A HIGH SPEED FAULT-TOLERANT MULTIMEDIA NETWORK AND CONNECTIONLESS GATEWAY FOR ATM NETWORKS

By

Patrick Lam Sze Fan

A Thesis

submitted in partial fulfillment of the requirements

for the Degree of Master of Philosophy

Division of Information Engineering

The Chinese University of Hong Kong

June 1997

Acknowledgement

It is my pleasure to acknowledge many people who have contributed to this project. Their suggestions have given me plenty of good ideas in designing and developing this prototypes.

I would like to express my warmest gratitude to my supervisor, Professor Cheung Kowk-wai for his supervision. His sincere advice and help have given me much valuable knowledge and trained me how to be a good researcher.

Moreover, I would like to thank my girl friend, Amanda S.M.Lau for her spiritual support and encouragement so as to alleviate the working pressure. I am indebted to my partner, Arion K.W.Ko for his participation in the fault-tolerant CUMLAUDE NET project and great contribution to the papers. Also, thanks to Calvin C.K.Chan for his benignant assistance and suggestions throughout these years. I would also like to thank Tai B K.N.Chung for his inspiration on Linux kernel hacking. Thanks are due to Ringo K.W.Lam who always produces joyful atmosphere and made the CUMLAUDE NET NICs. Thanks Wilson Y.H.Wang, my roommate for his advice on common C errors and the random generator routines. Lastly, I would like to express my gratitude to Alex W.H.Siu, Zoe C.Y.Tsang, Y.Gao, Mr. Simon S.B.Lau, Chris Wong and many more for their patronage within my two years research life.

Abstract

There are two contributions in this thesis. First is a high-speed distributed fault-tolerant and auto-healing network design and implementation. Second is a connectionless gateway design and implementation for ATM and LANs.

With the emerging of high-speed communication networks, the issues on network reliability and surviability become crucial since a single network failure can cause huge amount of data loss. Therefore, the network vendors and researchers are putting more efforts to improve the reliability and surviability of the existing networks.

In this thesis, we have proposed two distributed fault-tolerant and auto-healing algorithms to protect the network against failures. The algorithms are based on the inter-communications and hand-shaking processes between adjacent network nodes to facilitate distributed network restoration. They are characterized by the features of fast network restoration, short restoration messages and hot replacement of faulty network components. Thus the disturbance to the users caused by network failure is minimized while the network availability is maximized. The algorithms have been implemented and tested on a testbed called CUM LAUDE NET and a failure recovery time in milli-second range is achieved. CUM LAUDE NET is a 100-Mbit/s high-speed multimedia network

designed and prototyped at the Chinese University of Hong Kong. It is specially designed to provide various facilities for the algorithms implementation. Details about the CUM LAUDE NET will also be described in the thesis.

Asynchronous Transfer Mode (ATM) has been adopted as the standard for the Broadband Integrated Service Digital Network (B-ISDN) and may possibly serve the future Internet. ATM is working in connection-oriented mode which is different from the traditional connectionless LANs such as the Ethernet and the CUM LAUDE NET.

In order for the connectionless LANs to become accessible through the ATM networks, we have proposed a Connectionless Gateway which can connect these two broad types of networks. Since the demand for gateways will be high in the future, thus the proposed Gateway is "low-cost" in design. The designed Gateway is constructed base on the inexpensive PC platforms with a operating system called Linux which is a freeware. The Gateway functions as a connection manager and protocol converter to create connections and convert protocol data units between the communications parties in different kinds of networks.

Besides the primitive functions stated, we have additionally implemented a "Priority Queueing System" in the Connectionless Gateway to lower the delay of time sensitive traffic. Moreover, a real-time "Gateway Performance Monitor" application with Graphical User Interface (GUI) has been developed for the network administrators to monitor the status of the Connectionless Gateway interactively. Finally, the success of the "low-cost" Gateway has been demonstrated by testing it in a series of experiments and a satisfactory throughput of 52-Mbit/s has been obtained. Based on the results obtained, solutions are suggested to improve the gateway performance. The criteria for hardware selection

in the construction of a Connectionless Gateway with optimal performance has also been studied.

Acronyms

AAL	ATM Adaptation Layer
ABR	Available Bit Rate
ACTA	Adaptive-Cycle Tunable-Access
AFI	Authority and Format Identifier
AH	Auto-healing
API	Application Programming Interface
ASIC	Application Specific Integrated Circuit
ATM	Asynchronous Transfer Mode
ATMARP	ATM Inverse Address Resolution Protocol
B-ISDN	Broadband Integrated Service Digital Network
BEBP	Binary Exponential Backoff Polling
BLR	Burst Lost Rate
BT	Burst Tolerance
CBR	Constant Bit Rate
CDV	Cell Delay Variation
CDVT	Cell Delay Variation Tolerance
CLP	Cell Loss Priority
CLR	Cell Loss Ratio
CPCS	Common Part Convergence Sublayer
CTD	Cell Transfer Delay

DLL	Dynamically Linked Library
DSP	Digital Signal Processor
DSP	Domain Specific Part
ELAN	Emulated LAN
ESI	End System Identifier
FDDI	Fiber Distributed Data Interface
FIFO	First-In-First-Out
FPGA	Field Programmable Gate Array
FPR	Fast Packet Routing
FT	Fault-tolerant
GFC	Generic Flow Control
GUI	Graphical User Interface
GW	Gateway
IDI	Initial Domain Identifier
IETF	Internet Engineering Task Force
IISP	Interim Inter-Switch Signalling Protocol
ILMI	Interim Local Management Interface
IP	Internet Protocol
IPC	Interprocess Communication
IPv6	Internet Protocol Version 6
ITU-T	Telecommunication Standardization Sector of International Telecommunication Union
IWU	Interworking Unit
LAN	Local Area Network
LANE	LAN Emulation
LEC	LAN Emulation Client
LECS	LAN Emulation Configuration Server
LES	LAN Emulation Server

LIS	Logical IP Subnet
LSB	Least Significant Bit
MARS	Multicast Address Resolution Server
MCR	Minimum Cell Rate
MIB	Management Information Base
MSB	Most Significant Bit
MTU	Maximum Transfer Unit
NBMA	Non-Broadcast Multi-Access
NFS	Network File System
NHRP	Next Hop Resolution Protocol
NIC	Network Interface Card
NNI	Network Network Interface
NSAP	Network Service Access Point
PCR	Peak Cell Rate
PDU	Protocol Data Unit
PRM	Protocol Reference Model
PVC	Permanent Virtual Circuit
QoS	Quality of Service
RIP	Routing Information Protocol
RSVP	Resource Reservation Protocol
SAAL	Signalling ATM Adaptation Layer
SAR	Segmentation and Reassembly Sublayer
SCR	Sustainable Cell Rate
SDU	Service Data Unit
SEAL	Simple Efficient Adaptation Layer
SMDS	Switched Multimegabit Data Service
SNMP	Simple Network Management Protocol
SONET	Synchronize Optical Network

SSCF	Service Specific Coordination Function
SSCOP	Service Specific Connection Oriented Protocol
SSCS	Service Specific Convergence Sublayer
SVC	Switched Virtual Circuit
TAXI	Transparent Asynchronous Transmitter-Receiver Interface
TCP	Transmission Control Protocol
TOS	Type of Service
TTL	Time to Live
UBR	Unspecified Bit Rate
UDP	User Datagram Protocol
UNI	User Network Interface
VBR	Variable Bit Rate
VC	Virtual Channel
VCI	Virtual Channel Identifier
VPI	Virtual Path Identifier
WWW	World Wide Web

Contents

List of Tables

List of Figures

Chapter 1

Introduction

"Network" is a term that can be found in the newspapers or on TV almost everyday, especially on the world wide network "Internet". After the deployment of the World Wide Web (WWW), many people, other than the professionals, started using the Internet to browse information for both personal and business uses. Thus, the utilization of the Internet has increased at a rate much faster than predicted. Moreover, the bandwidth of the current Local Area Network (LAN) technologies cannot meet the bandwidth demand of the multi-media applications. Therefore, the design of a faster network seems to be indispensable.

In this thesis, three types of network technologies are considered, the CUM LAUDE NET, the Asynchronous Transfer Mode (ATM) and the Fast Ethernet.

CUM LAUDE NET is an experimental gigabit hierarchical dual-ring network developed at the Chinese University of Hong Kong. It is aimed at providing services for real-time multimedia applications. The MAC layer of the network is based on a novel protocol called the Adaptive-Cycle Tunable-Access (ACTA) Protocol [1, 2] which provides adaptive and fair allocation of bandwidth for

the users. Most importantly, fault-tolerant and auto-healing algorithms are also designed and implemented in the network such that the reliability and the survivability of the network can be greatly enhanced. Subsequently, the availability of the network is improved. In Chapter 2, the CUM LAUDE NET will be described in-depth and the details about the fault-tolerant and auto-healing algorithms will also be clearly presented.

Asynchronous Transfer Mode Network (ATM) is the de facto standard for the future Broadband Integrated Service Digital Network (B-ISDN). The idea of designing this technology is to integrate all types of (packet and switch) networks to form a single high speed network and to provide the capability of carrying different types of data (e.g., computer, video and voice data or multimedia) traffic seamlessly. Actually, ATM is a combination of the telephony and the packet computer network technologies. It is a connection-oriented network that dedicated virtual channel (VC) with enough resources (to provide Quality of Service) must first be established before any data transmission. This operation is analogous to the telephone dial up procedure. Data in ATM is transmitted in fixed size (53 bytes) packets which is called a "cell" in the ATM terminology. Details about ATM can be found in Chapter 3.

The term "Gateway" in the telecommunication industry generally means a device that interconnects similar or dissimilar networks. Some devices such as bridges or routers are also called gateways and their main difference is that they work at different OSI layers. In Chapter 4, the issues about the design of a connectionless Gateway connecting the connectionless legacy LANs and ATM networks will be explored. Five main issues that will be discussed are listed as follows:

- ATM Internetworking;
- Connections Management;
- Protocol Conversion;
- Bandwidth Assignment[3];
- Packet Forwarding Modes;

Among the issues, ATM Internetworking is one of the hottest topic that is under investigation. Two main proposals on this topic, "IP over ATM" and "LAN Emulation" will be discussed. These proposals are important in the path of technology migration from legacy connectionless LAN to connection-oriented ATM from the perspective of users. However, they cannot fully utilize the features provided by the ATM networks. Thus, in order to enhance the capability of ATM features utilization, new protocols such as IPv6 (Internet Protocol Version 6) and RSVP (Resource Reservation Protocol) are desirable, but it needs time for the development of these protocols.

After a thorough analysis on the gateway issues, the design and implementation of the Connectionless Gateway will be presented in Chapter 5. Figure 1.1 has depicted the working scenario of the Connectionless Gateways in a hybrid network. The designed gateway is a "low cost solution" for interconnecting traditional connectionless LANs with the ATM network. The construction of the Connectionless Gateway consists of a low cost PC equipped with ATM and connectionless LAN network interfaces cards (NICs). The "gateway engine" is implemented in the kernel of an operating system called Linux (which is another

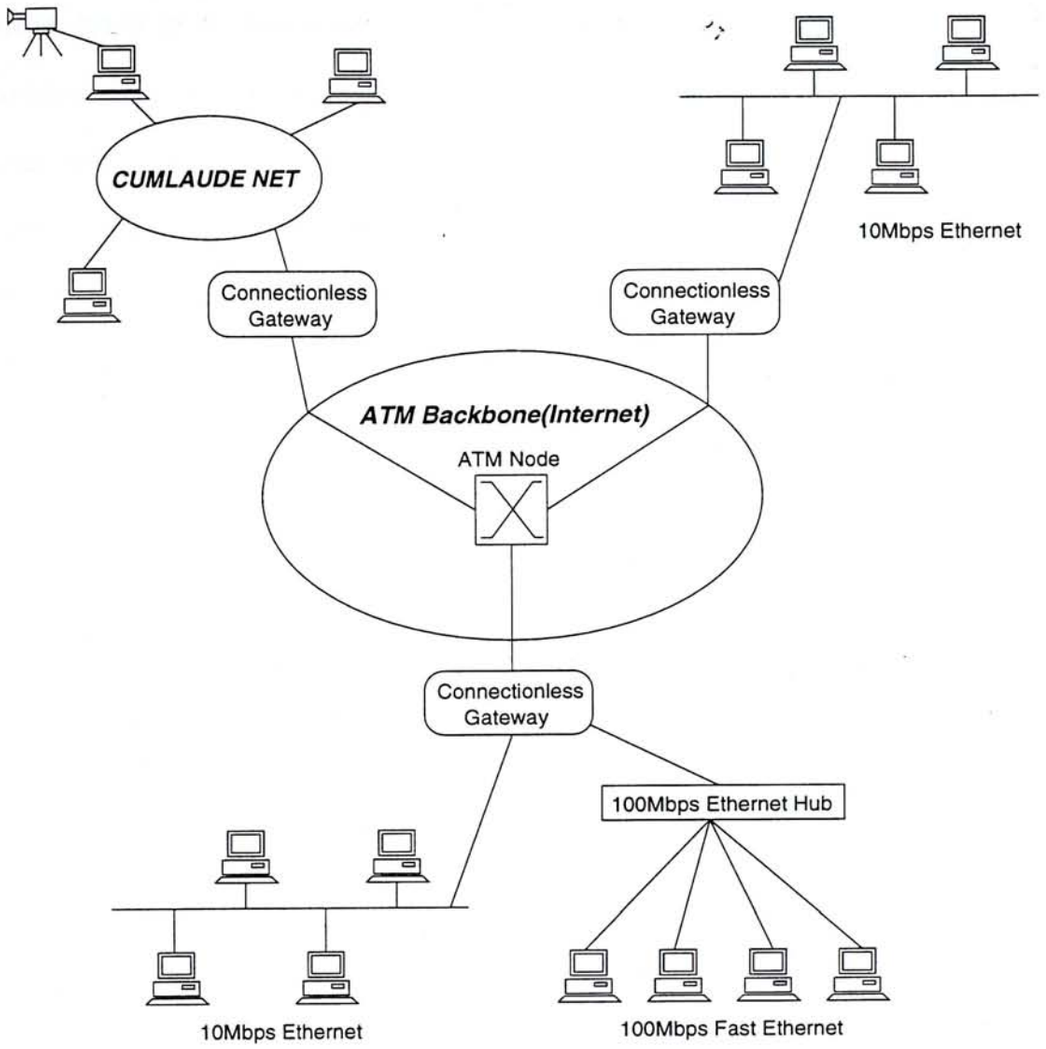


Figure 1.1: Operation scenario of the Connectionless Gateways.

UNIX clone). The engine is mainly responsible for filtering, converting and routing packets between different networks. Further, the Connectionless Gateway can connect up to n[1] networks in case there are n available buses slots.

Two additional features namely "Priority Output Queues System" and a "Gateway Performance Monitor" application, are also designed for the Gateway. The Priority Output Queues System allows the separation of IP packets with

[1]Number of available buses in the PC.

different *types of service* priorities and provides higher transmission priority for those time sensitive data. The "Gateway Performance Monitor" is designed for the network administrator to interactively monitor the working status of the Connectionless Gateway such as bandwidth utilization and the kernel status tables such as Routing Table. Moreover, this application is easy to use as it is designed with a user friendly interface operating under the X-window system.

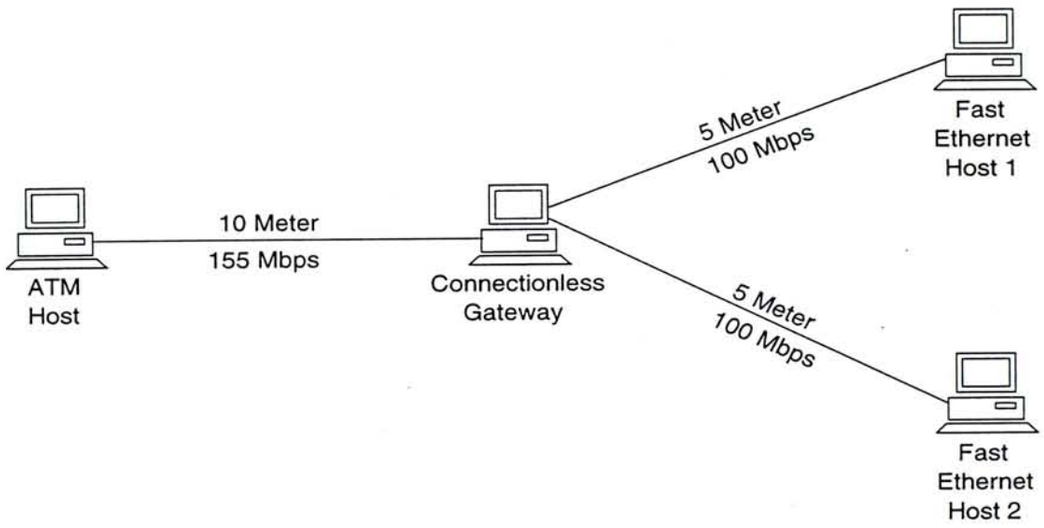


Figure 1.2: Experimental setup designed for the performance measurement experiments.

After the gateway implementation is completed, extensive experiments are designed and run over the experimental network in order to evaluate the gateway performance under different characteristics of data traffic. The dependency of the gateway performance on the gateway hardware configuration is also studied. The configuration of the experimental network is shown in Figure 1.2. As shown, 100-Mbit/s Fast Ethernet[2] PCI NICs are employed in the experiments

[2]Details about the Fast Ethernet technology can be found in [4, 5].

as they have higher throughput than the CUM LAUDE NET NICs[3]. Observations recorded during the experiments and details analysis on the results will be presented in Chapter 6. Some suggestions on improving the performance of the Gateway will also be discussed. A conclusion will be given in Chapter 7.

[3]Since the architecture of the CUM LAUDE NET NIC is based on the ISA bus, its throughput is limited by the bus bandwidth.

Chapter 2

Fault-tolerant CUM LAUDE NET

With the emerging needs of reliable high-speed communication networks, network vendors and researchers are putting more efforts to improve the reliability and survivability of the existing networks[6]. In this chapter, distributed fault-tolerant and auto-healing algorithms for dual-ring networks will be presented. The correctness of the algorithms have been tested on a multimedia network testbed called CUM LAUDE NET.

2.1 Overview of CUM LAUDE NET

CUM LAUDE NET is a group effort at the Chinese University of Hong Kong to prototype a multi-gigabit/sec multimedia integrated network[7]. It is designed to support high-speed, real-time multimedia services with maximum compatibility to IP-based networks, especially the Internet. The development is divided into

two phases. The objective of phase I is to demonstrate a practical, low-cost, high-speed and fault-tolerant integrated network that can provide real-time video and voice conferencing for a local area. The objective of phase II is to construct a gigabit/sec fault-tolerant dual-ring backbone for the provisioning of real-time multimedia services for a metropolitan area. The phase I construction has been completed in 1996. Various kinds of supporting software have been developed for the phase I network including a multimedia video and voice conferencing utility, voice mail services, a network management software called NETMAN and other Internet utilities (TELNET, FTP, etc.). Gateways have also been developed to interconnect with different networks such as public switched telephone network, FDDI, Ethernet and ATM networks. Works on the next phase construction has already started and is scheduled to be completed soon.

2.2 Network architecture of CUM LAUDE NET

The network architecture of CUM LAUDE NET is a multi-gigabit/sec hierarchical dual-ring[8]. The proposed prototype consists of two hierarchies as shown in Figure 2.1.

The level-2 hierarchy (phase II) is a 1-Gbit/s backbone connected in a fault-tolerant dual-ring with three access nodes. It aims at providing high-speed, real-time multimedia services for a metropolitan area. Each node will be able to route 1-Gbit/s bandwidth at each input and output ports. Assuming uniform traffic, the dual-ring backbone will have an aggregate capacity of 8 gigabit/sec.

Each level-1 hierarchy (phase I) is a 100-Mb/s dual-ring network aiming at providing the same services as the backbone but for a local area environment.

Different hierarchies are connected by routers, whose function is to pass packets from one hierarchy to another.

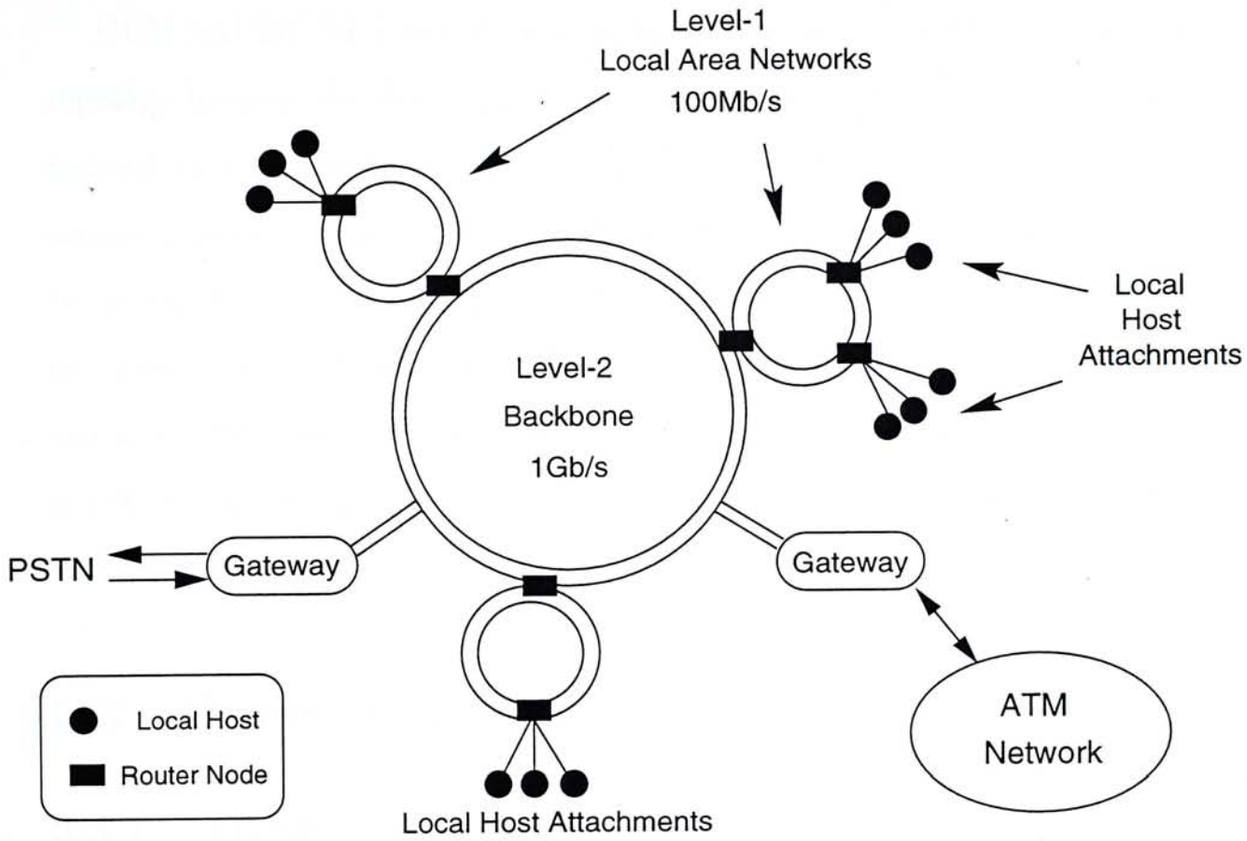


Figure 2.1: CUM LAUDE NET Network Architecture

The level-1 hierarchy is constructed as a fault-tolerant dual-ring which is formed by several 100-Mb/s router-nodes connecting together. Each 100-Mb/s router-node has three routers, two for dual-ring attachment and one for local attachment (either a local host or a hub[1][9]). The routers on these router-nodes also provide fault-tolerance and auto-healing functions which will be presented in the later sections.

Each local host is equipped with a network interface card (NIC) to process packets addressed between the network and the host. Due to the uniform design

[1]Hub is a device which serves as a concentrator/distributor to a number of local hosts.

of the different hierarchies, the hardware and software design of the routers are much simplified.

CUM LAUDE NET prototype is designed around a fault-tolerant dual-ring topology because the ring topology has several unique advantages over a centralized switching hub topology [1, 2]. This linear topology allows reserved service guarantee and fair sharing of bandwidth among all nodes. The distributive, sequential arrangement of the nodes also facilitates real-time protocol implementation. Distributive packet routing simplifies packet processing and introduces little packet delay. Another advantage of the linear topology is that it reduces the problems due to network congestion and complexities in control and management.

2.3 Design of Router-node

2.3.1 Architecture of the Router-node

Router-node has been designed and prototyped for the level-1 hierarchy as shown in Figure 2.2. On the router-node, there are three routers, two for the ring attachments (Ring-A and Ring-B Routers) and one for local attachment (Local Router).

As shown in Figure 2.2, the major hardware components on a router consists of a digital signal processor (DSP), two field programmable gate array (FPGA) ICs, four buffer memories (FIFOs) and a pair of transceiver. The DSP and FPGAs are grouped as a unit which is served a "Routing Controller". The major function of this "Routing Controller" is to control the on-board hardware

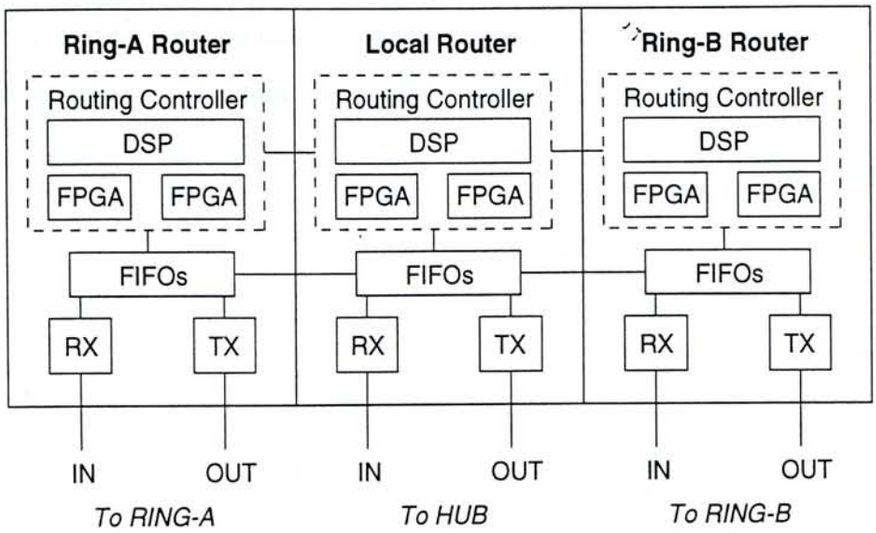


Figure 2.2: Router-node architecture of the CUM LAUDE NET.

in order to accomplish specific tasks such as deciding routes for the packets, inter-communicating with the other routers to exchange status information (through the serial ports on the same node), handling the problem of buffers congestion and executing the fault-tolerant and auto-healing functions.

Three out of the four buffer FIFOs are arranged as transmit buffers and the last one is used as receive buffer. The FIFOs of the routers are interconnected by shared data buses. This design is used to facilitate packet routing among different routers. Moreover, this design enables the dynamic constructions of logical paths inside the router-node. These FIFOs will also act as temporary storages during network restoration.

The transceivers used in the routers are some chipsets called TAXIs (Transparent Asynchronous Transmitter-Receiver Interface)[10] from the Advanced Micro Devices (AMD). They have maximum operation speed of 125 Mbaud on a serial link and use the robust 4B/5B and 5B/6B coding schemes to detect transmission errors. The physical media used to connect the router-nodes (ring

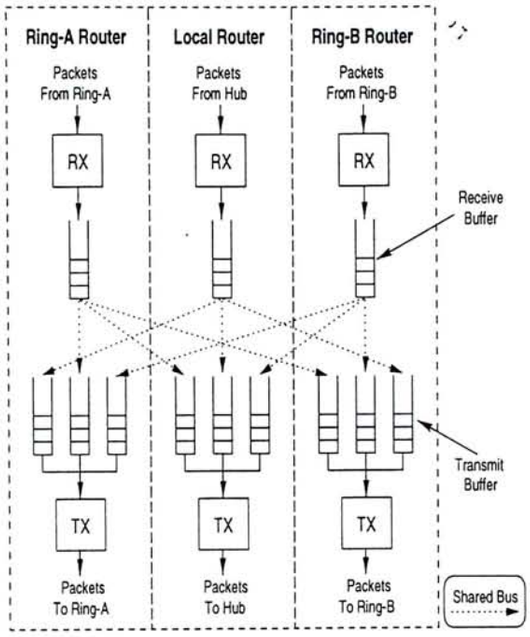


Figure 2.3: Buffers arrangement of a router-node.

routers) is optical-fiber and that used to connect the local router with the hub is coaxial cable.

2.3.2 Buffers Arrangement of the Router-node

Figure 2.3 shows the detail arrangement of the buffers in a router-node. The buffers used in the implementation are called FIFOs (First-In-First-Out) memories. Each router will have four FIFOs, three of them are allocated as transmit buffer and one is allocated as receive buffer. The sizes of the transmit and receive buffers are 8 Kbytes and 2 Kbytes respectively.

The receive buffers are used to store the packets coming from the respective rings/hub. Each receive buffer is connected to three transmit buffers, one on each router and we called these physical connections as "shared buses". Logically, the transmission channels between the routers can be interconnected by these buses.

Routing between the routers can thus be accomplished easily. Assume a packet is now coming from Ring-A, it will first be stored in the receiver buffer. Then, the routing controller will examine the packet's FPR (Fast Packet Routing) header to determine its destination. The packet will finally be forwarded to the transmission buffer of: Ring-A (normal case) or Ring-B (when the outgoing link of Ring-A is failed.). A copy of this packet will also be sent to the Local-Router if the destination of that packet is connected to the hub. The situation is similar for the Ring-B routers. The Local router may choose either forwarding the packets to Ring-A or Ring-B depend on the routing decisions made by the routing controller. It will also broadcast the packets back to the hosts that are connected to the hub.

2.3.3 Buffer transmission policies

As shown in Figure 2.3, the three transmit buffers on each router share one output channel. Therefore, some mechanisms must be designed to arrange the transmission sequences of the packets in these buffers. Two transmission policies have been tested in the implementation, they are called "Full-first" (which is dependent on the fullness condition of the buffers) and "Round-robin".

The FIFOs used in the routers have an indicator for reporting their fullness conditions and this indicator has been memory-mapped to the address space of the DSP. The fullness condition can be represented by four flags: *Full*, *Half-full*, *Non-empty* and *Empty*. The "Full-first" policy uses this indicator as a "transmission selector". Only the first three flags will be used to assign a transmission priority to each transmit buffer. When the routing controller desires to transmit a packet from the transmit buffers, it will examine the "transmission selector"

of each FIFO to find the one which is the fullest. Then, the packets inside such fullness buffer will be transmitted.

In case the fullness conditions of the buffers are the same, the routing-controller will select the FIFO in a so called "round-robin" manner. With the design of these transmission policies, the risk of buffer overflow can be minimized. It can also tackle the problem of resource contention and regulate the flow of traffic.

2.4 Protocols of CUM LAUDE NET

CUM LAUDE NET is designed to support high-speed, real-time multimedia services with maximum compatibility to IP-based networks. In order to achieve these goals, two protocols called "FPR (Fast Packet Routing)" and "ACTA (Adaptive-Cycle Tunable-Access)" are designed.

The FPR protocol employs the following features:

- Fixed size IP packets/FPR frames (576/582 octets);
- Fast packet routing (FPR) in the MAC and network layer;
- Direct IP addressing in the transport and routing of IP packets;
- Connectionless delivery of frames.

In the FPR protocol layer, the IP packets are encapsulated by a fixed-size header and trailer and all the routing information is available in the header. This allows each router to perform fast packet routing efficiently and simplifies the gateway design between the CUM LAUDE NET and Internet.

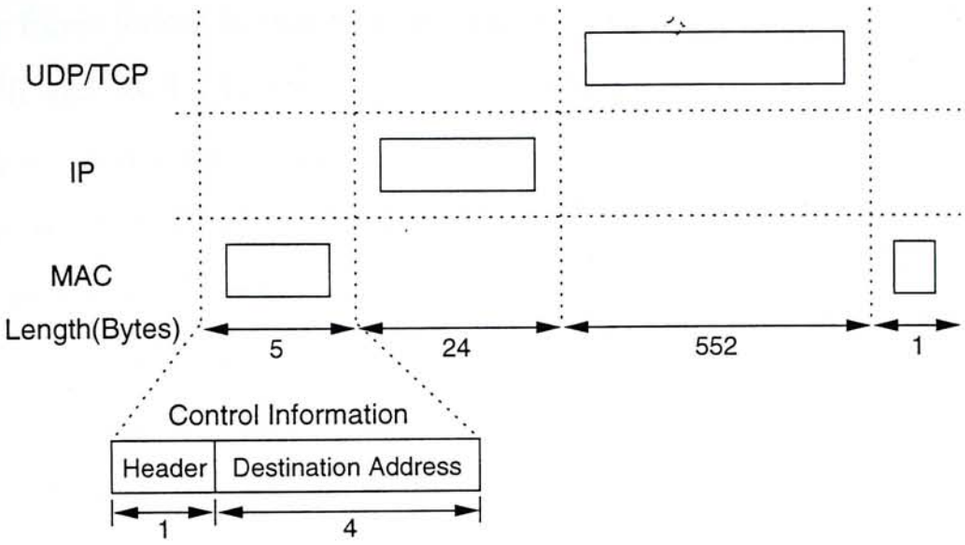


Figure 2.4: Frame format at the FPR protocol layer.

ACTA is a novel network protocol[1, 2] that is implemented in each Level-2 and Level-1 dual-ring hierarchy. Fair access is achieved by limiting the number of empty slots occupied by each router on each cycle. The cycle length is adjusted to reduce the packet latency and to increase the throughput.

2.5 Frame Format of CUM LAUDE NET

Fixed size frames (582 bytes) are used in the CUM LAUDE NET. In a FPR protocol layer frame, it contains four portions (Figure 2.4):

- Header (1 byte);
- Destination Address (4 bytes);
- Fixed size IP packet (576 bytes);
- Trailer (1 byte).

The frame format is chosen to facilitate the hardware and software design of the FPR and ACTA Layers. The control information required for fast packet routing is contained in the first 5 bytes (Header and Destination Address) of the FPR protocol layer frame as shown in Figure 2.4. The control header and the trailer are also used for frame synchronization.

Besides the FPR protocol, the ACTA also requires some bits to store up its protocol information such as "cycle start" and "slot occupied". These information are also stored in the first byte of the MAC header. The details about the one byte header is illustrated in Figure 2.5(a). The "NM" bit stands for Network Management which is used to carry information for the fault-tolerant and auto-healing functions. Figure 2.5(b) illustrates the contents of this one byte header when used in the Hub module where the media access protocol is different from the ACTA protocol which is called the Binary Exponential Backoff Polling(BEBP) protocol[9].

2.6 Fault-tolerant (FT) and Auto-healing (AH) algorithms

2.6.1 Overview of the algorithms

The FT and AH algorithms presented shortly are categorized as *distributed algorithms* because the fault detection, recovery and auto-healing processes are executed on individual network router-nodes. These algorithms are designed specifically for the dual-ring networks and are being implemented in the CUM LAUDE NET testbed. They are applicable in all kinds of dual-ring networks

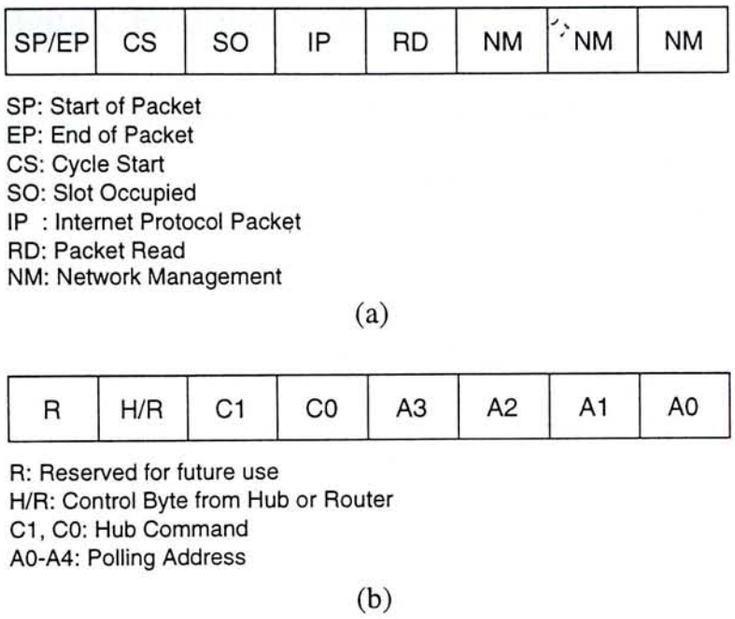


Figure 2.5: Details of the "Header" in the MAC header for (a) Router and (b) Hub.

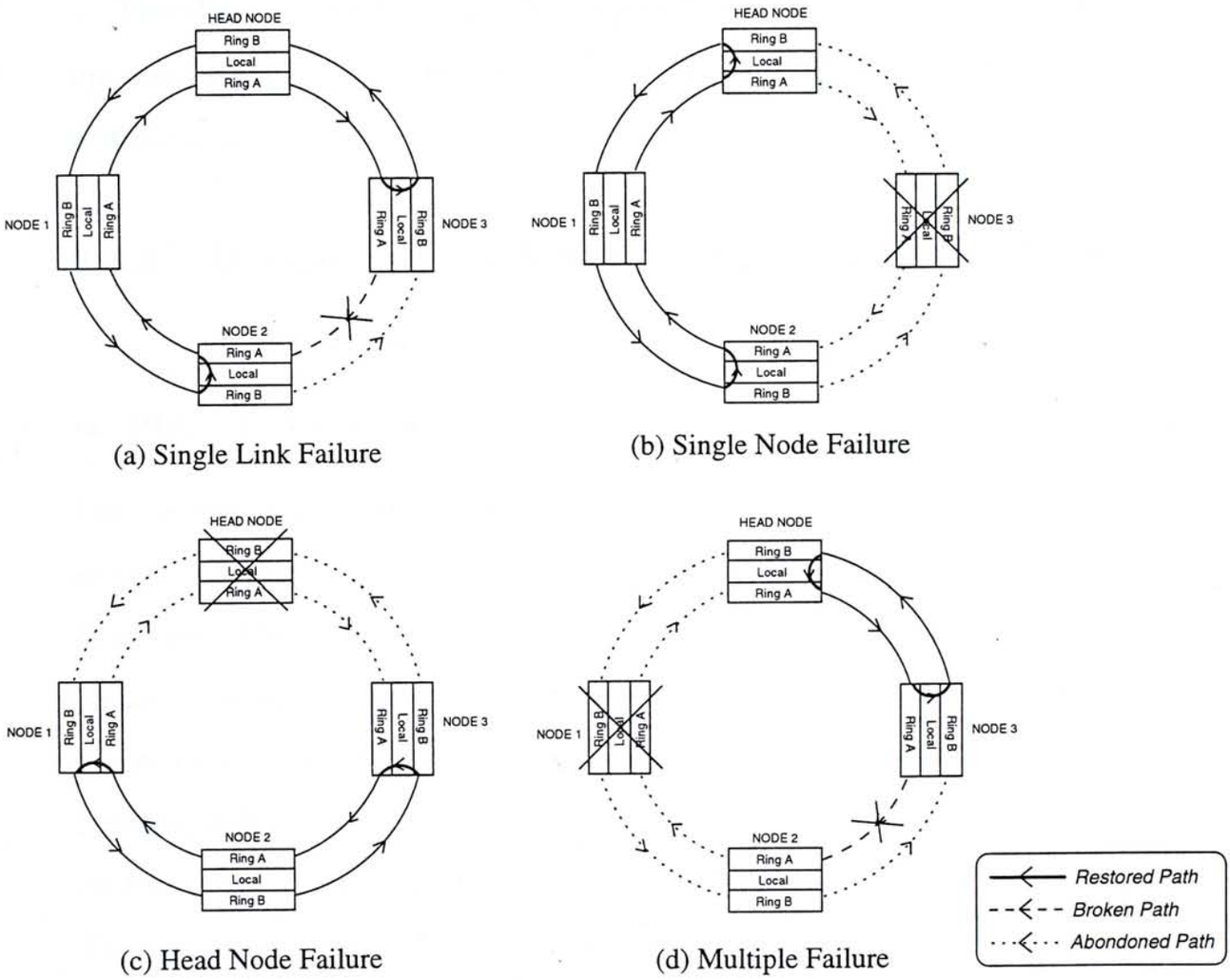
with their generic design.

The operations of the fault-tolerant algorithm can be divided into two phases, the "Fault-detection" phase and "Fault-Recovery" phase. In the "Fault-detection" phase, the ring routers will individually perform the detection process. When any fault is detected, the FT engine will call the "Fault-Recovery" function to recover the fault(s).

The AH function is another unique feature that has been implemented in the CUM LAUDE NET. The term "Auto-healing" defined here means "the network can automatically restore itself to normal state after it has detected the removal of the failure." It supports hot replacement of faulty network components and reduces the disturbance to the users. It thus simplifies the network maintenance and provides better network availability to the users in case of failure.

2.6.2 Network Failure Scenarios

Before the design and implementation of the proposed algorithms are presented, the possible network failure scenarios that may occur in a dual-ring network is discussed.



(a) Single Link Failure

(b) Single Node Failure

(c) Head Node Failure

(d) Multiple Failure

Figure 2.6: Network Failure Scenarios and Remedies.

In a dual-ring network, there are two major types of failures, i.e., the link and node failures as shown in Figure 2.6. The figure has illustrated the general failure scenarios that may occur in a dual-ring network and their respected remedies. As

depicted, the failures can be recovered by wrapping up the links. In the cases of slot-based networks like CUM LAUDE NET, additional restoration procedures are required to recover the possible failure of the headnode. It should also be noted that a node failure is equivalent to two pairs of link failure.

Therefore, in the design of the proposed algorithms, we will only consider the cases of link failures. Besides, it is assumed that the failures are occurred in a "fault-stop" manner[2].

2.6.3 Design and Implementation of the Fault Tolerant Algorithm

(a) Phase I - Fault Detection Phase

Two schemes are proposed in detecting the link failures. The first one is named as *Out-of-synchronization Detection* and the second one is *Packet Non-arrival Detection.* The implementation of the Out-of-synchronization scheme imposes certain criteria on the system hardware while the other scheme is hardware independent. Therefore, they are design as a "complete-solution" which can be implemented in most kinds of dual-ring networks.

(i) Out-of-synchronization Detection

This scheme requires the system hardware to perform link synchronization. Normally, the links between the router-nodes are kept synchronized by the transceivers. When the transmission quality of the links between the nodes decreases below a threshold (i.e., out-of-synchronization), the transceivers of the link will be aware of this problem and will generate some warning signals.

[2]i.e., the cases of which the routers/hosts generate garbage frames are not considered.

Appropriate restoration procedures will then be invoked after these signals are trapped.

The implementation of this "Out-of-synchronization Detection" is straight forward in the CUM LAUDE NET. Since the transceivers (TAXI) used in the routers (see Section 2.3) has provided a similar link synchronization function. Synchronization in TAXI is achieved by the transmission and reception of a special "K28.5"[3] *SYNC* symbol.

When the receiving TAXI does not receive the K28.5 symbols (i.e., link failed) for a short period, it will generate a "violation signal". This "violation signal" will then interrupt the DSP and the DSP will take up the responsibility to recover the fault(s) by invoking the *Restoration()* (to be described later) procedure.

(ii) Packet Non-arrival Detection

Although the *Out-of-synchronization Detection* scheme is simple and efficient, it cannot solved the cases of node components failure other than the transceivers. This algorithm is also not applicable to those systems that do not provide the function of link synchronization. Thus, the *Packet Non-arrival Detection* scheme is proposed. The philosophy behind this scheme is simple, when a node does not receive valid packet for a certain period, link failure is assumed occur somewhere in the ring.

The node that detects the occurrence of this failure should then send a *Test_Link* message to confirm whether its neighbour is still connected (i.e., to determine whether the failure is arounded it.). If a *Link_OK* message is responded, it signifies that its neighbour is still operative. Otherwise, the *Test_Link* message

[3]A K28.5 symbol consists of five consecutive ones follow by two zeros[10].

will be re-sent n times to confirm the occurrence of failure. This scheme can be used in the dual-ring networks that employing slot-based MAC because the circulated slots can be used as indicators for the nodes to detect the presence of traffic. The advantage of this scheme is that it does not impose any prerequisite on the hardware.

The correctness of this scheme has been proved in the CUM LAUDE NET. In the implementation, a timer routine is scheduled every 55 μs (which is equal to one ACTA slot time) to interrupt the DSP. When the DSP traps this interrupt, it will check whether packet is arrived at the node. If no packet has arrived, failures may have occurred in the network. An "failure locating" mechanism is proposed in order to find out the actual location of the failure. This failure locating mechanism is based on a hand-shaking inter-communications process between the router nodes. Figure 2.7 has depicted an example showing the operation of this mechanism.

In the example, the link of ring A between Router-Nodes M and M+1 is broken. All the upstream nodes of Router-Node M will aware the occurrence of this failure within the coming 55 μs. They will then individually start the failure locating mechanism - *Link Test* routine as shown in Figure 2.7(a). Hereafter, the other ring nodes (i.e., the ring B routers) will take up the responsibility of link testing as illustrated in the remaining figures. For example, Ring-B router of Router-Node M will issue a `T_HEADER` (*Link Test Header*[4]) to its neighbour (i.e., Ring-B router of Router-Node M+1) as shown in Figure 2.7(b). If the test header can reach Ring-B router of Router-Node M+1, the router will issue a command `RX_CORR` (*Receive the Header Correctly*) to the Ring-A router of

[4]It is generated by setting the NM bits in Figure 2.5(a) to '010' and it will not carry any address and user data information.

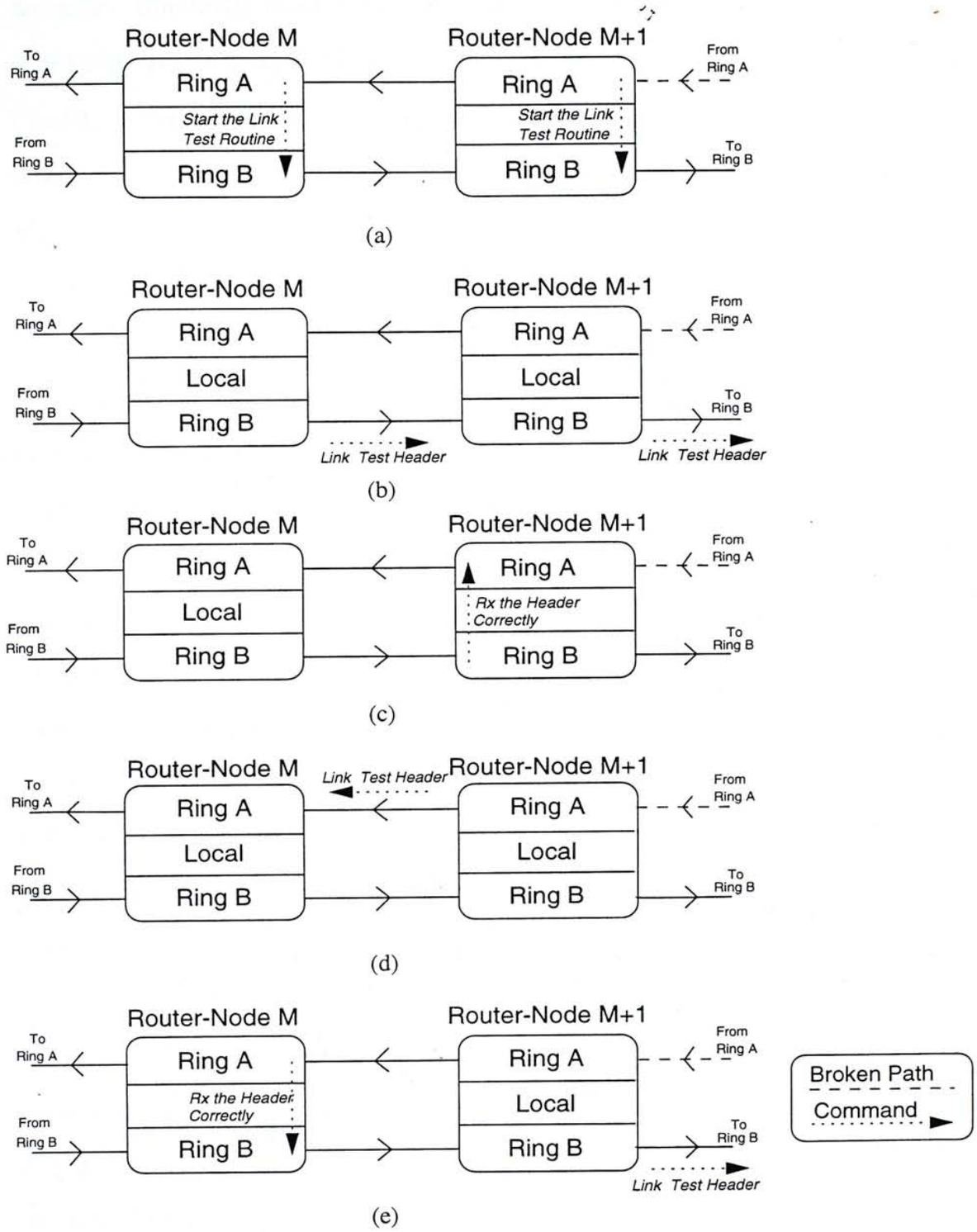


Figure 2.7: Diagram showing the link detection phase procedures in both single and multiple failures.

its node. Similarly, Ring-A router of Router-node M+1 will then transmit a T_HEADER back to Ring-A router of Router-Node M (i.e., the originating router). Finally, if the originating router does not receive any acknowledgment for n trials, the link/node is assumed to be failed and the *Restoration()* procedure will be invoked.

(b) Phase II - Fault Recovery Phase

As mentioned in Section 2.6.2, the fault recovery procedure bypasses the failure by wrapping the ring from one to the other. For isolated subnets without head node, a *Claim Head* process will be activated to regenerate a new head node. In the process, each node within the subnet will broadcast its registration number (an unique number assigned to each node during network initialization) and any node receives a registration number smaller than its own will generate an *Objection* message to the originating node.

In CUM LAUDE NET, the fault recovery procedure can be ported easily with the special design of the Router-nodes (see Section 2.3). Since the transmission buffers of the routers on a router-node are interconnected, then, the routers can accomplish the wrapping action by transferring the packets to the others buffer. For example, ring-A router can route their packets to ring-B router or local router by writing data to their transmit buffer. To facilitate this routing protocol, a signaling scheme to synchronize the nodes around the failure is required. Figure 2.8 has depicted an flow diagram showing the operations of the signaling protocol. It is based on the scenario of Figure 2.7. The Ring-A router of Router-Node M+1 will initiate the signaling scheme and inform other nodes of the failure event. During this synchronization process, any incoming

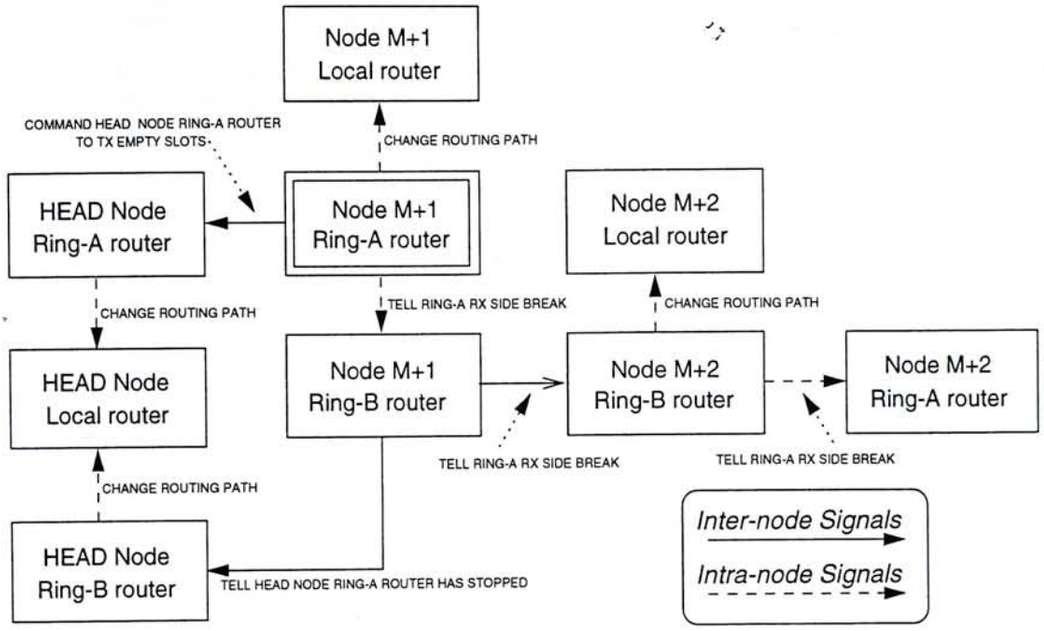


Figure 2.8: Diagram shows an example of Fault Recovery Signaling procedures

packets will be buffered in the input buffers.

(c) Pesudo-Code of the Fault-tolerant Algorithm

Figure 2.9 has shown a pseudo-code of the fault-tolerant algorithm. In *main()*, a *do ... while* loop is used to represent the time schedule of the processor. In the implementation, the two detection events (timer expired and outofsync detected) will not be explicitly checked because they are being implemented as interrupt driven events, but for the reason of clearness, they are included in the *main()* program. The function of the *Restoration()* routine is to re-configured the router-node to complete the broken path caused by the failure. The *Read_status()* routine handles all internodal communications and calls the *Restoration()* and *Claim_Head()* routines depending on the failure type.

```
Fault-Tolerant Algorithm
/* Begin */
main()
{
 Network Initialization
 Wait for registration packet to register
 Do
    if (timer expired) then Read_Status()
    if (out-of-sync detected) then Read_Status()
    Other network routines ...
 while (node is working)
}
Read_Status()
{
 switch(Status)
 /* Out-of-sync detection, suitable for all MAC protocols */
   case 'any of the receiving links is out-of-sync'
        if (Ring-A Receiving link is failed) then
          status(Ring-A receiving link) = 'failed'
        else
          status(Ring-B receiving link) = 'failed'
        Restoration()
   case 'both of the receiving links are out-of-sync'
        Claim_Head()
   /* Packet-non-arrival detection, suitable for slot-access protocols */
   case 'no valid packet arrives in period T1'
        Repeat
          Send Test_Link command to neighboring node
          status = 'failed'
          Wait for response until timeout
            if (response = 'yes') then (status = 'ok')
              counter = counter - 1
        Until ((counter = 0) or (status = 'ok'))
        if (status = 'failed') then Restoration()
   case 'no valid packet arrives from both rings in period T1'
        Claim_Head()
   case 'Test_Link command is received'
        Send Link_Ok command to neighboring node
   case 'Claim_Head command is received'
        if (Registration No. in command > Local Registration No.) then
          Send Objection to the node issuing the command
}
Restoration()
{
 if status(Ring-A Receive Link)='failed' then
    Wrap traffic to be sent on ring-A to ring-B instead
 if status(Ring-B Receive Link)='failed' then
    Wrap traffic to be sent on Ring-B to Ring-A instead
}
Claim_Head()
{
 Broadcast the Claim_Head command to the network
 Wait for objection until timeout
 if (no objection from other nodes) then
    Reconfigure as Headnode and re-initailize
 else
    Wait for empty slots
}
/* End */
```

Figure 2.9: Pesudo-code of the Fault-Tolerant Algorithm.

2.6.4 Design and Implementation of the Auto Healing Algorithm

Similar to the fault-tolerant algorithm, the operations of the auto-healing algorithm are also divided into two phases - the *Sensing Phase* and the *Healing Phase.*

(a) Phase I - Sensing Phase

When faults occurred, the healthy neighbours around the faults will continuously monitor the status of the faults (e.g., connectivity of the failed links) by sending *Status_Request* commands to the disconnected nodes and waiting for their response. If the responses from the disconnected nodes can be received, it implies that the faulty components have been replaced/repaired and returned back to the normal operation state. This sensing process will run continuously until the faulty components resume operations. The successfulness of this sensing algorithm has already been demonstrated in the CUM LAUDE NET.

(b) Phase II - Healing Phase

After the faults are fixed, the re-connected node(s) will response to the *Status_Request* commands send by their neighbours. The nodes around the faults will then alert that the faults have been fixed and they should restore their routing configurations by calling the *healing()* process. In the healing process, the operation of the re-connected node and its neighbours will be temporarily suspended, the logical paths will be re-arranged and the state variables will also be reset. In the period of suspension, the arriving packets will temporary be

stored in the FIFO queues of the nodes.

The *healing()* process implemented in the CUM LAUDE NET requires the usage of inter-communications in the router nodes. The signaling scheme of the healing process is depicted in Figure 2.10. The links between Router-Node M and M+1 are broken and *Status_Request* headers are sent between the ring routers in the same ring (Figure 2.10(a)). Without the loss of generality, the faulty path of ring-A is assumed to be replaced first and Ring-A router of Router-Node M will receive the *Status_Request* from the Ring-A router of Router-Node M+1. It will then try to signal the Ring-B router of Router-Node M+1 by sending it a *Rx_Link_Up* header through the Ring-B router on its node (Figure 2.10(b)). When the faulty path of ring B is also repaired, Ring-B router of Router-Node M+1 will receive the *Rx_Link_Up* signal and send a *Both_Link_Up* command to its neighbour node to inform it to restore the original logical path. The same operation occurs in Router-Node M when it receives the *Rx_Link_Up* signal (Figure 2.10(c)).

(c) Pesudo-Code of the Auto-healing Algorithm

The pesudo-code of the AH algorithm is illustrated in Figure 2.11. It directly reflects the implementation of the procedures and should be easily ported to any dual-ring networks.

2.6.5 Network Management Signals and Restoration Times

Table 2.1 summarizes the signals involved in the implementation of the fault-tolerant and auto-healing algorithms. As shown, a two-byte short message is sufficient for the implementation of the algorithms. Such a short restoration

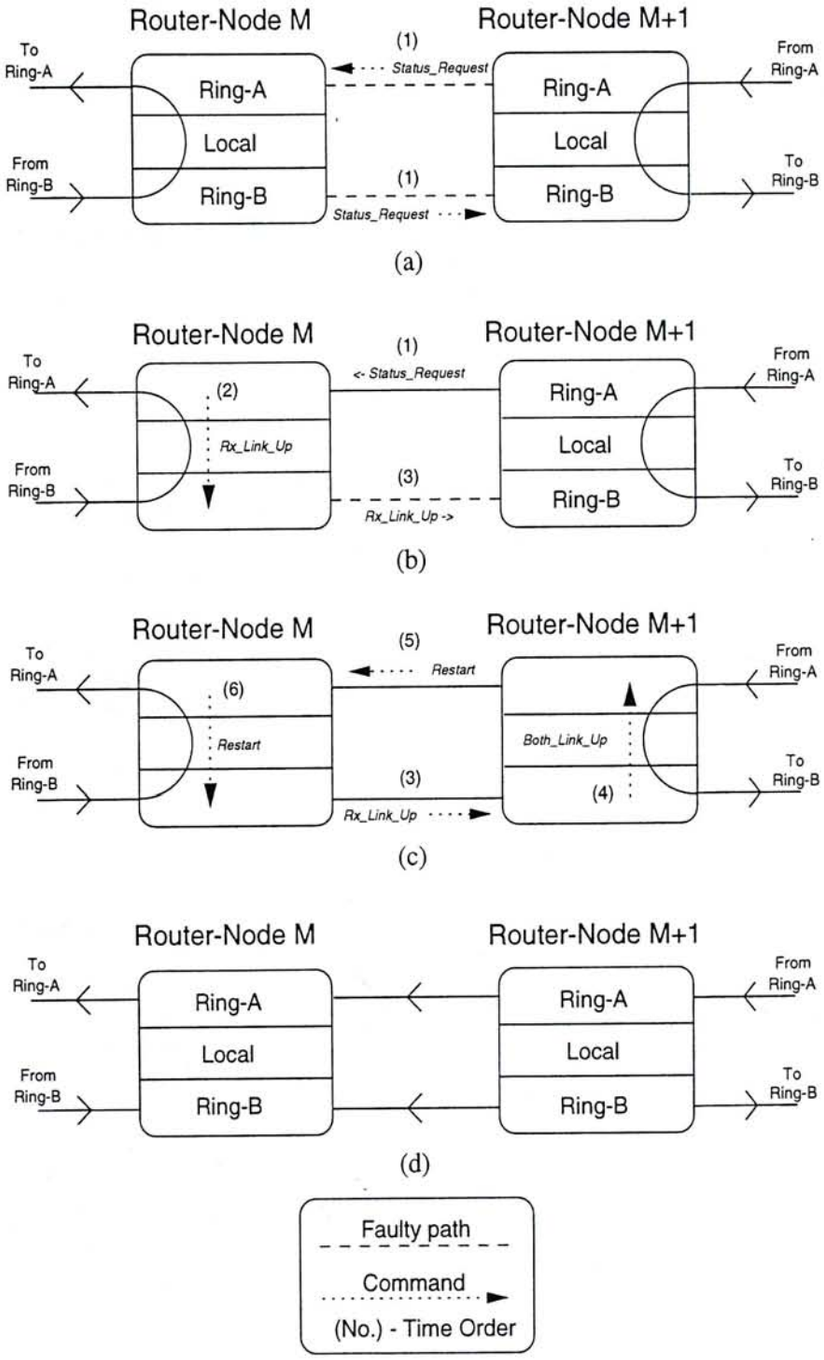


Figure 2.10: An example illustrates the Healing Signals.

Auto-Healing Algorithm

```
/* Begin */
sensing()
{
   Send Status_Request to neighboring nodes
   Monitor if any response has been sent back
   if (response is from in the same node) then
     switch (response)
        case 'Rx_Link_Up':
             Send Rx_Link_Up message to neighboring node
        case 'Restart':
             healing()
   else /* response is from neighboring node */
     switch (response)
        case 'Status_Request':
             Send Rx_Link_Up message to neighboring node
        case 'Rx_Link_Up':
             Send Restart command to neighboring node
   healing()
}

healing()
{
   Suspend operation of the node
   Store up incoming packets to transmission FIFO queues
   Reconstruct logical paths and reset state variables
   Resume normal operation
}
/* End */
```

Figure 2.11: Pesudo-code of the Auto-Healing Algorithm.

message is crucial in failure recovery procedures since it would not impose much extra work load on the damaged network and thus it can prevent the network from congestion.

Control Signals	No. of bits needed
Test_Link	1
Link_Ok	1
RingA_Rx_Break	1
RingB_Rx_Break	1
Change_Hop_Path	2
Status_Request	1
Rx_Link_Up	1
Both_Link_Up	1

Table 2.1: A summary of the control signals used in fault-tolerant and auto-healing algorithms of the CUM LAUDE NET and the number of bits required to represent them.

In order to show the efficiency of the FT and AH algorithms, Table 2.2 has shown the detection and restoration times of the algorithms in the CUM LAUDE NET. It is found that the total time required to complete the fault-recovery and auto-healing procedures is less than a millisecond.

Parameters	Value
packet non-arrival detection time	110.00 μs
out-sync detection time	55.00 μs
Message forwarding time of node	12 μs
FIFO delay	0.12 μs
Ring wrap time	3.00 μs
Node initialization time	6.10 μs
Router communication time	1.95 μs
Link capacity	100 Mb/s
Message traveling speed in links	1.99e8 m/s
Transmitter latency	0.74 μs
Receiver latency	1.22 μs

Table 2.2: Detection and Restoration Times of the algorithms in the CUM LAUDE NET. They are obtained from the measured value in the network prototype and hardware specifications given by network component manufacturer.

Restoration Techniques		Type of Networks		
		FDDI	SONET	CUMLAUDE
Dedicated Facility Restoration	Tech	Dual Homing	Diverse Protection(APS)	–
	Equip	DCS	1:N APS	–
	Time	second - minute	around 50ms	–
	Tech	Protective Switching	Self Healing Rings	–
	Equip	Nodal Bypass Switch	Add/Drop Multiplexer	–
	Time	less than a second	around 50ms	–
Dynamic Facility Restoration	Tech	Ring-Wrapping	Reconfigurable Mesh	Ring-Wrapping
	Equip	DCS	DCS	Reconfigurable Router
	Time	second - minute	second - minute	few hundred μs

Table 2.3: A comparison of restoration techniques for some common ring technologies based on the techniques used (Tech), the equipment required (Equip) and the recovery time (Time).

2.6.6 Comparison of fault-tolerance features of other networks with the CUM LAUDE NET

Among the existing fault-tolerant ring networks, two most well known technologies are the Fiber Distributed Data Interface (FDDI) and Synchronize Optical Network (SONET). A table comparing the restoration techniques of these two networks with the CUM LAUDE NET is shown in Table 2.3. As shown in the table, network fault tolerance can be provided either by dedicated facility restoration or dynamic facility restoration. It shows that dedicated facility generally provides faster restoration and the restoration time is ranges from micro-second to a few second. In the dual-ring technologies compared, CUM LAUDE NET can provide the fastest restoration.

2.7 Chapter Summary

In the Chapter, a high speed multimedia network - CUM LAUDE NET is introduced. CUM LAUDE NET is a multi-gigabit/sec hierarchical dual-ring which

consists of two hierarchies. The level-1 hierarchy (100 Mb/s) dual-ring network has been successfully deployed. Various multimedia applications (e.g., video conferencing) have been developed for the network.

Besides, a novel and efficient fault-tolerant and auto-healing algorithms for dual-ring networks have been proposed. These algorithms provide the functions of fault detection, fault recovery, hot replacement of failure components and automatic re-configuration. By using these algorithms, a highly reliable and survivable network can be produced. The successfulness of these algorithms have been proved by implementing them on the CUM LAUDE NET.

Chapter 3

Overview of the Asynchronous Transfer Mode (ATM)

In this chapter, we will present a brief review on the ATM technology to facilitate the illustration of the design of the ATM Connectionless Gateway.

3.1 Introduction

Asynchronous Transfer Mode (ATM) is a high-speed *connection-oriented* network technology[11, 12, 13] which employs a switching and multiplexing technique. The basic data transfer unit in ATM is called "cell", which is a fixed length packet[14]. With the connection-oriented nature, communications over the network require the pre-establishment of virtual connections with the admission control of the ATM switches[12, 13]. The path information is stored in each cell header as a combination of Virtual Path Identifier and Virtual Channel Identifier (VPI/VCI). Based on this pair of identifiers, the ATM switches can

switch (route) the incoming cells to their corresponding destined output port. This virtual connection is called a Virtual Circuit or Virtual Channel(VC)[13]. The VC can be created either manually, which is namely as Permanent Virtual Circuit (PVC), or upon demand by the signalling protocol, which is then called Switched Virtual Circuit (SVC). The type of connections to be used depend on the requirement of the users and the functionality provided by the network and the hosts.

Currently, the major organizations working on ATM standardization are the ATM Forum[1], the Telecommunication Standardization Sector of International Telecommunication Union (ITU-T)[2] and the Internet Engineering Task Force (IETF).

3.2 ATM Network Interfaces

Figure 3.1 depicts a typical ATM network which consists of private switches and end stations connected to a public ATM Network. Two types of interfaces are defined in ATM - UNI (User-Network Interface) and NNI (Network-Network Interface). In the UNI 3.1 specification[15], UNI comprises *Public UNI* and *Private UNI*. As stated in UNI 3.1, "Public UNI is typically used to interconnect an ATM user with an ATM switch deployed in a public service provider's network." and "Private UNI is typically used to interconnect an ATM user with an ATM

[1]The ATM Forum is a consortium including memberships of more than 750 companies representing all sectors of the communications and computer industries, as well as a number of government agencies, research organizations and users. This statistics is referenced from the ATM Forum Home Page http://www.atmforum.com

[2]Previously is known as CCITT

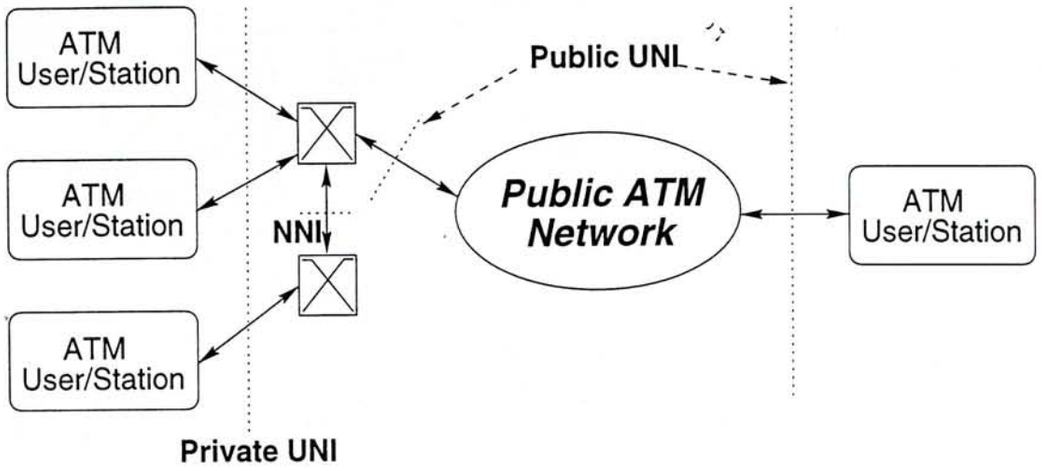


Figure 3.1: ATM Network Interfaces.

switch that is managed as part of a corporate network.". NNI is used to interconnect ATM switches within an ATM Network[16]. The difference between NNI and UNI is that routing information is exchanged between switches in NNI.

3.3 ATM Virtual Connections

As stated in the ITU-T Recommendation I.150[17], "Connection identifiers are assigned to each link of a connection when required and released when no longer needed". The connection identifiers referred in the recommendation are the VPI/VCI pairs. Figure 3.2 illustrates the relationship between the VPI/VCI pairs.

The Transmission Path is a physical medium (e.g. optical fiber and UTP) that is used to interconnect the UNIs or the NNIs. Inside the Transmission Path, there is one or more Virtual Paths which are identified by the VPIs[3]. Each VP contains one or more Virtual Channels which are identified by the VCIs. The

[3]there are two VPs in the example with VPI=1 and 2 respectively

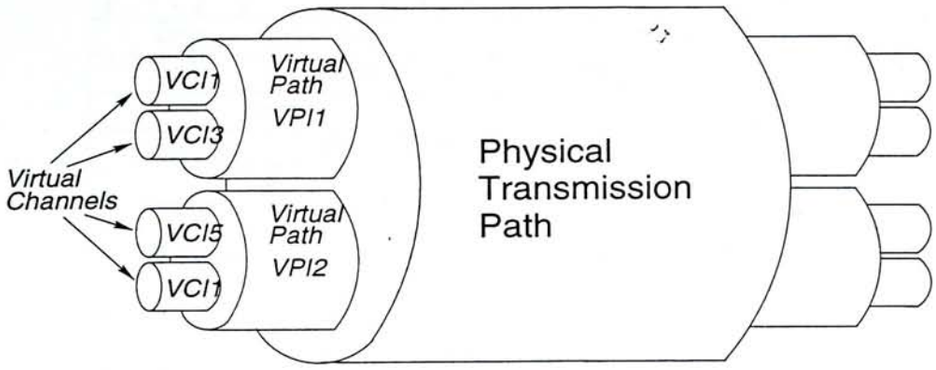


Figure 3.2: An example reveals the relationship between VPI and VCI.

VCI values in different VPs are independent to each others and any two VPs can contain two VCs with identical VCI value. Based on this concept, switching in ATM can be performed either on the virtual path or on the virtual channel.

3.4 ATM Cell Format

Figure 3.3 illustrates two types of ATM cell format are defined respectively for the UNI and the NNI. Only the first 5-byte cell header is used in performing cell-switching while the 48-byte payload is carried without any interpretation over the network. Among the six fields, only the VPI and the VCI fields are manually assigned in creating the PVCs. We should be careful in selecting the VCI number because there are some reserved values[4].

[4]The VCI values range from 1 to 31 are reserved. Therefore, the smallest configurable VCI value is 32.

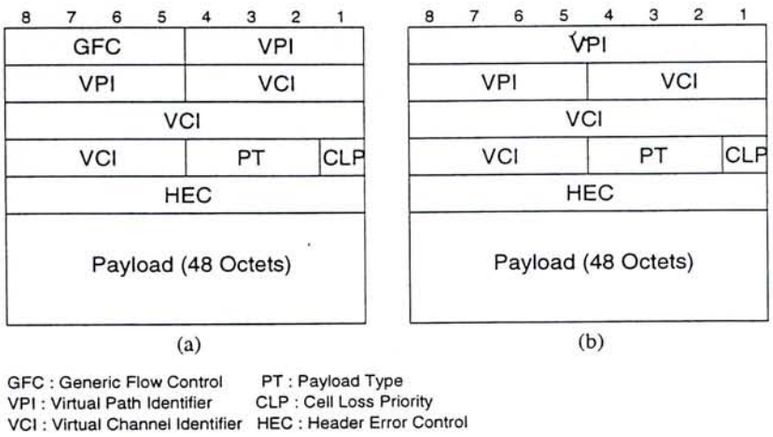


Figure 3.3: ATM Cell Format of (a) UNI and (b) NNI.

3.5 ATM Address Formats

ATM Forum has defined three types of private UNI address format as shown in Figure 3.4 and they are all 20 bytes in length. The ATM address format is modeled after the OSI Network Service Access Point (NSAP) address. Each address format is characterized by three identifiers: Authority and Format Identifier (AFI), Initial Domain Identifier (IDI) and Domain Specific Part (DSP).

AFI is used to identify which type of address format is being used and the respective values of these formats are shown in Table 3.1. Both IDI and DSP are assigned with values depending on the type of address format.

For Public Networks, Section 5.1.3.2 of UNI 3.1[15] specified that the public UNI should support either the E.164 address structure, aforementioned Private UNI Address Formats, or both of them.

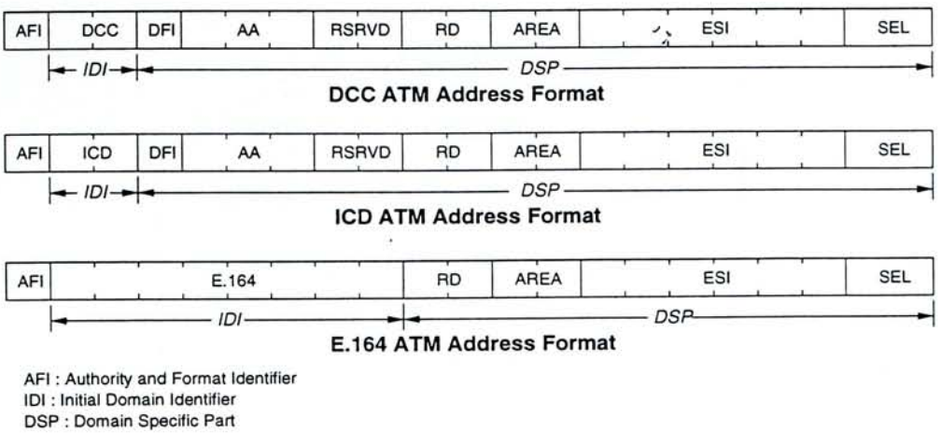


Figure 3.4: ATM Address Formats.

AFI	Format	Specified by
39	DCC ATM Format	Country
47	ICD ATM Format	Internation Organization
45	E.164 ATM Format	ISDN/Telephone number

Table 3.1: Classifications of ATM Address Formats

3.6 ATM Protocol Reference Model

Figure 3.5 shows the 3-dimensional protocol reference model (PRM) defined for the B-ISDN/ATM. This model is based on the standard developed by the ITU-T Recommendation I.121[18, 15]. The PRM consists of three planes: user, control and management planes. Two functions are defined for the *Management Plane* - "Plane Management[5]" and "Layer Management[6]". *User Plane* provides functions to transfer user information. *Control Plane* is responsible for the call and connection controls.

[5]It perform functions related to a system as a whole and provides coordination between all planes[5, 19]

[6]It perform management functions relating to resources and parameters residing in its protocol entities[5, 19]

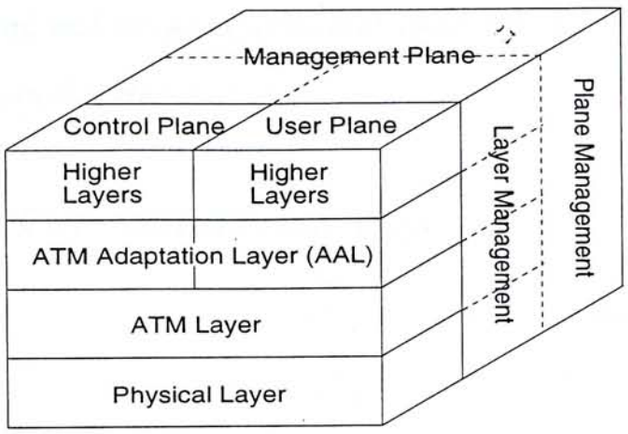


Figure 3.5: B-ISDN/ATM Protocol Reference Model.

3.6.1 The ATM Layer

ATM Layer serves as an interface for the AAL and the Physical Layers. It is independent of the Physical Layer. When it receives a cell from the AAL Layer, it appends a cell header to the cell and passes it to the Physical Layer for transmission. Conversely, when it receives a cell from the Physical Layer, it extracts the cell header and passes it to the AAL Layer. Besides, it provides the following functions:

- Generic Flow Control;
- Cell Header Generation/Extraction;
- Cell VPI/VCI Translation (in the ATM switches only);
- Cell Multiplex and Demultiplex.

Except the HEC field in the cell header, all the other fields are managed by the ATM Layer. Therefore, the ATM Layer in essence provides traffic management functions including: cell loss priority marking (i.e., CLP), congestion indication (i.e., PT) and generic flow control access (i.e., GFC). Moreover, it

buffers the incoming and outgoing cells and monitors the transmission rate so as to conform the service contract[13].

3.6.2 The ATM Adaptation Layer

ATM Adaptation Layer (AAL) interfaces the higher layer protocols to the ATM Layer. The design of the AAL is to enhance the adaptation of services provided by the ATM Layer to meet the requirements of the higher layers[19]. The main functions of the AAL are to segment and re-assembly the user data into or from the 48-byte ATM cells.

The AAL is divided into two sublayers: the Convergence Sublayer (CS), and the Segmentation and Reassembly Sublayer (SAR). The CS is further divided into Service Specific (SS) and the Common Part (CP) Convergence Sublayers (i.e., SSCS and CPCS).

(a) Classification of AAL Services

In order to define specific AAL protocols, ITU-T has proposed a "Classification of Services" for the AAL to handle different kinds of traffic. The proposed classes are shown in Figure 3.6. The first three rows are the attributes of the service classes and the fourth row shows the defined AAL Protocols for the respective classes.

AAL Protocols can be classified into five types: AAL-1 to AAL-5. Initially, only AAL-1 to AAL-4 have been defined and mapped to the service classes A to D. AAL 3 was developed for the connection-oriented service while AAL 4 was developed for the connectionless service. However, it was later realized that there were too many common parts between AAL-3 and AAL-4. Thus they were

	Class A	Class B	Class C	Class D
Timing relation between Source and Destination	Required		Not Required	
Bit Rate	Constant	Variable		
Connection Mode	Connection-oriented			Connection-less
AAL Protocol(s)	AAL1	AAL2	AAL3/4 or AAL5	AAL3/4 or AAL5

Figure 3.6: ATM AAL Service Classes.

combined and became AAL-3/4. Since the overhead in AAL-3/4 is large[7], thus it stimulates the development of a light-weight AAL – AAL-5.

There is no stipulation that an AAL designed for one class of traffic cannot be used for another[13]. Nowadays, AAL-5 becomes the predominant AAL adopted by the vendors to support all classes of services in their products. It is the standard AAL used in the ATM signalling system[8]. Moreover, it is the default AAL used in ATM internetworking such as the IP over ATM.

The ATM Forum has also recommended a set of *ATM Layer service classes* to cover the QoS (Quality of Services) requirements of the traffic expected to be run over the ATM network[20]. The QoS requirements are: Cell Loss Ratio (CLR), Cell Delay Variation (CDV), Cell Transfer Delay (CTD), Peak Cell Rate (PCR), Cell Delay Variation Tolerance (CDVT), Sustainable Cell Rate (SCR), Burst Tolerance(BT) and Minimum Cell Rate (MCR). Five classes of ATM Layer Service are defined as: Constant Bit Rate (CBR), Variable Bit Rate(VBR, Real Time and Non-real Time), Unspecified Bit Rate (UBR) and Available Bit Rate (ABR). The UBR service is designed to be used in computer communications

[7]Only 44-byte out of 48-byte payload are used for user data.

[8]Interface with the Service Specific Convergence Sublayer (SSCS) and the Service Access Point (SAP) which is called Signalling AAL (SAAL).

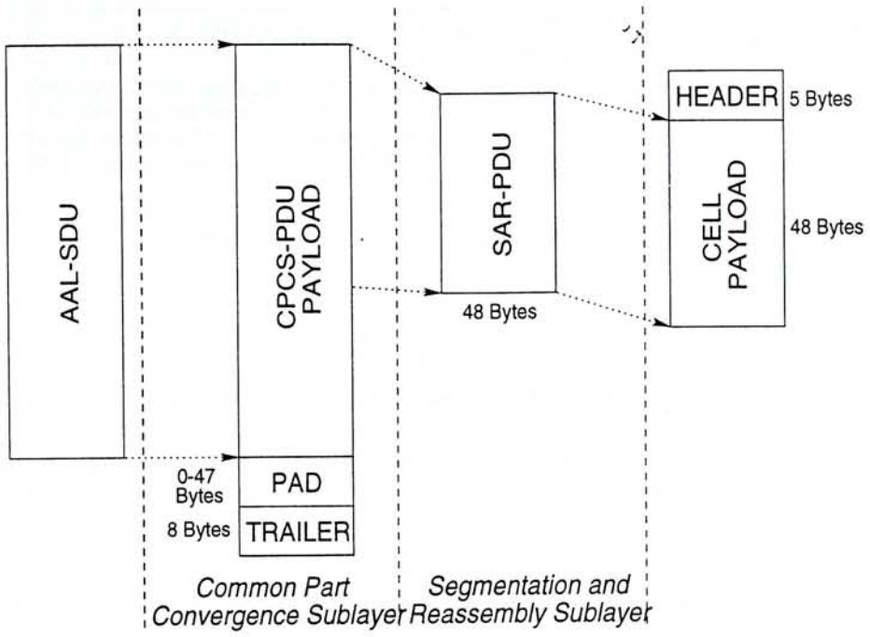


Figure 3.7: AAL-SDU Processing in AAL-5.

such as file transfer. This service does not provide any QoS guarantee at the ATM level. So the higher layer protocols such as Transmission Control Protocol (TCP) would be required to provide the guaranteed service. The ABR is one of the hot topics discussed in the ATM community. It is also designed to be used in the applications with vague throughputs and delays requirements.

(b) Data Processing in AAL-5

Since only AAL-5 is used in the implementation of the Connectionless Gateway, thus more details about this layer will be presented in this section. The processes of service data unit (SDU) encapsulation and segmentation in AAL-5 are illustrated in Figure 3.7, while the contents in an AAL-5 Protocol data unit (PDU) is shown in Figure 3.8. Features provided by AAL-5 will be described in the following parts.

(i) Error Detection and Correction

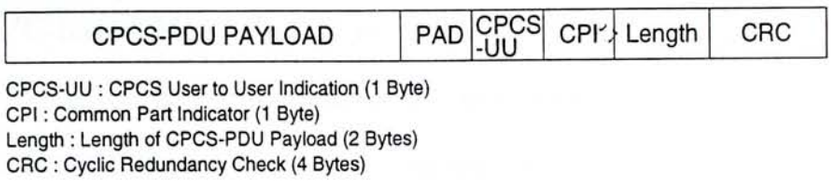


Figure 3.8: AAL-5 Protocol Data Unit.

It uses two fields (Length and CRC fields) in the trailer of the CPCS-PDU to perform the functions of error detection and correction. The Length field is used to determine whether the payload size of the received CPCS-PDU is correct. The CRC field carries the result of the CRC calculation. As illustrated, the error detection and correction mechanism of AAL-5 is simple. Thus, it is assumed that AAL-5 should be used in transmission media with low bit error rate.

(ii) Multiprotocol Encapsulation

Above the AAL, there are multiple higher layer protocols that request services from the network. Therefore, a mechanism is required to tackle the problem of multi-protocols supported over the ATM Layer.

In RFC 1483[21], a scheme called "Multiprotocol Encapsulation over ATM Adaptation Layer 5" is proposed to solve the multiprotocol encapsulation problem. In this document, two schemes called "LLC/SNAP encapsulation" and "VC-based multiplexing" are proposed.

In the LLC/SNAP scheme, the CPCS-PDU payload is encapsulated with the IEEE 802.2 LLC and 802.1a SNAP headers before it is passed to the SAR Layer. The protocol related information can be carried in these headers. The shortcoming of this scheme is that it does not support cell interleaving and only allows packet interleaving.

In the VC-based Multiplexing scheme, a VC is created for each protocol. Since a VC only carries information for one specific protocol, thus does not require any extra-header. The shortcoming of this scheme is that it uses the VCs inefficiently.

Nevertheless, AAL-5 is an efficient protocol with little overhead and easy to implement. As a result, AAL-5 is currently the most widely used AAL in the ATM community and it is sometimes regarded as "Simple Efficient Adaptation Layer (SEAL)".

3.7 ATM Signalling

In an ATM network, each user will establish a "Signalling Channel" to the network to exchange control information. It provides functions such as dynamic establishment or disconnection of the VCs and request of network resources. The VPI/VCI pair now reserved for this channel is 0/5.

Nowadays, most of signalling implementations are based on the ATM Forum's UNI Signalling Specification[18, 15]. This signalling standard is a variance of the ITU-T Recommendation Q.2931[9]. In the definition of the UNI Signalling in UNI 3.1, it supports both point-to-point[10] and point-to-multipoint connections[11].

A Signalling ATM Adaptation Layer (SAAL) has been defined for ATM Signalling. This SAAL provides reliable transport of signalling message between

[9]Q.2931 is based upon Q.931, the signalling standard for the N-ISDN.

[10]Point-to-point connection is a bi-direction or uni-direction connection between a pair of end-systems.

[11]Point-to-multipoint connection is a uni-directional one to many connection. One party is served as root and the other parties joint the session as leaves. Since only the root is allowed to transmit data, thus this type of connection is uni-directional.

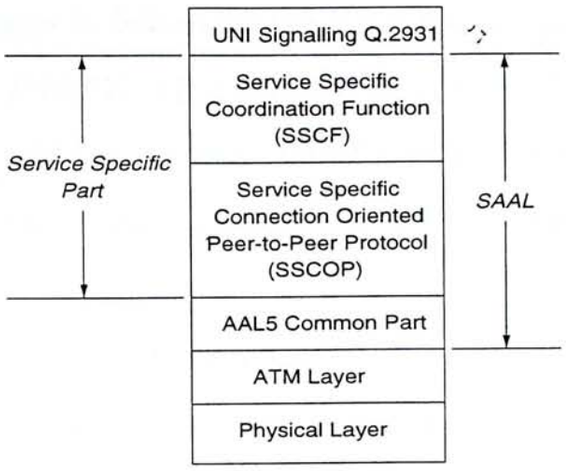


Figure 3.9: Structure of the Signalling ATM Adaptation Layer.

peer entities over the ATM Layer[15]. The structure of the SAAL is depicted in Figure 3.9. AAL-5 is currently adopted as the Common Part. The Service Specific Part consists of the Service Specific Coordination Function (SSCF) and Service Specific Connection Oriented Protocol (SSCOP). Since AAL-5 only provides unassured information transfer, the SSCOP is used to complement the lack of assurance of AAL-5 by providing functions to recover the lost or corrupted SDUs. SSCF and SSCOP are defined by Q.2130 and Q.2110 respectively.

3.7.1 ATM Signalling Messages and Call Setup Procedures

In Section 5.3 of UNI 3.1[15], a full set of signalling messages have been defined for different types of calls and connection controls. A set of messages is defined for the ATM point-to-point calls and connection control: *ALERTING, CALL PROCEEDING, CONNECT, CONNECT ACKNOWLEDGE, RELEASE, RELEASE COMPLETE, SETUP, STATUS* and *STATUS ENQUIRY.*

Another set of messages is defined for the Point-to-multipoint calls and connection control: *ADD PARTY, ADD PARTY ACKNOWLEDGE, ADD PARTY REJECT, DROP PARTY* and *DROP PARTY ACKNOWLEDGE.* The last two messages are called the Global Call Reference: *RESTART* and *RESTART ACKNOWLEDGE.*

Procedural modules have been defined with the messages, e.g., *Call Setup Procedure.* Please refer to Section 5.5[12] and Section 5.6[13] of the UNI 3.1[15] for details.

An example illustrates the "Call Setup Procedures" is shown in Figure 3.10. To initiate a call, the calling party first transmits a SETUP message to the called party. This SETUP message contains the Information Elements such as called Party Address, AAL Parameters, QoS Parameters, traffic contract parameters, etc. This message will travel through the ATM network to the destination upon the called-party ATM address. The network will then assign any available VPI/VCI values for that connection or it may reject the call if enough resource is not available. The network will return a CALL PROCEEDING message to the calling party and send a SETUP message to the called party if the required service is permitted. If the service is not allowed, the Call Clearing Procedures will be invoked.

If the called party accepts the call, it will send a CALL PROCEEDING and CONNECT message back to the network. The CONNECT message contains the valid connection identifiers (i.e., VPI/VCI), in addition to the AAL parameters such as Broadband Low-layer Information, etc. Then, the network will send a CONNECT message back to the caller and a CONNECT ACKNOWLEDGE

[12]Call/Connection Control Procedures For ATM Point-to-point Calls
[13]Call/Connection Control Procedures For ATM Point-to-multipoint Calls

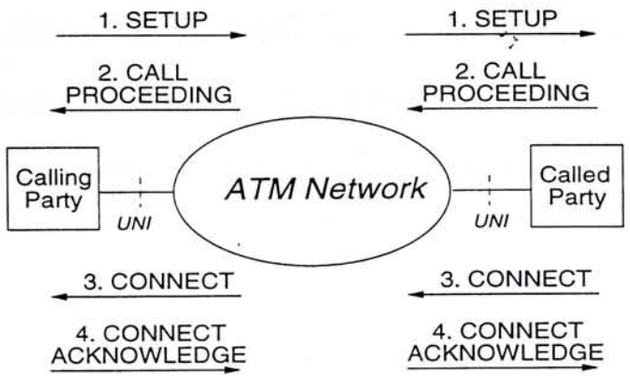


Figure 3.10: ATM Call Setup Procedures.

message to the called party. After the calling party receives the CONNECT message, it will send a CONNECT ACKNOWLEDGE message back to the network.

3.8 Interim Local Management Interface (ILMI)

The Interim Local Management Interface is defined in UNI3.0[18] and 3.1[15] as a temporary solution for controlling, managing, and exchanging status and configuration information between the connected ATM systems. This protocol makes use of the Simple Network Management Protocol (SNMP) associated with the ATM UNI Management Information Base (MIB) to provide information exchange facilities. A dedicated Virtual Channel with VPI/VCI values equal to 0/16 is assigned for the ILMI protocol to exchange data. This protocol provides functions for the nodes to exchange various configuration information such as the type of signalling used, hooks for network management autodiscovery[22] and *Address Registration.*

In the address registration procedure, the ILMI protocol transfers the ESI

(End System Identifier)[14] of the ATM end-station to the ATM switch. It then receives the remainder of the node's full ATM address (i.e., the network prefix) in return and completes the address in the end-station.

[14]Sometimes called the MAC (Media Access Control) address.

Chapter 4

Issues of Connectionless Gateway

4.1 Introduction

One of the challenges in interconnecting LANs and MANs with the ATM network is the support of *connectionless traffic* in the connection-oriented ATM network[23]. ATM operates in connection-oriented mode and its basic requirement is a virtual connection which must be created between the communicating parties. The conventional LANs are mostly *packet-switched*[11] networks (e.g. Ethernet, FDDI) and provide connectionless services. As ATM networks will most likely be installed in the public domain or campus backbone networks[3], the interconnection of LAN/MANs with ATM network will become a major networking issue (Figure 4.1). In order to accomplish the task of internet-working, a dedicated agent is required to create connections in the ATM networks for the

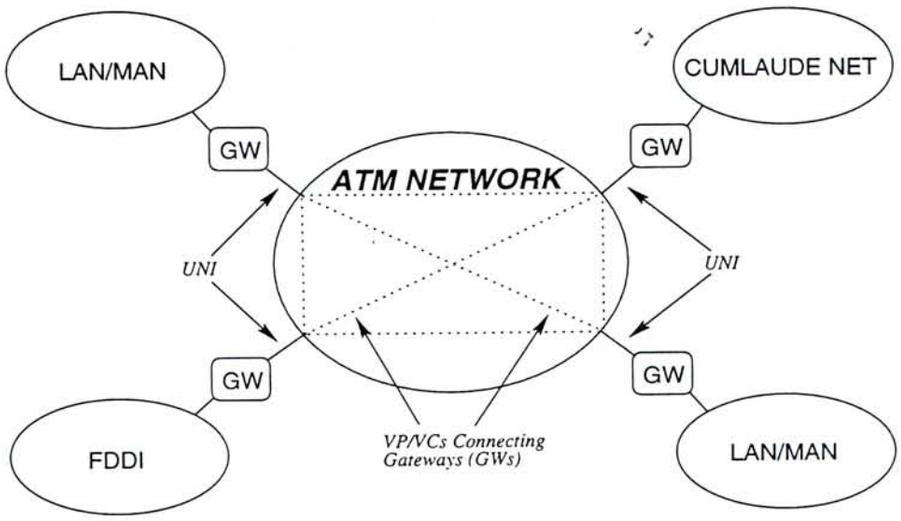


Figure 4.1: LAN/MANs and ATM Internetworking Scenario

LANs/MANs. It is one of the functions of the proposed *Connectionless Gateway*. Problems and solutions on internetworking the LAN/MANs and the ATM networks will be discussed in the following sections.

4.2 The Issues

The issues that needed to be considered are:

- ATM Internetworking;
- Connections Management;
- Protocol Conversion;
- Bandwidth Assignment[3];
- Packet Forwarding Modes;
- Traffic Control[3];

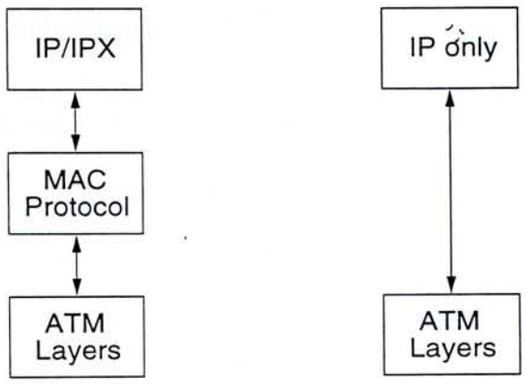


Figure 4.2: ATM Interworking Models - (a) LAN Emulation and (b) IP over ATM

- Handling of Mulit-cast and Broadcast Traffic[3].

Among these issues, we will focus on the first five in the following discussions. The last two are pertinent to ATM[3], thus they will not be discussed.

4.3 ATM Internetworking

The key to the success of ATM is the ability to support the existing protocols over ATM. Currently, two schemes have been proposed for supporting existing protocols over ATM. One is *IP over ATM* which is proposed by the Internet Engineering Task Force (IETF)[1] and the other one is *LAN Emulation (LANE)* which is proposed by the ATM Forum. Figure 4.2 depicts the models used in IP over ATM and LAN Emulation.

[1]The IP over ATM is also called Native Mode in [22].

4.3.1 LAN Emulation

In LAN Emulation, an extra MAC layer is placed on top of the AAL to emulate the behaviour of the traditional connectionless LAN. The LANE end-station is identified by a MAC address instead of the ATM address. The existing protocols will only address the end-stations through this MAC address. The packets are also required to encapsulate to an appropriate MAC packet format. As a result, the functions of the ATM layers are completely hidden by this MAC layer. The existing network layer protocols such as IP, IPX and NETBIOS can thus directly run over the ATM network in similar way that they run over the Ethernet.

The LAN Emulation standard - *LAN Emulation Over ATM Version 1.0*[24] is worked out by a technical committee in the ATM Forum called LAN Emulation Sub-working Group (established in November 1993[13]) which was released in January, 1995.

The LANE specification[24] has defined two emulation interfaces for the traditional LAN technologies: the IEEE 802.3/Ethernet and IEEE 802.5/Token Ring. The concept of Emulated LAN (ELAN) is also introduced in the specification. ELAN allows a physical network to be divided into multiple logical domains. Several Emulated LANs can be configured within one ATM network and the memberships in an ELAN are not constrained by the physical locations of the end-systems[24]. Bridges or Routers are used to connect multiple ELANs. In each ELAN, there consists of *LAN Emulation Clients (LE Clients or LECs)* and LAN Emulation Service (LE Service). The LE Service consists of a *LAN Emulation Configuration Server (LECS)*, a *LAN Emulation Server (LES)* and a *Broadcast and Unknown Server(BUS)*. These servers may be centralized or distributed over a number of stations. The LE Clients request services from the

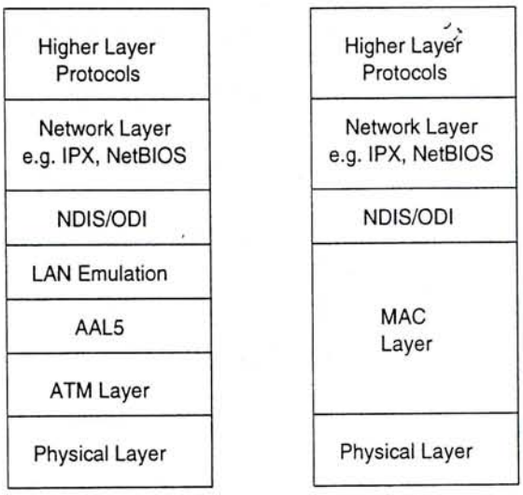


Figure 4.3: Protocol stacks of (a) ATM LAN Host and (b) Legacy LAN Host

LE Service through the LAN Emulation User-Network-Interface (LUNI).

Figure 4.3(a) shows the protocol stacks of an LANE ATM Host while Figure 4.3(b) shows the protocol stacks of a conventional LAN host. It illustrates that the ATM specific layers can be logically considered equivalent to the MAC layer of the conventional LAN host[22, 25, 26].

There are many good relevant references [22, 16, 13, 27, 28, 29] which described LANE in details.

4.3.2 IP over ATM

IP over ATM is also called the *Native Mode Protocol*[22, 28]. There is no extra MAC layer as an LANE has. The IP address is now directly mapped to the ATM address. The higher layer protocols are now directly communicating with the ATM layers and can use the features provided by the ATM network such as QoS.

Currently, this mechanism only supports the IP protocol run over ATM, but

other vendors, such as Novell has announced that they will develop a protocol known as Connection Oriented IPX which supports IPX protocol run over ATM[22]. Nevertheless, IP is the first protocol that is widely used on the Internet, thus it can provide "world-wide" connectivity.

Three RFCs have been proposed concerning IP over ATM: RFC 1626[30] - "Default IP MTU for use over ATM AAL5", RFC 1577[31] - "Classical IP and ARP over ATM" and RFC 1755[32] - "ATM Signaling Support for IP over ATM".

Since IP over ATM is used in our gateway implementation, thus it is further presented in details. There are four main aspects that should be considered in IP over ATM: *Multiprotocol Encapsulation*, *Default IP MTU for use over ATM AAL-5*, *Logical IP Subnet* and *ATM Address Resolution Protocol.*

(a) Multiprotocol Encapsulation

In RFC 1577, AAL-5 is chosen as the default AAL for transferring the IP over ATM packets. Thus, the schemes proposed by RFC 1483 can be adopted in carrying IP packets with multiprotocol over the ATM network. Among the proposed schemes (refer to Section 3.6.2), the LLC/SNAP scheme is the most commonly used. Figure 4.4 shows the format of a LLC/SNAP encapsulated IP over ATM packet and the well-defined value of each field in the LLC/SNAP header. The PID shown is defined for IP packet. When ARP packets are multiplexed on the VC, the PID value is 0x806.

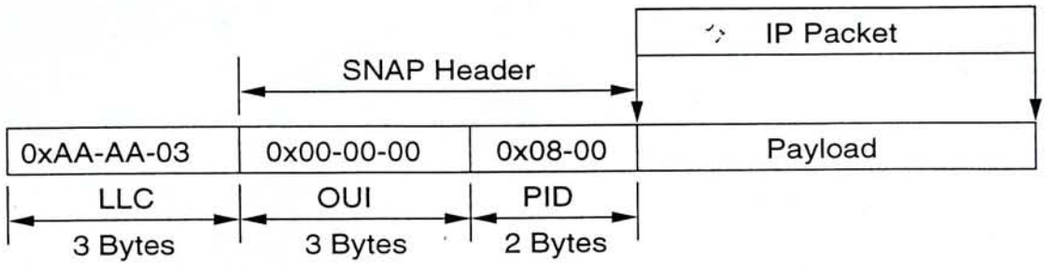


Figure 4.4: LLC/SNAP Encapsulated IP over ATM Packet Format

(b) Default IP MTU

RFC 1626[30] has defined the MTU (Maximum Tranfer Unit) of an IP over ATM packet using AAL-5 to be 9180 bytes[2]. With a larger packet size, the probability of IP packet fragmentation in the routers is lowered. This size is also aligned to the MTU defined for the IP over Switched Multimegabit Data Service (SMDS) packet which is also 9180 bytes. Thus, it facilitates the interworking of SMDS and ATM.

(c) Logical IP Subnet

In the traditional IP network, a bigger network would normally be divided into smaller networks called subnets. Each set of hosts in a subnet will be assigned the same subnetwork address. The assignment of these conventional IP subnets are restricted by the physical network.

In IP over ATM, the concept of "logical IP subnet" is applied. An ATM

[2]The MTU of the LLC/SNAP encapsulated IP over ATM packet is 9188 bytes.

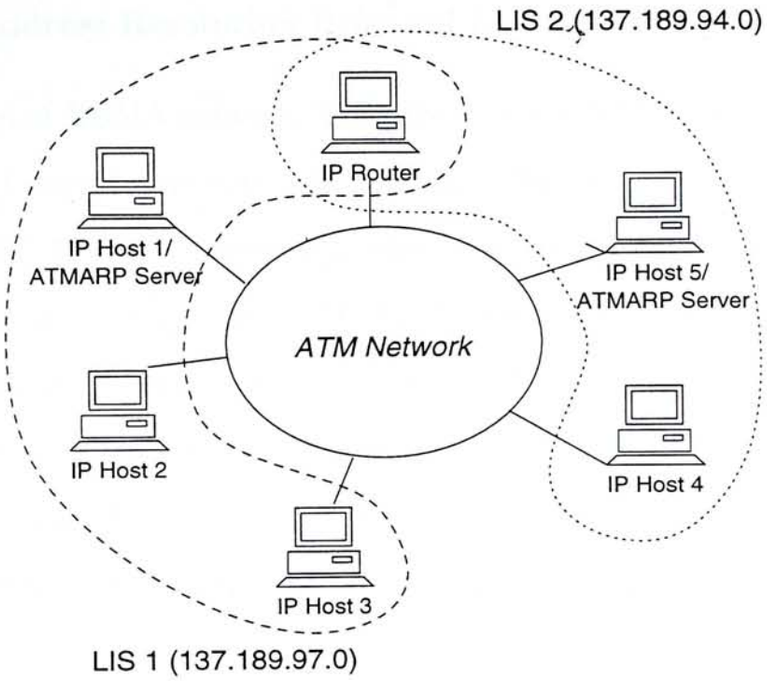


Figure 4.5: A simple ATM network shows the concept of LIS

network can be divided into multiple administrative domains[3]. Figure 4.5 illustrates an example of an ATM network with multiple logical domains.

The LISs formulated in IP over ATM networks follow the conventional IP model where the hosts in different LISs are required to communicate through the IP routers. This rule has restricted the performance of the IP over ATM networks because the IP routers are normally the bottle-neck. Therefore, other schemes are proposed on supporting direct connection between the hosts in different LISs of the Non-Broadcast Multi-Access(NBMA) networks such as ATM, frame relay or X.25. One of them is the Next Hop Resolution Protocol(NHRP). For further information, please refer to [22].

[3]All the hosts in one domain will have the same subnetwork address and at least one ATMARP server(to be discussed later) will be allocated per domain.

(d) ATM Address Resolution Protocol (ATMARP)

ATM is a kind of NBMA network. Thus the existing Address Resolution Protocol (RFC 826) cannot work over it because the ARP only works on broadcasting networks. RFC 1577 has proposed a scheme to resolve this address resolution problem. It suggests that one ATMARP Server[4] should be allocated per LIS which is responsible to resolve addresses. It also recommends that the ATMARP Server should be an IP workstation. In Figure 4.5, this ATMARP Server is shown to be situated in each LIS. Similarly, an ATM specific Inverse Address Resolution Protocol (ATMInARP)[5] is defined.

Figure 4.6 shows an example to illustrate the operations of the ATMARP and InATMARP protocols. Two operations scenarios[6] can be derived from this example: 1. ATM Station A is booting up; 2. ATM Station A is going to make a connection with Station B.

(i) System Boot Up Scenario

Firstly, the ATMARP Server must be started and its ATM address must be well-known to all the hosts in its LIS. Assume ATM Station A is now powered up and the following happens:

1. ATM station A creates a point-to-point connection with ATMARP Server by using SVC[7];

2. ATMARP Server will then be aware of the presence of ATM Station A

[4]But it does not preclude the usage of other robust approaches such as multiple servers per LIS.

[5]The broadcasting InARP is defined in RFC 1293.

[6]These scenarios only applicable to the systems use SVCs. If the hosts are using PVC, ATMARP is not required and only InATMARP will be implemented.

[7]To establish this VC, ATM signaling is required and this VC should use LLC/SNAP encapsulation.

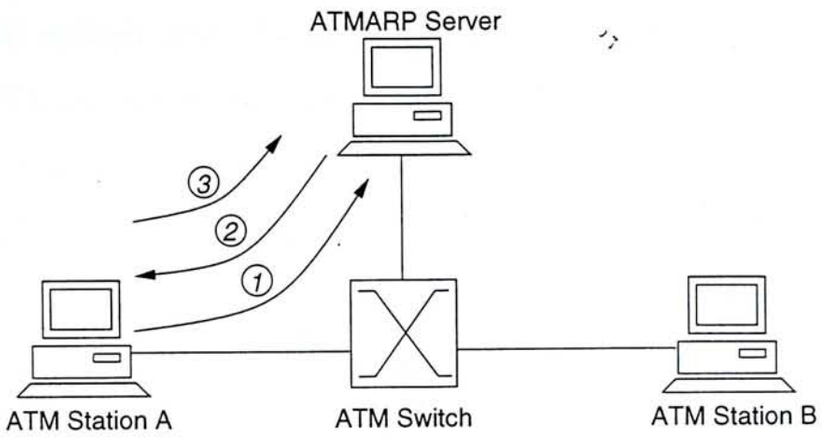


Figure 4.6: An example illustrates the operations of the ATMARP and InATMARP protocols.

and issue an InATMARP_request to request ATM Station A's IP address;

3. Finally, ATM Station A replies with an InATMARP_reply so that the server can update its ATMARP Table.

(ii) Scenario of VC Establishment

In this scenario, two steps are performed to resolve ATM Station B's IP address for ATM Station A.

1. ATM Station A sends ATM Station B's IP address in an ATMARP_request to ATMARP Server;

2. ATMARP Server receives this request, it checks whether ATM Station B's address entry is inside its ATMARP Table. If the entry exists, it will reply ATM Station A with the ATM address of Station B in an ATMARP_reply. Otherwise, an ATMARP_NAK[8] is replied to inform ATM Station A that the address cannot be resolved.

[8]ATMARP_NAK is used to distinguish the cases of server failure and unresolvable address.

In a PVC environment, the ATMARP server described is not required. It is because PVCs are created administratively and their VPI/VCI values are pre-assigned. Thus the ATM hosts do not need to know other ATM Addresses. However, as stated in section 6.1 of RFC 1577[31], the ATM host should have a mechanism to determine the PVCs it is connected to and uses the InATMARP_request messages to request other IP addresses.

In the ATMARP Server, an ATMARP Table is required to store the ATM-IP addresses information. The entries in this ATMARP Table are required to be refreshed/updated periodically in order to guarantee data integrity. RFC 1577 has defined this update period to be 20 minutes[31]. If the entry of a connected host does not have any enquiry in this period, the entry should then be updated by transmitting an InATMARP_request from the server to the connected host and wait for the reply. If there is no reply from that host, its entry will be deleted.

In each ATM client, there is a cached ATMARP Table. With this local cache, the time used in address resolution can be reduced. Similarly, this table is required to be updated periodically. The entries in this table are valid for a maximum period of 15 minutes as defined in RFC 1577. After the time period is expired, the entries will be updated by sending an ATMARP_request to the Server for a SVC entry and an InATMARP_request to the connected host for a PVC entry.

4.3.3 Comparing IP over ATM and LAN Emulation

In comparing IP over ATM with LAN Emulation, the following points can be considered:

1. Efficiency – IP over ATM is more efficient since it directly supports IP layer over AAL while LANE requires an extra MAC layer.

2. Latency – LANE provides the mechanism of using BUS to flood packets in parallel with address resolution to reduce the latency of packets arrival to the destination. IP over ATM has not considered this issue.

3. Multicast – LANE has been designed with the capability of supporting multicast and broadcast traffic while the "Classical IP over ATM" has not. However, there are some other proposals concerning the issue of multicast support in IP over ATM networks such as the Multicast Address Resolution Server(MARS)[9][33].

4. Multiprotocols Support – LANE is stronger in this aspect since it can support more protocols than IP over ATM. However, in [33] a modification of the IP over ATM specification was proposed to support other network layer protocols. They proposed to change the PID of the LLC/SNAP header from 0x800 to other values for other protocols, e.g. PID of value 0x809B for AppleTalk.

5. QoS Support – IP over ATM is a *native protocol* which can support QoS in the ATM networks. LANE does not consider the use of ATM QoS feature.

6. Complexity – We can hardly conclude on this aspect. In the current stage, IP over ATM is a simpler technology and more vendors have adopted this approach, such as the Fore System[10] and ATM on Linux.

[9]It controls the memberships of multicast groups and provides the capability of creating multicast mesh connections or using multicast server.

[10]However, they have their own implementation approach[34]

These are some general comparisons and the arguments may not hold in other systems. Therefore, we can hardly find an absolute conclusion on the question of "Which approach is better ?". However, from my own point of view, IP over ATM is a better solution. Firstly, the IP protocol is directly implemented over the ATM Layer which does not rely on the traditional network layers as LANE does (i.e., No MAC layer). Thus, the higher layer applications can take advantages of the unique features provided by the ATM networks such as "Guarantee of Service" in real-time data transmission. Secondly, the IP over ATM model is simpler than the LANE which means that the implementation is more simpler and efficient. This is an advantage to the construction of the Gateway since the Gateway has to process many number of IP packets per second and an efficient protocol processing system can enhance the packet processing capability of the Gateway. The delay caused in the transmission of packets can also be minimized. Moreover, IP over ATM works well with the world-wide "Internet" protocol. Thus, it enhances the possibility of upgrading the Internet to use the ATM technology and the existing IP networks/applications do not need to be changed.

4.4 Connection Management

In Figure 4.1, different types of LANs are shown to be interconnected through an ATM Backbone with the Connectionless Gateways[11]. It shows the case which the Gateways are interconnected by a *Full VP/VC Mesh*[23, 35, 36, 37]. There are other approaches that may be used in interconnecting the Gateways and they

[11]Some periodical calls them *Interworking Units (IWU)*.

will be presented in this section. The ITU-T has recommended two approaches in providing connectionless service in an ATM network: *Indirect Approach* and *Direct Approach*[38]. The approach employed in Figure 4.1 can be classified as Indirect Approach.

4.4.1 The Indirect Approach

In Indirect approach[12], connectionless service is offered by the network through the use of virtual connections established between each pair of ATM Interworking Units(IWUs)[38] or Gateways(GWs)(In Figure 4.1 is shown an example of this approach.). The Gateway becomes the interfacing agent between the connectionless LANs and the connection-oriented ATM network.

Two types of connections can be created between the IWUs/GWs, the Permanent/ semi-permanent Virtual Connection (PVC) and the Switched (Demand) Virtual Connection (SVC). The use of PVCs to interconnect the Gateways is simple but inflexible. It is because the setup of PVCs requires administrative configuration and the scale of the network is limited by the numbers of PVCs. The advantage of using PVC is that the network topology can be controlled easily, for example, *Ring* topology can be used to connect the GWs.

Another alternative is using SVCs to interconnect the Gateways which the connections are created on demand. The disadvantage of this method is the introduction of overhead in latency in the connection setup procedures. During the connection setup period, the Gateway is also required to buffer the incoming packets. This buffering operation adds extra delay to the transmission and the memory requirement on the Gateway becomes stringent. Nevertheless, SVC is

[12]Also called connection oriented approach[14, 39].

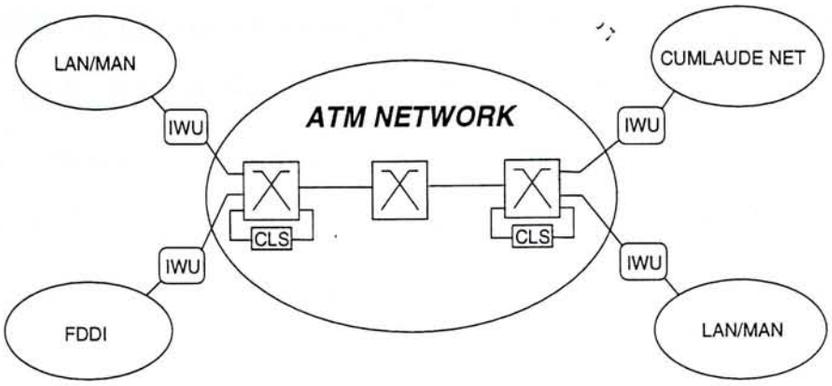


Figure 4.7: An example shows the configuration of the Connectionless Servers (CLSs) in an ATM/LANs Internetworks.

a more feasible solution in planning large network with many Gateways.

The network resources (especially the bandwidth) can hardly be managed in this approach because it is difficult for a Gateway to predict the characteristic of the bursty connectionless traffic. Thus the Gateways could hardly acquire the correct amount of bandwidth from the ATM network. This topic will be further explored in the last section.

This project has employed this approach in the implementation and the reasons of choice will become clear after the other approach is narrated. Both LANE and IP over ATM also employ a similar approach as the Indirect Approach.

4.4.2 The Direct Approach

Direct Approach[13] provides connectionless service through the Connectionless Servers (CLSs). Figure 4.7 depicts a network installed with CLSs. This figure is similar to Figure 4.1, except that the ATM switches are equipped with CLSs.

The main functions of the CLSs are to routing cells and manage the usage of bandwidth. The GWs/IWUs provide the services of packet segmentation

[13]Also called connectionless approach[14, 39].

and reassembly for the connectionless LANs. The GWs are now interconnected through the CLSs with either PVCs or SVCs. While the CLSs can be interconnected with different topologies such as ring, complete mesh, hierarchy, etc[38].

Although the CLSs are located near the ATM switches as shown in Figure 4.7, there are two proposals in arguing where the CLSs should be placed. The first proposal suggested that the connectionless service function should be integrated inside the switch[14] and the second proposal suggested appending the service to the switch externally (i.e., the approach shown in the figure.).

A Connectionless Network Access Protocol (CLNAP) layer is placed on top of the ATM layers in the IWUs and CLSs to provide the function of datagrams encapsulation. The default AAL used by this approach is AAL3/4.

Currently, both the Connectionless Broadband Data Service (CBDS) as well as Switched Multimegabit Data Service (SMDS) use this approach in providing connectionless service within the B-ISDN[19].

4.4.3 Comparing the two approaches

As claimed in [38], the public carriers would like to choose the Direct Approach because:

- The IWU requires only one connection to the CLS and does not need to perform routing. Then, the IWUs can be managed by the users. The complexity on VCs management is reduced. For the Indirect Approach, there requires *n(n-1)/2* connections for a network of n users connected in a full mesh[15], but the number of connections is reduced to n for the Direct

[14]Called option A in I.364[38]
[15]i.e. $O(n^2)$

Approach[16].

- All connectionless traffic is aggregated onto a smaller number of VCs, thus the statistical multiplexing gain is improved.
- The public carriers measure the bandwidth consumed by the users easily.

Despite the direct approach is said to be better, it yet has disadvantages. The bottlenecks in the Direct Approach are both IWUs and CLSs, but the number of bottleneck is reduced to one for the Indirect Approach. The CLSs should have high processing power in order to keep in pace with the switch[38] and serve plenty numbers of IWUs. It thus implied the cost of the CLSs will be high.

In small scale implementation (e.g., within a department), the Indirect Approach should be the better solution because the high cost CLS is not required. Moreover, both IP over ATM and LANE can be employed in the networks using Indirect Approach. AAL-5 can be used in these two protocols, which is more efficient than the AAL that adopted by the Direct Approach (AAL-3/4).

4.5 Protocol Conversion

When packets transfer from one type of network to another, their format may have to be changed because the networks may work on different protocols. This process of changing is called "Protocol Conversion".

In the OSI PRM[17], Internetworking is done in the network layer (i.e., layer 3)[40]. However, it can also be done in other layers in regarding of network

[16] i.e., $O(n)$

[17] Although B-ISDN/ATM has defined its Protocol Reference Model(PRM), most discussions in the periodicals chose to use the conventional OSI PRM instead (respect to the user plane of B-ISDN PRM only). Therefore, the OSI PRM convention is used.

relaying. Generally speaking, network relaying devices can be classified into four categories: Repeater, Bridge, Router and Application Gateway. Out of the four categories, only the Bridge and the Router will be considered.

Bridge and Router are sometimes called "Gateway". Gateway with common LLC layer, but different MAC layer is classified as Bridge; Gateway with common network layer is classified as Router[41]. In Figure 4.8(a), the protocol stacks of a *bridge* works in a LANE network is shown and Figure 4.8(b) shows the protocol stacks of a *Router* works in an IP over ATM network. Actually, both bridge and router can be used in a LANE network because the LANE network works like an IEEE802.3/802.5 network. However, only IP router can be used in an IP over ATM network[18].

The protocol stack of the network components (IWUs, CLSs and ATM switch) used in a network employing the Direct Approach is illustrated in Figure 4.9. There are three main layers that should be explained - the CLNAP, CLNIP and ME. CLNAP stands for Connectionless Network Access Protocol, this protocol is resided in the IWUs for encapsulating datagrams before transmission. The CLNAP frame format is similar to the DQDB Initial MAC PDU format[38]. When the CLNAP frame arrives the CLS, it is further encapsulated with another 4-byte header by the Connectionless Network Interface Protocol (CLNIP). ME stands for Mapping Entity which is responsible to perform the encapsulation/decapsulation processes.

[18]It is because the ATM layers in IP over ATM is not treated as a MAC layer as LANE do. Therefore, the IP packet should first be extracted in the router and then retransmitted in the ATM network after the LLC/SNAP headers are encapsulated.

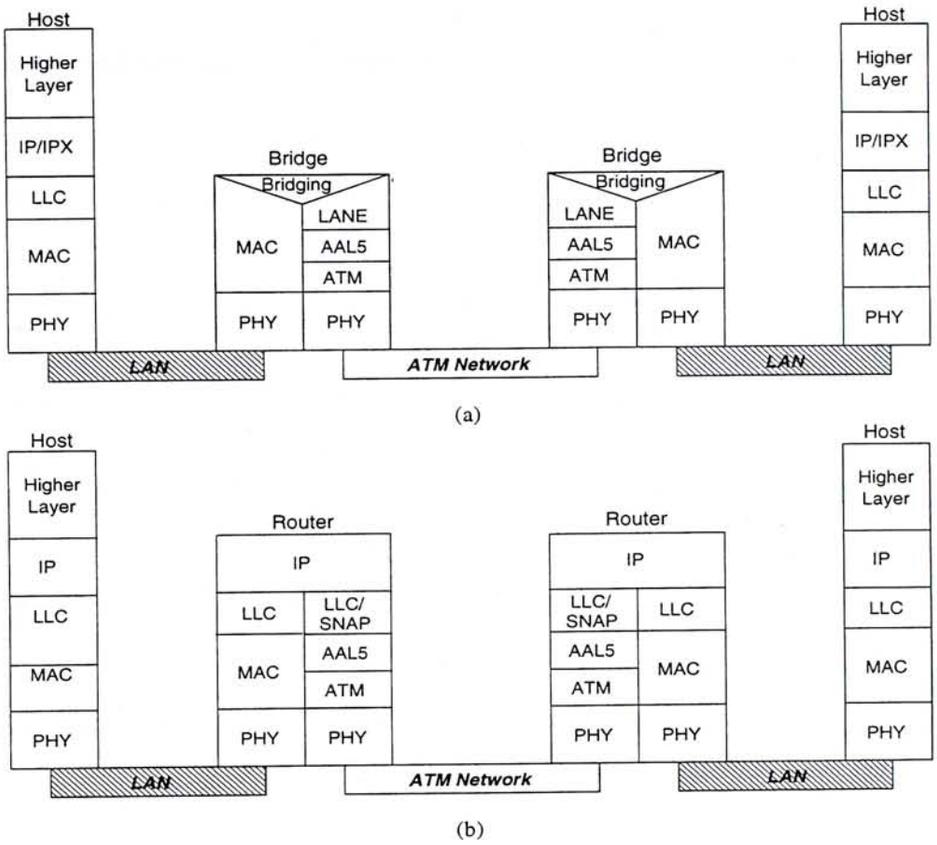


Figure 4.8: Protocol Stacks of (a) Bridge in a LANE network and (b) Router in an IP over ATM network.

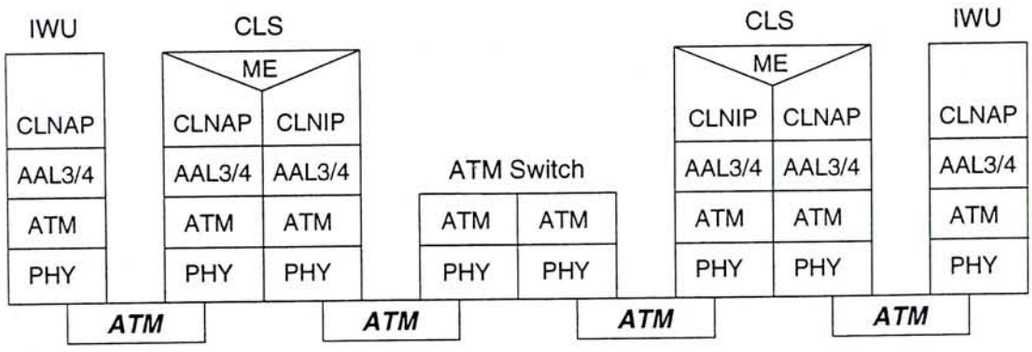


Figure 4.9: Protocol stacks of the components in a network employs the direct approach.

4.5.1 Selection of Protocol Converter

The selection of protocol converter is depended on the implementation environment. Bridge can only be used to interconnect the LANs with similar layer 2 protocols[5]. Therefore, bridge is generally used in connecting LAN segments, e.g., LANs in different laboratories connect to the departmental network through the bridges. While Router can be used to interconnect similar or dissimilar networks which are using the same network layer protocol. Therefore, routers are widely used on the Internet to interconnect networks which are using similar or dissimilar network technologies.

In the view of routing, Router is a more efficient device. However, due to the computational intensive (header conversion, route table searching, etc.) nature of Router, the implementation complexity is high and the cost is also higher than the Bridge. Therefore, bridges should be used in the intra-networking environment in which the employed network technologies are similar. Routers are used in the inter-networking environment in which different types of network technologies are employed.

In our implementation, we should properly choose Router as the protocol converter since the designed Gateway will be used in inter-networking networks of different technologies.

4.6 Packet Forwarding Modes

There are two general types of packet forwarding modes: *Re-assembly Mode* and *Cut-through Mode*.

The Gateway using the "Re-assembly Mode" will first re-assemble the segmented PDUs to the original data unconditionally and re-segmented the data back to some suitable sized PDUs for transmission. While "Cut-through Mode" operates in this way, the Gateway will directly transmit the incoming PDUs to the adjacent network without waiting the arrivals of all the PDUs for re-assembly. However, the Gateway still requires to do segmentation on the incoming PDUs if the size of the PDUs are larger than the MTU size of the destination network. These two modes in the CLS of the Direct Approach are respectively named as "Frame-based Forwarding" and "Cell-based Forwarding"[38]. Since in a CLS, they are manipulating cells rather than packets.

In comparing the two forwarding modes, the Re-assembly Mode will impose the following problems:

- More memories is required to buffer up all the PDUs of a SDU. Moreover, the Gateway may suffer from the lacking of memory when the data is large in size.
- The transmission latency is increased.
- The processes of segmentation and reassembly consume much CPU time and thus increase the cost.
- It imposes a larger queueing delay when the sources are contenting for the same output port.

Although the "Re-assembly mode" seems not to be a good solution, it stills has its advantages. It can lower the bandwidth wastage by discarding the error data in advance. In considering an OC3-SONET Interface(155.52-Mbit/s), if the

packets transmitted are all 100 bytes in size, then the Gateway should process the packets within 5.1 μs in order to keep in pace with the flow of traffic[19]. Thus, the processing speed of the Gateway can be lightened by using the Re-assembly mode.

Nevertheless, the processing delay and buffering problems of the "Re-assembly Mode" out-weight its advantages. Thus, the "Cut-through Mode" should be the more appropriate solution in actual implementation.

4.7 Bandwidth Assignment

ATM network is designed to provide guaranteed services. In order to provide guaranteed services, *Bandwidth Allocation/Assignment* for the connections is an important issue. The connectionless networks are designed to carry data traffic which is bursty in nature and the bandwidth requirement is changing continuously. Thus, when an ATM network is connected with the connectionless LANs, a control scheme is required in order to protect the ATM network from congestion. It can thus protect the ATM users whose have requested certain type of services guarantee suffering from resource shortage.

Preceding to the proposal of the Available Bit Rate(ABR) Service, there are many schemes proposed to achieve the means of Bandwidth Allocation for connectionless traffic in the ATM network[3, 37, 23, 35, 42, 36]. These proposals all assumed that the bottle-neck is in the ATM network. Conversely, [43] considered the problem in another way, the authors assumed that the destined

[19]This example is referenced from [38] where it addressed about the processing speed of the CLS. The paper mentioned that for a full speed OC3-SONET interface transmits in the above rate, the CLS should processes each cells within 2.7 μs

Gateway is the bottle-neck.

In this section, only the first issue (i.e., ATM truck is the bottle-neck) will be considered. There are five proposals and each of them will be reviewed:

- Bandwidth Reservation[42];
- Fast Bandwidth Reservation[3];
- Bandwidth Advertising[37, 23, 35];
- Bandwidth Advertising with Cell Drop Detection[36];
- Bandwidth Allocation on Source Demand[3].

4.7.1 Bandwidth Reservation

In this proposal, the ATM network is designed to interconnect the FDDI networks through the Gateways. It focuses on a bandwidth allocation algorithm that allows the Gateways to negotiate bandwidth per ATM link on demand (i.e., Bandwidth Re-negotiation) so as to prevent the ATM network and the destination Gateways from congestion. The Gateways are interconnected with a full mesh of virtual connections[20] and worked as brouters[21].

Each VC (pipe) of the mesh is allocated a few Mbit/s of bandwidth and the Gateways will keep track of the incoming traffic from the LANs by monitoring the Gateway transmission buffer length. When the incoming LAN traffic exceeds the allocated bandwidth, more bandwidth will be acquired from the ATM network by using the re-negotiation messages to signal the switches along the

[20]They called these VCs as pipes.

[21]Operates the interworking at layer 2 of OSI PRM for data transfer and performs layer 3 functions for the signaling.

path and the destination Gateway. The acquired bandwidth will be released by the similar procedure when the incoming LAN traffic decreases.

4.7.2 Fast Bandwidth Reservation

Initially, virtual circuits with zero bandwidth are setup for the interconnected Gateways. When data arrives at the Gateway, it will send a resource request packet with estimated bandwidth on to the desired virtual connection. The ATM switches along this connection will further verify the bandwidth acquisition request. If all the components along this connection have enough resources to support this request, a positive acknowledgment will be sent back to the source Gateway. The Gateway will then transmit the buffered traffic burst. Otherwise, a negative acknowledgment will be sent back to the source Gateway. The Gateway will then drop the buffered traffic burst.

4.7.3 Bandwidth Advertising

In order to use the resources of the ATM network efficiently and lower the transmission delay, "Bandwidth Advertising" is proposed with the ability to use the residual bandwidth of the ATM network in transmitting the excessive data traffic without buffering.

It presumes the existence of a "Bandwidth Advertising (BA)" mechanism which provides up to date information for the Gateways about the available bandwidth of the VPs they created in the ATM network. The authors have proposed two mechanisms to achieve the BA function: Bandwidth Probing and Bandwidth Broadcasting Schemes. Based on the information provided by these

BA mechanisms, the Gateways could decide how much volume of the excessive traffic burst could be transmitted.

If the traffic burst overflows the Gateway for a certain period, the Gateway would still perform "Bandwidth Re-negotiation" to acquire more bandwidth from the network.

4.7.4 Bandwidth Advertising with Cell Drop Detection

This is a modified version of the Bandwidth Advertising Approach described previously with the addition of a cell drop detection function.

The scheme enhanced the BA approach by properly discarding the data burst which portions of cells has been discarded by the switches. The philosophy of this scheme is that the importance of Burst Loss Rate (BLR) is more significant than the Cell Loss Rate (CLR) that the original BA scheme concerned.

The operation of this approach is based on the BA approach. However, when cells are dropped at the ATM switches, cell drop signals will be reported to the Gateway and the Gateway will discard the cells of the same violating burst (i.e., Traffic burst which cells has been discarded). This scheme can significantly lower the garbage traffic from the violating bursts.

4.7.5 Bandwidth Allocation on Source Demand

This scheme has adopted the concept of connection establishment between the hosts of the connectionless LANs. It also employed the BA mechanism described so that the Gateways could have knowledge on the available bandwidth in the connections of the ATM network.

When a host in the connectionless LAN wants to communicate with a remote host in another LAN through the ATM network, it should first enquire its attached Gateway to make a connection for it. The Gateway will use the information from the BA mechanism to decide whether this connection can be accepted. If enough resources are available, a positive acknowledgment will be replied to the source and the source can start the transmission immediately. Otherwise, a negative acknowledgment will be replied and the source should try again later.

4.7.6 The Common Problems

There are a few major drawbacks in the *Bandwidth Reservation* approach. Firstly, if the bandwidth re-negotiation process takes a long time, the delay in the data transmission will be increased substantially. Second, the amount buffer allocation in the Gateway is another problem. The amount of buffer memory required is a heuristic issue as it depends on the delay of the re-negotiation processes. Thirdly, if too many bandwidth re-negotiation processes are run in parallel, workload on the ATM switches will be increased. Also, the resources cannot be used efficiently because there will be a fixed amount of bandwidth allocated per connection even there is no traffic flow. Lastly, the burst/cell lost rate cannot be guaranteed. It is because when the Gateway cannot acquire enough resources in case of bandwidth suffocation, the burst in the buffer will be lost.

The main improvement of the *Fast Bandwidth Reservation* approach over the Bandwidth Reservation approach is that it does not waste bandwidth as it allocates zero bandwidth in the idle connections. It still imposes transmission delay because of cells buffering before the resources are acquired. Consequently, the

same buffer allocation problem of the Bandwidth Reservation approach occurs in this scheme.

The *Bandwidth Advertising* proposal has an improvement in the buffering delay problem because the excessive cells are not required to wait in the buffer. The BA traffic may cause congestion to the ATM network if their broadcasting period is high. If the coverage of the ATM network is large, the propagation delay of the BA cells may be large and the information may become outdated. Similar to the Bandwidth Reservation approach, the idle connections in this approach will waste bandwidth.

The *Bandwidth Advertising with Cell Drop Detection* approach is only an improved version of the original BA approach, thus it still has the similar problems as the original approach. There is one more disadvantage with this approach. If the cell drop signals cannot reach the Gateway on time, the whole traffic burst may have been transmitted. So, this signal just imposes further workload to the Gateway rather than improvement.

The *Bandwidth Allocation on Source Demand* is the most ideal solution. However, it requires modification of the existing LAN protocols to support the connection setup function with the Gateway. Moreover, the proposal requires a VC to be created for each end-to-end connection. This may not be achieved in the current implementation[22]. Further, it may not be possible for a user to predict the bandwidth he actually required.

Nevertheless, they are just some proposals and the feasibility of implementation have not been carried. For example, the BA approach requires the switches to broadcast bandwidth information. It requires the switches to be modified

[22]It is because LLC/SNAP encapsulation is used in carrying multi-protocols over an ATM connection.

and may not be possible in certain extent. The standard of Bandwidth Renegotiation has still not been yet defined in the UNI 3.1[15], so the Bandwidth Reservation Proposal can not be implemented. However, they should be implemented in the near future when the standards of ATM are settled down.

Chapter 5

Design and Implementation of the Connectionless Gateway

This chapter combines the solutions of the discussed issues from the previous chapter (Chapter 4) to propose a feasible solution for the implementation of the "Connectionless Gateway".

5.1 Introduction

This project was initiated by Prof. Kwok-Wai Cheung in late 1995. The goal of this project is to design and implement a *Connectionless Gateway* for internetworking the CUM LAUDE NET (See Chapter 2) and the ATM networks.

We foresee that in the near future, ATM will become the major technology in building network backbones. Therefore, in order to increase the connectivity of our CUM LAUDE NET with the future ATM backbones (Internet), the "Connectionless Gateway" is an indispensable element. Figure 5.1 has depicted the

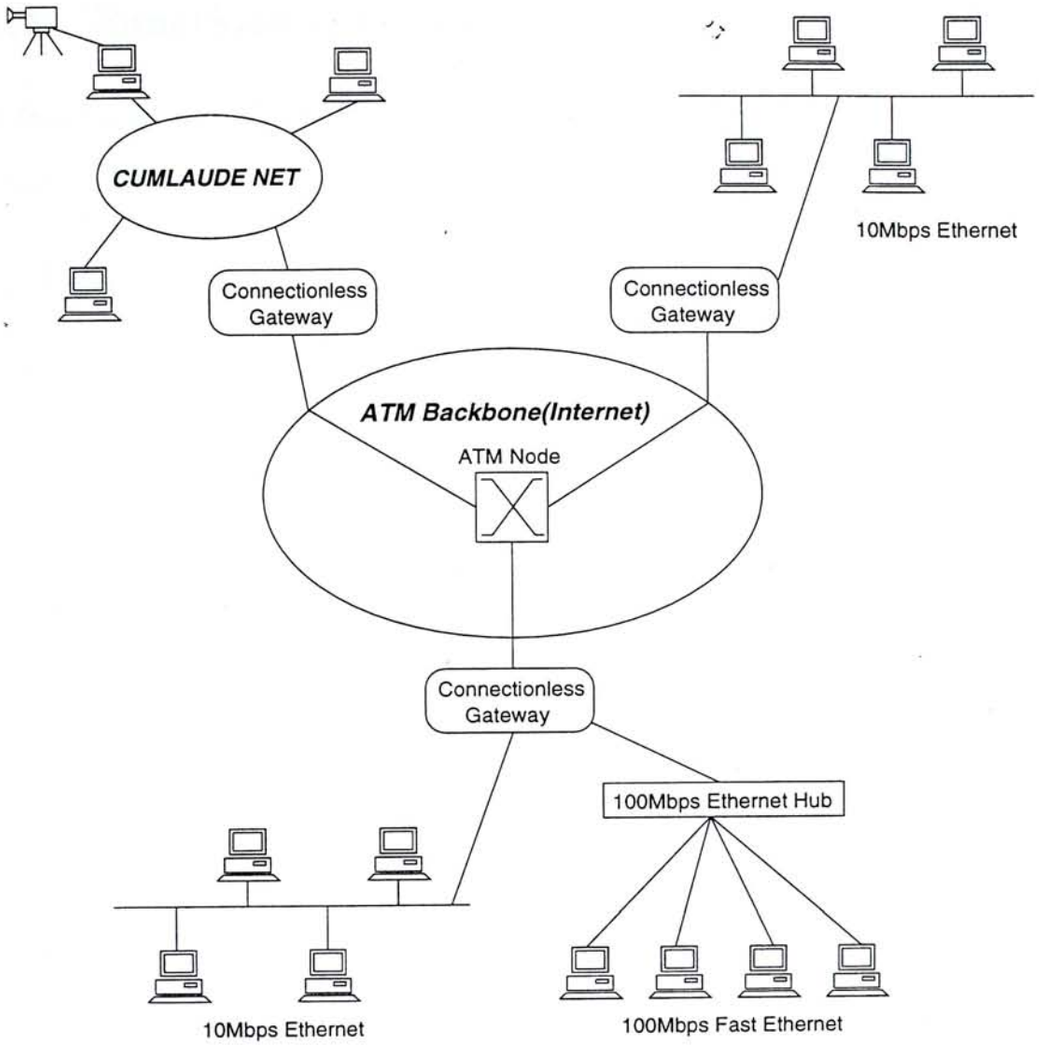


Figure 5.1: Scenario of internetting the CUM LAUDE NET to an ATM backbone.

scenario of internetworking the CUM LAUDE NET with the Ethernet networks through an ATM backbone using the Connectionless Gateways.

Besides, the designed Gateway should be able to interconnect other types of conventional LANs with the ATM networks such as Ethernet, FDDI, Token Ring, etc. The last, but not the least, the design should be a low-cost and feasible solution.

5.1.1 Functions Definition of Connectionless Gateway

The functions that the Connectionless Gateway should be implemented are as the following:

1. The Gateway should be able to store, filter and forward the incoming packets. If any possible route/port can be determined for the incoming packets, they will be forwarded to the corresponding output port (network). Otherwise, the packets will be discarded.

2. The Gateway is responsible for performing any conversion on the Protocol Data Units (PDUs).

3. The Gateway should be able to segment the IP packets to the right sizes smaller than or equal to the MTU sizes of the destination networks[1].

4. The Gateway should be able to create connections to the destination Gateways which are determined in the routing procedures.

5. Lastly, the Gateway should support the ATM specific functions such as ATM signalling, SAR of PDUs, etc.

5.2 Hardware Architecture of the Connectionless Gateway

In the early time of development, ATM Network Interface Cards (NICs) were mostly designed for the workstations using the SBus architecture[2]. It thus slowed

[1]The Gateway is designed to operate in "Cut-through" mode (See Section 4.6 of Chapter 4) and thus will not perform re-assembly operations to the segmented packets.

[2]Sbus is developed by Sun Microsystems, Inc.

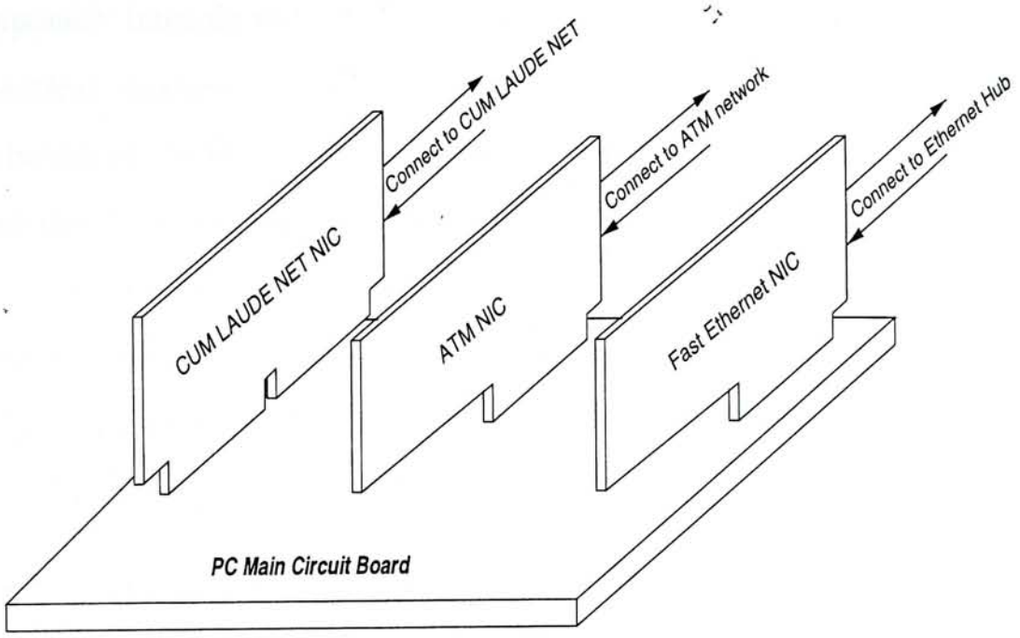


Figure 5.2: Hardware Configuration/Architecture of the Connectionless Gateway.

down the deployment of ATM to the desktop PCs since the PCs do not support SBus. Until early 1995, vendors started promoting ATM NICs that work with PCI bus on the PCs[44]. Then, many projects were shifted to use the PCs in the ATM applications development. It is because PCs are much cheaper and their computational power are also good compared with the expensive workstations[3]. Therefore, we also choose to use the PCI-based PCs as our implementation platforms.

The hardware architecture of the Connection Gateway is illustrated in Figure 5.2. The Gateway is constructed by plugging different network interface cards (a CUM LAUDE NET NIC, an ATM NIC and a Fast Ethernet NIC) into a PC platform.

The latest and fastest bus technology used in the PC system is the Peripheral

[3]However, their floating point computation power is just satisfactory in comparing with the workstations[45].

Component Interconnect (PCI) bus[4] which the bus width is 32/64 bits and its theoretical maximum transfers rate is 133 Mb/s[45]. Therefore, the IO bus bandwidth of the PCI bus is much better than the plain old ISA or EISA buses which the maximum transfer rate are just 8 Mb/s and 30 Mb/s respectively[45]. Thus, almost all the IO bandwidth demanding cards today are designed based on the PCI bus (e.g., the 155-Mb/s ATM NIC used in the Gateway[5]).

The components of the Connectionless Gateway used in the project are summarized as following:

- PCs equipped with Pentium (100,120 and 166 MHz) or Pentium Pro (200 MHz) Processors and 64 MByte of memory;
- 155-Mb/s ATM NICs[6] with 2MB (Server) and 512KB (Client) memories using multi-mode fiber as the transmission medium with SONET interface[46];
- CUM LAUDE NET NIC;
- 100-Mb/s Fast Ethernet Adapters [7] using category 5 unshielded twist pair (UTP) as the transmission medium;
- Other 10-Mb/s ISA bus Ethernet Adapters.

The above list only includes those essential components that is significant to the gateway design and other indispensable components such as the display card, hard-disk, etc. are assumed to be present.

[4]PCI is a bus standard which is not designed solely for PC, for example, the Digital Dec Alpha workstations also use the PCI bus system.

[5]It requires bus bandwidth of 16.8 Mb/s[45].

[6]Manufacture by the Efficient Networks, Inc. of model ENI-155e-MF.

[7]Manufactured by 3Com of model 3c595-Tx.

A general purpose PC consists of 4 PCI bus slots and 2 ISA bus slots. One of the PCI bus slots is occupied by the display adapter and three spare PCI bus slots are left for the NICs. The distribution of these slots are: one for the ATM NIC and two for the Fast Ethernet NICs. The two ISA slots are left for the CUM LADUE NET NICs and 10-Mb/s Ethernet adapters. Thus, the Connection Gateway can connect up to five networks.

5.2.1 Imposed Limitations

Since the functions of the ATM layers are computation intensive, in order to achieve high speed operation without imposing much burden on the host processor[8], these functions are built into the Application Specific Integrated Circuits (ASICs) of the ATM NICs. However, not all of the ATM layers and functions defined by the ATM specifications are implemented in these ASICs.

The ATM NIC used in our implementation only support the CPCS and SAR functions of AAL-5, which is sufficient for data communication. Moreover, the only VPI value supported by the NIC is zero. Thus, the Gateways can only be used in small areas because the NIC has a limit of 1024 virtual channels but may not be sufficient for larger areas.

[8]Because in early implementation of the ATM NICs, the SAR functions are performed in software on the host.

5.3 Software Architecture of the Connectionless Gateway

We have selected an OS called "Linux" to run on the development platforms. Linux is a free UNIX clone developed by volunteers in the Internet Community[9]. The source codes and applications of Linux can be freely downloaded from many ftp sites, thus it is a good OS for development. It also provides many powerful features such as TCP/IP networking, BSD socket API, etc.

This section mainly illustrates the architecture of the operating system "Linux" used in the implementation.

Linux is initially designed to run over the Intel x86 machines and is now ported to other machines with different architectures such as Dec Alpha and Sun Sparc. Linux is conformed with a number of UNIX standards including IEEE POSIX.1, System V and BSD. The main features of Linux can be summarized as follows[16]:

- Supports multitasks and multi-users;
- Supports the use of Dynamically Linked Library (DLL) for applications;
- Fully supports TCP/IP internetworking and many network device drivers were written;
- Have a specially designed file system called *ext2fs* and supports other file systems such as DOS FAT, Win95 VFAT, Minix FS, etc. It also supports the use of Network File System (NFS).

[9]The initiator is Linus Torvarlds.

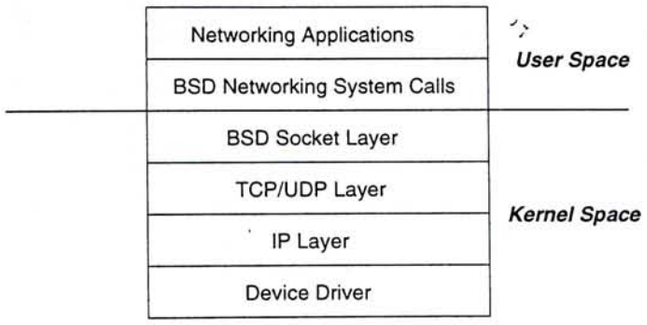


Figure 5.3: Linux Networking Protocol Stacks.

5.3.1 TCP/IP Internals

The current implementation of the TCP/IP protocols in Linux is named as *NET-3*. All the TCP/IP codes are written in the kernel of Linux and provide services to users through the well-known BSD Socket Application Programming Interface (API). Figure 5.3 has depicted the relationship between the components of the user and kernel spaces in Linux Networking.

The BSD API has provided a number of system calls for the user applications to request services from the kernel. Since the kernel of Linux is non-preemptive[10], so direct access of the kernel services from user space is dangerous and it should be done through the system calls. The BSD API has provided the system calls such as `int socket(int family, int type, int protocol)`, `int bind(int sockfd, struct sockaddr *myaddr, int addrlen)` and `int listen(int sockfd, int backlog)` for creating connections between applications/machines. Chapter 6 of [47] has provided a detailed description on BSD Sockets. The Socket concept makes network programming easier as it is treated as a UNIX file I/O[11],

[10]It is the magic of preemptive mode operation in user space that makes the user applications run in parallel. It also means that if any operation holds up the kernel for a long time, the users applications may be suspended.

[11]Thus, the value returned by the `socket()` system call is normally called socket file descriptor.

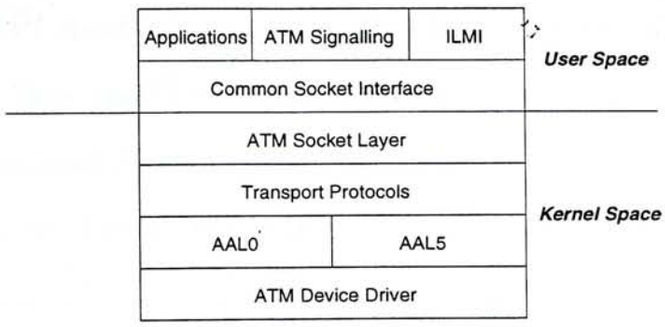


Figure 5.4: Protocol Stacks of ATM on Linux.

so the system calls designed for file access such as `read()`, `write()` can also be used in network communication.

Besides, the BSD Socket API supports other protocol suites such as the UNIX domain socket and Xerox NS domain socket. Therefore, in referencing the system calls, parameters are required to distinguish the type of sockets being created.

5.3.2 ATM on Linux

ATM is now being supported on Linux and the arrangement of the components of "ATM on Linux"[12] is illustrated in Figure 5.4.

The figure has depicted the main components. In order to access the ATM NIC, the ATM device driver is indispensable. The AAL layer shown is not going to provide the AAL functions but acts as an interface between the transport layer and the ATM driver. The AAL functions are provided by an ASIC on the ATM NIC. The BSD Socket API is extended to support the creation of ATM socket. However, this ATM Socket API does not conform with ATM Forum's Native

[12] "ATM on Linux" is developed by an interest group in Laboratoire de Réseaux de Communication(LRC) which is headed by Werner Almesberger[44]. This project is started in the beginning of 1995.

ATM Services API specification. Since the Signalling and ILMI protocols are rather complex, thus they are implemented as user processes (daemons) and they just keep minimal data structures in the kernel (not shown in the figure).

The ATM Device Driver has maintained a data structure `atm_dev` for each ATM device which is used to describe the device and physical operations supported, parameters related to the VCs opened by the device and the ATM device registers such as the ESI value (End System Identifier which is the Hardware address of the NIC) and allowable VPI, VCI ranges. It should also cooperate with the ATM NIC, e.g., in opening a VC. Therefore, the ATM device operations defined provide abstract calls to access the device such as `int (*open)(struct atm_vcc *vcc, int vpi, int vci)` and `void (*close)(struct atm_vcc *vcc)` for VCC creation and termination. Details of the operations can be found in [48].

In Linux, the notion "socket" is basically used to represent the interface between the networking modules. Socket buffer (`sk_buff`) is used to transfer the network data between these sockets. The definition of `sk_buff` is extended to include the entries for the ATM specific sockets. The extended part is shown as below:

```
struct {
        int                 size;
        unsigned long       pos;
        struct atm_vcc      *vcc;
        int                 iovcnt;
        struct timeval      timestamp;
} atm;
```

In order to create native ATM connections, two protocol family identifiers are added to the `socket()` system call: `PF_ATMPVC` and `PF_ATMSVC` for PVC and SVC connections respectively. The structure of the protocol addresses for these connections are also defined. Details about the ATM API is not included here and interested readers should refer to [49].

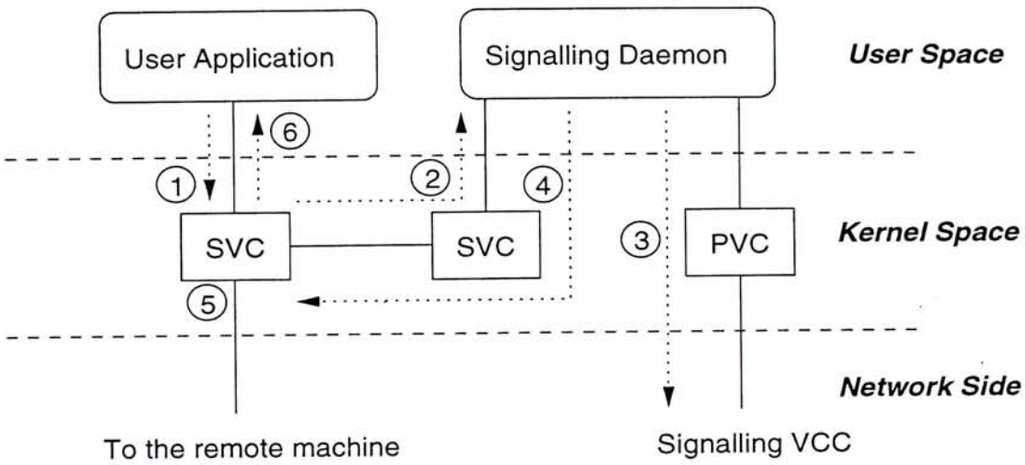


Figure 5.5: An example showing the SVC setup procedures.

The Signalling Daemon mainly provides the connection setup service. It consists of a Q.93B (Q.2931) message handler and uses SSCOP as the transport protocol over AAL-5. Figure 5.5 illustrates the SVC setup procedures for ATM on Linux machine. These procedures can be stepwised as follows:

1. The application sends a connection setup request to the remote party.
2. The kernel part signalling daemon forwards this request to the signalling daemon.
3. The user part signalling daemon will contact the ATM network to create the connection.

4. When the connection is accepted and established, the signalling daemon will notify the kernel.

5. The local part of the data connection will be setup.

6. Lastly, the application will be acknowledged for the establishment of connection.

The current ILMI daemon provides the function of address auto-configuration.

The current ATM software and hardware can only support limited type of services (or Quality of Service (QoS)): Constant Bit Rate (CBR) and Unspecified Bit Rate (UBR) services. Before the launch of the Available Bit Rate (ABR) service, the feasible type of service that can be used for bursty data traffic should be the UBR service.

5.4 Network Architecture

The Connectionless Gateway is assumed to be deployed in the network scenario illustrated in Figure 5.6. This scenario will also be used as reference for the subsequent sections.

Three networks are connected to the ATM backbone using the "Direct Approach" with "IP over ATM". A dedicated ATMARP Server is assigned for the Logical IP Subset (LIS) of the Connectionless Gateways as shown. It also functions as a *Routing Server* which will be described later. The connections between the Gateways are SVCs[13].

[13]If PVCs are used, the ATMARP Server is not needed.

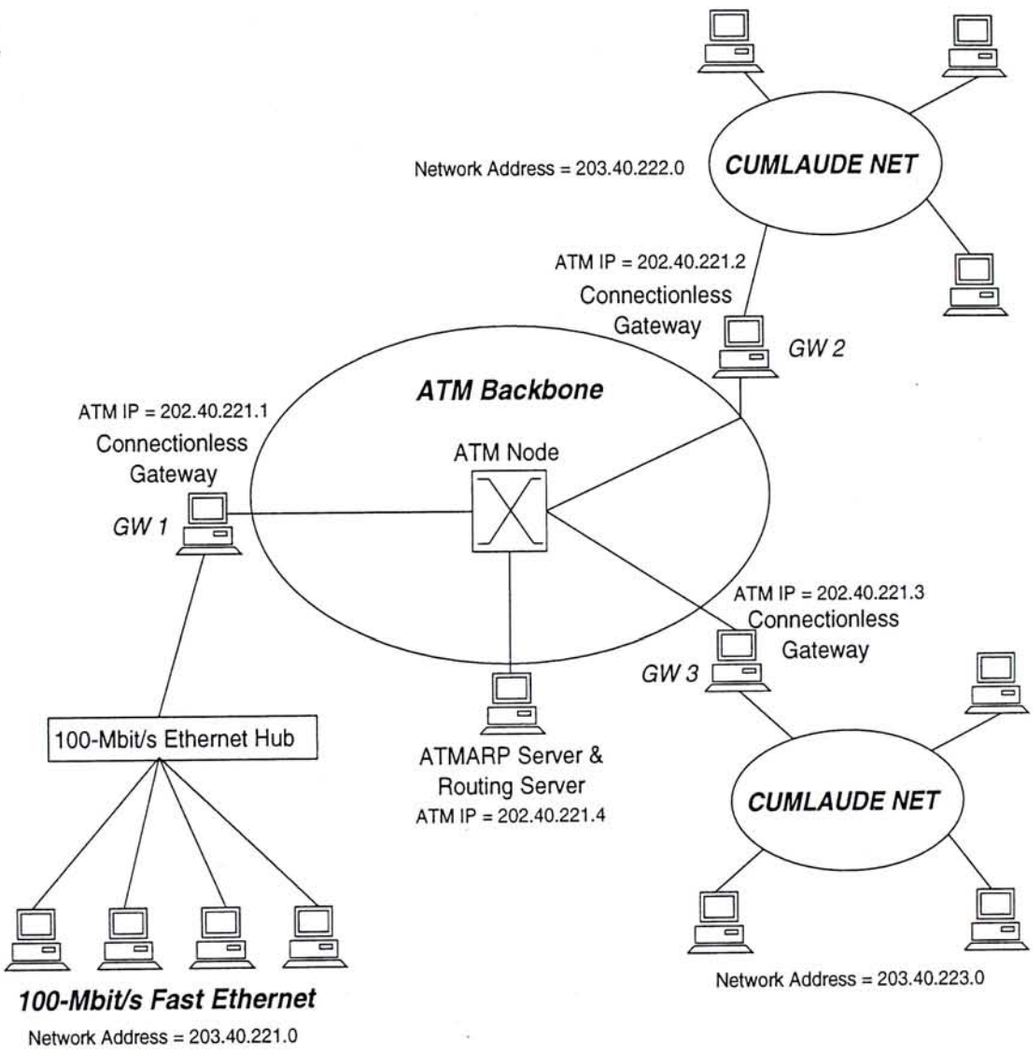


Figure 5.6: Network Architecture for the designed Connectionless Gateway.

The Connectionless Gateway is a multi-homed machine which has at least two IP addresses, namely ATM IP addresses and Ethernet/CUMLAUDE IP addresses[14]. These conventions will also be used in the following discussions.

5.4.1 IP Addresses Assignment

The connectionless networks and gateways illustrated are all assigned with IP addresses. In order to eliminate the requirement of extra IP routers (See Section 4.3.2(b) of Chapter 4) and ATMARP Servers (Because each LIS should have an ATMARP Server.), thus the ATM IP addresses of the Connectionless Gateways are purposely allocated to the values shown. They all belong to the same Logical IP Subnet (LIS) with subnet address of 202.40.221.0.

The Ethernet/CUMLAUDE IP addresses of the Gateways are not specifically assigned since they are insignificant to the discussion. The only selection criterion is that the address should belong to the corresponding IP domain.

5.5 Internal Structure of Connectionless Gateway

5.5.1 Protocol Stacks of the Gateway

Figure 5.7 illustrates the protocol stacks of the designed Gateway. The functions of the Gateway is provided by the *Gateway Engine*. The Gateway Engine is a part of the IP layer. The first three functions defined in Section 5.1.1 have been implemented in the Gateway Engine and the remainding functions are provided

[14]Depends on which type of network the gateway is connected to.

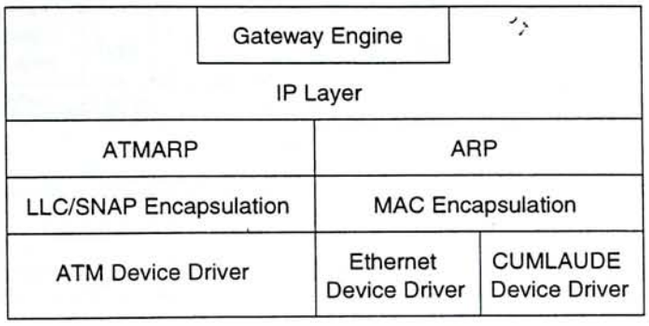


Figure 5.7: Protocol Stacks of the Connectionless Gateway.

by the ATM Signalling and ATMARP daemons.

In Figure 5.7, the ATMARP and ARP layers are shown to be situated below the IP layer. Actually, they are situated in parallel to the IP layer because the ATMARP/ARP messages are sent directly without relay on the IP layer. They are arranged as shown because we want to illustrate the procedures of data flow, i.e., before the transmission of IP packets, ATMARP/ARP procedures should first be proceeded.

The MAC Encapsulation layer is used to encapsulate/decapsulate the packets from/to the higher layer with the MAC addresses of the source and destination for the connectionless LAN[15]. While the LLC/SNAP Encapsulation layer is used to encapsulate/decapsulate the packets from/to the higher layers with the LLC/SNAP header for the ATM network.

Figure 5.8 depicts the types of encapsulation employed in the implementation. In both Figure 5.8(a) and (b), the Data field can either be an IP packet

[15]In Ethernet/most of the shared-medium LAN technologies networks, before the transmission of an IP packet, it must first be encapsulated with the source and destination MAC addresses. It is because the devices of these networks will receive all the frames from the shared-medium and they have to use the MAC addresses to sort out the frames that are really sent to them.

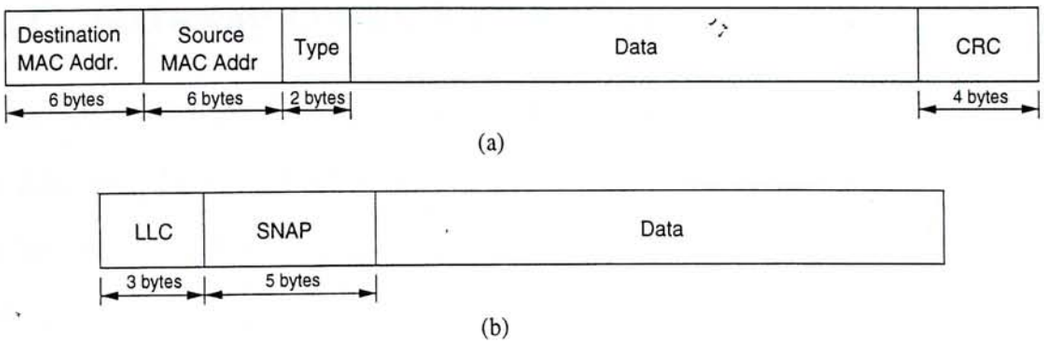


Figure 5.8: Packet encapsulation of (a) Ethernet and (b) IP over ATM (LLC/SNAP)

or ARP/ATMARP requests. Figure 5.8(a) has illustrated where the MAC addresses resolved from the ARP procedure should be placed. The LLC/SNAP encapsulated IP over ATM packet can be considered as a type of IEEE 802.3 encapsulated packet with the MAC addresses and Type fields removed.

The device drivers are written to provide high level function calls for the higher layers to access (e.g., transmit or receive data) the network device. It takes care all the hardware specific tasks for the higher layers.

There are three important components that have not been shown in the figure: the *Routing, ARP* and *ATMARP Tables.* The ARP and ATMARP Tables are used to store the mapping of IP with MAC addresses and mapping of IP with ATM addresses respectively. The Routing Table is used to record the destinations that the Gateway can reach. The Gateway can determine which outing device/port the incoming packets should be routed by examining the entries in this table. If no route can be found for the packets, they will be dropped. Details about these tables can be found in the next subsection.

5.5.2 Gateway Operation by Example

Figure 5.9 shows an example showing the communication scenario between a Fast Ethernet Host and a CUMLAUDE NET Host through an ATM network[16]. The internal structure (up to the network layer) of the Connectionless Gateway (GW 1) is expanded with the higher layers hidden[17].

Since the space of Figure 5.9 is limited, the detailed operation flow diagram of the Connectionless Gateway is shown in a separate figure (Figure 5.10).

Assume Host 1 (a Fast Ethernet Host) wants communicate(e.g. ftp) with Host 2 (a CUMLAUDE Host). Host 1 is situated in a network different from that of Host 2. Thus, the frames send by Host 1 can only reach Host 2 through the GWs and the ATM network.

Figure 5.9 has shown the header of the frames generated by Host 1. The Ethernet Interface of GW 1 will accept these frames (sent by Host 1) upon the MAC addresses in the header of the frames. The process for Host 1 to obtain GW 1's Ethernet MAC address is achieved by the Address Resolution Protocol(ARP)[18]. After the frames have entered GW 1, GW 1 will take up the responsibility to forward these frames to their destination(s).

Figure 5.9 and 5.10 illustrate the packet processing steps. After a complete frame has been received by the Ethernet device driver, the device interrupts the host. The kernel device driver then picks up this frame and put it into the input backlog queue[19]. The IP Input function is called after the interrupt. The IP Input function determines whether the frame is valid and only valid frame will

[16]Represented by an ATM node.

[17]It is because they do not have any significance in the explanation.

[18]Details of ARP can be found in [11].

[19]All the network device will share one backlog queue and its current size is set to store up 300 packets.

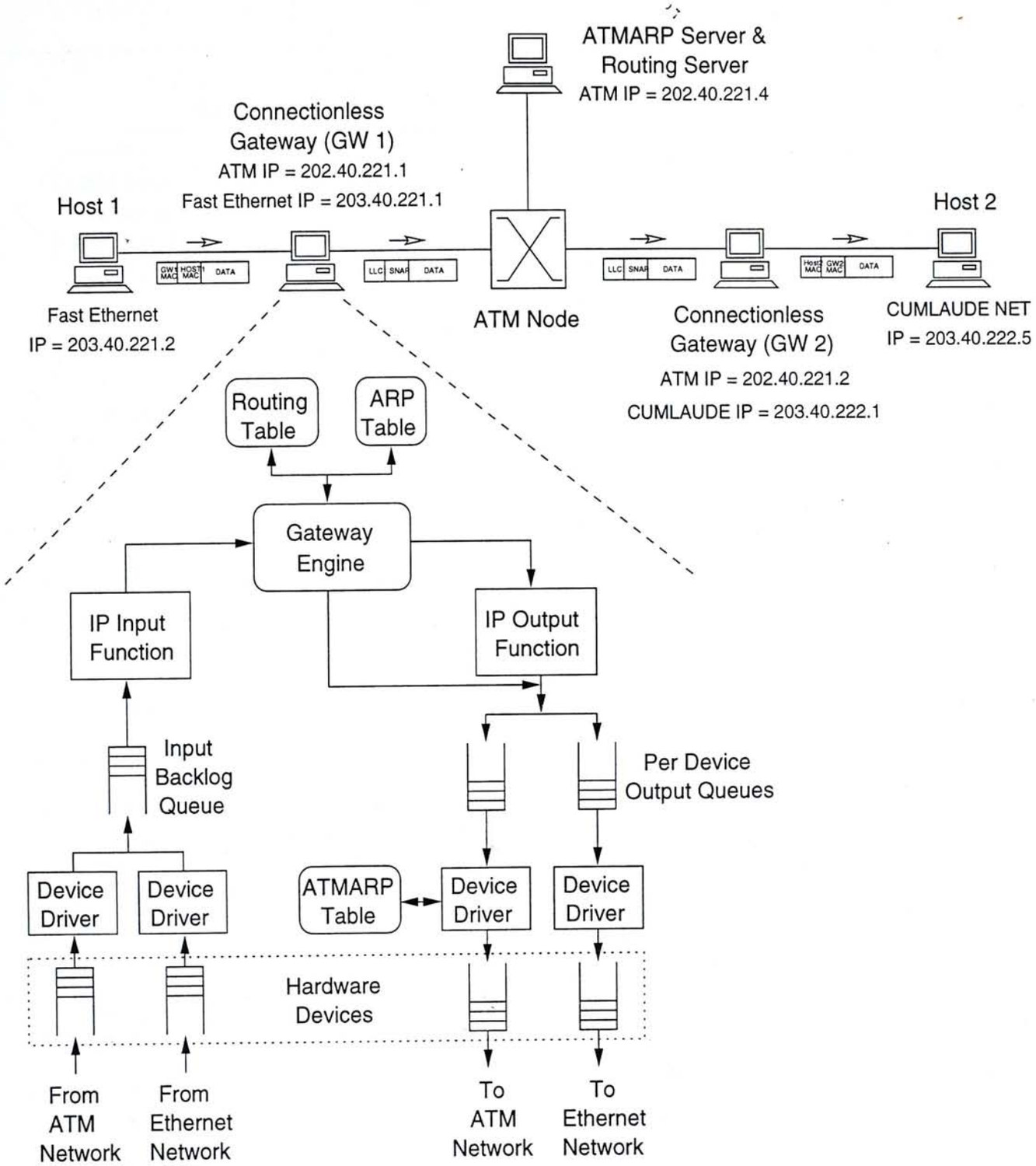


Figure 5.9: Example shows the communication scenario and internal structure of the Connectionless Gateway.

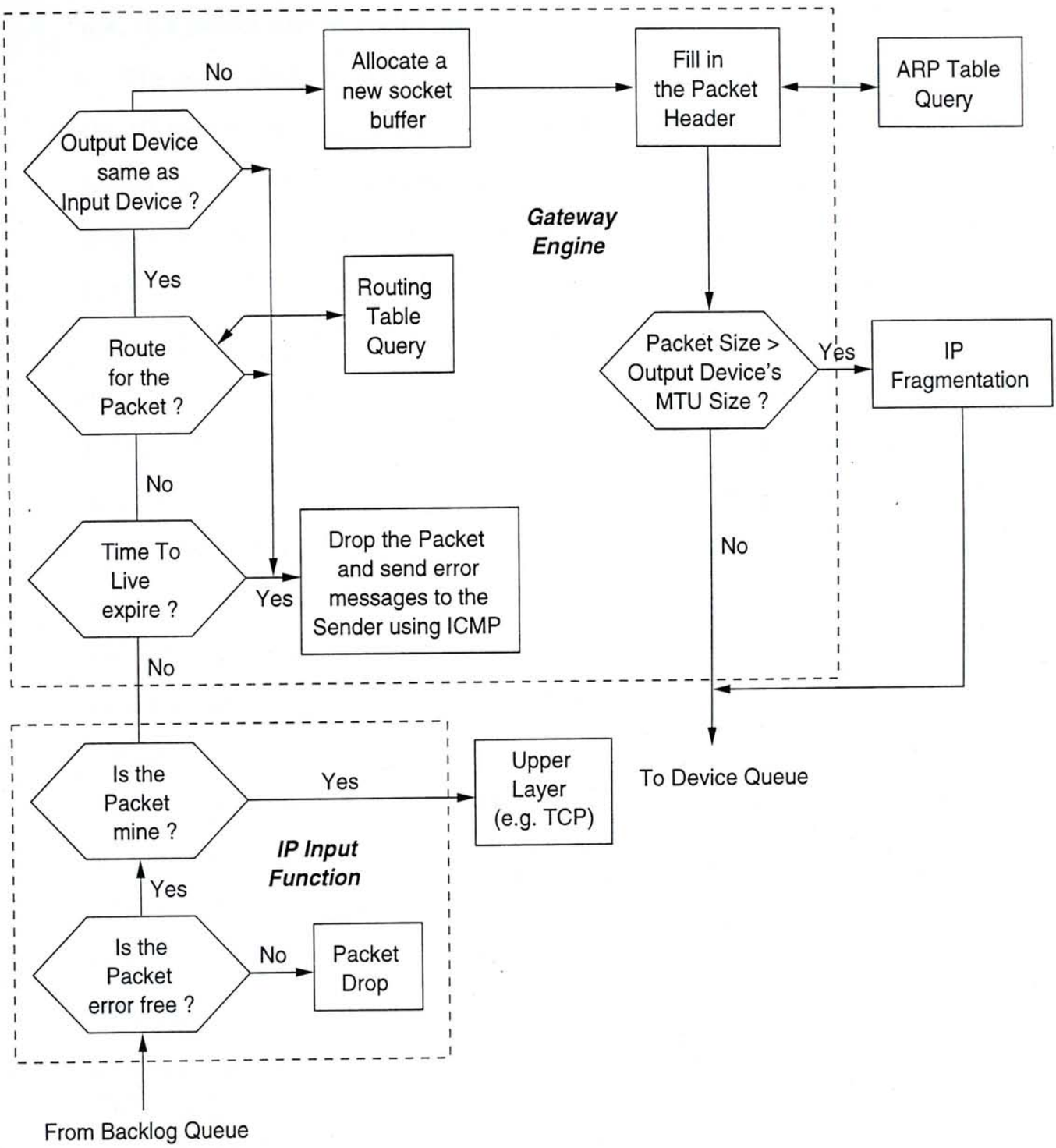


Figure 5.10: Operation flow diagram of the Connectionless Gateway

be further processed. It also determines whether this frame is destined for it, if not, this packet will be passed to the Gateway Engine.

The main function of the Gateway Engine is to construct new headers for the IP packets, but not modifying the contents. When the Engine receives a packet, it first examines whether its Time to Live (TTL) field has expired, if it does, the packet should be discarded. The Engine then issues an ICMP message to inform the sender that the packet has been dropped because of time expiry[20]. The purpose of the TTL field is to prevent the packet from being routed in a loop [11], but this case would rarely happen in the connection-oriented ATM network since the routes are all pre-assigned. The TTL field will decrease by 1 and the Engine will find a route for the destined IP host of the packet from its routing table. If the Gateway can find a route for the packet and the output device is not equal to the input device[21], it will allocate a new socket buffer (`sk_buff`) for that packet if the header length of the input packet is shorter than the header length of the output packet, otherwise, the original input buffer will be used. Then, a new MAC or LLC/SNAP header is appended to the packet depend on the type of output device. In appending a MAC header, the Gateway should determine the MAC address of the destination. That is the reason for the existence of an "ARP Table Query" entity shown in Figure 5.10. If the size of this "completed packet" is greater than the MTU size of the output network, the IP fragmentation function will be called. Finally, the Gateway Engine *directly* queues the packet to the output queue of the destination device[22].

[20] Actually, it is hop count expiry.

[21] It is the case for the conventional LAN since if the input device same as the output device, the packet should be directly route to the destination than to the Gateway. However, if the Gateway is configured as an Router for an ATM LAN, this may not be true since the notion of LIS allows this operation.

[22] Because it can enhance the operation speed of the Gateway.

The packet is still stored in the buffer of GW 1 in this stage. The next step is to establish a VC between GW 1 and GW 2[23] and transmit the packet. As illustrated in Figure 5.9, when the ATMARP Table is queried in the device driver level, the connection is also established by this driver. Actually, the ATM device driver shown is a virtual (IP) device driver which is operated and written in the similar way as the well-known Ethernet Device Driver written by Donald Becker. After the packet for the ATM device is enqueued, the function `int hard_start_xmit(struct sk_buff *skb, struct dev *dev)`[24] of the virtual device driver is invoked to send the packet out. This is the point where the required VC is established if it has not yet existed. An ATMARP Daemon in user space is designed to handle the signalling of IP/ATM addresses resolving and SVC establishing. Finally, the `send()` system call of the ATM device driver will be invoked to deliver the packet. More details about the ATM device driver can be found in [48]. The pseudo code of the Gateway Engine algorithm is shown in Figure 5.11.

5.5.3 Routing Table Maintenance

Figure 5.12 illustrates three example routing tables of Host 1, GW 1 and ATMARP Server. The values in the entries are assigned with respect to the scenarios of Figure 5.6 and 5.9. Figure 5.12 shows that each Routing Table contains three columns, but there are actually thirteen entities (columns) in the implementation. The first entity is *Destination*, it represents the destination networks

[23]GW 1 and GW 2 have registered their IP and ATM addresses with the ATMARP Server.
[24]This is a generic name and it is called `clip_xmit()` in the ATM IP device driver.

Gateway Engine Algorithm

```
/* Begin */
Gateway_Engine_Function(Socket Buffer, Input Device, Target Address)
{
 Time to Live = Time to Live – 1
 Recompute the IP Header Checksum
 if (Time to Live <= 0) then Drop the Packet and Release Buffer
                             ICMP_Send(Time to Live)
                             return error
 Route = Check_Route(Target Address)
 if (no Route) then ICMP_Send(Route)
                    return error
 /* Route has been found for the packet and the Output Device is determined */
 if (Ouput Device = Input Device) then ICMP_Send(Device)
                                       return error
 if (Output Device is operating) then
   if (Socket Buffer is too small) then create Socket Buffer 2
   else Socket Buffer 2 = Socket Buffer
   Release the memory of Socket Buffer
   Fill_Header(Output Device, Route, Socket Buffer 2)
   if (length of Socket Buffer 2 > MTU of Output Device) then
      IP_Fragmentation(Socket Buffer 2, Output Device)
      Release the memory of Socket Buffer 2
   else Put Socket Buffer 2 to the Output Device Queue
 else return error
}

ICMP_Send(Error)
{
 switch (Error)
   case 'Time to Live'
        Compose an ICMP message with 'Time to Live Exceed' field set
   case 'Route'
        Compose an ICMP message with 'Change the Route for Destination' field set
   case 'Device'
        Compose an ICMP message with 'Destination Unknown' field set
        /* i.e, Tell the sender: You should not send me this packet, you can contact the user
           directly !! */
   case 'MTU'
        Compose an ICMP message with 'Invalid Packet Length(Fragmentation Error)' field set
        /* This Packet is so small and should not be fragmented */
   case 'MAC'
   Compose an ICMP message with 'Destination cannot be reached' field set
 Send the ICMP message
}

Check_Route(Target Address)
{
 Point to the First Entry of the Routing Table
 Do
    if (Destination Address in the Routing Table Entry = Target Address) then
       return the entry
    Point to the next Entry
 while (Current Entry is not the last entry)
}
```

```
IP_Fragmentation(Socket Buffer, Output Device)
{
 Calculate the Link-Layer Header Length
 Data Size = IP Packet Length - IP Header Length
 Header Length = IP Header Length + Link-Layer Header Length
 Maximum Transfer Unit = Output Device Maximum Transfer Unit Length - Header Length
 if (Packet cannot be fragmented field set) then return
 if (Maximum Transfer Unit < 8) then ICMP_Send(MTU)
                                     return
 Set Fragement Offset Value
 Do
    Length = Data Size
    if (Length > Maximum Transfer Unit) then
       Length = Maxmimum Transfer Unit
    Allocate Socket Buffer of size Length
    Put Packet Data into Socket Buffer
    Fill the IP Header Value into Socket Buffer
    Calculate the IP Packet Offset
    Data Size = Data Size - Length
    Put Socket Buffer to the Output Device Queue
 while (Data Size > 0)
}

Fill_Header(Output Device, Route, Socket Buffer)
{
 Extract the Next Hop address from Route
 switch (Output Device)
   case 'ATM'
        Fill in the Header with the Corresponding LLC/SNAP Value
        /* The Type filed of the SNAP Header is different for IP and ATMARP Packets */
   case 'Ethernet or CUMLAUDE NET'
        Search the MAC address of the Next Hop from the ARP Table
        if (MAC address cannot be found) then
           Send ARP Packet to query the Next Hop for its MAC address
        if (No response from the Next Hop) then ICMP_Send(MAC)
                                                return error
        Fill in the Header with the MAC addresses
}
/* End */
```

Figure 5.11: Algorithms of the Gateway Engine

or hosts that the *Host* owning the table can reach. *Gateway* means the Gateway/Router that routes packets for Host to Destination. *Interface* shows the output device that the packet should be routed for the Destination of that entry. Another important field that has not been shown is the *Network Mask* entity which is used in hierarchical routing[11].

Here is a simple explanation of the routing operation in the scenario of Figure 5.9. With reference to Figure 5.12(a), Host 1 has gotten two entries in its routing table. The Destination network address of the first entry in that table is the network address that Host 1 is directly connected to and the Gateway is "x", it means that there is no Gateway available. This type of routing is called *Direct Delivery*[11] because the host can directly connect to other hosts that belong to its network[25] and should never cross any Gateway. The remaining entry shows a Destination of *Default* which means that any outgoing packet from Host 1 to Default are sent to GW 1 for routing. In fact, this "Default" route is a wild card address such that all non-local addresses that do not have any entries in the routing table will use this route. With this generic route, the size of the host routing table can thus be minimized. This type of routing is known as *Indirect Delivery*[11].

The Routing Table of GW 1 (Figure 5.12(b)) is a little bit more complicated. There are two entries in the table that contained no Gateway which correspond to the networks that GW 1 is directly connected to and the other two entries are pointed to the other Gateways. There is no default route in GW 1's routing table because all routes are well-known. However, for example, if there is one exit Gateway (say GW 4) that connects to the Internet for the ATM Network,

[25]However, there may be bridges connect them together because it makes no difference in IP routing.

Destination	Gateway	Interface
203.40.221.0	x	ETH
Default	GW 1	ETH

(a) For Host 1 (IP = 203.40.221.2)

Destination	Gateway	Interface
203.40.221.0	x	ETH
202.40.221.0	x	ATM
203.40.222.0	GW 2	ATM
203.40.223.0	GW 3	ATM

(b) For GW 1 (IP = 202.40.221.1)

Destination	Gateway	Interface
202.40.221.0	x	ATM
203.40.221.0	GW 1	ATM
203.40.222.0	GW 2	ATM
203.40.223.0	GW 3	ATM

(c) For ATMARP Server (IP = 202.40.221.4)

Figure 5.12: Example Routing Tables of : (a) Host 1, (b) GW 1 and (c) ATMARP Server

a Default entry is required to be added into GW 1' Routing Table. This is used to direct GW 1 to point to GW 4 for Internet access.

The cited example network has only shown a small ATM Backbone with a few connectionless LANs connected. Therefore, there exists only a few entries in the routing tables and could be configured manually by the network manager. If the number of connectionless LANs is large (i.e., many gateways connected to the ATM network), it may not be feasible for the network manager to manually adding or deleting any routing entry in all the Gateways whenever a new network is connected to or removed from the backbone. Moreover, if a connectionless LAN can be connected to a backbone ATM network through two Gateways, the problem of optimal routes determination will also arise.

In conventional Internet, there are standard network routing protocols[26] and

[26]Such as Exterior Gateway Protocol of Vector Distance (Bellman-Ford) Routing, Gateway-To-Gateway Protocol (GGP), and Interior Gateway Protocol of Routing Information Protocol, etc. as shown in chapter 14, 15 and 16 of [11].

software[27] that provide the functions of optimal routes determination and routing tables updating. However, the software have normally used the broadcasting capability of the shared-medium networks which cannot be provided by the connection-oriented ATM networks. Therefore, the proposal of using a *Routing Server* (Calling it Routing Table Server should be more appropriate) to maintain the routing table database for the Connectionless Gateways is raised as depicted in Figure 5.9. In this proposal, only the issue of routing table maintenance is concerned (i.e., not including the cases of LANs interconnection through multiple ATM networks with multiple gateways.). The issue of route optimization is out of the scope of this thesis. It is assumed that the Gateways only contact the nearest egress switches they are connected to and the routing in the ATM network is left to the ATM routing/signalling protocol such as the Interim Inter-Switch Signalling Protocol (IISP).

The Routing Server is built into the ATMARP Server and the Routing Clients are run in the Gateways. The main reason for using this configuration is that the Connectionless Gateway is assumed to have a SVC connected with the ATMARP Server. This configuration can save connections and workstations.

The operation of the Routing Server and Clients is simple, an example is shown in Figure 5.13. The steps can be summarized as follow:

1. GW 1 is powered up, its routing table contains the entries of the direct routes. It then transmits its routing table to the Routing Server[28].

2. When the Routing Server receives this table, it will sort out the required

[27]Such as "routed" (route daemon) and "gated" (gateway daemon).

[28]The initialization processes such as ATMARP Server registration, address registration, etc. have been performed.

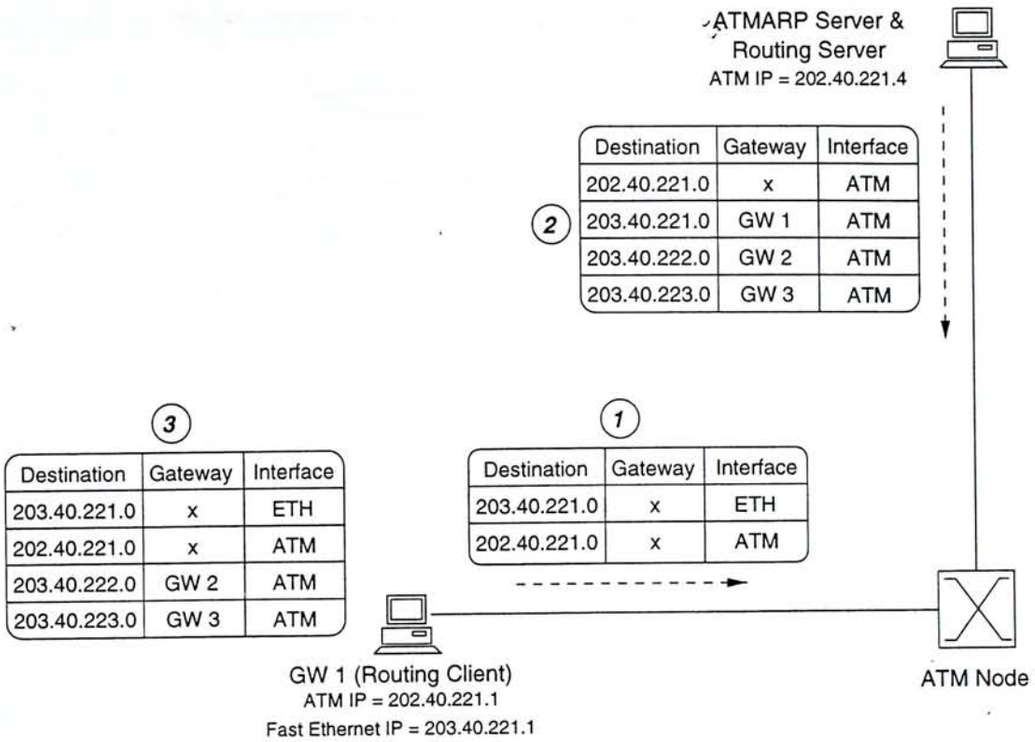


Figure 5.13: A simple example shows the process of routing table setup for a Connectionless ATM Gateways

entries that should be added into its routing table. In this example, it is the entry of the Fast Ethernet network.

3. After the new route(s) has(have) been added to the Server's Routing Table, it will respond by sending back its Routing Table to GW 1.

4. When GW 1 receives the routing table from the server, it will similarly find out the entries of the destination network addresses that do not appear in its routing table and those entries will be added into its table.

In order to ensure the information in the routing tables are up-to-date, a life timer of 30 seconds is set in each table. The Routing Server will periodically request the Gateways for their routing table every 30 seconds. The Routing Server will then transmit its routing table to all the connected Gateways in

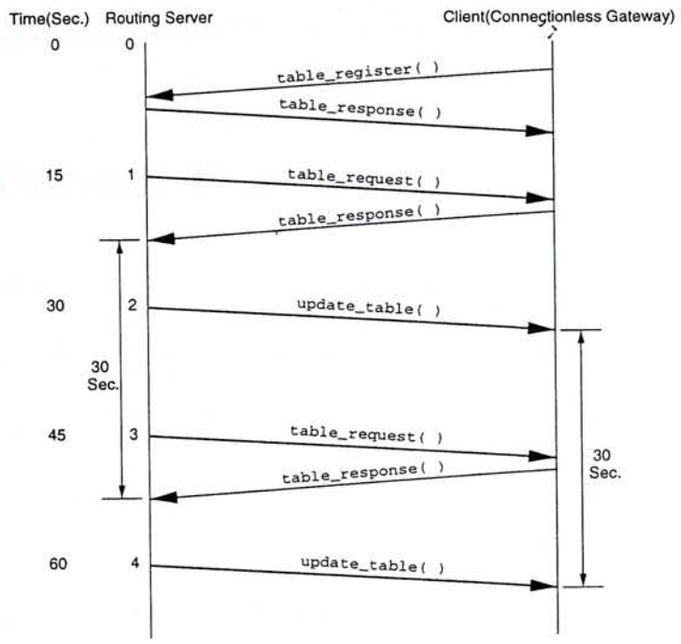


Figure 5.14: Timing relations between the events of routing table expiring and updating

another 30 seconds. The timing relation between these 30 seconds periods is illustrated in Figure 5.14.

The figure depicts an example in which a client (Connectionless Gateway) is started and registered with the Server within the first 15 seconds time interval. The events of tables expiring and updating are also shown. The example of client registration illustrates the fact that the timing of the clients should be synchronous with the expiring and updating events of the Server. It also shows that the routing tables in both clients and server will be valid and updated for every 30 seconds[29]. The choice of the 30 seconds period is based on the specification of the Routing Information Protocol (RIP). Except for the described events operations, other details such as message formats and transport protocol used in this Routing Table Maintenance Program are all referenced from the

[29]Only applicable to networks with constant transmission delay.

RIP. Details of RIP can be found in chapter 16 of [11].

5.6 Additional Features

Besides the implementation of the primitive functions, there are two additional features that have been implemented in the Connectionless Gateway: the *Priority Output Queues System* and the *Gateway Performance Monitor*. They will be described in separate sections.

5.6.1 Priority Output Queues System

As described in Section 5.3.2, only UBR service is provided in the implemented IP over ATM network. Therefore when congestion occurs, all the packets from the Connectionless LANs will suffer from the same transmission delay and dropping probability. In order to introduce some kind of QoS, a *Priority Output Queues System* is implemented in the Connectionless Gateway.

With reference to Figure 5.9, there is one output queue per network device. The output queue receives packets/frames from the Gateway Engine and IP Output Function. In this proposal, two more output queues are reserved for each device. These queues are now named as High, Normal and Low Priority Queues.

With multiple priority queues, a mechanism is required to distinguish the priorities of the packets such that they can be output to the right priority queues. In the IP header, there is an 8-bit field called *Type Of Service(TOS)* which can provide such a mechanism. Figure 5.15 depicts the position of this field in an IP header.

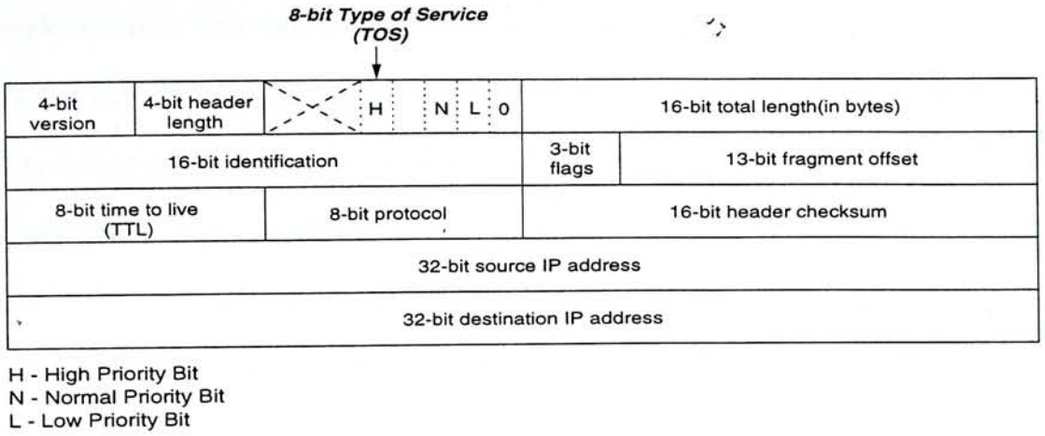


Figure 5.15: IP Header Format.

Reference [50] stated that the first three bits of this TOS field is ignored nowadays and the last bit should be set to zero as shown in the figure. Thus, only four out of the eight bits can be used in assigning priorities for the IP packets. However, each packet can only have one bit set, thus there can be totally four types of priority packets. With reference to RFC 1340[50], the naming convention of the type of service corresponds to the set bits are: Minimize Delay (MSB set), Maximize Throughput, Maximize Reliability, and Minimize Monetary Cost (LSB set). Since we have designed to use three priority queues only, thus only three out of the four TOS bits will be used in the implementation. The used bits are as shown in Figure 5.15. The High, Normal, and Low Priority Packets in this proposal are equivalent to the Minimize Delay (MSB), Maximize Reliability (the third bit), and Minimize Monetary Cost (LSB) type of services respectively.

With sufficient knowledge on the priority/TOS scheme of IP, the next issue of concerned is how a user decide the priority of the outgoing IP packets. With the provision of the BSD Socket API, applications/users can easily access the kernel

network services through the sockets. Similarly, applications/users can also access/alter the IP TOS through the system calls provided by the Socket API. The system call provides such service is `int setsockopt(int sockfd, int level, int optname, char *optval, int *optlen)`, it is pronounced as "Set Socket Option". This is a generic system call which is used in setting many options. The first argument *sockfd* is the socket identifier (ID)/file descriptor (FD) which is used to specify which socket it is going to be modified[30]. The value of *level* is `SOL_IP` which means that the call is going to set the IP options. *optname* is set to `IP_TOS` which means that the call is going to set the IP TOS. Finally, *optval* and *optlen* contain the option (TOS) value and the data size of the option value respectively. The option (TOS) values are: `0x10`(High Priority), `0x4`(Normal Priority) and `0x2`(Low Priority). In a newly created socket, the default priority is set to Normal. Therefore, applications/users must set the desired priority explicitly by calling the `setsockopt()` system call. Currently, only a few applications will set the priority of the packets other than Normal, e.g., File Transfer Protocol (FTP) sets the data packet to Low Priority while other request packets are set to Normal Priority.

Figure 5.16 shows the block diagram of the Priority Output Queues System. A "Device Queueing Function" which is not shown in Figure 5.9 is placed in between the IP Output Function/Gateway Engine and the Priority Queues. The two output functions are responsible for selecting the right output queues and devices, while the "Device Queueing Function" passively places these frames to the respected device queues.

The Device Queueing Function is also responsible for dequeueing frames from

[30]The socket must first be created.

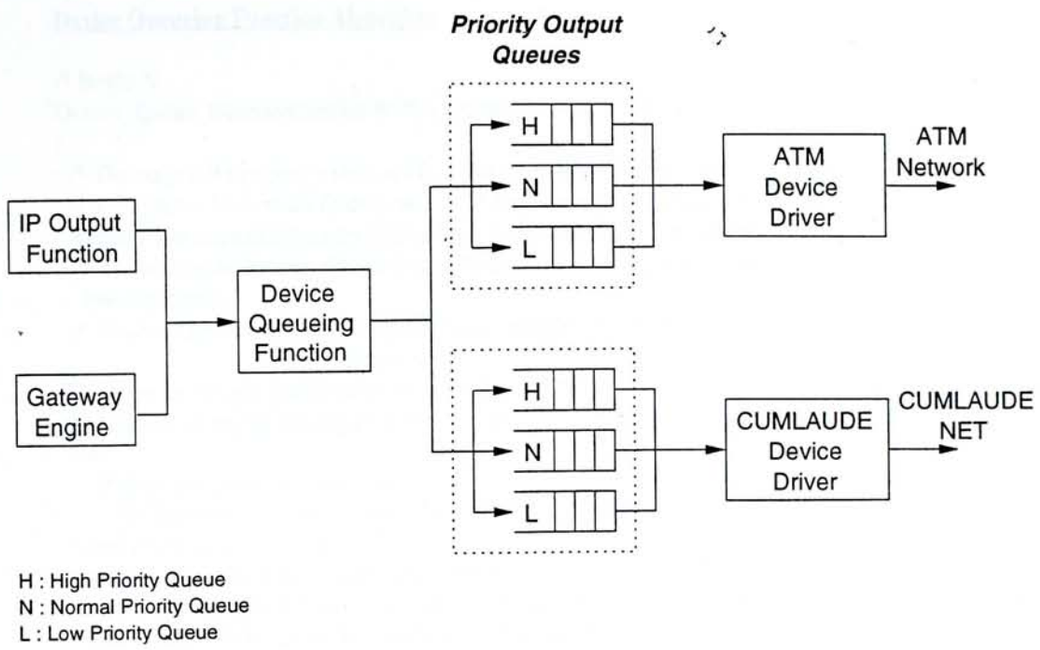


Figure 5.16: Block diagram of the Device Priority Output Queues.

these queues and transferring them to the device drivers for transmission.

The queueing discipline employed by the Device Queueing Function is First-In-First-Out (FIFO). In the first phase of implementation, the transmission discipline used is *Absolute Priority Order*. The frames in the higher priority queues will always be served/transmitted first although they may come later than the low priority frames.

After the Device Queueing Function places a frame to the priority queue of a network device, it will immediately dequeue a frame from the High Priority Queue and transfer it to the device driver if there are frames in the queue. Otherwise, the Function will search other lower priority queues. The algorithm of the Device Queueing Function is shown in Figure 5.17.

The implementation of this algorithm is simple and straight forward. However, in case some users transmit substantial amount of High Priority packets, the Normal or Low Priority packets from other users may never (or delay a lot)

Device Queueing Function Algorithm

```
/* Begin */
Device_Queue_Function(Socket Buffer, Output Device, Priority)
{
 /* The only valid priority values are 0, 1 and 2 for High, Normal and Low Priority */
 if (Priority >= 3) then set Priority to 1    /* Priority invalid and set it to Normal(1) */
 Retrieve the respected Priority Queue from the Queue List of the Output Device
 /* The size of the Priority Queue is specified by the device driver and it is not fixed in
   initialization*/
 if (Priority Queue full) then Drop the Packet and Release Buffer
                               return
 Enqueue the Socket Buffer to the Priority Queue
 Set Priority Queue to the High Priority Queue
 Do
    if (Priority Queue is empty) then
       Set Priority Queue to the next lower Priority Queue
 while (there is lower queues)
 Dequeue the first frame from Priority Queue
 if (Pass that frame to the respected Device Driver for output is OK) then return
    else Queue this frame back to the head of Priority Queue
         return
}
/* End */
```

Figure 5.17: Algorithm of the Device Queueing Function

reach their destinations because the Device Queueing Function is always sending out the High Priority packets first. Then, the Normal and Low priorities packets may be dropped seriously and affect the flow of data between users. Therefore, a better frame transmission discipline is designed.

The authors of [51] also observed the similar problem of service starvation for the lower priorities packets when using the similar servicing discipline. They has proposed a dynamic priority scheme by adjusting the relative priorities of the packets according to their mean waiting time. When the time limit for those low priorities packets are expired, they will be dequeued and move to the end of the next higher queues. Thus, the packets should have a chance to be sent as time passes by. However, the problems of this proposal are that if time limits are kept for all the packets, it will impose substantial processing overhead to

the operating system. Moreover, the movement of packets from one queue to another increases the number of memory copy operation and it costs dearly in some low memory bandwidth platforms such as the PCs.

In view of this problem, another priority queue servicing algorithm is proposed which may be regarded as a light weight version of [51]. The working principle of this algorithm is based on *weighting factor*. The Weighting Factor is set between pair of queues, for example, if the Weighting Factor between High and Normal Priority queues is set to 5, then after every 5 High Priority packets are sent, the sixth transmission should be a Normal Priority packet. Similarly, if the Weighting Factor between the Normal and Low Priority queues is also set to 5, then the Low Priority packet may have to wait for 25 transmission of High Priority packets and 5 Normal Priority packets before it can transmit a packet. Consequently, the selection of these Weighting Factors depends on the service parameters such as the maximum tolerable queueing delay. Moreover, if the Weighting Factors could be adjusted dynamically, the likelihood of packet dropping could be reduced, but this requires extensive evaluation and will be considered in the future. The algorithm of this new "Priority Queueing Servicing Function" is shown in Figure 5.18.

Indeed, the algorithm has been directly reflected in the implementation. However, a new data structure is required to be added into the kernel network device data structure stated in `netdevice.h`. This structure is shown below and the meaning of the components have been stated in the algorithm. If all the devices use the same Weighting Factor values, M and N in the structure can be omitted. Otherwise, these values must be initialized/handled by the respective device driver.

Device Queueing Function Algorithm with Weighting Factors

```
/* Begin */
Device_Queue_Function(Socket Buffer, Output Device, Priority)
{
 /* The only valid priority values are 0, 1 and 2 for High, Normal and Low Priority */
 if (Priority >= 3) then set Priority to 1 /* Priority invalid and set it to Normal(1) */
 Retrieve the respected Priority Queue from the Queue List of Output Device
 /* The size of the Priority Queue is specified by the device driver and it is not fixed in
    initialization*/
 if (Priority Queue full) then Drop the Packet and Release Buffer
 return
 Enqueue Socket Buffer to Priority Queue
 /* Transmission Begin */
 /* Noted: LQ_xmit, HQ_xmit_cnt, NQ_xmit_cnt, Low, Normal,
          High Priority Queues, M and N are all specified for Output Device */
 if (LQ_xmit is set) then
 /* LQ_xmit is used to determine whether Low Priority Packet should be transmitted. */
   if (Low Priority Queue is not empty) then Dequeue Frame from Low Priority Queue
                                             Set LQ_xmit back to 0
 else if (High Priority Queue is not empty) then
      /* HQ_xmit_cnt is used to record the number of High Priority Packets transmitted */
      /* M is the relative Weighting Factor between High and Normal Priority Queue */
        if ((HQ_xmit_cnt modulate M) = 0) then
          if (Normal Priority Queue is not empty) then
            Dequeue Frame from Normal Priority Queue
            NQ_xmit_cnt = NQ_xmit_cnt + 1
            if ((NQ_xmit_cnt modulate N) = 0) then
            /* NQ_xmit_cnt is used to record the number of Normal Priority Packets transmitted */
            /* N is the relative Weighting Factor between Normal and Low Priority Queue */
            Set LQ_xmit
            /* Then, Low Priority Packet gains the next transmission right */
        else Dequeue Frame from High Priority Queue
             HQ_xmit_cnt = HQ_xmit_cnt + 1
             if ((HQ_xmit_cnt modulate (MxN)) = 0) then Set LQ_xmit
             /* This is used to avoid the case when only the High and Low Priority queues have
                frames waiting for transmission */
      else if (Normal Priority Queue is not empty) then
             Dequeue Frame from Normal Priority Queue
             NQ_xmit_cnt = NQ_xmit_cnt + 1
             if ((NQ_xmit_cnt modulate N) = 0) then Set LQ_xmit
   if (Pass Frame to the respected Device Driver for output is OK) then return
   else Queue Frame back to the head of its priority queue
   return
}
/* End */
```

Figure 5.18: Algorithm showing the "Device Queue Servicing Function" with the introduction of the "Weighting Factor".

```
struct dev_priority_q
{
        unsigned int        M;
        unsigned int        N;
        unsigned int        LQ_xmit;
        unsigned int        HQ_xmit_cnt;
        unsigned int        NQ_xmit_cnt;
};
```

5.6.2 Gateway Performance Monitor

Gateway Performance Monitor is an application program that runs under the X-window system which measures the bandwidth transfer across the Connectionless Gateway in real-time. Figure 5.19 depicts the main and control panels of this application.

As illustrated in Figure 5.19, the bandwidth transfer across the ATM, Ethernet and CUMLAUDE NET are displayed in numerical figures and in a relative bar graph. The average size of the ATM and Ethernet packets are also displayed, but the average packet size of the CUMLAUDE NET device is not shown because the CUMLAUDE packet is fixed in size. The number of packets forwarded by the Gateway is also displayed.

The application also provides monitoring functions on the Routing Table, ATMARP Table and Connections (SVC and PVC) Information as illustrated in Figure 5.20.

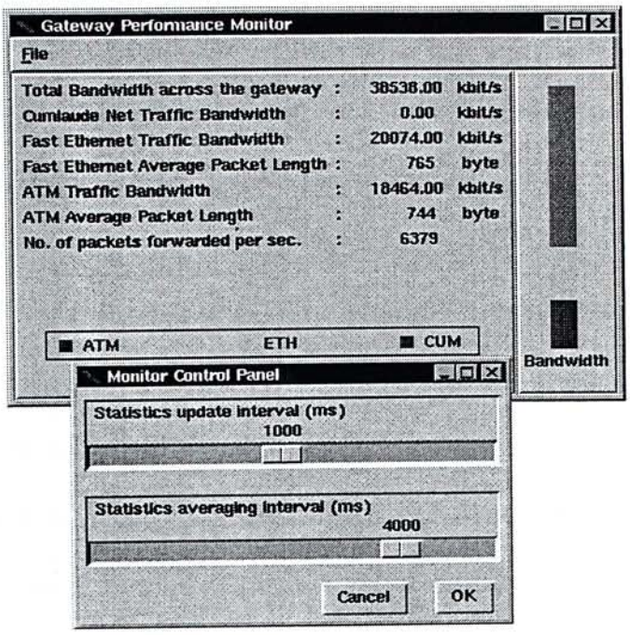


Figure 5.19: Main and Control Panels of Gateway Performance Monitor.

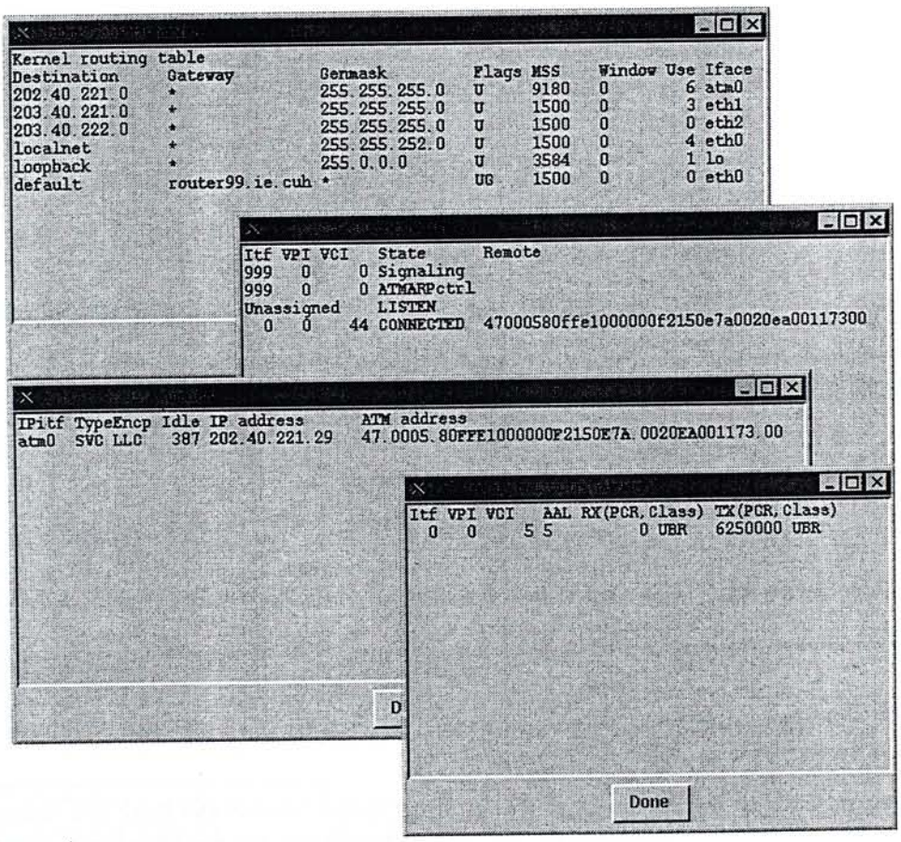


Figure 5.20: Other monitoring panels of Gateway Performance Monitor.

(a) Software Architecture of the Gateway Performance Monitor

The software architecture can be divided into three separate modules: the *Graphical User Interface (GUI) Module*, *Statistics Collection Module* and *Data Updating Module.*

The *Graphical User Interface (GUI) Module* is written by using a programming language known as *Tcl/Tk*[31]. The interfaces depicted in Figures 5.19 and 5.20 are the visual results. The GUI Module only provides the graphical interface for the users. It accepts data from the other modules to display.

The *Statistics Collection Module* is a stand alone process written in *C*. It is executed as a child process of the main modules (i.e., the other two modules). A special `socket()` with type `SOCK_PACKET` and protocol `ETH_P_ALL` is created[32] in this modules to listen to all the IP packets received by all the network devices. By examining the socket address of the received packet, the input device of the packet can be determined and the statistic of the respective device can also be updated. A data structure is created to handle the statistical information for the devices is shown below:

```
struct req_data
{
        unsigned long       ATM_recvbytes;
        /* Accumulate bytes received by the ATM Network */
        unsigned long       ATM_pkcnt;
        /* Accumulate number of packets received by the ATM Network */
```

[31]Tcl stands for Tool Command Language and Tk is a toolkit for X-window programming. Details about this language can be found in [52, 53, 54].

[32]This socket type is specifically designed in the Linux system and only super-users can created this type of socket.

```
    unsigned long       CUM_recvbytes;
    unsigned long       CUM_pkcnt;
    unsigned long       ETH_recvbytes;
    unsigned long       ETH_pkcnt;
};
```

Lastly, but most importantly is the *Data Updating Module*. This module is the bridge between the GUI and Statistics Collection Modules. It periodically retrieves data from the Statistics Collection Module, performs functions such as data averaging and re-calculation and finally passes the results to the GUI Module for displaying. The statistics updating time is default to 1 second and can be changed in the control panel (Figure 5.19). Moreover, *moving window averaging algorithm* is implemented in this module such that smooth reading can be observed. The default averaging time is set to 4 second as shown in Figure 5.19. It could also be altered in the Control Panel. The working principle of the moving window averaging algorithm is simple and it is illustrated in Figure 5.21.

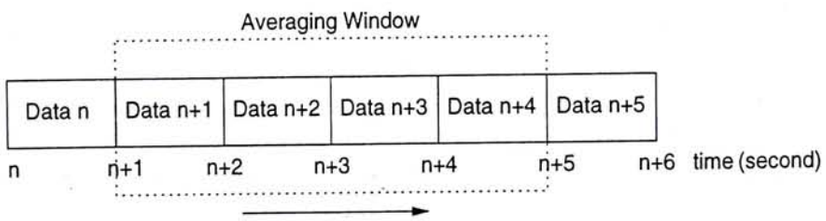


Figure 5.21: Example of the Moving Window Averaging Algorithm.

If the "Statistics Averaging Interval" is 4 second and "Statistics Update Interval" is 1 second, then there will always be 4 (i.e., Statistics Averaging Time/Statistic Update Interval) data to be averaged in every Statistic Update

Interval. In the implementation, a circular buffer of size "Statistics Averaging Time/Statistics Update Interval" is reserved to store the raw data. Since the raw data is accumulative in nature[33], by proper indexing the buffer space, the most recent data can be arranged to be subtracted from the data stored "Statistics Averaging Time" second before. The average can then be obtained by further dividing the result with the value "Statistics Averaging Time". Finally, this most recent data will be stored into the circular buffer with index same as the subtraction data.

In order for the Data Updating Module to gather information from the Statistics Collection Module, an interprocess communication (IPC) channel is created between them. The IPC technique adopted in the implementation is *Shared Memory*[55]. Since the Data Updating Module is written in C while the GUI Module is written in Tcl/Tk, thus a "middle-man" is required to stick them together. This "middle-man" is an application called *Embedded Tk system (ET)*[56] which provides the facilities for data communication between C and Tcl/Tk programs.

(b) A Simple Manual

This Gateway Performance Monitor application is compiled as a single execution file. It is invoked under the X-window system by typing the program name "guigateway". This program can only be executed by the super-user since it has a special socket that can only be called by the super-user.

The Monitor Control Panel is popped up by double clicking the bar graph. The values can be changed by sliding the two buttons. The other monitoring

[33] i.e., the most recent data is always larger than the last one.

functions can be called under the *File* pull-down menu. Finally, the program is terminated by select *Exit* under the File menu.

5.7 Setup an Operational ATM LAN

5.7.1 SVC Connections

If SVCs are used in connecting the Connectionless Gateways, only two types of components (endstations) are required to be setup specially: the Connectionless Gateway and the ATMARP Server.

Among them, the ATMARP Server should first be started[34]. The Gateways are started on demand. Two simple scripts (`atmconfig`) have been written to accomplish the task of initialization. The following is the content of the `atmconfig` script for the ATMARP Server:

```
atmsigd -b
ilmid -b
atmarpd -b
atmarp -c
ifconfig atm0 202.40.221.4 up
route add -net 202.40.221.0
```

The line `atmsigd -b` starts the ATM signalling daemon of the Server. This signalling daemon must first be started, otherwise the Server cannot communicate with the ATM switch and the script lines follow will then be invalid. The

[34]This is because the ATM address of the ATMARP Server is required to be registered in the Gateway setup procedure.

line `ilmid -b` starts the ILMI daemon which provides the function of address auto-configuration with the switch[35]. After this daemon is started, the ATMARP Server will obtain its ATM address from the switch. Then, other ATM nodes in the network can locate this Server with this address.

The line `atmarpd -b` starts the ATMARP daemon which is used to generate the ATMARP queries and generates replies to these queries. While the line `atmarp -c` initializes and configures an IP-ATM interface for the ATM device. Finally, `ifconfig` and `route` are the general commands used in configuring the IP devices. In the line `ifconfig atm0 ...`, the parameter `atm0` stands for the name of the IP ATM interface just created by `atmarp -c`. `202.40.221.4` is the IP address of this Server and `up` means turn on this interface. The command `route` is used to add a route of the newly created interface into the routing table of the host. If this last step is not done, the host will never know it is connected to the network of address 202.40.221.0.

The `atmconfig` script for the Connectionless Gateway is similar to the previous one but contains additional script lines. At least one more script line must be added:

`atmarp -s 202.40.221.0` *ATM_address_of_the_ATMARP_Server* `arpsrv`

By invoking this command line, the Connectionless Gateway will create a connection to the ATMARP Server. Then, the Gateway will perform the address registration procedure. After that, some more routes may be required to be added into the routing table of the Connectionless Gateway. Cite "GW 1" of Figure 5.6 as an example, two more routes must be added to its routing table:

```
route add -net 203.40.222.0 gw 202.40.221.2
```

[35]Refer to Section 3.8 of Chapter 3 for the description of address registration process.

`route add -net 203.40.223.0 gw 202.40.221.3`

The keyword of these command lines is **gw**, it stands for Gateway. The first command line tells GW 1 that: "If the packet wants to go to the network of address 203.40.222.0, it should first be forwarded to the Gateway of address 202.40.221.2 (i.e., GW 2)". The second line tells the similar truth.

The order of execution of these command lines must be preserved, otherwise, errors may result. For more information about the options of these commands, please consult [57] or their on-line helps.

Lastly, the hosts of the connectionless LANs should have a "default" route added into their routing tables.

5.7.2 PVC Connections

If PVCs are established between the Connectionless Gateways, a few modifications are required in the setup procedures. Firstly, the ATMARP Server is not needed. However, the ATMARP daemon on each machine is still required in order to perform the InATMARP (Inverse ATMARP) processes. The configuration script of the Gateways is similar to the Gateways using SVC, but the script line used in creating the connection to the ATMARP Server is deleted and replace with the following line:

`atmarp -s` *ip_of_remote_host itf.vpi.vci*

It contains the IP address of the connected host, itf is the interface number of the local ATM interface (normally 0). The administrator should also choose an unused VPI/VCI pair (only a VPI of 0 is supported by the Efficient ATM Interface and the maximum VCI supported is 1023). This can better be explained by an example based on the connections of GW 1:

```
atmarp -s 203.40.221.2 0.0.88
atmarp -s 203.40.221.3 0.0.89
```

Similar commands should also be invoked by the Gateways that GW 1 is connected to. Besides, the administrator should configure the switches/nodes to support these PVCs. Therefore, PVC should only be used in the networks with a few connections which do not need to be altered frequently.

5.8 Application of the Connectionless Gateway

Commercial ATM service in Hong Kong is still not yet available, thus this Gateway will mostly be used in the campus or departmental networks where an ATM backbone is available. The laboratories/departments which originally use the traditional Ethernet network can be connected to the Connectionless Gateway for the connections outside their networks. Then, the machines inside the original networks will not be affected by the changes of the backbone architecture. Whenever the public service is available, the Gateway can also be acted as the front-end for the local network and connected to the public switches by using PVCs. The Connectionless Gateway can be modified as an IP Switch Gateway in connecting the LANs with the IP switch[36]. This technology is very suitable for upgrading the Hong Kong Internet Exchange (HKIX)[37] to an ATM-based high-capacity switch.

[36]IP Switch is a technology that uses the ATM switch to switch IP packets instead of cells.
[37]For detailed information, please refer to their web-page at http://www.cuhk.edu.hk/hkix/.

Chapter 6

Performance Measurement of the Connectionless Gateway

6.1 Introduction

After the implementation of the Gateway has been completed, a number of experiments are performed to test its performance. The experiments are mainly designed to measure the maximum *throughput* and locate the *bottleneck* of the Connectionless Gateway. Additionally, the behaviour of the prioritized packets are also studied.

6.2 Experimental Setup

The experimental setup (Figure 6.1) consists of four hosts: two Fast Ethernet hosts, one ATM host and one Connectionless Gateway. The hosts are all

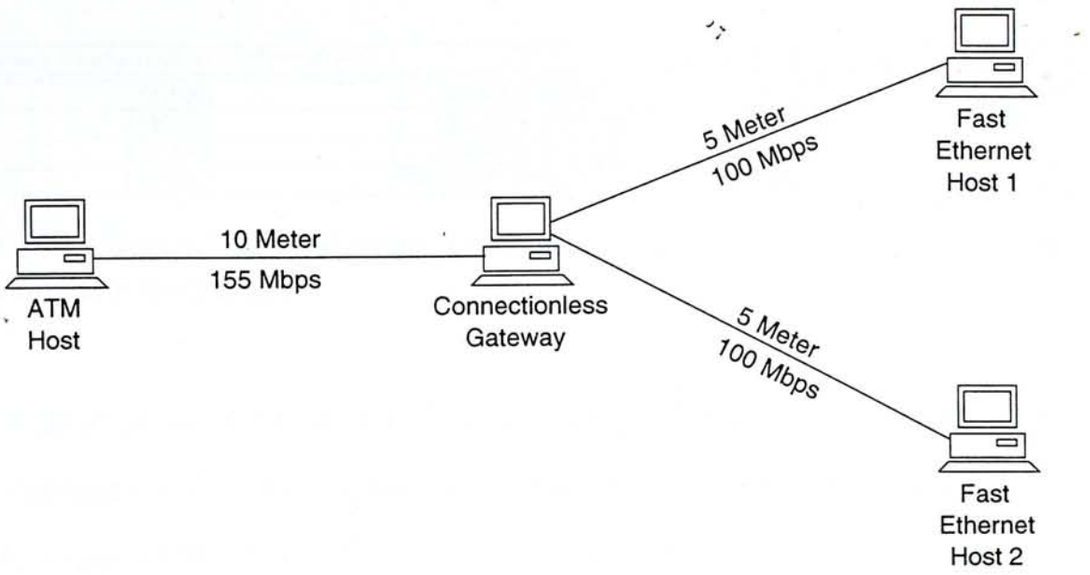


Figure 6.1: Experimental setup designed for the performance measurement experiments.

connected point-to-point purposely to minimize/eliminate the possible performance degradation imposed by the extra equipment such as ATM-switches and Ethernet-hubs. Moreover, the network resources contention problem has been eliminated. As the length of the connection cables are relatively short (5 m for Ethernet-to-Ethernet connections and 10 m for ATM-to-ATM connection), so the delay-bandwidth products are becoming insignificant[1].

The ATM Host (ATM-Host) and Fast Ethernet Host 1 (ETH-Host-1) are both PCs equipped with a 120-MHz Pentium CPU, mainboard using the Intel 430HX PCI chip set[2] and 32-MB of memory (ram). Fast Ethernet Host 2 (ETH-Host-2) is idle most of the time in the experiments and it is configured with a

[1]For the ATM Link, the delay-bandwidth is about 0.036 cell or 15 bit, assuming the speed of light in the optical fiber is approximately $2 * 10^8$ meters/sec and the maximum cell rate is 353207.547 cells/sec. For the Ethernet link, the delay-bandwidth product is about 5 bit assuming an effective data rate of 90 Mb/s.

[2]PCI Chip Set is a set of ASICs (Application Specific Integrated Circuits) that interconnect the CPU with the memory and buses. For detail information, please refer to http://www.intel.com.

Gateway Configuration (Abbreviation)	CPU	Main Board Chip Set	Memory Size
1 (P166)	166-MHz Pentium	430HX	64-MByte
2 (P120)	120-MHz Pentium	430HX	64-Mbyte
3 (P100)	100-MHz Pentium	430FX	64-Mbyte
4 (PPro)	200-MHz Pentium Pro	440FX	64-Mbyte

Table 6.1: System configurations/components of the Connectionless Gateways used in the experiments.

100-MHz Pentium CPU on a mainboard using 430FX chip set. While the system configuration (components) of the Connectionless Gateway is not fixed in all the experiments, four different configurations are used as shown in Table 6.1. Gateway Configuration 1 is mostly used in the experiments (i.e., the one uses a 166-MHz Pentium CPU). The hardware configurations of the hosts are emphasized because in the experiments, we will try to find out the relationships between the hardware configurations of the Gateway and its performance.

6.3 Measurement Tools of the Experiments

In testing the throughput of the Gateway, a network performance bench-marking software called "Netperf"[3] is used. Netperf's primary focus is on bulk data transfer and request/response performance tests using TCP/UDP, BSD Socket and other communication protocols[58].

Netperf consists of two executables, "`netserver`" and "`netperf`". `netserver` is a concurrent server that runs as a daemon[4] to listen to the default TCP port of 12865. `netperf` is the client software that requires the supply of parameters to

[3]This tool is developed by the IND Networking Performance Team of Hewlett-Packard Company. The source code of it can be retrieved from http://www.cup.hp.com[58].

[4]Or installs under the system's `inetd` (Internet daemon).

specify the type of the test (e.g., UDP or TCP, transmission rate, etc.). Before it starts a test, a control connection is made to pass test configuration information and results to or from the server (`netserver`) through the same known port . There are numerous parameters that can be specified and the details can be found in [58]. Since `netperf` is executed using line commands, thus a number of UNIX shell scripts are written to run different tests with different parameters.

Two more measurement tools are designed to measure the delay characteristic of the prioritized packets transmitting through the priority queues. The first one is a classical UDP client-server program (thereafter call it *Traffic-Generator*) in which the client generates Poisson distributed packets to the server and supports the transmission of prioritized packets. The purpose of this program is to generate the background traffic. The main tool is developed by modifying the well-known "Ping" program to support the transmission of prioritized probing packets (thereafter call it *Delay-test*) to determine the network round trip delay.

6.4 Descriptions of the Experiments

In summary, there are five types of experiments that should be performed in order to:

1. determine throughput of the Gateway under different traffic condition,
2. study the relationship between the hardware configuration of the Gateway and its performance,
3. study the dependency of the Gateway performance on different packet sizes,

Experiment Group	Name of the Test
1	UDP Control Rate Test, UDP Maximum Rate Test, TCP Maximum Rate Test
2, 3	UDP Control Rate Test
4	Request/Response Test
5	Priority Queue System Verification Test

Table 6.2: Usage of the designed tests with respect to the experiment groups.

4. measure the delay imposed by the Gateway and,

5. verify the effectiveness of the Priority Queue System in the Connectionless Gateway.

Five tests have been designed to achieve these five goals. They are namely as *UDP Control Rate Test*, *UDP Maximum Rate Test*, *TCP Maximum Rate Test*, *Request/Response Test* and *Priority Queue System Verification Test*. Table 6.2 has shown the usage of each test with respect to the experiment groups. Details about each test are described in separate sections. The results and the analysis of each test will also be presented immediately following the descriptions.

6.4.1 Log Files

In the experiments, different log files are saved for analysis. Two log files `dev` and `snmp` are recorded from the directory `/proc/net/`. The file `dev` saves the run-time statistics such as the number of healthy packets received/transmitted, number of packets dropped with reasons of error of the network devices. The file `snmp` saves the processing statistics of the IP, TCP, UDP and ICMP protocol layers.

The last log file is generated by a program called `vmstat`. It saves information about the OS/host such as the number of interrupts, context switching and CPU

utilization. These information are valuable for later analysis. The program saves the information every 20 second during the tests.

6.5 UDP Control Rate Test

By transferring a fixed amount of UDP data traffic between the ATM and Ethernet hosts (say ETH-Host-1 to ATM-Host) in a fixed time period, the responses of the Gateway to different amount of traffic can be studied. This can determine the bottleneck of the Gateway with respect to different sizes of packets.

In this test, seven sizes of test packets (packets in generating the data traffic) are chosen. The data size of these test packets are 1024, 1448, 1472, 1473, 4098, 9152 and 9153 bytes. With the addition of the UDP (8 bytes) and IP (20 bytes) headers, the size of these packets become 1052, 1476, *1500*, *1501*, 4126, *9180* and *9181* bytes respectively. The third and fourth figures are the critical points where fragmentation of IP packets occur in the Ethernet network. The sixth and seventh figures are the points of IP packets fragmentation in the Classical IP over ATM network.

In order to control the traffic rate, the packets are transmitted in bursts. Netperf supports a function of sending packets in burst (*burst-length*) and provides a timer to control the *inter-burst* transmission time. Thus, these two parameters can be used to control the traffic rate. On a PC, the timer resolution that can be set by Netperf is 10ms. Thus, in the experiments, the inter-burst time is fixed to 10ms or equivalently, 100 bursts per second. The variable parameters are then decreased from two to one, the choice of burst-length is dependent on the desired rate of traffic, e.g., if 20-Mb/s of data traffic with size 1472 bytes is

```
Socket  Message  Elapsed  Messages
Size    Size}    Time     Okay      Errors  Throughput
bytes   bytes    secs     #         #       10^6 bits/sec

Burst Length=45, Loop time=7
-----------------------------------------------------------
65535   1472     59.99    269955    0       52.99
65535            59.99    269953            52.99
```

Figure 6.2: Experimental Result of UDP Control Rate Test.

required, the "burst-length" can be calculated as:

$Burst - length = 20M/(100 * 1472 * 8) \approx 17$

Therefore, a 20-Mb/s traffic can be supplied by sending 17 packets of data size 1472 bytes per burst and the number of burst is 100 per second. The final parameter required to choose is the desired traffic rate and they are chosen to vary from 1 to 50-Mb/s with 5-Mb/s per step. We have also designed to generate this set of traffic rates for each packet size in order to observe the dependency of the Gateway performance with different data sizes. For each experiment in the test, 10 one-minute experiment are run for better average results. We observe from Table 6.2 that the experiments of this test can give us valuable results.

6.5.1 Results and analysis of the UDP Control Rate Test

Figure 6.2 depicts one of the experimental results. The first and the second lines shows the statistics of the sender and the statistics of the receiver respectively. In fact, thousands of this kind of result have been collected, but for the simplicity of analysis, they are averaged and plotted as graphs.

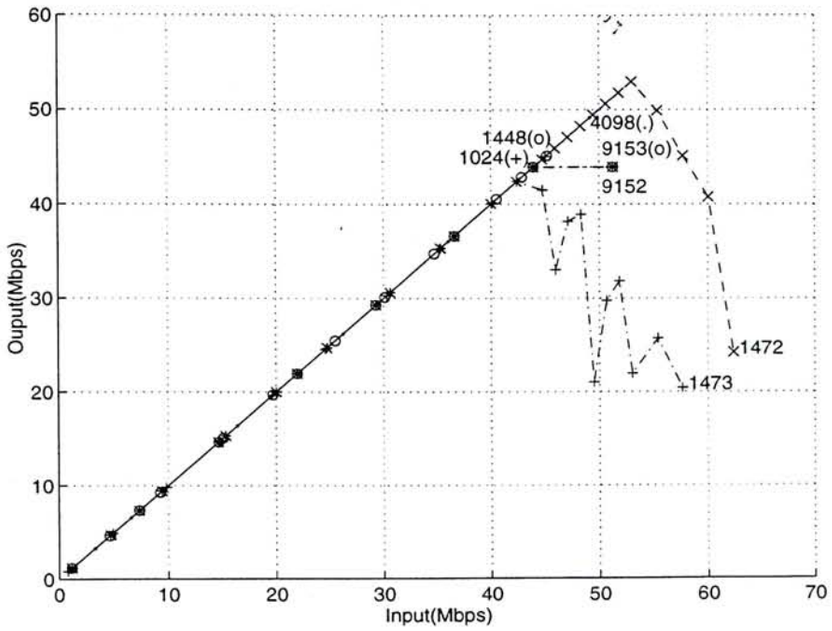


Figure 6.3: Results of the UDP Control Rate Test with ETH-Host-1 as the sender and ATM-Host as the receiver.

(a) Throughput Measurement

Figure 6.3 and 6.4 illustrate the results obtained in the experiments when ETH-Host-1 acts as a sender and ATM-Host acts as a receiver, and the converse respectively. The Connectionless Gateway is configured with a Pentium 166-MHz CPU (i.e., the first Gateway configuration of Table 6.1.) and is connected to the ATM-Host by SVC.

There are totally seven curves shown in each graph and the corresponding datagram sizes used in collecting the data are shown next to the end point of each line. To analyse these graphs, the main concerns are the following:

- Frequency of input/output queues overflow;
- Number of fragmentations;
- CPU usage;

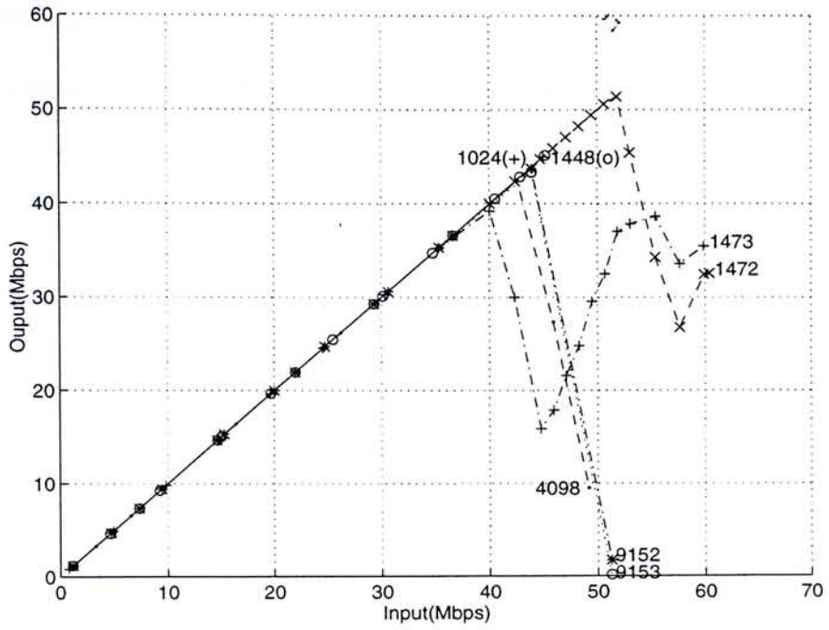


Figure 6.4: Results of the UDP Control Rate Test with ATM-Host as the sender and ETH-Host-1 as the receiver.

- Number of interrupts generated;
- Number of context switching;

(i) Test from Ethernet to ATM

For the curves as shown in Figure 6.3, it can be observed that only four curves have significant changes when the input rate varies. These curves are obtained when the testing traffic are generated by the packets with critical sizes (i.e., 1472, 1473, 9152 and 9153 bytes). Before the analysis, we should note that the amount of traffic that can be generated by the transmission system is limited by the number of frames generated rather than the bandwidth. This phenomenon arises because the system resources (e.g. time) in the OS is insufficient to send out an excessive number of packets. Moreover, Netperf limits the number of packets transmitted based on the run-time system information provided by the system call `sysconf()`. This can explain the trend of the curves of the experiments using

1024 and 1448 bytes packets. In both Figure 6.3 and 6.4, we can only observe that these curves are monotonic increasing straight (nearly) lines. However, we should expect there will be turning points (sudden drop) when the input traffic increases to a certain amount as the other curves shown. The occurrance of this sudden drop is because the Gateway cannot process such substantial amount of traffic. This does not occur in the experiments using relatively small sized packets because the generated traffic rate of the source are limited by the frame rate.

Figure 6.3 shows that the maximum throughput the Gateway can reach is *52.97 Mb/s* (Input rate is 52.98 Mb/s) when the datagram size is 1472 bytes, while its throughput drops to *42.405 Mb/s* (Input rate is 42.415 Mb/s) when the datagram size becomes 1473 bytes. The discrepancy is as much as 20% (i.e., 10.565 Mb/s), but the packet size differs only by 0.067% (i.e., 1 byte). The main cause for the discrepancy is the occurrence of packet fragmentation in ETH-Host-1, but not in the Gateway. When the input rate of 1472 bytes datagram is 52.98-Mb/s, there is approximately 4500 packets sent by ETH-Host-1 to the Connectionless Gateway. In the case where the datagram size is 1473 bytes, the number of packets sent by ETH-Host-1 to the Gateway is 7200 for an output rate of 42.415-Mb/s. It means that the Connectionless Gateway are busy in serving the hardware interrupts generated by the Ethernet network device. Since the routines (network device driver) which provide services to the hardware interrupts are situated in the bottom half[5] of the kernel, they are never scheduled and cannot be blocked[59]. The IP protocol is situated in the

[5]The kernel of the UNIX system is divided into two halves. In a layer structure, the "bottom half" is situated above the physical layer (i.e., hardware device) and the "top half" is on top of the bottom half.

top half of the kernel and may be interrupted by the routines of the bottom half[59]. Therefore, when the number of arrived packets increases, it implies that the number of interrupts increases and the IP protocol service time will be decreased. The input packets may then have to be queued in the *backlog queue*[6] (Refer to Figure 5.9 of Chapter 5.). However, when most of the processing time is occupied by the interrupt routines, the upper layer may not have enough time to process all these incoming/queued packets. The backlog queue will finally be overflowed and packets will be dropped.

When the number of packets inputs to the Gateway further increases, more problems are observed and they are described sequentially as follows:

- Packets arrived at the Connectionless Gateway with a rate higher than its service rate;
- The un-serviced packets are queued up in the backlog queue;
- During packet processing, if new packets arrive, the Connectionless Gateway will be forced[7] to pick them up;
- During the pick up period, the packet processing will be temporarily suspended;
- Deadlock may then occur if packets are continuously arriving at the Gateway and all these packets will be dropped;
- If the Gateway is too busy to pick up the newly arrived packets, the buffer of the network interface will be overrun and packets will be dropped also;

[6]The current maximum queue size is 300 packets.

[7]It is because the hardware interrupt the Gateway, and the priority of this interrupt is higher than the current running process.

- The ATM Connection will even break down because the Service Specific Connection Oriented Protocol(SSCOP) of the ATM Signaling AAL of ATM Host cannot receive the Timer Poll reply from the Gateway[8]. Thus the network connection is assumed broken by SSCOP and the existing connection will also be reset.

By examining the run time kernel messages and log files, the causes of throughput drop for the curves using packets of size 1472 and 1473 bytes can be explained. Firstly, at the points where the throughput rate starts to drop, the backlog queue is found to be full and any newly arrived packets are dropped. Moreover, the buffer of the Ethernet network interface is overflowed and the new incoming packets are dropped before being processed by the kernel.

By examining the recorded results of `vmstat`, it is found that the number of context switching in the experiments using datagrams of size 1473 byte is approximately constant at 400 times/second. It shows that context switching is not the major factor that affects the performance of the Gateway though it is always claimed as a time consuming process. The CPU usage is also found to be low. Therefore, the major cause of packet drop is the presence of a substantial number of packets that cause too many interrupts to the Gateway and perform too many memory copy operations (from the network interface to the kernel). Since the memory bandwidth of PC is claimed to be low[45], thus these memory copy operations in the Gateway become a major bottleneck.

The result curves of the experiments use datagrams of sizes 9152 and 9153 bytes are explored. There is no significant throughput difference between the curves since the numbers of fragmented packets generated are the same for these

[8]The Gateway acts as the network side (i.e., switch node) in the test.

two data sizes. However, it is found that the curves use 9152 and 9153 bytes packets become flattened at the points *43.917*-Mb/s and *43.921*-Mb/s respectively. By inspecting the `dev` and `snmp` files recorded from the testing machines, it is found that the Gateway can successfully forward all the packets transmitted by ETH-Host-1. However, the data of the `snmp` files in the ATM-Host show that the *UDP* datagrams decapsulated from the packets of ETH-Host-1 have errors and are discarded. These errors are caused by the overflow of the socket buffer queue[9]. It thus shows that the receiver's application (i.e., `netperf`) cannot process such huge amount of UDP datagram (packets).

(ii) Test from ATM Host to Ethernet Host

Next, the experimental results obtained when ATM-Host acts as the sender and ETH-Host-1 acts as the receiver are examined. This test (result shows in Figure 6.4) tests whether the performance of the Gateway is symmetrical with respect to the previous test (i.e., Test direction is from Ethernet Host to ATM Host). In this test, the packet fragmentation processes is done in the Gateway when the data size is greater than 1472 bytes, but they are done in the Ethernet Host in the previous test. The maximum achievable throughput for the curves of experiments using datagrams of sizes 1472, 1473, 9152 and 9153 bytes are 51.387 Mb/s (51.804 Mb/s), 39.209 Mb/s (40.039 Mb/s), 43.735 Mb/s (43.928 Mb/s) and 43.350 Mb/s (43.930 Mb/s) respectively. The figures in the brackets are the corresponding input rates.

In Figure 6.4, it shows that the curve for the data size of 1473 bytes fluctuates when the input traffic rate ranges from 40 Mb/s to 60 Mb/s. After analysing

[9]This socket is the BSD socket, not the socket of the Linux Kernel.

the log files (`snmp` and `dev`), the trend of this situation can be explained. The causes for the dropping of output rate (i.e., approximately from 40-45 Mb/s) are as follows:

- Some packets are dropped by the Gateway as they overflow its input backlog queue;
- Most of the packets can pass through the Gateway and arrive at the receiver host (ETH-Host-1). However, the socket buffer queue of ETH-Host-1's application (i.e., `netserver`) is flooded by this substantial amount of input packets.

When the traffic output from the ATM-Host further increases, more packets are dropped by the Connectionless Gateway because of input backlog queue overflow. Some of these dropped packets do not cause any warnings or errors. However, overflow of socket buffer in ETH-Host-1 is alleviated until the input rate becomes 55.368 Mb/s (Output is 38.635 Mb/s). This is because some of the packets are now dropped by the Gateway and thus effectively reduce the input traffic to ETH-Host-1. A further increase in the input rate will trigger the occurrence of the SSCOP problem described previously.

The sharp drop of the curves for data sizes of 9152 and 9153 bytes are due to the input backlog queue overflow at the receiver (i.e., ETH-Host-1) which causes a deadlock situation and the kernel cannot allocate sufficient kernel memory for the Ethernet device driver to store up the packets.

Lastly, by comparing the shape of the curves in Figures 6.3 and 6.4, it can be found that the performances of the Gateway is approximately symmetrical except for the case with the 1473-byte probing packets. Table 6.3 shows the

Data Size (Bytes)	Throughput(Mb/s) (E→A)[10]	Throughput(Mb/s) (A→ E)[11]	Percentage Difference (E→A > A→E)[12]
1472	52.970	51.387	3.08
1473	42.405	39.209	8.15
9152	43.917	43.735	0.42
9153	43.921	43.350	1.32

Table 6.3: Comparison of the throughput of the Connectionless Gateway in different test directions and with different data sizes.

throughputs gathered and their relative degradation in these experiments. It has been revealed that the fragmentation process in the Connectionless Gateway is not the determining factor for the Gateway performance.

(b) Performance Comparison of the Connectionless Gateways

The configuration of the Connectionless Gateway will be changed in order to find out the relations between the performance of the Gateways and their hardware configurations.

Figures 6.5 and 6.6[13] show the performance curves when the data sizes are 1472, 1473 (Figure 6.5), 9152 and 9153 (Figure 6.6) bytes with different Gateway configurations (the respective Gateway used in obtaining each curve is stated in the bracket). In these experiments, ETH-Host-1 is the sender and ATM-Host is the receiver. The configuration of the Gateways under tests are P166, P100 and PPro (Refer to Table 6.1 for explanation).

Normally, it is expected that the machines with faster CPUs should perform better. However, as shown in the figures, the performance of the PPro machine is worse than the P166 Gateway. While the PPro CPU is nearly 0.423 times

[13]These curves are drawn from the same set of results but separated because a clear presentation is desired.

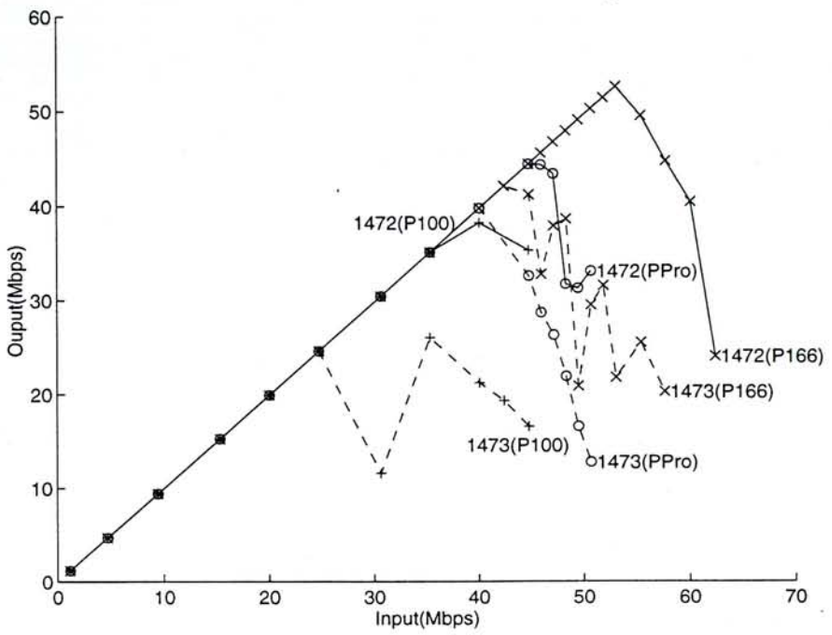


Figure 6.5: Comparing the Performance of the Gateways when the datagram sizes are 1472 and 1473 bytes.

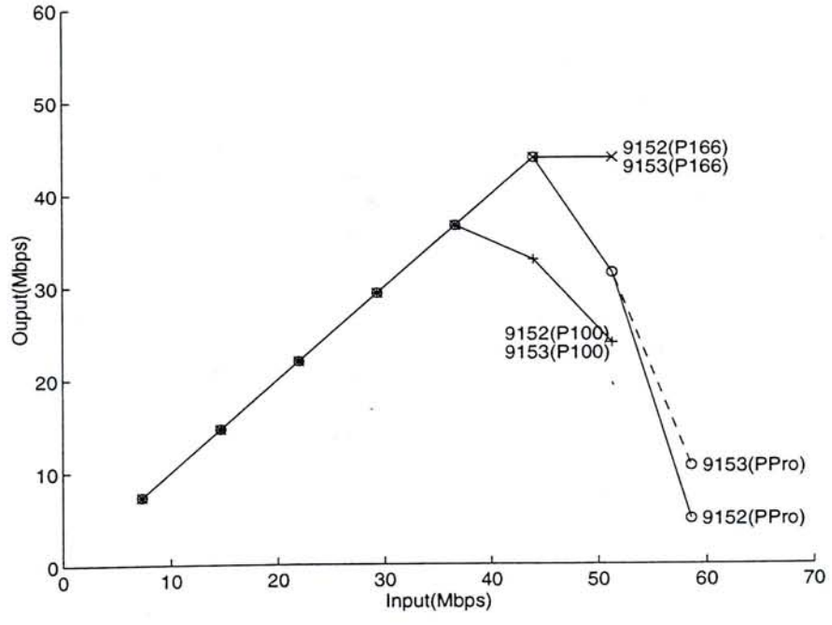


Figure 6.6: Comparing the Performance of the Gateways when the datagram sizes are 9152 and 9153 bytes.

Data Size (Bytes)	Throughput PPro (Mb/s)	Throughput P166 (Mb/s)	Throughput P100 (Mb/s)	Percentage Difference PPro and P166	Percentage Difference P100 and P166
1472	44.730	52.970	38.497	18.42	37.60
1473	40.051	42.405	26.242	5.88	61.59
9152	43.917	43.921	36.584	0.009	20.00
9153	43.921	43.914	36.607	-0.02	19.96

Table 6.4: Throughput of the Gateways

faster than the P166 CPU[14].

Therefore, it reveals that there are other factors affecting the system performance. Beside of the CPU, other determining factors are memories(EDO main memory and cache memory), system buses(PCI and ISA), PCI chipset and the interfaces. Except for the PCI chipset, other components used in the Gateways in the experiments are the same, thus it is one of the major factors that degrades the system performance. When packets are transmitted to or received from other systems, the memory copying processes between network devices and the kernel is one of the major processes, reference [60] has named this type of copying process as *DataMove* which is a data-touching operation. It means that the time required for the completetion of this process is packet size dependent. Therefore, we can conclude that the operations in the Connectionless Gateway is I/O intensive rather than computation intensive. More evidences will be provided in the subsequent analysis to support this conclusion. Table 6.4 summarizes the achievable throughput and the comparison for each configuration of Gateway.

[14]Each CPU's Benchmark is: PPro/220, P166/127 and P100/90. This is the official value supplied by Intel(`www.intel.com`) under the iCOMP Index2.0 standard.

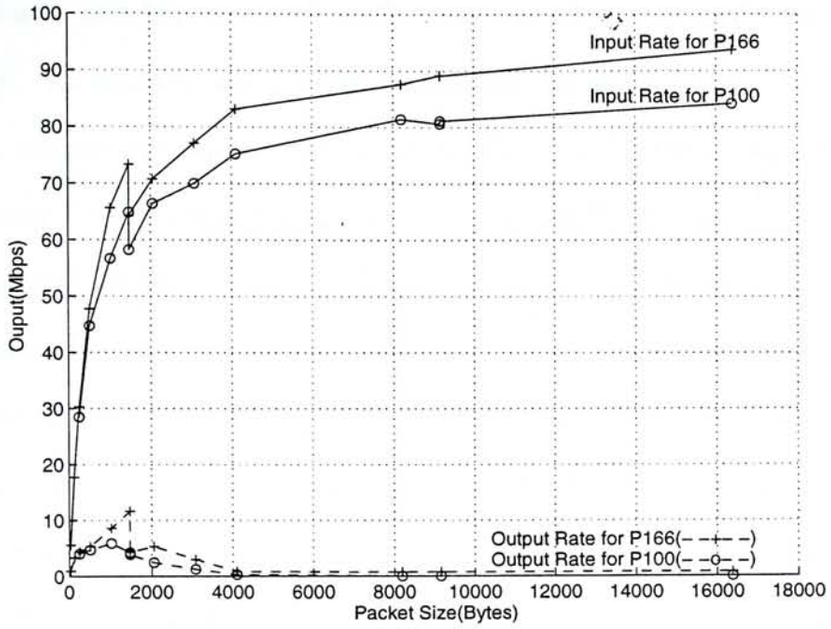


Figure 6.7: Performance results of the P166 and P100 Gateways in the UDP Maximum Rate Test.

6.6 UDP Maximum Rate Test

The difference between this test and the UDP Control Rate Test is that the burst-length and inter-burst time are not specified, i.e., the sender will send as much datagrams as possible. The data sizes chosen for this test also vary, they are 36, 128, 256, 512, 1024, 1472, 1473, 2048, 3072, 4096, 8192, 9152, 9153 and 16384 bytes. The number of one-minute experiment run for each data size is increased to 25.

6.6.1 Results and analysis of the UDP Maximum Rate Test

Figure 6.7 depicts the results of the test on the P166 and P100 Gateways. In the figure, the traffic rate output by the sender are shown in solid lines (lines

in the top half) and the received traffic rate are shown in dash lines (lines near the X-axis). As shown, the output rates are extremely low. Only a limited throughput can be obtained when the data size is 1472 bytes (approximately 11 Mb/s for using P166 Gateway and 5 Mb/s for using P100 Gateway) and a sudden drop is observed when the data size increases to 1473 bytes (approximately 5 Mb/s for using both P166 and P100 Gateways). Moreover, it shows that the performance of the P166 Gateway is much better than the P100 Gateway as the maximum throughput processed by the P166 Gateway is 97.7% higher than that of the P100 Gateway (The respected throughput when using the P166 and P100 Gateways are 11.651 Mb/s and 5.894 Mb/s).

Other experiments have also been done by segmenting the testing loop into two (i.e., ETH-Host-1 to Gateway and Gateway to ATM-Host). It is found that the achievable throughput in the ATM to ATM segment is relatively high (maximum 76.856 Mb/s). However, when the packet size increases up to 9153 bytes (fragmentation point), the throughput is dropped to nearly zero. The achievable rate for the ETH to ETH segment is extremely low when compare with the ATM segment, it is approximately equal to the achievable throughput of the test result previously presented (Figure 6.7). Thus, we can conclude that the Ethernet components (device and device drivers) are one of the other major bottlenecks.

6.7 TCP Maximum Rate Test

As a matter of fact, most of the applications[15] run in the Internet are using TCP as the transport layer protocol instead of UDP. This is because TCP can provide reliable services and flow control function[50], thus it is desirable to determine the maximum bandwidth of TCP traffic that can be transported through this Connectionless Gateway.

Similar to the previous tests, pre-defined packets of size 24, 128, 256, 512, 1024, 1460, 1461, 1472, 1473, 2048, 3072, 4096, 8192, 9152, 9153 and 16384 bytes are chosen. Since the TCP header is 12 bytes larger than the UDP header, thus the critical points have been changed to 1460 and 1461 respectively.

TCP uses byte-stream transportation mode and provides flow control on the stream, thus the bandwidth of the TCP traffic cannot be controlled. Thus, this test is called "TCP Maximum Rate Test". It can also explain why we do not choose to use TCP in the Control Rate Test. It is because we would like to study the responses of the Gateway at different traffic rate. The number of one-minute experiment run for each data size is 20.

6.7.1 Results and analysis of the TCP Maximum Rate Test

Figure 6.8 shows the test results for the PPro, P166 and P120 Gateways. The performance of the P166 Gateway is still the best, followed by the P120[16] Gateway and the PPro Gateway.

[15] Such as `telnet`, `ftp`, etc.

[16] The processing power of the P120 CPU is only approximately 45% of the PPRO CPU.

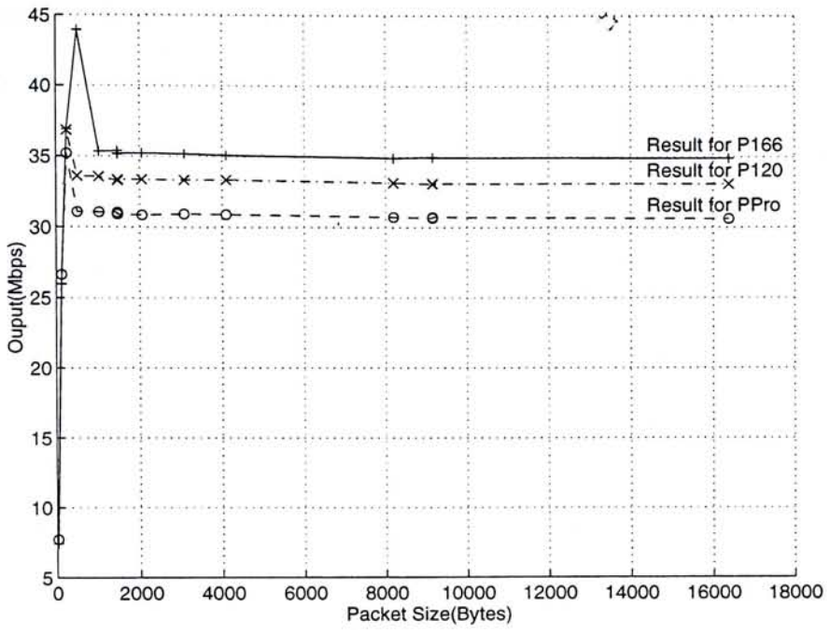


Figure 6.8: Results of the TCP Maximum Rate Test for the PPro, P166 and P120 Gateways.

The maximum achievable throughput for the PPRO, P160 and P120 is 35.196 Mb/s (256 bytes), 43.949 Mb/s (512 bytes) and 36.828 Mb/s (256 bytes) respectively. The data sizes that lead to the maximum throughput are shown in the brackets. The throughputs are then decrease to a steady value in all the cases and the respective values are approximately 31 Mb/s (PPro), 35 Mb/s (P166) and 33 Mb/s (P120).

The throughput drops that occurred in both UDP Control Rate Test and UDP Maximum Rate Test in the specific points (1472, 1473, 9152 and 9153 bytes in UDP) has not been observed in this test. It is because TCP is a stream (unstructured) oriented and buffered transfer protocol[11], it does not preserve the datagram boundaries[61] as in UDP. The TCP layer in the originating host will perform the processes of data fragmentation before data are transferring to the IP layer. Therefore, the IP packet fragmentation process will not occur in

the Gateway and thus no degradation in performance is observed.

The processing delay added by the Gateway may be the cause for the low throughput. In TCP, a rate control mechanism is implemented based on the feedback acknowledgment from the destination host, so a longer round trip delay means the originating host needs to wait longer before another transmission. Although TCP has used the sliding window mechanism to maximize the throughput, it still incurs some problem degradation on the system. Moreover, the congestion control mechanism provided by TCP has limited the rate because the devices (mainly Gateway in this case) along the transmission path cannot support more traffic. The impact of the TCP window size on the performance is not studied in this thesis although it is claimed to be a major factor that affects the network performance in TCP/IP over ATM network[61].

The processing overhead in TCP is more than in UDP, but it should only affect the sender and the receiver because the operation of the Gateway is independent of the network layer. The additional transmission delay in sending the larger TCP header should be small since it differs only by 12 bytes with respect to the UDP header.

The test loop is also divided into two in this test as in the previous test. Figure 6.9 has shown the test results for the PPro and P120 Gateways. It is found that the throughput of the ATM loop is about 69 Mb/s, and the throughput of the Ethernet loop is 51 Mb/s for the PPro Gateway and 47 Mb/s for the P120 Gateway. Curiously, the performance of the PPro Gateway is better than the P120 Gateway for the Ethernet loop. It can be concluded that the dominant factor affecting the results of these experiments is the processing speed of the CPU.

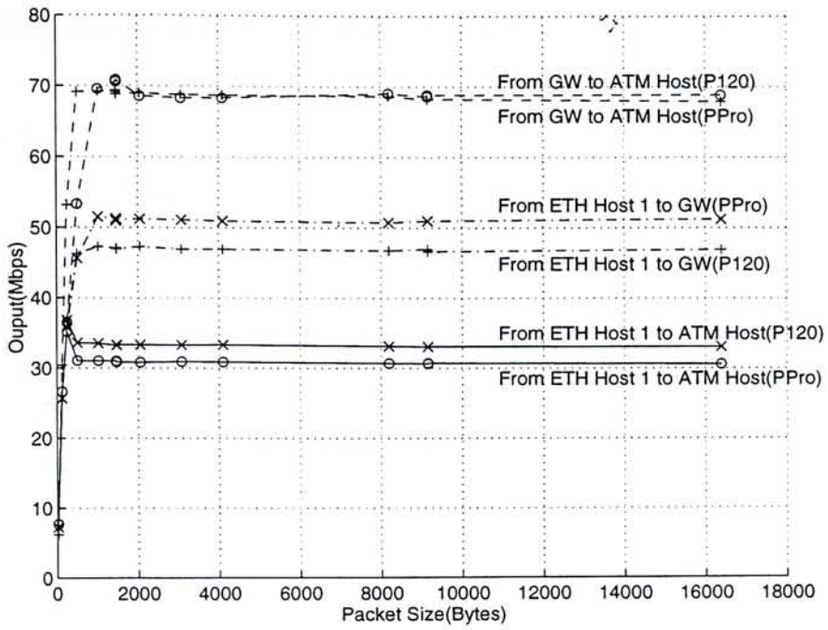


Figure 6.9: Results of the TCP Maximum Rate Test for the segmented loops of the PPro and P120 Gateways.

For the Ethernet loop, the Gateway acts as the receiver. By examining the log files of `vmstat`, it is found that the number of context switching per second for a receiver[17] is about two times more than a gateway when the data size is greater than 1024 bytes. Moreover, it is found that the numbers of interrupts generated by a receiver is about half of that by a gateway, but the CPU usage for system processes is about 10 times more. It thus signifies that the context switching processes is CPU time consuming and a better CPU should give better results. Therefore, when the PPro is acted as the receiver, a better result is achieved.

This test also confirms the assumption "Operations in Connectionless Gateway is I/O intensive rather than computation intensive" that we have made in section 6.5.1.

[17]It means the machine that receives frames and passes to the upper layers.

6.8 Request/Response Test

This experiment is designed to measure the delay imposed by the Connectionless Gateway. The sender in this test will transmit UDP datagrams of sizes 1, 64, 128, 256, 1024, 1472, 1473, 2048, 3072, 4096, 8192, 9152, 9153 and 16384 bytes to the receiver and wait for the 1-byte response from the receiver. The sender will try to send as much request as possible until the experiment ends. The result records the number of successful request and response in each second. The reciprocal of this result will be the round trip delay. The number of one-minute tests run for each data size in this test is 20.

6.8.1 Results and analysis of the Request/Response Test

(a) Round Trip Delay from ETH-Host-1 to ATM-Host

Considering only the forward trip delay from ETH-Host-1 to ATM-Host, the delay components include in a trip are as follows:

- Delay of data transfers from the user process to the network interface through the kernel in the sender;
- Packet transmission delay;
- Data Propagation delay;
- Additional delay added by the Connectionless Gateway;
- Delay of data transferred from the network interface back to the user process through the kernel in the receiver.

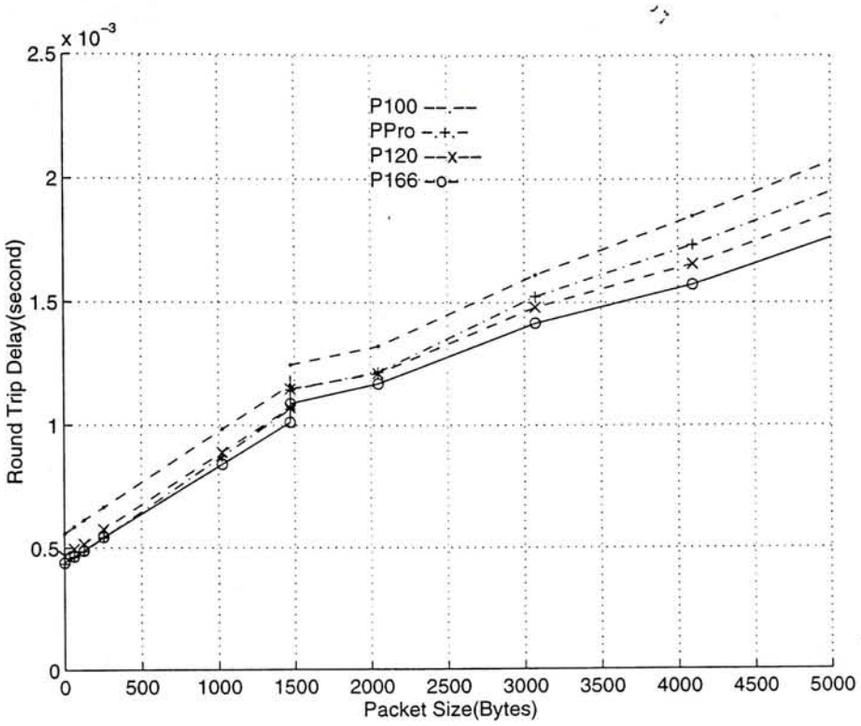


Figure 6.10: Calculated round trip delay from ETH-Host-1 to ATM-Host.

Among the five main components, propagation delay is constant in the experiments. It is the delay of the electrical signal transmitted in the UTP cable and optical signal transmitted in the optical fiber. The delay of both UTP cable and optical fiber is approximately 25.16 ns and 48.66 ns respectively[18] in the experiments. The other delay components are dependent on the size of the request-data. The first and the last components are the processing delays in the hosts including memory copying processes between the user and the kernel spaces, the kernel space and the network interface. The sources of delay are header processing time, checksum processing time and packet generation time.

[18]Assume the velocity of electrical signal in UTP cable is 2/3 of light velocity in free space, i.e., 1.987e8 m/s and the velocity of light in optical fiber can obtained by dividing the velocity of light in free space to the refractive index of fiber core(1.45), i.e., 2.055e8 m/s.

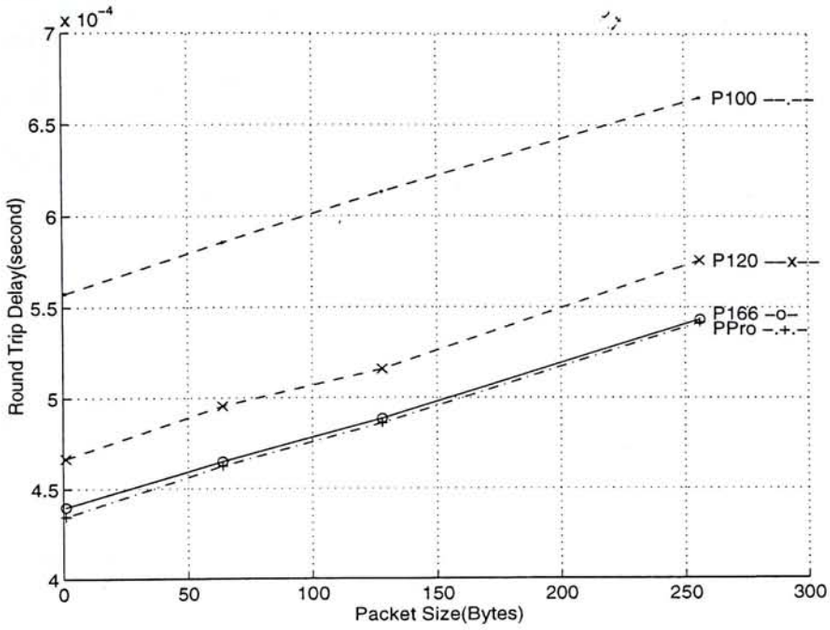


Figure 6.11: Calculated round trip delay from ETH-Host-1 to ATM-Host zooms at smaller data sizes.

Figures 6.10 and 6.11 illustrate the *round trip delay* from ETH-Host-1 to ATM-Host with Gateways of various configurations. They are calculated by taking the reciprocal of the request/response per second obtained from the Request/Response experiments. Figure 6.10 illustrates the dependency of the round trip delay on different request-data sizes in using different Gateway configurations. While Figure 6.11 is the magnified version of Figure 6.10 on small data sizes. Figure 6.10 clearly shows the Gateway performance under different configurations. The results further confirm our previous assumption that the CPU computation power is less important than the IO bandwidth because the performance ranking of the Gateways still hold.

Figure 6.10 has shown that when small sized packets (smaller than 1472 bytes) are transferred, the delay is larger because of the larger protocol overhead. It also clearly illustrates the sudden increase in delay at the critical data points

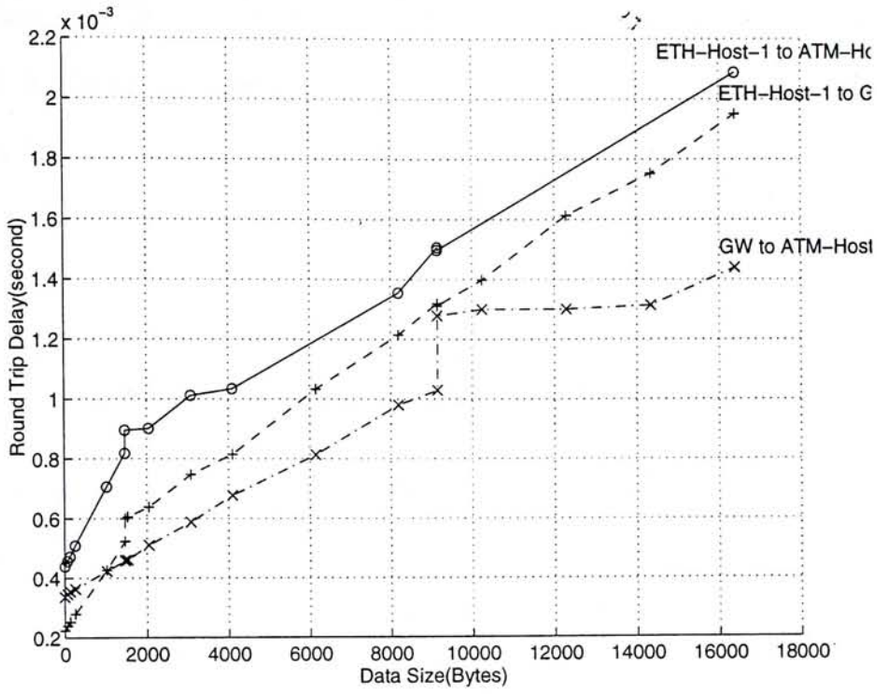


Figure 6.12: Comparison of the round trip delay between ETH-Host-1 and ATM-Host with the segmented loops when using the P166 Gateway.

(1472 and 1473 bytes). In order to verify that this additional delay is added by the Gateway, two other experiments are run (with P166 Gateway) on the segmented test loops. The results are shown in Figure 6.12. In order to shows the relative difference between the tests, the data used in drawing the curves have been subtracted from the packet transmission and propagation delay. It is found that the additional delay is actually added by the fragmentation processes in the sender rather than by the Gateway. By calculation, the increase in delay around the critical points on the curve "ETH-Host-1 to ATM-Host" and "ETH-Host-1 to Connectionless Gateway" is approximately 76.79 μs and 75.95 μs respectively. Their difference is only 0.84μs, thus we believe that the sender is the main source of the additional delay.

This test also illustrates that the efficiency of the ATM subsystem (ATM Card and ATM Driver) performs better in transmitting large packet (greater than 1024 bytes). Moreover, the major factor that affects the performance is the size of the MTU in the two systems (Ethernet and ATM). This is because the MTU of Classical IP over ATM is much larger than that of Ethernet, thus the frequency of packet fragmentation is decreased.

Figure 6.11 shows when the request data size is small, the performance of the PPro Gateway is the best. It means that when the request data size is small, the protocol overhead is greater than the I/O overhead, thus the PPro Gateway performs better because of its greater computation power.

(b) Round Trip Delay from ATM-Host to ETH-Host-1

The RR Test has also been done in the reverse direction(i.e., ATM-Host → ETH-Host-1) with Gateways of different configurations. Figure 6.13 compares the round trip delay for the two test directions.

Figure 6.13 depicts that the fragmentation overheads in the Gateway is greater in the transmission direction from ATM-Host to ETH-Host-1. Before reaching the critical size (1472 bytes of UDP datagram) of fragmentation, the total delay in the two directions is approximately the same but with a constant deviation of 20 μs. When the data size is increased to 1473 bytes, packets transmitted by ATM-Host are required to be fragmented by the Gateway (These packet fragmentation processes are done by the sender in the transmission direction from ETH-Host-1 to ATM-Host.). The time used in this fragmentation process is approximately 200 μs, which is 163.33 % larger than the process done

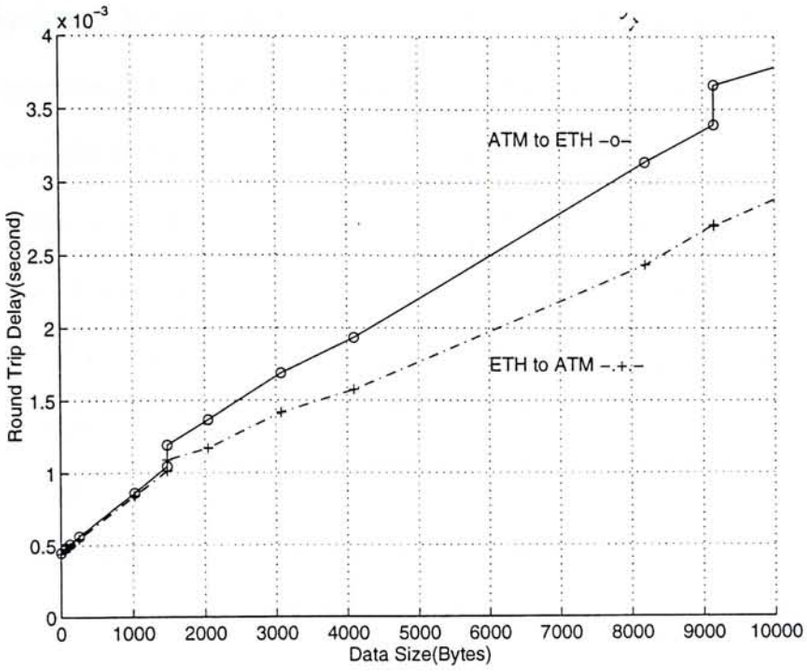


Figure 6.13: Comparison of round trip delay between the two test directions.

in the sender. When the data size further increases, the overhead for the fragmentation process also increases. When the packet size exceeds the MTU (9180 bytes) of the Classical IP over ATM network, an additional overhead is appended by the sender. For data sizes from 2048 to 9152 bytes, the fragmentation delay is increasing at a rate approximately equals to 0.07 μs/byte. However, this rate may change in those boundary cases such as the data point of 1473 bytes.

6.9 Priority Queue System Verification Test

In this experiment, both fast Ethernet machines will send prioritized packets (background traffic and test packets) "simultaneously"[19] to the ATM-Host. These prioritized packets will first arrive at the Connectionless Gateway. Then,

[19]ETH-Host-2 executes the Traffic Generator and ETH-Host-1 executes the Delay-test.

these packets will be served by the Connectionless Gateway in the order defined by their priorities. It means that the arrival times of the packets with different priorities are different although they are sent at the same time. Thus, the effectiveness of the priority queue system can be examined by measuring the delay of the test packets under different volume of background traffic.

In each experiment, ETH-Host-2 executes the Traffic-Generator and ETH-Host-1 executes the Delay-test simultaneously for a period of 1000 seconds. Traffic-Generator generates different volume of traffic with the three different priorities and the Delay-test also generates probing packets with the same three priorities at a rate of one packet/second. Thus the total number of test combinations per bandwidth of background traffic is 9. The selected volume of background traffic are 6, 12, 24, 36 and 48 Mb/s which are generated by 5, 10, 20, 30 and 40 UDP/IP datagrams of size 1500 bytes[20] per 10 ms respectively. The data size of the test packet is 56 bytes.

6.9.1 Results and analysis of the Priority Queue System Verification Test

One of the results obtained in this test has been shown in Figure 6.14.

The background traffic generated in this experiment is approximately 24 Mb/s (i.e., Sending of 20 packets per 10 ms with a data size of 1472 bytes) and the legends for each priority of probing packet is shown. On average, the round trip time for the Delay-test packets is approximately 500 μs.

Although the round trip delays of the high priority packets are expected to be the smallest, they still grow high for some time and even higher than the

[20]Data size is 1472 bytes.

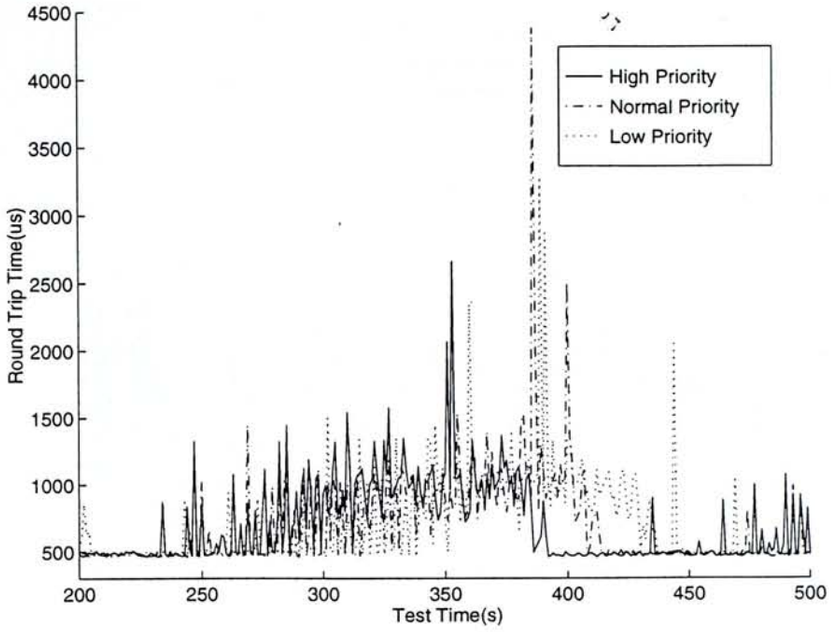


Figure 6.14: Round trip time measurement using Delay-test with 24 Mb/s normal priority background traffic.

lower priority packets. Nevertheless, in comparing the peak values (i.e., the longest round trip time), the round trip delay for the high priority packets is the smallest, followed by packets with low priority, and then packets with normal priority.

The reason for the occurrence of this special phenomenon is that there is only one receiver (i.e., ATM-Host) in the setup. The packets from the two traffic source, Traffic-Generator and Delay-test may content for the same resource from the receiver (e.g., packet processing) at the same time, thus the delay experienced in the receiver cannot be guaranteed especially when there is amble of background traffic. For example, if a high priority Delay-test packet arrives in the receiver which input queue has just been filled by the packets from the Traffic-Generator, then this high priority packet must wait until the queued packets are processed. This kind of queue waiting is the cause of the long round

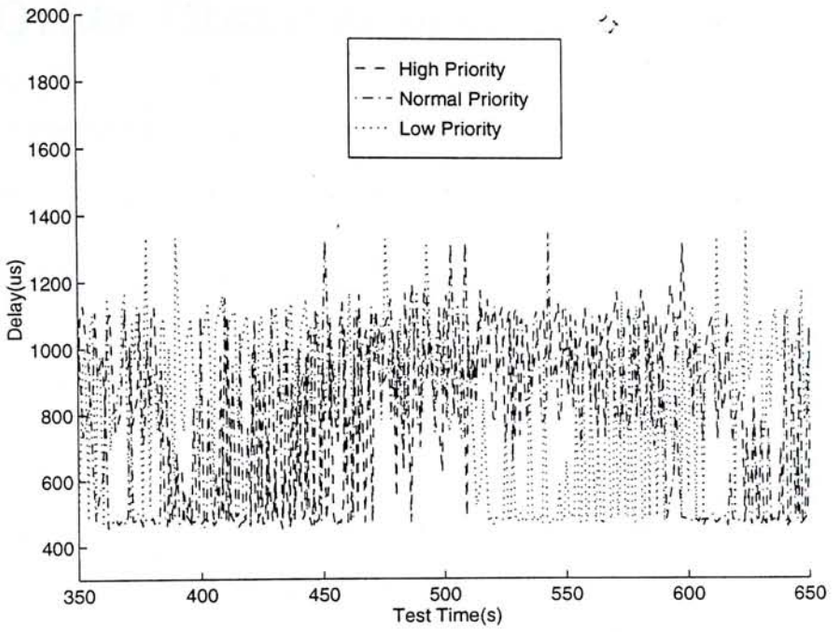


Figure 6.15: Round trip time measurement using Delay-test with 36 Mb/s normal priority background traffic.

trip time for the high/normal priority packets. Thus for an accurate test, each test clients should send packets to individual server. However, it requires an ATM switch and an extra ATM interface card to complete such setup. These testing equipment have been ordered at the time of writting and will be done in the future developments.

For comparison, Figure 6.15 shows the result obtained for normal priority background traffic with a rate of 36 Mb/s. The average round trip delay for the probing packets is still approximately 500 μs. However, at "busy time", the delay of the probing packets fluctuate around a high level because of the increase in background traffic. Simultaneously, the probability of resource contention at the Gateway and receiver is increased.

6.10 Other Observations

During the experiments, two more phenomenon related to the performance impact of the OS on the Gateway are observed. They are:

- "Infinite loop in interrupt" in the Ethernet device driver;
- The kernel is running out of buffer memory.

In order to explain the first observation, the operation principle of the Ethernet device driver should first be explained. Whenever an Ethernet device receives a frame, it will call the interrupt service routine of its device driver. The interrupt service routine will then stay in a loop to wait until the whole frame is received by the device. However, when the routine finds that the loop has been running for a long time, it will assume that a problem has occurred in the interface and will display the warning of "Infinite loop in interrupt" to inform the administrator. Then, the driver will reset the device and return back to the main routine before the interrupt. This warning message can be interpreted as "The device is running into a dead lock situation". This situation is caused by the excessive incoming of packets and the device is unable to handle all of them.

The second problem is related to the implementation of the Linux Kernel. It is because only a limited amount of host memory[21] is allocated as kernel buffer. Moreover, the kernel can only allocate buffer from contiguous memory locations within this kernel buffer segment. Therefore, when the kernel memory space is fragmented, the kernel may not be able to acquire enough memory to create the socket buffers for the protocol processing routines and the network device

[21] Approximately 1.5 MBytes.

drivers. Then, the only action the kernel can do is to drop the packets and to print out the warning messages.

6.11 Solutions to Improve the Performance

There are several problems encountered in our experiments that degrade the performance of the Gateway. They are listed as follows:

- Overflow of input backlog queue;
- Overflow of user space socket buffer;
- Timer expiry of SSCOP;
- Kernel memory shortage;
- Infinite loop in Ethernet device.

Actually, the problems listed above will only occur when the clients/servers do not behave well, e.g., the client in the UDP Maximum Rate test transmits the amount of traffic that exceeds the network capacity. Thus, the best way to prevent the occurrence of these problems is to control the maximum transmission rate of the users by the higher layer protocols such as TCP. However, previous researches had shown that the performance of TCP over ATM was poor because of the TCP/IP packets segmentation process at the ATM layer[62, 63]. Moreover, as indicated in the TCP Maximum Rate Test, the delay/timing relations between data[61] will also affect the performance. Therefore, a better transport protocol layer should be developed for the ATM/high speed networks.

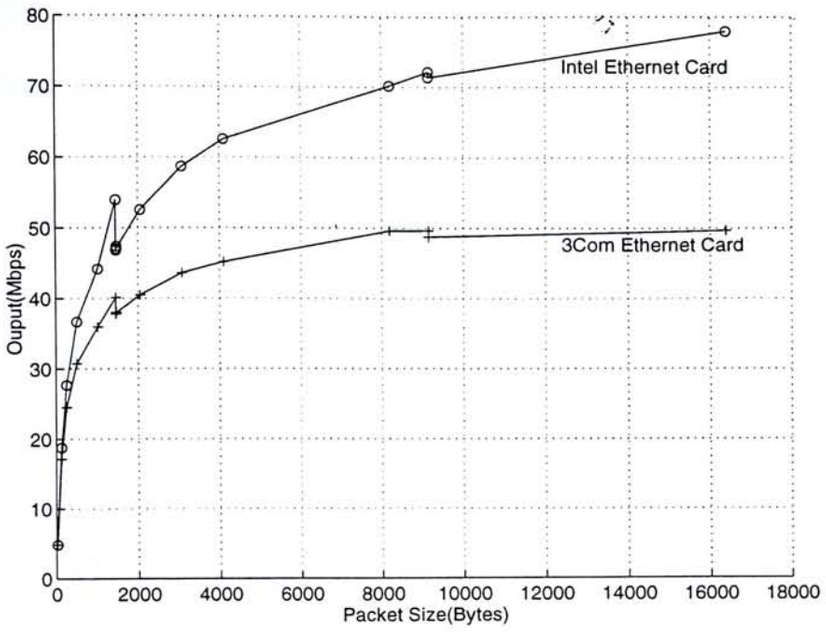


Figure 6.16: Compare the achievable TCP throughput of the Intel and 3Com Fast Ethernet Adapters.

Referring back to the throughput issue, it is found that different combinations of hardware devices could affect the throughput of the Gateway. Therefore, in order to improve the performance of the Gateway, the selection of hardware is crucial. It has been proved in our experimental results that a better CPU with a good PCI chip set should increase both the memory bandwidth and the computation power.

Second, the time spend in the interrupt routine to receive packets/frames is directly related to the network adapter, thus a better structured network adapter and driver could alleviate the burden brought by the interrupts on the operating system. Figure 6.16 has illustrated the result of an experiment that is intended to compare TCP throughput between two point-to-point machines which both use the Intel or 3Com 100-Mb/s Fast Ethernet Adapters. It is found

that the TCP throughput of the Intel Ethernet Adapter[22] is much higher than the 3Com Ethernet Adapter (56.91 % higher when the data size of the packets is 16384 bytes). Further, a white paper[64] published by Intel[23] also showed that the design of the adapter and driver should affect the system performance and efficiency.

Therefore, with better system components, the number of packets a gateway can process will increase (i.e., higher throughput). The probability of backlog queue overflow will be reduced and the chance of kernel buffer shortage will be lowered. Subsequently, the probability of the occurrence of the race situation will also be minimized.

Finally, a new concept called *single/zero-copy*[65, 45] has been introduced into the Linux community that could improve the system throughput. This technique eliminates the heavy burden of memory copy between the *kernel* and the *user space* by allocating a shared memory space for them to access. However, in the Gateway, memory copy operation between the kernel and user space is not required. It is because all the packet processing operations in the Gateway are done in the kernel only. Nevertheless, it should improve the performance of the end systems (receivers). Thus, as a whole, a better system performance could be achieved.

[22]The Intel Ethernet Card is a brand new product and device driver is also new and not fully operational. Thus in the previous experiments, the older, but stable 3Com 3c595-Tx Ethernet Adapters are used instead.

[23]This white paper was written by LANQuest Lab for Intel.

6.12 Future Development

In the current stage, the maximum achievable throughput of our Connectionless Gateway is approximately 52 Mb/s and this capacity will be sufficient for the departmental networks. However, for the bandwidth hungry multimedia applications, this capacity may not be enough. Nevertheless, the suggested solutions stated in the previous section could improve the Gateway performance and meet the future requirements.

Chapter 7

Conclusion

In this thesis, two important and interesting topics, namely the "Distributed Fault-Tolerant and Auto-Healing Algorithms" and "Connectionless Gateway for the ATM network" have been presented. Both of them have been implemented and valuable results have been obtained from various field tests.

The distributed fault-tolerant and auto-healing algorithms are designed to improve the reliability and the survivability of data communication networks. They have been implemented in the CUM LAUDE NET which is an experimental high-speed multimedia network supporting high speed multimedia services such as teleconferencing. The level-one hierarchy of the CUM LAUDE NET is a 100-Mbit/s high speed dual ring network of which the network nodes and network interfaces are developed by a team effort in the Lightwave Communications Laboratory, CUHK since 1993.

The operations of the distributed fault-tolerant and auto-healing algorithms are based on the inter-communications and hand-shaking processes between the network nodes. In the fault-tolerant algorithm, two individual fault-detection

schemes namely the *Out-of-synchronization Detection* and *Packet Non-arrival Detection* are designed for different types of ring networks. Each node in the network executes its own detection routines and thus no central controller is required. These schemes are implemented in the CUM LAUDE NET and they have been proven to successfully detect the types of failure that stated in Section 2.6.2. The design of these algorithms are generic in nature and can be ported to other types of networks. Since the CUM LAUDE NET has a dual ring network architecture, thus the fault can be simply bypassed by wrapping the links around the failure. The algorithm is efficient and the time to recover from a fault is just a few milli-seconds.

The Auto-Healing algorithm is designed to provide the feature of "Hot replacement" of faulty components. The idea is different from most of the previously published work. It has been successfully implemented in the CUM LAUDE NET and the auto-healing time is also only a few milli-seconds. Therefore, the disturbance of any network failure to the users is negligible with the provision of these algorithms.

ATM network is currently a hot topic in both academia and industry. Therefore, we foresee that ATM networks will soon become the dominant technology used to build the public/campus backbone networks. However, ATM works in connection-oriented mode which is different from the common connectionless networks such as Ethernet, thus a Connectionless Gateway is proposed to connect the ATM networks with the connectionless networks such as CUM LAUDE NET, Ethernet and FDDI.

The design concept of ATM is similar to the traditional telephone network that connection establishment is required before any communication can take

place and quality of service (QoS) is provided. Moreover, it also works similarly to the packet type computer networks which transfer data in packet like units called "cell", but its packet length is only 53 bytes for efficient processing in the switching nodes. However, routing is done according to the VPI/VCI values specified in each cell header of which is different from the IP routing networks. In view of the importance of ATM, a dedicated chapter (Chapter 3) is written to review the general ATM issues.

Another chapter (Chapter 4) is written to discuss the design issues on Connectionless Gateway which can be summarized as follows:

1. ATM Internetworking;
2. Connections Management;
3. Protocol Conversion;
4. Bandwidth Assignment[3];
5. Packet Forwarding Modes;

Different approaches of solutions are described for each issue and comparisons are made to select the best solution in the implementation of the Connectionless Gateway.

Finally, the design and implementation of the Connectionless Gateway is described in Chapter 5. The Connectionless Gateway is implemented using the inexpensive PCs plugged with the ATM, CUM LAUDE NET and Fast Ethernet network interface cards (NICs). The operating system run in the Gateway is a free UNIX clone called Linux which is developed by many volunteers around the

world. The core of the Gateway is the "Gateway Engine" which is responsible for filtering, converting and routing packets between different types of networks.

The designed network infrastructure is an ATM backbone with Connectionless Gateways connected as edge devices. The connectionless LANs work in their usual way and all traffic outside their own domain is sent to the Connectionless Gateway for further processing. Since the gateway is presumed to be connected to the Internet, thus it is currently designed to route IP packets only. The proposal of Classical IP over ATM is used to connect the Connectionless Gateways through the ATM backbone. The connection type used in the implementation is SVC with the ATMARP server. The ATMARP server is also configured as a Routing (Routing Table) Server which is responsible for propagating the Connectionless Gateways' routing table to each others.

Further, an additional "Priority Output Queues System " is designed and implemented in the Gateway to serve the time delay sensitive data. Three priorities have been defined for the IP packets. The system queues the packets in the order according to their priorities. Two queueing disciplines have been proposed for this Priority Output Queues System. Besides, a user friendly "Gateway Performance Monitor" operates under the X-window system is designed for real-time measurement and monitoring of the amount of traffic passing through the Gateway. The application also provides other functions to monitor the status of the routing tables and ATM connections.

Finally, a number of experiments are designed to evaluate the performance of the gateway. Different types of self developed tools are written to facilitate the tests. Satisfactory results are obtained and it is found that the throughput of the Connectionless Gateway can reach 52-Mbit/s approximately. From the

observations recorded in the experiments, the causes of bottleneck are found. The impact of packet fragmentation on the gateway has also been extensively studied and it was revealed that the process of fragmentation only increases the delay (0.07 μs/byte) but does not cause other serious problems. Lastly, it is concluded that too much interrupts will cause deadlock problems because of theirs high service priority in the operating system. Interestingly of all, it is found that a faster CPU does not always mean a higher throughput, instead, a machine with higher memory bandwidth performs better.

After detailed evaluation, solutions are suggested to improve the Gateway's performance. The results implies a better NIC coupled with a fast CPU and good CPU interface chipset can properly increase the throughput of the Gateway. Further improvement can also be made to the OS such as the implementation of "Zero/Single Copy" mechanism to minimize the time spent in the slow memory copy operations between the kernel and the user spaces.

Currently, there is no QoS supported in the ATM network for the IP traffic until the deployment of the IPv6 (which is under construction by a group of volunteers in the Internet). After the launching of IPv6, the Connectionless Gateway may required to be re-designed to support this protocol and other protocols specially designed for it.

Bibliography

[1] K. W. Cheung, "Adaptive-Cycle Tunable-Access(ACTA) Protocols: A Simple, High-Performance Protocol for Tunable-Channel Multi-Access(TCMA) Networks," *ICC '93*, pp. 166 – 171, May 1993.

[2] K.W.Cheung, L.K.Chen, C.Su, et al., "Tunable-Channel Multi-Access(TCMA) Networks: A New Class of High-Speed Networks Suitable for Multimedia Integrated Networking," *SPIE' 93 - Multigigabit Fiber Communication Systems, San Diego, CA*, July 1993.

[3] S. Varada and R. Sankar, "Bandwidth Allocation for Connectionless Traffic in ATM Networks," *Proceedings IEEE Southeastcon '95.*, pp. 128–132, 1995.

[4] H.-G. Hegering and A. Lapple, *Ethernet - Building a Communications Infrastructure.* Addison-Wesley Publishing Company, Inc., 1993.

[5] W. Stallings, *Local & Metropolitan Area Networks.* Prentice-Hall, Inc., 5 ed., 1997.

[6] K.W.Ko, S.F.Lam, K.W.Cheung, et al., "Fast Distributed Fault-Tolerant and Auto-Healing Algorithms on Dual-Ring Networks," *SICON '97*, 1997.

[7] K. W. Cheung, C. T. Yeung, W. K. Lam, et al., "CUM LAUDE NET - A High-Speed Multimedia Integrated Network Prototypes," *1st ISMM International Conference Multimedia and Distributed Applications, Hawaii*, 1994.

[8] Y. C. Toa, "Design and implementation of high speed multimedia network," Master's thesis, The Chinese University of Hong Kong, June 1994.

[9] R. W.-K. Lam, "An Integrated Broadband Concentration/Distribution Network for Multimedia Application compatible with the Hybrid Fiber-Coax(HFC) Architecture," m.phil thesis, The Chinese University of Hong Kong, June 1995.

[10] I. Advanced Micro Devices, *Am7968/Am7969 TAXI TM - 125 Technical Manual.* 1993.

[11] D. E. Comer, *Internetworking with TCP/IP, Volume I: Principles, Protocols, and Architecture.* Prentice Hall, Inc., 3 ed., 1995.

[12] M. de Prycker, *Asynchronous Transfer Mode Solution for Broadband ISDN.* Ellis Horwood Ltd., 2 ed., 1993.

[13] K.-Y. Siu and R. Jain, "A Brief Overview of ATM: Protocol Layers, LAN Emulation, and Traffic Management," *ACM SIGCOMM, Computer Communication Review*, vol. 25, pp. 6 – 20, Apr. 1995.

[14] T. W. W. Fischer, E. Wallmerier and et al., "Data Communications Using ATM: Architectures, Protocols, and Resource Management," *IEEE Communication Magazine*, pp. 24 – 33, Aug. 1994.

[15] T. A. Forum, "Atm user-network interface specification version 3.1," tech. rep., Sept. 1994.

[16] K. Marko, "Implementation of LAN Emulation Over ATM in Linux," master's thesis, TAMPERE UNIVERSITY OF TECHNOLOGY, Oct. 1996.

[17] G. Armitage, *The Application of Asynchronous Transfer Mode to Multimedia and Local Area Networks.* PhD thesis, The University of Melbourne, June 1994.

[18] T. A. Forum, *ATM User-Network Interface Specification Version 3.0.* PTR Prentice Hall, Sept. 1993.

[19] M. N. H. Rainer Handel and S. Schroder, *ATM Networks Concepts, Protocols, Application.* Addison-Wesley Publishing Company, Inc., 2 ed., 1994.

[20] M. Schwartz, *Broadband Integrated Networks.* Prentice Hall PTR, 1996.

[21] Juha, "RFC 1483 - Multiprotocol Encapsulation over ATM Adaptation Layer 5," tech. rep., July 1993.

[22] A. Alles, "ATM Internetworking," May 1995.

[23] T. Y. C. T. M. Gerla and G. Gallassi, "Internetting LAN's and MAN's to B-ISDN's for Connectionless Traffic Support," *IEEE Journal on Selected Areas in Communications*, vol. 11, pp. 1145 – 1159, Oct. 1993.

[24] T. A. F. T. Committee, "LAN Emulation Over ATM Version 1.0," tech. rep., Jan. 1995.

[25] L. Staalhagen, "A Comparsion Between the OSI Reference Model and the B-ISDN Protocol Reference Model," *IEEE Network*, pp. 24 – 33, Jan. 1996.

[26] D. S. H. Jonathan Chao, Dipak Ghosal and S. K. Tripathi, "IP on ATM Local Area Networks," *IEEE Communications Magazine*, pp. 52 – 59, Aug. 1994.

[27] N. Finn and T. Mason, "ATM LAN Emulation," *IEEE Communications*, vol. 34, pp. 96–100, June 1996.

[28] N. Kavak, "Data Communication in ATM Networks," *IEEE Network*, pp. 28 – 37, 1995.

[29] H. L. Truong, William W. Ellington Jr., et al., "LAN Emulation on an ATM Network," *IEEE Communications Magazine*, pp. 70 – 85, May 1995.

[30] R. Atkinson, "RFC 1626 - Default IP MTU for use over ATM AAL5," tech. rep., Navel Research Laboratory, May 1994.

[31] M. Laubach, "RFC 1577 - Classical IP and ARP over ATM," tech. rep., Jan. 1994.

[32] M. Perez, F. Liaw, A. Mankin, D. Grossman, et al., "RFC 1755 - ATM Signaling Support for IP over ATM," tech. rep., Feb. 1995.

[33] G. J. Armitage, "Multicast and Multiprotocol support for ATM Based Internets," *ACM SIGCOMM, Computer Communication Review*, vol. 25, pp. 34 – 46, Apr. 1995.

[34] D. E. McDysan and D. L. Spohn, *ATM Theory and Application*. McGraw-Hill, New York, 1995.

[35] M. Gerla and T. Y. Tai, "Interconnecting LANs and MANs to ATM," *IEEE Proceedings 16th Conference on Local Computer Networks*, pp. 259 – 270, 1991.

[36] V. C. S. L. Edward Chan and J. M. Ng, "On the Performance of Bandwidth Allocation Strategies for Interconnectiong ATM and Connectionless Networks," *ACM SIGCOMM*, vol. 26, pp. 29 – 38, Jan. 1996.

[37] G. G. P. Crocetti and M. Gerla, "Bandwidth Advertising for MAN/ATM Connectionless Internetting," *IEEE INFOCOM '91*, vol. 3, pp. 1145 – 1150, 1991.

[38] B. J. Vickers and T. Suda, "Connectionless Service for Public ATM Networks," *IEEE Communication Magazine*, pp. 34 – 42, Aug. 1994.

[39] S. ichi Iisaku and M. Ishikura, "ATM network architecture for supporting the connectionless service," *INFOCOMM '90 Proceeding*, vol. 2, pp. 796 – 802, 1990.

[40] A. S. Tanenbaum, *Computer Networks.* Prentice Hall International, Inc., second ed., 1989.

[41] A. S. Acampora, *An Introduction to Broadband Networks LANs, MANs, ATM, B-ISDN, and Optical Networks for Integrated Multimedia Telecommunications.* Plenum Press, New York, 1994.

[42] M. F. L. Mongiovi and V. Trecordi, "A Proposal for Interconnecting FDDI Networks through B-ISDN," *IEEE INFOCOM '91*, vol. 3, pp. 1160 – 1167, 1991.

[43] A. A. E. H. S. Hassanein and P. H. Roe, "Congestion Avoidance in Interconnected Local Area Networks," *IEEE*, pp. 262 – 268, 1993.

[44] W. Almesberger, "ATM on Linux." Mar. 1996.

[45] W. Almesberger, "High-speed ATM networking on low-end computer systems." Aug. 1995.

[46] *ATM Adapter Installation Guide EISA/PCI Windows NT 3.51 Orca 2.0*, 1996.

[47] W. R. Stevens, *UNIX Network Programming.* Prentice Hall, Inc, 1991.

[48] W. Almesberger, *Linux ATM device driver interface.* EPFL, LRC, 0.1 ed., Feb. 1996.

[49] W. Almesberger, *Linux ATM API.* EPFL, LRC, 0.4 ed., July 1996.

[50] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols.* Addison-Wesley Publishing Company, Inc., 1994.

[51] J. A. Manan and D. A. Harle, "Performance of An FDDI-ATM Gateway Supporting Multi-class Traffic," *Singapore ICCS' 94*, pp. 500 – 504, 1994.

[52] E. F. Johnson, *Graphical Applications with Tcl & Tk.* M&T Books, 1996.

[53] J. K. Ousterhout, *Tcl and the Tk Toolkit.* Addison-Wesley Publishing Company, Inc., 1994.

[54] B. B. Welch, *Practical Programming in Tcl and Tk.* Prentice Hall PTR, 1995.

[55] S. B. Sven Goldt, Sven van der Meer and M. Welsh, *The Linux Programmer's Guide*, 0.4 ed., Mar. 1995.

[56] D. R. Hipp, *Embedded Tk*, 1996.

[57] W. Almesberger, *ATM on Linux User's Guide Release 0.18*, 1996.

[58] I. N. Division, *Netperf: A Network Performance Benchmark Revision 2.1.* Hewlett-Packard Company, Feb. 1996.

[59] M. J. K. Sammuel J. Leffler, Marshall Kirk McKusick and J. S. Quarterman, *The Design and Implementation of the 4.3BSD UNIX Operating System.* Addison-Wesley Publishing Company, 1989.

[60] J. Kay and J. Pasquale, "Profiling and Reducing Processing Overheads in TCP/IP," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 817–828, Dec. 1996.

[61] E. K. K. Moldeklev and O. Kure, "TCP/IP Behavior in a High-Speed Local ATM Network Environment," *Proceedings 19th Conference on Local Computer Networks*, pp. 176 – 185, 1994.

[62] A. Romanow and S. Floyd, "Dynamics of TCP Traffic over ATM Networks,"

[63] G. J. Armitage and K. M. Adams, "How Inefficient is IP over ATM Anyway ?," *IEEE Network*, pp. 18 – 26, Jan. 1995.

[64] "LANQuest Study: 100Mbps PCI Fast Ethernet Adapter Performance," tech. rep., LANQuest Labs, http://www.intel.com, 1997.

[65] H. S. Hiroshi Kitamura, Kunihiro Taniguchi and T. Nishida, "A New OS Architecture for High Performance Communication over ATM Networks.".

Appendix A

List of Publications

1. K.W. Ko, S.F. Lam, K.W.Cheung et al. Distributed Fault-Tolerant and Auto-Healing Algorithms on Dual-Ring Networks. *IEEE SICON'97. April, 1997.*

2. Kin-Wa Ko, Sze-Fan Lam and Kwok-Wai Cheung. A Revertive Fault-Tolerant Scheme for Dual-Ring Networks. *IEEE ISCC'97. July, 1997.*