映放在 Server 端的 DVD 檔。

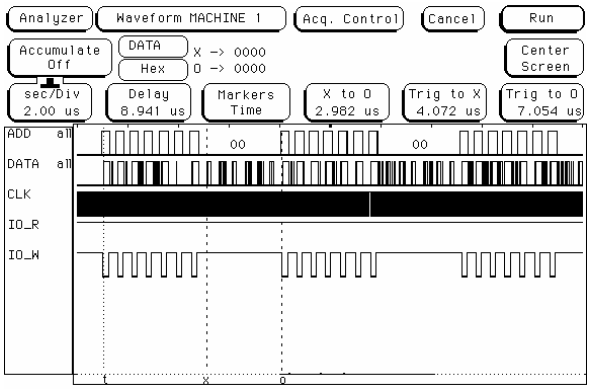
Trumpet Winsock在 Windows95 未廣泛使用前是最受歡迎的 TCP/IP 的上網工具，但在 Windows95 推出後因內建 TCP/IP Stack，使得 Trumpet Winsock 漸漸乏人問津，但是使用 Windows95 內建 TCP/IP Stack 以前必須備便網路卡的 NDIS Driver，因此，在本網路卡之 NDIS Driver 尚未發展成功以前，透過執行 Trumpet Winsock 的方式，才能使用到為 Windows95 撰寫的諸多網路應用程式。

透過圖十七可以看到，以 Trumpet Winsock 方式存取的 FTP 軟體只能有大約 3Mbps 的效能，這個結果顯示 TCP/IP Stack 是個人電腦在 Internet 的應用上的一大瓶頸，也就是說，如何改善 TCP/IP 的效能，是改善 Internet 端對端品質的一大課題。

7. 結論

我們已成功的發展出一個 PCI ATM/AAL5 網路卡，這個網路卡上有三顆自行設計的 FPGA，其中兩顆 FPGA，ATM_Tx 及 ATM_Rx，共同執行 ITU-T 規格所訂定的 ATM 層及 AAL 層的功能。第三顆 FPGA 則是實現 PCI Bus Interface 的功能。這三顆晶片及 TAXI 晶片組構成了此網

路卡的主要硬體元件。另外我們也依據 Packet Driver 撰寫出網路卡驅動程式，使得 DOS, Windows 的網路應用程式均能藉此網路卡傳輸資料。因為我們把 AAL5 的功能放在網路卡上執行, 這使得 PC 的 CPU 有較多的時間處理上層的協定，另外網路卡上的傳送端採用 Pipeline 的運作，可免除 Store-and-Forward 所造成的 Delay，提高整體 Throughput, 這個 PCI ATM 網路卡經驗証實現了我們所設計的 Physical，ATM，AAL 層的功能，我們並且將之與 Fore ATM Switch 相連，成功的展示了基本的 FTP, Telnet 的功能，以及利用 QuickLAN 展示簡易 Video On Demand 的應用。



圖十八 邏輯分析儀顯示Data在PCI Bus上傳送的 Timing

8. 參考資料

[1] Advanced Micro Devices, Inc. AM7968/AM7969-125 TAXI chip Integrated Circuits Data Sheet and Technical Manual.

[2] ATM Forum. ATM User-Network Interface Specification.

[3] C. L. Chang, T. C. Hou. Y. S. Chu, and K. J. Chen. Throughput Characterization of a PC with a High Speed ATM Network Interface. In Proc. IEEE Globecom'96 Workshop on Transport Protocols for High-Speed Broadband Networks, Nov. 1996.

[4] K. J. Chen. and C. L. Chang. Universal Parallel CRC Circuit and Algorithm. ROC Patent.

[5] FTP Software, Inc. PC/TCP Packet Driver Specification， VERSION 1.11， June 1994.

[6] ITU-T. I.361 B-ISDN ATM Layer Specification

[7] ITU-T. I.363 B-ISDN ATM Adaptation Layer (AAL) Specification.

[8] T. Shanley and D. Anderson, PCI system architecture. Addison Wesley,1995.

[9] Actel Corporation. Actel FPGA databook, 1995.

[10] Quick Logic Corporation. Quick Logic FPGA application note, 1995.

[11] Simple Connections Company, QuickLAN, Quick Installation and User's Guide, 1992.

[12] PCI Special Interest Group, PCI Local Bus Specification , Revision 2.1, June 1, 1995.