An Introduction to

ATM Networks

by

Harry Perros

An Introduction to ATM Networks

Harry Perros

To

Helen, Nick, and Mikey

Foreword

ATM networks was the subject of intense research and development from the late 1980s to the late 1990s. Currently, ATM is a mature networking technology and it is taught regularly in Universities and in short professional courses. This book was written with a view to be used as a text book in a second course on computer networks at the graduate level or senior undergraduate level. Also, it was written for networking engineers out in the field who would like to learn more about ATM networks. A pre-requisite for this book is basic knowledge of computer networking principles.

The book is organized into the following four parts:

Part One: Introduction and Background
Part Two: The ATM Architecture
Part Three: Deployment of ATM
Part Four: Signalling in ATM Networks.

Part One "*Introduction and Background"* contains a variety of topics which are part of the background necessary for understanding the material in this book. It consists of Chapters 1, 2, and 3. Chapter 1 contains a discussion of what caused the development of ATM networks, and a brief description of the various standards committees that feature prominently in the development of ATM networks. Chapter 2, gives a review of basic concepts of computer networks that are used in this book. This Chapter can be skipped by the knowledgeable reader. Chapter 3 is dedicated to frame relay, where we describe the motivation behind the development of frame relay and its basic features, the frame relay UNI, and congestion control. It is educationally constructive to understand how frame relay works since it is a very popular networking solution and it has many common features with ATM networks, such as, layer two switching, no error or flow control between two adjacent nodes, and similar congestion control schemes.

Part Two "*The ATM Architecture"* focuses on the main components of the ATM architecture. It consists of Chapters 4, 5, 6, and 7. In Chapter 4, the main features of the ATM architecture are presented. An ATM packet, known as *cell*, has a fixed size and it is equal to 53 bytes. We start with a brief account of the considerations that led to the

decision to use such a small packet. Then, we describe the structure of the header of the ATM cell, the ATM protocol stack, and the various ATM interfaces. We conclude this Chapter with a description of the physical layer that supports ATM networks and the various public and private interfaces. In Chapter 5, we describe the ATM adaptation layer. The purpose of this layer is to isolate higher protocol layers and applications from the specific characteristics of ATM. Four different ATM adaptation layers are described, namely ATM adaptation layers 1, 2, 3/4, and 5. Chapter 6 is dedicated to ATM switch architectures, and the following three different classes of ATM switch architectures are presented: space-division switches, shared memory switches, and shared medium switches. We describe various architectures that have been proposed within each of these three classes. Also, to give the reader a feel of a real-life switch, the architecture of a commercial switch is described. We conclude this Chapter by describing various algorithms for scheduling the transmission of cells out of an output port of an ATM switch. Finally, Chapter 7 deals with the interesting problem of congestion control in ATM networks. We first present the various parameters used to characterize ATM traffic, the various quality of service (QoS) parameters, and the standardized ATM classes. In the rest of the Chapter, we focus on the two classes of congestion control schemes, namely, the preventive and reactive congestion control. We introduce the preventive congestion control scheme, and we present various call admission control algorithms, the GCRA bandwidth enforcement algorithm, and cell discard policies. Finally, we present the available bit rate (ABR) scheme, a reactive congestion control scheme standardized by the ATM Forum.

Part Three *"Deployment of ATM"*, deals with the two different topics, namely, how IP traffic is transported over ATM, and ADSL-based access networks. It consists of Chapters 8 and 9. In Chapter 8 we describe various schemes used to transport IP traffic over ATM. We first present ATM Forum's LAN emulation (LE), a solution that enables existing LAN applications to run over an ATM network. Then, we describe IETF's schemes classical IP and ARP over ATM and next hop routing protocol (NHRP) designed for carrying IP packets over ATM. The remaining of the Chapter is dedicated to the three techniques IP switching, tag switching, and multi-protocol label switching (MPLS). IP switching inspired the development of tag switching, which at this moment is

being standardized by IETF under the name of multi-protocol label switching. Chapter 9 is dedicated to the asynchronous digital subscriber line (ADSL) technology which can be used in residential access networks to provide basic telephone services and access to the Internet. We describe the discrete multi-tone (DMT) technique used to transmit the information over the telephone twisted pair, the seven bearer channels, the fast and interleaved paths, and the ADSL super frame. Finally, we discuss architectures for accessing network service providers.

Part Four *Signalling in ATM Networks* focuses on the signalling protocols used to set-up a switched virtual connection (SVC). It consists of Chapters 10 and 11. In Chapter 10, we review the signalling protocols used to establish a point-to-point connection and a point-to-multipoint connection over the private UNI. The signalling protocol for establishing a point-to-point connection is described in ITU-T's Q.2931 standard, and the signalling protocol for establishing a point-to-multipoint connection is described in ITU-T's Q.2971 standard. We first describe a specialized ATM adaptation layer, known as the signalling AAL (SAAL), that is used by both protocols. Then, we discuss in detail the signalling messages and procedures used by Q.2931 and Q.2971. In Chapter 11, we examine the private network-network interface (PNNI) used to route a new call from an originating UNI to a destination UNI. PNNI consists of the PNNI routing protocol and the PNNI signalling protocol. We first describe the PNNI routing protocol in detail and then we briefly discuss the PNNI signalling protocol.

At the end of each Chapter there are problems given. Also, in some Chapters 6 and 7, there are three simulation projects designed to help the reader understand better some of the intricacies of ATM networks.

To develop a deeper understanding of ATM networks, one has to dig into the various documents produced by the standards bodies. Most of these documents are actually very readable! A list of standards which are relevant to the material in this book can be found at the end of the book.

Finally, in ATM networks there is an abundance of abbreviations, and the reader is strongly encouraged to learn some of them.. When in doubt, the glossary of abbreviations given at the end of the book may be of help!

Harry Perros

Cary, February 13th, 2001

Contents

PART TWO: THE ATM ARCHITECTURE

PART ONE:
INTRODUCTION AND BACKGROUND

In Part One, we present several topics which are part of the background necessary for understanding ATM networks. It consists of Chapters 1, 2 and 3. Some of the material presented in these Chapters can be skipped by the knowledgeable reader.

Chapter 1: Introduction

In this Chapter we identify the various forces that gave rise to ATM networks, and describe some of the well-known standards bodies.

Chapter 2: Basic Concepts From Computer Networks

This Chapter gives a review of some basic concepts from computer networks that we will make use in this book.

Chapter 3: Frame Relay

In this Chapter, we present frame relay, a very popular networking technique for transporting data over a wide area network, which has many common features with ATM networks.

CHAPTER 1

Introduction

In this Chapter, we introduce the *Asynchronous Transfer Mode* (ATM) networking technique, and discuss the forces that gave rise to it. Then, we describe some of the well-known national and international standards committees involved with the standardization process of networking equipment.

1.1 The Asynchronous Transfer Mode (ATM)

ATM is a technology that provides a single platform for the transmission of voice, video, and data at specified quality of service and at speeds varying from fractional T1, i.e., nX64 Kbps, to Gbps. Voice, data and video are currently transported by different networks. Voice is transported by the public telephone network, and data by a variety of packet-switched networks. Video is transported by networks based on coaxial cables, satellites, and radio waves, and to a limited extent, by packet-switched networks.

In order to understand what caused the development of ATM, we have to go back to the 80's! During that decade, we witnessed the development of the workstation and the evolution of the optical fiber. A dramatic reduction in the cost of processing power and associated peripherals, such as main memory and disk drives, lead to the development of powerful workstations capable of running large software. This was a significant improvement over the older "dumb terminal". These workstations were relatively cheap to buy, easy to install and interconnect, and they enabled the development of distributed systems. As distributed systems became more commonplace, so did the desire to move files over the network at a higher rate. Also, there was a growing demand for other applications, such as, video conferencing, multi-media, medical imaging, remote

processing and remote printing of a newspaper. At the same time, optical fiber technology evolved very rapidly, and by the end of the 80s there was a lot of optical fiber installed. Optical fiber permitted high bandwidth and very low bit-error rate.

These technological developments coupled with the market needs for faster interconnectivity, gave rise to various high-speed wide-area networks and services, such as *frame relay, Asynchronous Transfer Mode* (ATM), and *Switched Multimegabit Data Services* (SMDS).

ATM was standardized by ITU-T in 1987. It is based on packet-switching and it is connection oriented. An ATM packet, known as a *cell*, is a small fixed-size packet with a payload of 48 bytes and a 5-byte header. The reason for using small packets was motivated mostly by arguments related to the transfer of voice over ATM.

Unlike IP networks, ATM has built-in mechanisms that permits it to provide different quality of service to different types of traffic. ATM was originally defined to run over high-speed links. For instance, in North America, the lowest envisioned speed was OC-3, which corresponds to about 155 Mbps. It should be noted that the fastest network in the late 80s was the FDDI which ran at 100 Mbps. However, as ATM became more widely accepted, it was also defined over slow links, such as fractional T1, that is, nX64 Kbps.

In the early 90s, ATM was poised to replace well-established local and wide area networks, such as Ethernet and IP networks. ATM was seen as a potential replacement for Ethernet because it ran faster and it also provided quality of service. We note that at that time Ethernet ran at 10 Mbps, but due to software bottlenecks its effective throughput was around 2 Mbps. Also, since ATM has its own addressing system and it can set-up and route connections through the network, it was seen as a potential foe of IP networks. In view of this, Ethernet and IP networks were declared by the ATM aficionados as “dead”!

Interestingly enough, Ethernet made a dramatic come-back when it was defined to run at 100 Mbps and later on at 1 Gbps. As a result, ATM lost the battle to the “desk-top”. That is, it never became the preferred networking solution for interconnecting workstations and personal computers at a customer’s premises. Also, in the mid-90s, we witnessed a new wave of high-speed IP routers and a strong effort to introduce quality of

service in IP networks. As a result, one frequently hears cries that it is the ATM technology that is now "dead"!

ATM is a mature networking technology and it is still the only networking technology that provides quality of service. ATM networks are used in a variety of environments. For instance, it is widely used in the backbone of *Internet service providers* (ISP) and in campus networks to carry Internet traffic. ATM wide area networks have also been deployed to provide point-to-point and point-to-multipoint video connections. Also, there are on going projects in telecommunication companies aiming at replacing the existing trunks used in the telephone network with an ATM network.

On a smaller scale, ATM is used to provide *circuit emulation,* a service that emulates a point-to-point T1/E1 circuit and a point-to-point fractional T1/E1 circuit over an ATM network. ATM is the preferred solution for ADSL-based residential access networks used to provide access to the Internet and basic telephone services over the phone line. Also, it is used in *passive optical networks* (PON) deployed in residential access networks.

We conclude this section by noting that arguments in favour and against existing and emerging new networking technologies will most likely continue for a long time. There is no argument, however, that these are indeed very exciting times as far as communication systems are concerned!

1.2 Standards committees

Standards allow vendors to develop equipment to a common set of specifications. Providers and end-users can also influence the standards so that the vendors' equipment conform to certain characteristics. As a result of the standardization process, one can purchase equipment from different vendors without being bound to the offerings of a single vendor.

There are two types of standards, namely *de facto* and *de jure*. De facto standards are those which were first developed by a single vendor or a consortium, and then they were accepted by the standards bodies. De jure standards are those generated through consensus within national or international standards bodies. ATM, for instance, is the result of the latter type of standardization.

Several national and international standards bodies are involved with the standardization process in telecommunication, such as the *International Telecommunication Union* (ITU), the *International Organization for Standardization* (ISO), the *American National Standards Institute* (ANSI), the *Institute of Electrical and Electronics Engineering (IEEE)*, the *Internet Engineering Task Force* (IETF), the *ATM Forum*, and the *Frame Relay Forum*. The organizational structure of these standards bodies is described below.

The ITU-T and the ATM Forum are primarily responsible for the development of standards for ATM networks. ITU-T concentrates mainly on the development of standards for public ATM networks, whereas the ATM Forum concentrates on private networks. The ATM Forum was created because many vendors felt that the ITU-T standardization process was not moving fast enough, and also because there was an emerging need for standards for private ATM networks. In general, ITU-T tends to reflect the view of network operators and national administrations, whereas the ATM Forum tends to represent the users and the customer premises equipment (CPE) manufacturers. The two bodies compliment each other and work together to align their standards with each other.

The International Telecommunication Union (ITU)

ITU is a United Nations specialized agency whose job is to standardize international telecommunications. ITU consists of the following three main sections: the *ITU Radiocommunications Sector* (ITU-R), the *ITU Telecommunications Standardization Sector* (ITU-T), and the *ITU Development Sector* (ITU-D).

The ITU-T's objective is the telecommunications standardization on a worldwide basis. This is achieved by studying technical, operating and traffic questions, and adopting recommendations on them. ITU-T was created in March 1993, and it replaced the former well-known standards committee *International Telegraph and Telephone Consultative Committee*, whose origins are over 100 years old. This committee was commonly referred to as CCITT, which are the initials of its name in French.

ITU-T is formed by representatives from standards organizations, service providers, and more recently by representatives from vendors and end users.

Contributions to standards are generated by companies, and they are first submitted to national technical coordination groups, resulting to national standards. These national coordinating bodies may also pass on contributions to regional organizations or directly to ITU-T, resulting in regional or world standards. ITU more recently started recommending and referencing standards adopted by the other groups, instead of re-writing them.

ITU-T is organized in 15 technical study groups. At present, more than 2500 recommendations (standards) or some 55,000 pages are in force. They are non-binding standards agreed by consensus in the technical study groups. Although, non-binding, they are generally complied with due to their high quality and also because they guarantee the inter-connectivity of networks, and enable telecommunications services to be provided on a worldwide scale.

ITU-T standards are published as *recommendations*, and they are organized into series. Each series of recommendations is referred to by a letter of the alphabet. Some of the well-known recommendations are the I, Q, and X. Recommendations I are related to integrated services digital networks. For instance, I.321 describes the B-ISDN protocol reference architecture, I.370 deals with congestion management in frame relay, and I.371 deals with congestion management in ATM networks. Recommendations Q are related to switching and signalling. For instance, Q.2931 describes the signalling procedures used to establish a point-to-point ATM switched virtual connection over the private UNI, and Q.2971 describes the signalling procedures used to establish a point-to-multipoint ATM switched virtual connection over the private UNI. Recommendations X are related to data networks and open system communication. For instance, X.700 describes the management framework for the OSI basic reference model, and X.25 deals with the interface between a DTE and a DCE terminal operating in a packet mode and connected to a public data networks by dedicated circuit.

The International Organization for Standardization (ISO)

ISO is a worldwide federation of national standards bodies from some 130 countries, one from each country. It is a non-governmental organization established in 1947. Its mission is to promote the development of standardization and related activities in the world with a

view to facilitating the international exchange of goods and services, and to developing cooperation in the spheres of intellectual, scientific, technological and economic activity.

It is interesting to note, that the name ISO does not stand for the initials of the full title of this organization, which would have been IOS! In fact, ISO is a word derived from the Greek *isos*, which means "equal". From "equal" to "standard", was the line of thinking that led to the choice of ISO. In addition, the name ISO is used around the world to denote the organization, thus avoiding a plethora of acronyms resulting from the translation of "International Organization for Standards" into the different national languages of the ISO members, such as IOS in English, and OIN in French (from Organization International de Normalization).

ISO's standards covers all technical fields. Well-known examples of ISO standards are: the ISO film speed code, the standardized format of telephone and banking cards, ISO 9000 which provides a framework for quality management and quality assurance, paper sizes, safety wire ropes, ISO metric screw threads, and the ISO international codes for country names, currencies and languages. In telecommunications, the *open system interconnection* (OSI) reference model (see Chapter 2) is a well-known ISO standard.

ISO has co-operated with the *International Electronical Commission* (IEC) to develop standards in computer networks. IEC emphasizes hardware while ISO emphasizes software. In 1987 the two groups formed the *Joint Technical Committee 1* (JTC 1). This committee developed documents that became ISO and IEC standards in the area of information technology.

The American National Standards Institute (ANSI)

ANSI is a non-governmental organization and it was formed in 1918 to act as a cross between a standards setting body and a coordinating body for US organizations that develop standards. ANSI represents the US in international standards bodies such as ITU-T and ISO. ANSI is not restricted to information technology. In 1960 ANSI formed X3, a committee responsible for developing standards within the information processing area in the US. X3 is made up of 25 technical committees, of which X3S3 is the committee

responsible for data communications. The main telecommunications standards organization within ANSI is the T1 secretariat, sponsored by the *Exchange Carriers Standards Association*. ANSI is focused on standards above the physical layer. Hardware oriented standards are the work of the *Electronics Industries Association* (EIA) in the US.

The Institute of Electrical and Electronics Engineering (IEEE)

IEEE is the largest technical professional society in the world, and it has been active in developing standards in the area of electrical engineering and computing through its *IEEE Standards Association* (IEEE-SA). This is an international organization with a complete portfolio of standards. The IEEE-SA has two governing bodies: the Board of Governors, and the Standards Board. The Board of Governors is responsible for the policy, financial oversight, and strategic direction of the Association. The Standards Board has the charge to implement and manage the standards process, such as approving projects.

One of the most well-known IEEE standards body in the networking community is the *LAN/MAN Standards Committee*, or otherwise known as the *IEEE project 802*. They are responsible for several well-known standards, such as CSMA/CD, token bus, token ring, and the *logical link control* (LLC) layer.

The Internet Engineering Task Force (IETF)

The IETF is part of a hierarchical structure that consists of the following four groups: the *Internet Society* (ISOC) and its Board of Trustees, the *Internet Architecture Board* (IAB), the *Internet Engineering Steering Group* (IESG), and the *Internet Engineering Task Force* (IETF) itself.

The ISOC is a professional society that is concerned with the growth and evolution of the Internet worldwide. The IAB is a technical advisory group of the ISOC, and its charter is to provide oversight of the Internet and its protocols, and to resolves appeals regarding the decisions of the IESG. The IESG is responsible for technical management of IETF activities and the Internet standards process. It administers the standardization process according to the rules and procedures which have been ratified by the ISOC Trustees.

The IETF is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet. It is divided into the following eight functional areas: applications, Internet, IP: next generation, network management, operational requirements, routing, security, transport, and user services. Each area has several working groups. A working group is made-up of a group of people who work under a charter in order to achieve a certain goal. Most working groups have a finite lifetime, and a working group is dissolved once it has achieved its goal. Each of the eight functional areas has one or two area directors, who are members of IESG. Much of the work of IETF is handled via mailing lists, which anyone can join.

The IETF standards are known as *request for comments* (RFC), and each of them is associated with a different number. For instance, RFC 791 describes the internet protocol (IP), and RFC 793 the transmission control protocol (TCP). Originally, an RFC was just what the name implies, that is, a request for comments. Early RFCs were messages between the ARPANET architects about how to resolve certain procedures. Over the years, however, RFCs became more formal, and they were cited as standards, even when they were not. There are two sub-series within the RFCs, namely, *for your information* (FYI) RFCs and *standard*(STD) RFCs. The FYI RFC sub-series was created to document overviews and topics which are introductory in nature. The STD RFC sub-series was created to identify those RFCs which are in fact Internet standards.

Another type of Internet document is the *Internet-draft*. These are work-in progress documents of the IETF, submitted by any group or individual. These documents are valid for six months, and they may be updated, replaced, or become obsolete.

Finally, we note that the ISOC has also chartered the *Internet Assigned Numbers Authority* (IANA) as the central coordinator for the assignment of "unique parameters" on the Internet including IP addresses.

The ATM Forum

During the late 80s, many vendors felt that the ATM standardization process in ITU-T was too slow. The ATM Forum was created in 1991 with the objective of accelerating the use of ATM products and services in the private domain through a rapid development of

specifications. The ATM Forum is an international non-profit organization, and it has generated very strong interest within the communications industry. Currently, it consists of over 600 member companies, and it remains open to any organization that is interested in accelerating the availability of ATM-based solutions.

The ATM Forum consists of the *Technical Committee*, three *Market Awareness Committees* for North America, Europe and Asia-Pacific, and the *User Committee*.

The ATM Forum Technical Committee works with other worldwide standards bodies selecting appropriate standards, resolving differences among standards, and recommending new standards when existing ones are absent or inappropriate. It was created as one, single worldwide committee in order to promote a single set of specifications for ATM products and services. It consists of several working groups, which investigate different areas of ATM technology, such as, the ATM architecture, routing and addressing, traffic management, ATM/IP collaboration, voice and multimedia over ATM, control signalling, frame-based ATM, network management, physical layer, security, wireless ATM, and testing.

The ATM Market Awareness Committees provide marketing and educational services designed to speed the understanding and acceptance of ATM technology. They coordinate the development of educational presentation modules and technology papers, publish the *53 Bytes*, the ATM Forum's newsletter, and coordinate demonstrations of ATM at trade shows.

The ATM Forum User Committee, formed in 1993, consists of organizations which focus on planning, implementation, management or operational use of ATM based networks, and network applications. This committee interacts regularly with the Market Awareness Committees and the Technical Committee in order to ensure that ATM technical specifications meet real-world end-user needs.

The Frame Relay Forum

The Frame Relay Forum was formed in 1991, and it is an association of vendors, carriers, users, and consultants committed to the implementation of frame relay in accordance with national and international standards.

The Forum's technical committees take existing standards, which may not be sufficient for full interoperability, and create *implementation agreements* (IA). These IAs represent an agreement by all members of the frame relay community as to the specific manner in which standards will be applied. At the same time, the Forum's marketing committees are chartered with worldwide market development through education as to the benefits if frame relay.

Problems

5 Visit the web sites of ITU-T, ATM Forum, and IETF. Familiarize yourself with their organizational structure, and the type of standards that are available on these web sites.

6 Read some of the issues of *53 Bytes*, ATM Forum's newsletter, available on ATM Forum's web site.

CHAPTER 2

Basic Concepts From Computer Networks

In this Chapter, we review some basic concepts from computer networks that we will make use of in this book. First, we discuss the various communication networking techniques and the OSI reference model. Then, we present the data link layer of the OSI model, the *high-level data link control* (HDLC), the *synchronous time division multiplexing* (TDM) technique, and the *logical link control* (LLC) layer. Finally, we examine the network access protocol X.25 and we conclude this Chapter with the very popular and important *internet protocol version 4* (IPv4) .

2.1 Communication networking techniques

Communication networking techniques can be classified into the following two broad categories: *switched communication networks* and *broadcast communication networks*. Examples of switched communication networks are circuit-switched networks, such as the public telephone system, and packet-switched networks, such as computer networks based on TCP/IP. Examples of broadcast communication networks are packet radio networks, satellite networks, and multi-access local networks such as Ethernet. ATM networks belong to the packet-switched networks.

Circuit switching and packet switching are two different technologies that evolved over a long period of time. Circuit switching involves three phases: *circuit establishment*, *data transfer*, and *circuit disconnect*. These three phases take place when we make a phone-call. Circuit establishment takes place when we dial-up a number. At that moment, the public network attempts to establish a connection to the phone set that we dialed. This involves finding a path to the called party, allocating a channel on each transmission link

on the path, and alerting the called party. The data transfer phase follows, during which we converse with the person we called. Finally, the circuit disconnect phase takes place when we hang-up. At that moment, the network tears down the connection, and releases the

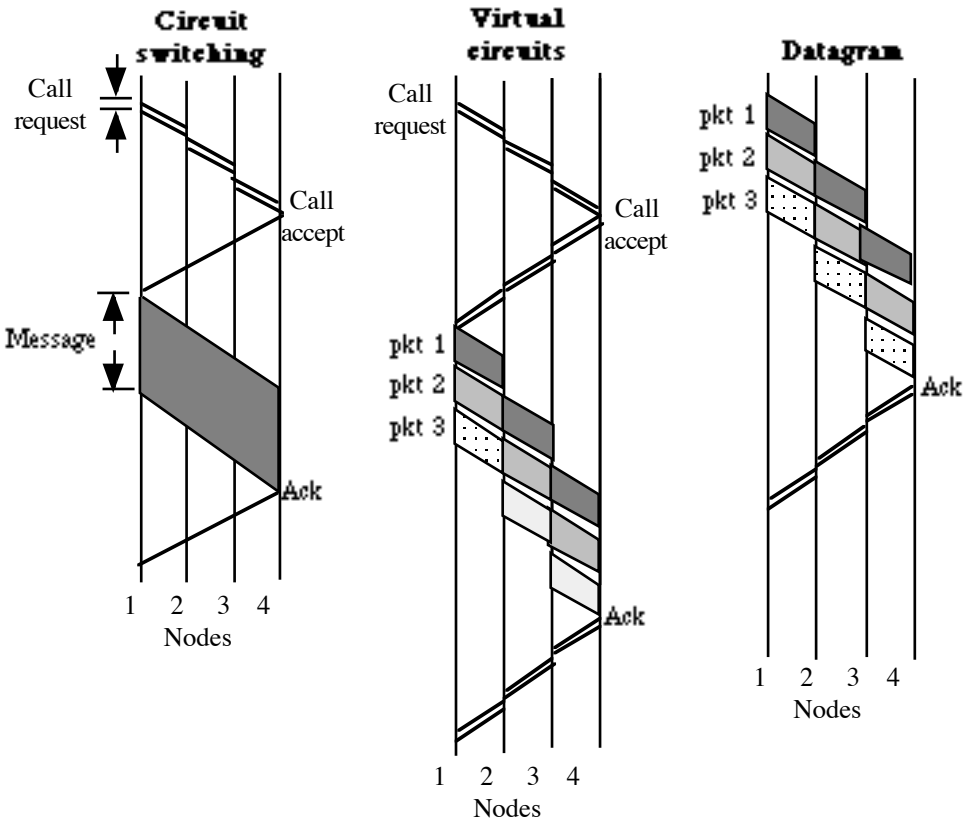


Figure 2.1: A comparison between circuit-switching, virtual circuits and datagrams

allocated channel on each link on the path. In circuit switching, channel capacity is dedicated for the duration of the connection, even when no data is being sent. For instance, when we make a phone-call, the channel that is allocated on each transmission link along the path from our phone to the one we called, is not shared with any other phone-calls. Also, in circuit switching both stations must be available at the same time in order to establish a connection. Circuit switching is a good solution for voice, since it involves exchanging a relatively continuous flow of data. However, it is not a good solution if the data is bursty. That is, the source emitting the data is active transmitting for

a period of time and then it becomes silent for a period of time during which it is not transmitting. This cycle of being active and then silent repeats until the source completes its transmission. Such intermittent type of transmission occurs in data transfers. In such cases, the utilization of the circuit-switched connection is low.

Packet switching is appropriate for data exchange. Information is sent in packets and each packet has a header with the destination address. A packet is passed through the network from node to node until it reaches its destination. Error and flow control procedures can be built into the network to assure a reliable service. In packet switching, two different techniques can be used, namely, *virtual circuits* and *datagrams*.

A virtual circuit imitates circuit switching and it involves the same three phases, that is, call set-up, transfer of packets, and call termination. In call set-up, a logical connection is established between the sender and the receiver before any packets are allowed to be sent. This is a path through the nodes of the computer network which all packets will follow. Unlike circuit switching, channel capacity on each transmission link is not dedicated to a virtual circuit. Rather, the transmission link is shared by all the virtual circuits that pass through it. Error control assures that all packets are delivered correctly in sequence. Flow control is used to assure that sender does not overrun the receiver's input buffer. The X.25 network is a good example of a packet-switched network with virtual circuits. Also, as we will see in Chapter 4, ATM networks are also packet-switched networks and they use virtual circuits.

In datagrams, no call set-up is required, and each packet is routed through the network individually. Because of this, it is possible that two successive packets transmitted from the same sender to the same receiver may follow different routes through the network. Since each packet is routed through the network individually, a datagram service can react to congestion easier. The datagram service provided by the early packet-switched networks was in some cases more primitive than that provided by virtual circuits. For instance, there was no error control, no flow control, and no guarantee of delivering packets in sequence. The IP network, used in the Internet, is a packet switched network based on datagrams. However, due to the use of static routes in the IP routers, IP packets follow the same path from a sender to a destination, and

therefore, they are delivered in sequence. Also, unlike earlier packet-switched networks with datagram services, TCP/IP provides error and flow control.

An example of how two nodes communicate using circuit switching, virtual circuits, and datagrams is given in figure 2.1. In this example, node 1 communicates with node 4 through intermediate nodes 2 and 3. The passage of time is indicated on the vertical lines, and there is one vertical line per node. In the circuit–switching case, the time it takes node 1 to transmit the call request packet and the message, is indicated vertically between the two arrows, on the first line associated with node 1. The two diagonal parallel lines between the vertical lines of the first and the second node show the propagation delay of the call request packet between these two nodes. Similar notation is used for the virtual circuit and datagrams cases. As we can see, the datagram scheme takes less time to transmit the three packets than the virtual circuit scheme.

A broadcast network has a single communication channel that is shared by all the stations. There are no switching nodes as in circuit or packet switching. Data transmitted by one station is received by many, and often by all. An access control technique is used to regulate the order in which stations transmit. The most widespread example of a broadcast network is the Ethernet.

2.2 The Open System Interconnection (OSI) Reference Model

In the early days of packet switching, the various communications software suites that were available could not communicate with each other. In order to standardize the communications protocols and also facilitate their development, the International Organization for Standardization (ISO) proposed a model known as the *Open Systems Interconnection (OSI) Reference Model*. The functionality of the software for packet switching was grouped into seven layers, namely, the *physical* layer, the *data link* layer, the *network* layer, the *transport* layer, the *session* layer, the *presentation* layer, and the *application* layer. These layers are shown in figure 2.2. Each layer provides service to the layer directly above it, and receives service from the layer directly below it.

The physical layer is concerned with the transmission of raw bits over a communications channel. The data link's function is to transform the raw transmission link provided by the physical layer into a reliable communications link. This was deemed

necessary since early transmission links were inherently unreliable. Modern fiber-based communications links are highly reliable, and as will be seen later on in this book, there is

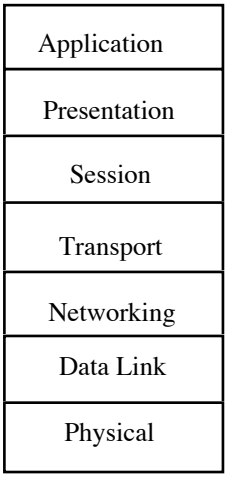


Figure 2.2: The OSI reference model

no need for all the data link functionality. The network layer is concerned with routing packets from source to destination, congestion control, and internetworking. The transport protocol is concerned with the end-to-end packet transfer, that is, between an application in the source computer and an application in the destination computer. Some of its main functions are establishment and deletion of connections, reliable transfer of packets, and flow control. The session layer allows users in different computers to set up sessions between themselves. One of the services of the session layer is to manage dialogue control. The presentation layer is concerned with the syntax and semantics of the information transmitted. In general, two heterogeneous computers may not have the same way of representing data types internally. The presentation layer facilitates the communication between two such computers, by converting the representation used inside a computer to a network standard representation and back. Finally, the application layer contains protocols that are commonly used, such as, file transfer, electronic mail, and remote job entry.

2.3 Data link layer

This protocol layer was designed to provide a reliable point-to-point connection over an unreliable link. The main functions of the data link layer are: window flow control, error control, frame synchronization, sequencing, addressing, and link management. At this layer, a packet is referred to as a *frame*. Below, we examine the window-flow control mechanism, error detection schemes, and the error control mechanism.

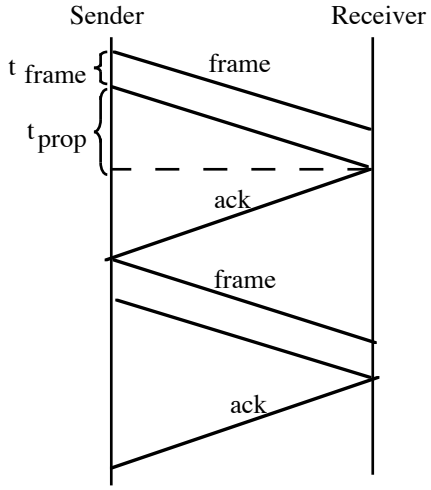


Figure 2.3: The stop-and-wait scheme

Window-Flow Control

This is a technique for assuring that a transmitting station does not overrun the receiving station's buffer. The simplest scheme is *stop-and-wait*. The sender transmits a single frame and then it waits until the receiver gets the frame and sends an acknowledgment (ACK). When the sender receives the ACK, it transmits a new frame. This scheme is shown in figure 2.3. The link's utilization U depends on the propagation delay, t_{prop}, and on the time to transmit a frame, t_{frame}. Let

$$a = \frac{t_{prop}}{t_{frame}}$$

Then,

$$U = \frac{t_{frame}}{t_{frame} + 2t_{prop}} = \frac{1}{1 + 2a} .$$

If a<<<1, that is the propagation delay is significantly less than the time to transmit a frame, then the link's utilization U is large. If a>>>1, that is the propagation delay is significantly greater than the time to transmit a frame, then U is small. As an example, let us consider a satellite link transmitting at 56 Kbps, and let us assume 4000-bit frames and a propagation delay of 270 msec. Then, the time to transmit a frame is 71 msec, a = 270/71 = 3.8, and U = 0.116.

In the stop-and-wait protocol, only one frame is outstanding, i.e. unacknowledged, at a time. A more efficient protocol is the *sliding window-flow control* protocol where many frames can be outstanding at a time. The maximum number of frames, W, that a station is allowed to send to another station without acknowledgment is referred to as the maximum window. To keep track which frames have been acknowledged, each frame is numbered sequentially, and the numbers are re-usable. An example of the sliding window-flow control scheme is shown in figure 2.4. The maximum window size W is fixed to 8. In

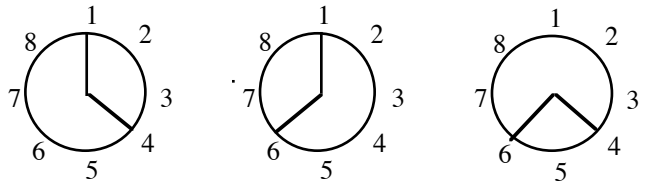


Figure 2.4: An example of the sliding window-flow control scheme

figure 2.4(a), station A transmits 4 frames with sequence numbers 1, 2, 3 and 4, and its window is reduced to 4 consisting of the sequence numbers {5,6,7,8}. In figure 2.4(b), station A sends 2 more frames with sequence numbers 5 and 6, and its window is down to 2 consisting of the numbers {7,8}. In figure 2.4(c), station A receives an ACK from station B for the frames with sequence numbers 1, 2 and 3, and its window opens up to 5 frames consisting of the sequence numbers {7,8,1,2,3}.

The efficiency of this protocol depends upon the maximum window size and the round-trip delay. Let $t_{frame} = 1$. Then,

$$a = \frac{t_{prop}}{t_{frame}} = t_{prop}.$$

The time to transmit the first frame and receive an acknowledgment is equal to $t_{frame} + 2\,t_{prop} = 1+2a$. If $W>1+2a$, then the acknowledgment arrives at the sender before the window has been exhausted, and we have that $U = 1$. If $W<1+2a$, then the acknowledgment arrives after the window has been exhausted, and we have

$$U = \frac{W}{1+2a}.$$

Error detection

The simplest error detection scheme is the *parity check*. In this scheme, a parity bit is appended to the end of each frame. A more complex error detection scheme based on the parity check is the *longitudinal redundancy check*. The data is organized into a matrix, as shown in figure 2.5. There are eight columns, and as many rows as the number of bytes. Each matrix element contains one bit. An even parity check is applied to each row and each

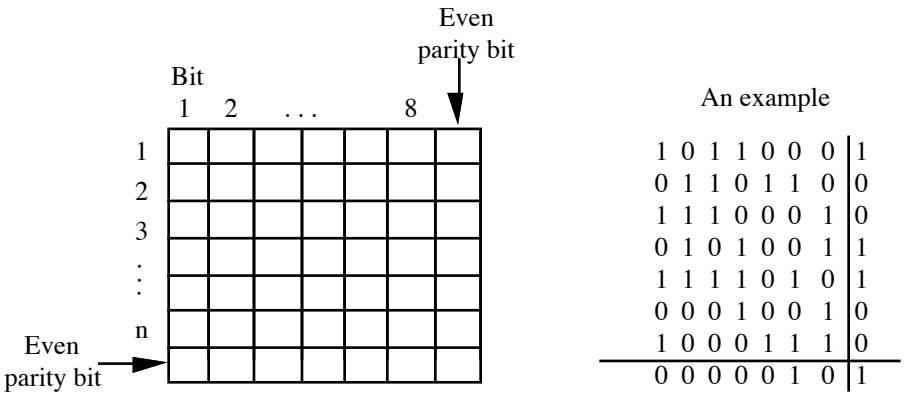


Figure 2.5: The longitudinal redundancy check

column. We observe that the parity bit applied to the last column which contains the parity bits of all the rows, is the same as the one applied to the last row which contains the parity bits of all the columns!

The *cyclic redundant check* (CRC) is a commonly used error detection scheme, and it is used extensively in ATM networks. The CRC scheme utilizes a pre-determined bit pattern P, which is known to both the sender and the receiver. Let n+1 be the length of this bit pattern. Now, let us assume that we have a k-bit message M to transmitted. The sender shifts M to the left by n bits to obtain the quantity 2^nM, and then divides 2^nM by P. The remainder of that division is an n-bit sequence, known as the *frame check sequence* (FCS). The FCS is added to 2^nM and the entire (k+n)-bit message is transmitted to the receiver. The receiver divides the message by the same bit pattern P. The message has been received correctly if the remainder of that division is zero. All single bit errors, and some combinations of erroneous bits can be detected and corrected.

As an example let M=1010001101 and P=110101. Then, the FCS will be 5 bits long and it is calculated as follows. M is first shifted to the left by 5 positions, that is 2^5M=101000110100000. Then, 2^5M is divided by P resulting to an FCS equal to 01110. Finally, the transmitted message is 101000110101110. If this message is correctly received, when divided by P=110101, it should give a zero remainder.

It is customary to express the bit pattern P in polynomial form. This is done as follows. Each bit is represented by a term x^n, where n is the location of the bit in the pattern, counting from the right-hand-side towards the left-hand side. That is, the rightmost bit corresponds to the term x^0, the second rightmost bit corresponds to the term x^1, and so on. The value of the bit is the coefficient of its corresponding polynomial term. For instance, the pattern 110101 used above is expressed as $x^5+x^4+x^2+1$.

The *checksum* is another error detection technique that is used in the TCP/IP suite of protocols. The data to be sent is treated as a sequence of binary integers of 16 bits each, and the sum of these 16-bit integers is computed. The data could be of any type or a mixture of types. It is simply treated as a sequence of integers for the purpose of computing their sum. The 16-bit half-words are added up using 1's compliment arithmetic. The 1's compliment of the final result is then computed, which is known as the checksum. 32-bit integers can also be used. The checksum is used in TCP to protect

the entire packet. That is, it is calculated using the header and the payload of the TCP packet. It also used in IP to protect the IP header only. Computing the checksum in TCP is a time-consuming operation, and a considerable speed up can be achieved if it is done in hardware.

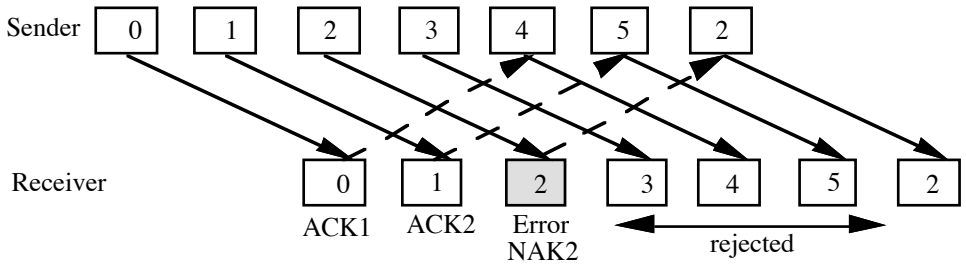


Figure 2.6: The go-back-n scheme

Error control

Error control refers to the mechanism used to detect and correct errors occurred in the transmission of frames. This mechanism is known as the *automatic repeat request* (ARQ) and it uses error detection, the window-flow control mechanism, positive and negative acknowledgments, and timers. Errors in the transmission of frames occur because a frame is lost or because it is damaged, that is one or more of its bits have been flipped. Damaged frames are detected by the ARQ mechanism using CRC, and lost frames are detected by observing out-of-sequence frames. Recovery of a lost or damaged frame is done by requesting the sender to re-transmit the frame. Three different versions of the ARQ have been standardized, namely *stop-and-wait ARQ*, *go-back-n ARQ*, and *selective-reject ARQ*. The stop-and-wait ARQ is based on the stop-and-wait window-flow control scheme, whereas the go-back-n ARQ and the selective-reject ARQ are based on the sliding window-flow control scheme.

In the go-back-n scheme, the sender sends a series of frames using the sliding window-flow control technique. Let us assume that station A is transmitting to station B. If

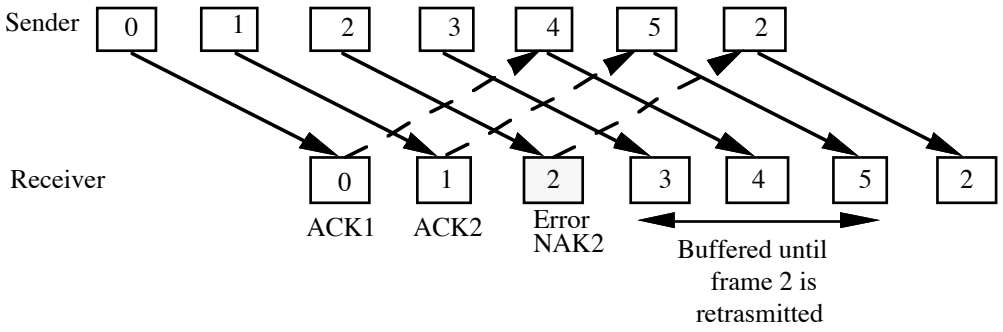


Figure 2.7: The selective-reject scheme

B receives a frame correctly, then it sends an ACK with the next frame number that it expects to receive. An ACK may be for several successive frames that have been correctly received. If B receives a damaged frame, say frame i, and it has previously received correctly frame i-1, then B sends a negative acknowledgment (NAK) indicating that frame i is in error. When A receives the NAK, it retransmits frame i plus all other frames after i that it has already transmitted. An example of this scheme is shown in figure 2.6.

Now, let us consider the case where frame i is lost. If B correctly receives later on frame i+1, then it will realize that frame i+1 is out-of-sequence, and it will deduce that frame i is lost. B will then send a NAK indicating that the ith frame has to be re-transmitted. A retransmits frame i plus all other frames after i that it has already transmitted. If frame i is lost and no other frames arrive, then B cannot detect the lost frame. However, for each transmitted frame, A sets a timer. If the timer expires before A receives an ACK or a NAK, A retransmits the frame. In the above case, the lost frame's timer will expire and A will re-transmit it.

In the selective-reject ARQ scheme only the frame that is in error is retransmitted. All subsequent frames that arrive at B are buffered, until the erroneous frame is received again. This is a more efficient procedure, but it is more complex to implement. The selective-reject scheme is used in TCP. An example of the selective-reject ARQ scheme is shown in figure 2.7.

2.4 The high data link control (HDLC) protocol

This protocol is widely used and it is the basis for many other important data link protocols. It was derived from IBM's data link protocol *synchronous data link control* (SDLC). It was later on modified and standardized by ISO as the *high data link control* (HDLC) protocol. HDLC was designed to satisfy different types of stations, link configurations, and transfer modes. The following three types of stations were defined: *primary*, *secondary*, and *combined*. A primary station is responsible for controlling the operation of the link, a secondary station operates under the control of a primary station, and a combined station has the features of both the primary and the secondary station.

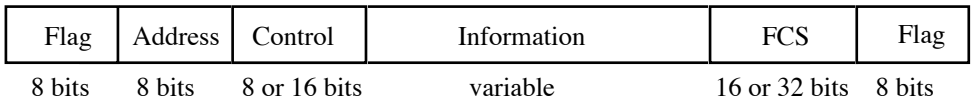


Figure 2.8: The HDLC frame

Also, the following types of link configurations were defined: *unbalanced* and *balanced*. An unbalanced configuration consists of one primary and one or more secondary stations and it supports both full-duplex and half-duplex transmission. A balanced configuration consists of two combined stations and it supports both full-duplex and half-duplex transmission. Based on these station types and configurations, the following three data transfer modes were defined: *normal response time mode* (NRM), *asynchronous balanced mode* (ABM), and *asynchronous response mode* (ARM). NRM is used with an unbalanced configuration. The primary station initiates data transfers to the secondary stations, and a secondary station may only transmit data in response to a command from the primary. NRM is used in multi-drop lines connecting terminals to a host. ABM is used with a balanced configuration, and it is the most widely used transfer mode for a full-duplex point-to-point link. Either combined station may initiate a transmission without receiving the permission from the other combined station. Finally, ARM is based on an unbalanced configuration and it is rarely used.

HDLC is a bit-oriented protocol and it uses the frame structure shown in figure 2.8. A single format is used for all data and control exchanges. The frame is delimited by a flag which contains the unique pattern 01111110. If frames are transmitted back-to-

back, a single flag may be used to indicate the end of one frame and the beginning of the next one. Obviously, the pattern 01111110 can be easily encountered within a frame, in which case, it will be interpreted as the end of the frame. In order to avoid this from happening, a technique known as *bit stuffing* is used. The sender always inserts an extra 0 after the occurrence of five consecutive 1's. The receiver monitors the bit stream looking for five consecutive 1's. When this pattern appears, the receiver examines the sixth bit. If it is a 0, it is deleted from the bit stream. If it is a 1 and the seventh bit is a 0, the receiver interprets the bit pattern as a delimiting flag. If the sixth bit is a 1 and the seventh bit is also a 1, then it is an error.

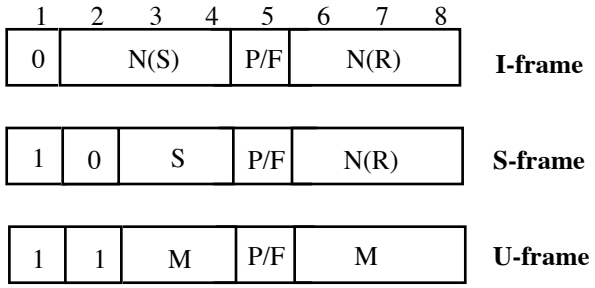


Figure 2.9: The control field of the HDLC frame

The second field in the HDLC frame is the address field. This is an 8-bit field used in multi-drop lines and it is used to identify the secondary station to which the frame is transmitted. It is not necessary in a point-to-point link.

The third field in the HDLC frame is the control field. It is an 8-bit field, extendible to a 16-bit field, and its structure is shown in figure 2.9. It is used to identify the following three types of frame: *information frame* (I-frame), *supervisory frame* (S-frame), and *unnumbered frame* (U-frame). An I-frame is used to carry data and ARQ control information, an S-frame is used to carry only ARQ control information, and a U-frame is used to provide supplemental link control functions. If the first bit of the control field is 0, then the frame is an I-frame. Otherwise, depending on the value of the second bit, it may be an S-frame or a U-frame. The meaning of the remaining sub-fields is as follows:

N(S): send sequence

N(R): receive sequence

S: supervisory function bits

M: unnumbered function bits

P/F: poll/final bit.

During a typical exchange of information between two stations, say A and B, both stations receive and send data. This means that there are two separate ARQ mechanisms, one for the data sent from A to B and another for the data sent from B to A. The fields N(R) and N(S) in the I-frame are used to carry information for both the ARQ mechanisms piggy-backed on the frames carrying data. N(R) is used by station A to indicate to station B the current status of the ARQ from B to A, and N(S) is used by station A to indicate the sequence number of the frame that it is transmitting to B. S-frames are used when no I-frames are exchanged and also to carry supplementary control information.

The information field is only present in the I-frames and in some U-frames. The FCS is calculated using a 16-bit CRC. A 32-bit CRC is optional.

2.5 Synchronous time division multiplexing (TDM)

Time division multiplexing permits a data link to be utilized by many sender/receiver pairs, as shown in figure 2.10. A multiplexer combines the digital signals from N incoming links into a single composite digital signal, which is transmitted to the demultiplexer over a link. The demultiplexer breaks out the composite signal into the N individual digital signals and distributes them to their corresponding output links. In the multiplexer, there is a small

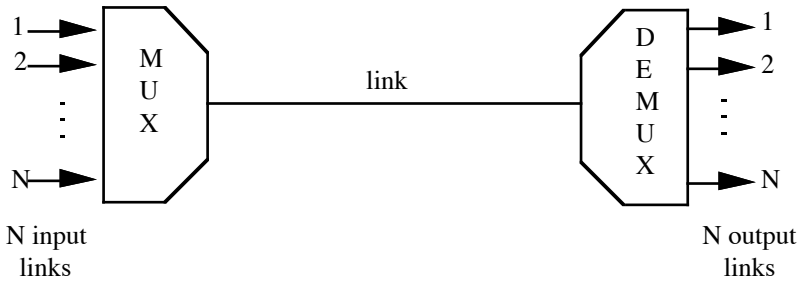


Figure 2.10: Synchronous time division multiplexing (TDM)

buffer for each input link that holds incoming data. The N buffers are scanned sequentially and each buffer is emptied out fast enough before new data arrives.

The transmission of the multiplexed signal between the multiplexer and the demultiplexer is organized into frames. Each frame contains a fixed number of slots, and each slot is pre-assigned to a specific input link. The duration of a slot is either a bit or a byte. If the buffer of an input link has no data, then its associated slot is transmitted empty. The data rate of the link between the multiplexer and the demultiplexer that carries the multiplexed data streams is at least equal to the sum of the data rates of the incoming links. A slot dedicated to an input link repeats continuously frame after frame, and it is called a *channel*.

TDM is used in the telephone system. The voice analog signals are digitized at the end office using the *pulse code modulation* (PCM) technique. That is, the voice signal is sampled 8,000 times per second, or every 125 μsec, and the amplitude of the signal is approximated by a 7- or an 8-bit number. At the destination end office, the original voice signal is reconstructed from these samples. As a consequence of this sampling mechanism, most time intervals within the telephone system are multiples of 125 μsec.

The standard that specifies how to multiplex several voice calls onto a single connection is known as the *digital signal level* standard or the *DS* standard. This is a generic digital standard and it is independent of the medium over which it is transmitted.

Digital signal number	Voice channels	Data Rate (Mbps)
DS-1	24	1.544
DS-1C	48	3.152
DS-2	96	6.312
DS-3	672	44.736
DS-4	4032	274.176

Table 2.1: The North American Hierarchy

Level number	Voice channels	Data Rate (Mbps)
1	30	2.048
2	120	8.448

3	480	34.368
4	1920	139.264
5	7680	565.148

Table 2.2: The international (ITU-T) hierarchy

The DS standard specifies a hierarchy of different data rates, as shown in table 2.1. The nomenclature of this hierarchy is DS followed by the level of multiplexing. For instance, DS-1 multiplexes 24 voice channels and it has a data rate of 1.544 Mbps. The higher levels in the hierarchy are integer multiples of the DS-1 data rate. This hierarchy is known as the *plesiochronous digital hierarchy* (PDH). Plesiochronous means frame synchronous (from the Greek word *plesio* which means frame).

The DS standard is a North American standard and it is not the same as the international hierarchy standardized by ITU-T. Table 2.2 gives the international hierarchy, which consists of different levels of multiplexing. For instance, level-1 multiplexes 30 voice channels and it has a data rate of 2.048 Mbps. As in the DS standard, the higher levels are integer multiples of the level-1 data rate.

The digital signal is carried over a *carrier system*, or simply a *carrier*. A carrier consists of a transmission component, an interface component, and a termination component.. The T carrier system is used in North America to carry the DS signal, and the E carrier system is used to carry the international digital hierarchy. T1 carries the DS-1 signal, T2 the DS-2 signal, T3 the DS-3 signal, and so on. Similarly, E1 carries the level-1 signal, E2 carries the level-2 signal, and so on. Typically, the T and DS nomenclatures are used interchangeably. For instance, one does not distinguish between a T1 line and the DS-1 signal. The same applies for the international hierarchy.

The DS-1 format, shown in figure 2.11, consists of 24 8-bit slots and a 1-bit slot for frame synchronization. On the 1-bit slot channel, the frame synchronization pattern 1010101... is transmitted. Each of the 24 slots carries a single voice. For five successive frames, an 8-bit PCM sample is used. In the sixth frame, a 7-bit sample is used and the 8th

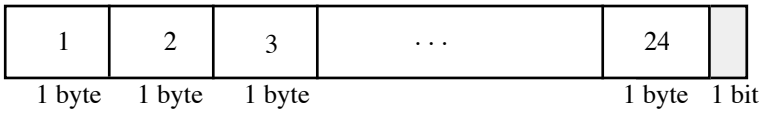


Figure 2.11: The DS-1 format

extra bit is used for signalling. The total transmission rate of the DS-1 format is 24x8+1=193 bits per 125 μsec corresponding to 1.544 Mbps, with each voice channel carrying a 64 Kbps voice.

The DS-1 format can be also used to carry data. In this case, 23 8-bit slots are used for data and the remaining slot is used for control and frame synchronization. Each data slot carries 7 bits of data amounting to a channel of 56 Kbps. The extra bit per slot is used for control.

In the international hierarchy, the level 1 format for voice consists of 32 8-bit slots, resulting to a total transmission rate of 2.048 Mbps. Of these slots, 30 are used for voice and the remaining two are used for synchronization and control.

2.6 The logical link control (LLC) layer

A *local area network* (LAN) or a *metropolitan area network* (MAN) consists of a transmission medium which is shared by all the stations that are attached to it. Access to the transmission medium is achieved through a *medium access control* (MAC) protocol.

The IEEE LAN/MAN Standards Committee, has produced several standards for local and metropolitan area networks, such as the IEEE 802.3 standard for Ethernet, the IEEE 802.4 standard for the token bus, and the IEEE 802.5 standard for the token ring. The *logical link control* (LLC) protocol was defined in the IEEE 802.2 standard, and it runs over several different MACs.

The OSI stack for stations communicating over the same LAN or a MAN is shown in figure 2.12. The data link layer in the OSI reference model corresponds to the LLC and MAC layers. As can be seen, the networking layer is not present. Typically, layer 3 carries out functions, such as routing, addressing, flow control and error control, over a sequence of links. In a LAN or a MAN, however, there is no need for routing when transmitting between two stations which are attached to the same shared medium.

The other functions of layer 3 are performed by the LLC. This simplifies considerably the OSI stack.

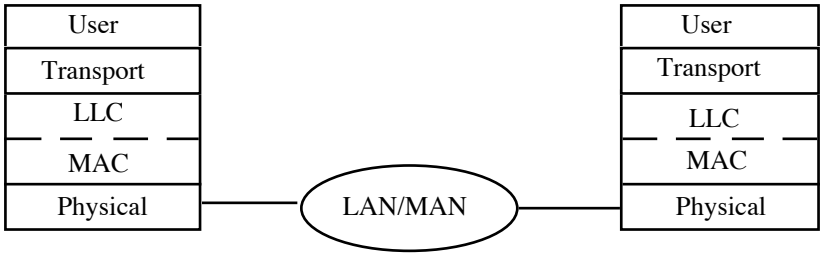


Figure 2.12: The OSI stack for LANs/MANs

LLC is concerned with the transmission of link-level PDUs between two stations. Addressing in LLC is achieved by specifying the source and destination LLC users. An LLC user is typically a high-level protocol or a network management function. An LLC user address is referred to as a *service access point* (SAP). The following services are provided by LLC.

Unacknowledged connectionless service: This is a datagram type of service. It does not involve any flow or error control and the delivery of data is not guaranteed.

Connection-mode service: This is similar to the service offered by X.25. A logical connection is first set-up between two users before any exchange of data takes place. Flow and error control is provided.

Acknowledged connectionless service: This is a service which is a cross between the above two services. Datagrams are acknowledged as in the connectionless mode service, but a logical connection is not set-up.

LLC is modelled after HDLC. It makes use of the asynchronous, balanced mode of operation of HDLC in order to support the connection-mode service. The unacknowledged connectionless service is supported using the unnumbered information PDU, and the acknowledged connectionless service is supported using two new unnumbered PDUs.

The LLC and MAC encapsulation is shown in figure 2.13. The LLC header contains the following fields. I/G is a 1-bit field indicating whether the destination address is an individual address or a group address. DSAP and SSAP are 7-bit fields

indicating the destination and source *service access points* (SAP). C/R is a 1-bit field indicating whether the frame is a command or response frame. The LLC control field is identical to that of the HDLC with extended sequence numbers. The MAC header contains a MAC control field, the destination address (DA) and the source address (SA), and the MAC trailer carries the FCS value. The address of a station is the physical attachment point on the LAN.

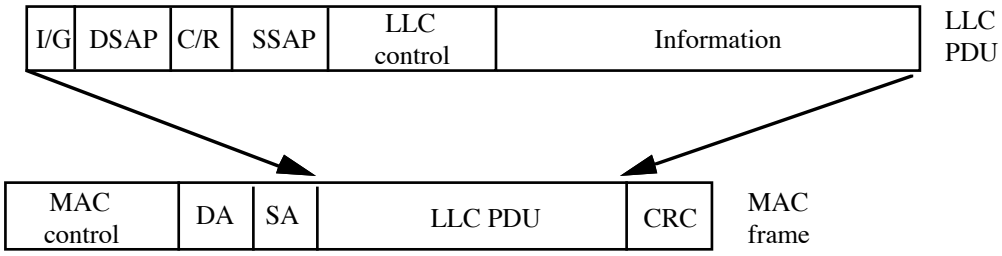


Figure 2.13: LLC and MAC encapsulation

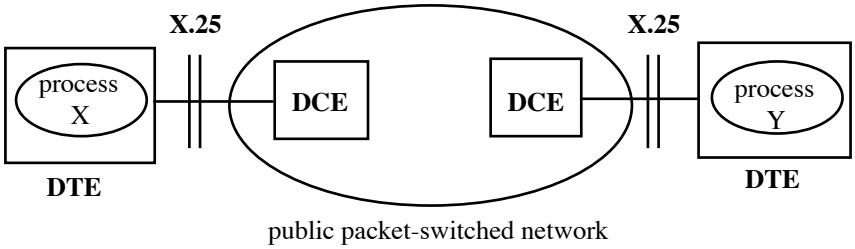


Figure 2.14: The X.25 interface

2.7 Network access protocol X.25

X.25 was originally approved by ITU-T in 1976 to provide an interface between public packet-switched networks and their customers. Since then, it has been revised several times. It has been widely used and it has also been employed for packet switching in ISDN. The X.25 standard specifies only the interface between a user's machine, referred to as the *data terminal equipment* (DTE), and the node in the packet-switched network to which it is attached, referred to as the *data communication equipment* (DCE), as shown in figure 2.14. The standard is not concerned with the internal architecture of the packet-switched network. This is done deliberately so that vendors can use their own network

architectures while at the same time they are compatible with the end users. The standard specifies the first three layers of the ISO model. As shown in figure 2.15, X.21 is the standard for the physical layer, LAP-B (a subset of HDLC) is the standard for the data link layer, and X.25 is the standard for the network layer. Below, we review some of the basic features of X.25.

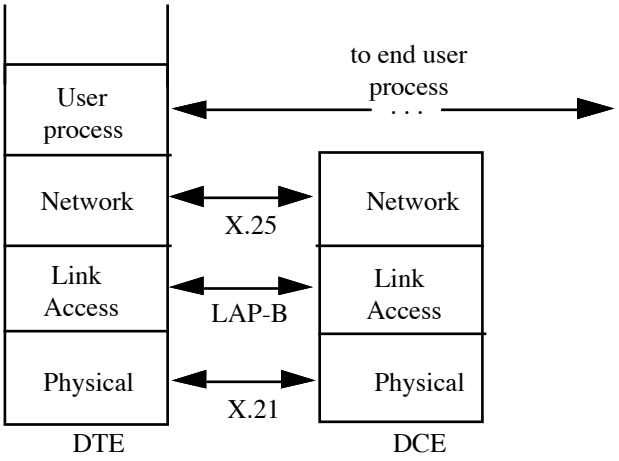


Figure 2.15: The X.25 suite

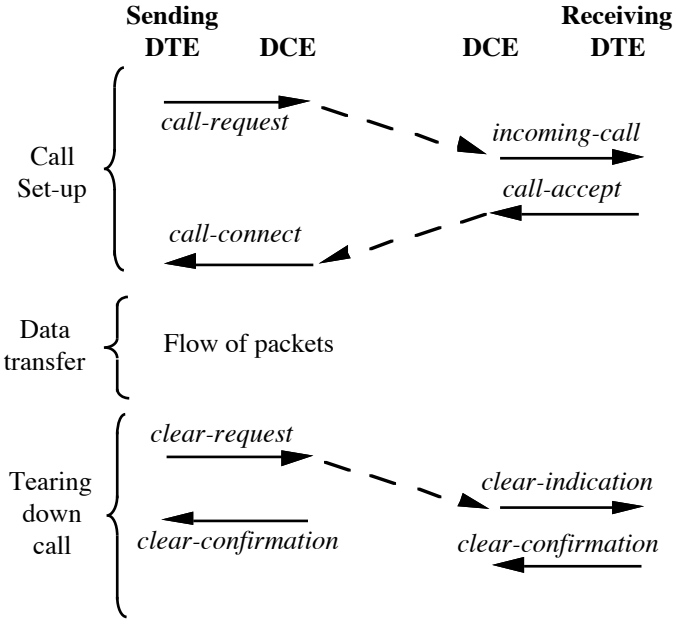


Figure 2.16: Call set-up and tearing down in X.25

X.25 provides a virtual circuit service. Two types of virtual circuits are allowed: *switched virtual circuits* (SVC) and *permanent virtual circuits* (PVC). The following events take place in order to set-up an SVC. A pictorial view is shown in figure 2.16.

1. The sending DTE sends a *call-request* packet to its DCE requesting to establish a virtual circuit to a specific DTE. The packet contains the source and destination addresses and a virtual circuit number selected by the DTE.
2. The network routes the call-request packet to the receiver's DCE, which sends an *incoming-call* packet to the receiving DTE. This packet has the same format as the call-request packet but it utilizes a different virtual circuit number selected by the receiver's DTE.
3. The receiving DTE, upon receipt of the incoming-call packet, indicates acceptance of the call by sending a *call-accept* packet to its DTE using the virtual circuit number used in the incoming-call packet.
4. The network routes the packet to the sender's DCE, which sends a *call-connected* packet to the sending DTE. This packet has the same format as the call-accept packet and it has the virtual circuit number used in the original call-request packet.
5. The sending and receiving DTEs exchange data and control packets using their respective virtual circuit numbers.

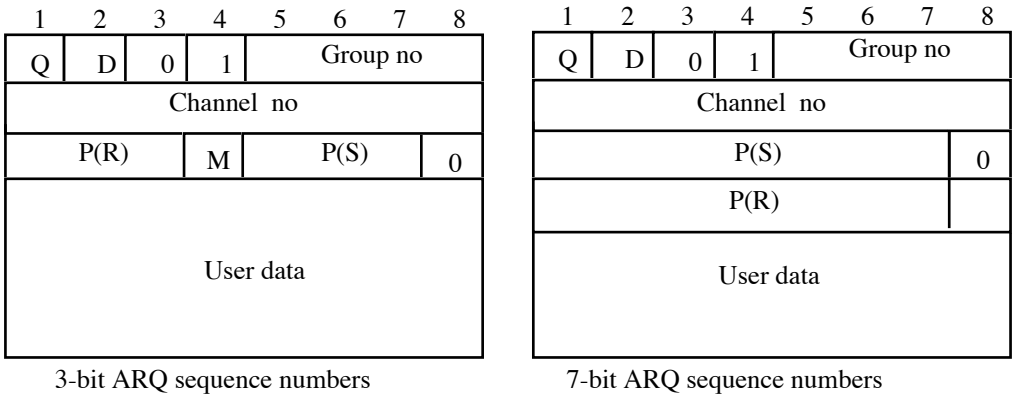


Figure 2.17: X.25 data packet formats

6. The sending (or the receiving) DTE sends a *clear-request* packet to terminate the virtual circuit. Its DCE sends back a *clear-confirmation* packet, and forwards the clear-request packet to the destination DCE, which issues a clear-indication packet to its DTE and from which it receives a clear-confirmation packet.

Several types of packets are used in X.25. The format for data packet with 3-bit and 7-bit sequence numbers is shown in figure 2.17. Q is a 1-bit field which is user specified, and D is a 1-bit field used to indicate whether the acknowledgments are local or remote. If D=0, the acknowledgments are between the DTE and its local DCE or the network. If D=1, the acknowledgments come from the receiver DTE. The 12-bit field obtained by combining the fields Group no and Channel no, is used to indicate the virtual circuit number, which has local significance, i.e., it is valid only between a DTE and its local DCE. M is a 1-bit

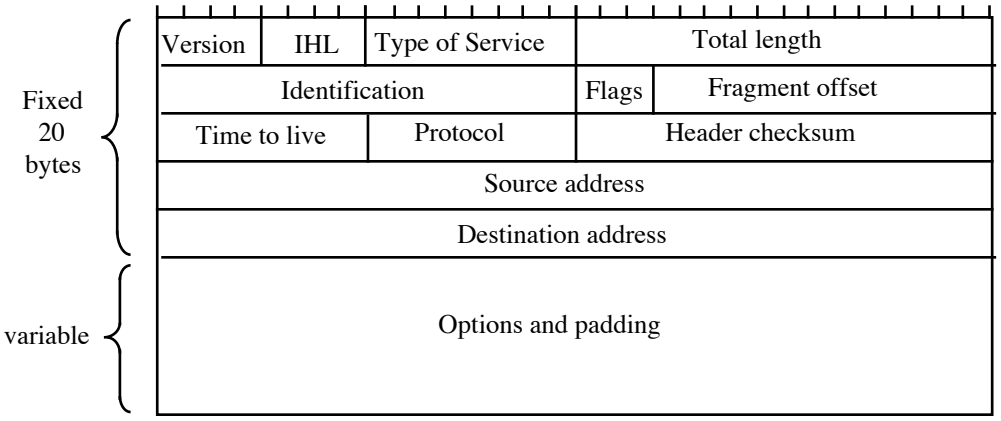


Figure 2.18: The IPv4 header

field used when packet fragmentation is employed. The P(R) and P(S) contain the receive and send ARQ sequence numbers.

2.8 The internet protocol (IP)

IP is part of the TCP/IP suite of protocols used in the Internet. TCP corresponds to the transport layer of the OSI model, and IP corresponds to the network layer of the OSI

model. In this section, we describe the current version of IP, known as *IP version 4* (IPv4).

IP provides a connectionless service using packet switching with datagrams. Packets in a connectionless network, such as the IP network, are referred to as *datagrams*. An IP host can transmit datagrams to a destination IP host without having to set-up a connection to the destination, as in the case of X.25, frame relay and ATM networks. IP datagrams are routed through the IP network independently from each other, and in theory, they can follow different paths through the IP network. In practice, however, the IP network uses routing tables which remain fixed for a period of time. In view of this, all IP packets from a sender to a receiver typically follow the same path. These routing tables are refreshed periodically, taking into account congested links and hardware failures of routers and links.

IP does not guarantee delivery of IP datagrams. In view of this, if the underlying network drops an IP datagram, IP will not be aware of that. Also, IP does not check the payload of an IP datagram for errors, but it only checks its IP header. IP will drop an IP datagram, if it finds that its header is in error. Lost or erroneous data is recovered by the destination's TCP using the selective-reject ARQ scheme described in section 2.3.

2.8.1 The IP header

An IP datagram consists of a header and a payload. The IP header is shown in figure 2.18, and it consists of a 20-byte fixed part and an optional part which has a variable length. The following fields are defined in the IP header:

Version: A 4-bit field used to indicate which version of the protocol is used.

Internet Header Length (IHL): This is a 4-bit field and it gives the length of the header in 32-bit words. The minimum header length is five 32-bit words or 20 bytes.

Type of service: This is an 8-bit field that is used to indicate whether the sender prefers the datagram to travel over a route with minimal delay or a route with maximal throughput.

Total length: A 16-bit field used to indicate the length of the entire datagram, i.e., header and payload. The default value for the maximum length is 65,535 bytes.

Identification: A 16-bit field used by the receiver to identify the datagram that the fragment belongs to. All fragments of a datagram have the same value in the identification field.

Flags: This is a 3-bit field, but only two bits are used, namely, the "more fragments" and the "don't fragment". All fragments except the last one, have the "more fragments" bit set. This information permits the receiver to know when all the fragments have arrived. The "don't fragment" bit is used to disallow fragmentation.

Fragment offset: The 13-bit field contains an offset that points where in the original datagram this fragment belongs to.

Time to live: This is an 8-bit field that specifies in seconds how long a datagram is allowed to live in the network. The maximum lifetime is 255 sec. Every router that processes the datagram must decrease this field by one second, and by several seconds if the datagram is queued in the router for a long time. This field can be seen as being similar to a hop count. When the time to live field becomes equal to zero, the datagram is discarded. This prevents a datagram from moving around in the network forever.

Protocol: This field is 8 bits long and it specifies the next higher level protocol, such as TCP and UDP, to which the datagram should be delivered..

Header checksum: A 16-bit field used to verify whether the IP header has been correctly received. The transmitting host adds up all the 16-bit half-words of the header using 1's compliment arithmetic, assuming that the checksum field is zero. The 1's compliment of the final result is then computed and placed in the checksum field. The receiving host calculates the checksum, and if the final result is zero, then the header has been correctly received. Otherwise, the header is erroneous and the datagram is dropped. The checksum is recomputed at each router along the path of the datagram, since at least one field of the header (the time to live field) is changed.

Source address: A 32-bit field populated with the network and host number of the sending host.

Destination address: A 32-bit field populated with the network and host number of the destination host. The IP addressing scheme is discussed below.

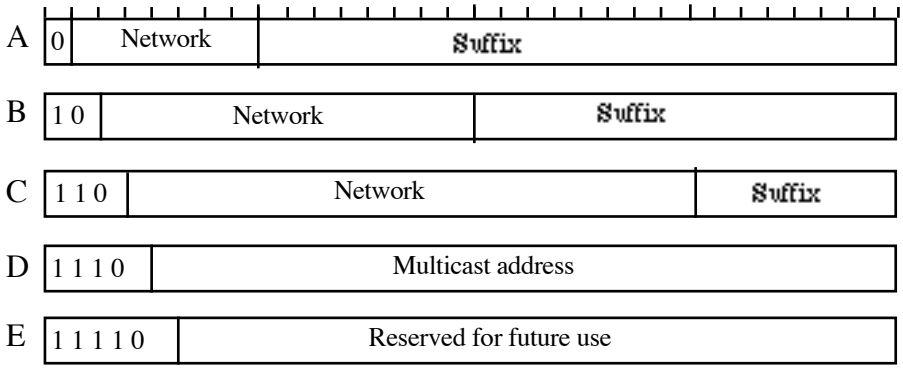


Figure 2.19: The IP address classes

Options: A variable-length field used to encode the options requested by the user, such as security, source routing, route recording, and time stamping.

Padding: A variable-length field used to make the header of the datagram an integral multiple of 32-bit words.

2.8.2 IP addresses

As we saw above, IP addresses are 32-bit long. An IP address is divided into two parts, namely, a network and a suffix. The network identifies the physical network to which the host computer is attached, and the suffix identifies the host computer itself. The size of these two fields may vary according to the class of the IP address. Specifically, five different classes of addresses have been defined, referred to as class A, B, C, D, and E, as shown in figure 2.19.

Classes A, B and C are called the *primary classes* because they are used for host addresses. Class D is used for multicasting, and class E is reserved for future use. The first field determines the class of the IP address, and it ranges from 1 bit for a class A address to five bits for a class E addresses. The second field gives the network address, and the third field is the suffix which gives the host address.

In class A, there is a 7-bit network address and a 24-bit host address, resulting to 128 network addresses and 16,777,216 host addresses. In class B, there is a 14-bit network address and a 16-bit host address, resulting to 16,384 network addresses and

65,536 host addresses. In class C, there is a 21-bit network address and a 8-bit host address, resulting to 2,097,152 network addresses and 256 host addresses.

Network addresses are usually written in the *dotted decimal notation*. That is, each byte is written in decimal, ranging from 0 to 255. As an example, the IP address 00000111 00000010 00000000 00000010 will be written as 7.2.0.2. Using this notation, we have that the range of class A addresses is from 1.0.0.0 to 127.255.255.255, for class B we have a range of values from 128.0.0.0 to 191.255.255.255, and for class C we have a range of 192.0.0.0 to 233.255.255.255.

Class C is very common, whereas class A is rarely used since there are only few networks with that large number of hosts. IP reserves host address zero to denote the address of a network. For instance, in the class B address 128.32.0.0 the network field 128.32 and the suffix is 0.0. This indicates the address of the network 128.32. For broadcasting within the network, IP uses the address 128.32.255.255.

IP assigns multiple IP addresses to routers, since a router is attached to multiple networks. Specifically, a router has one IP address for each network that it is attached to. An individual host connected to multiple networks has also multiple IP addresses, one for each network connection. Such a host is referred to as *multihomed*.

Subnetting

The IP address structure described above introduces a two-level hierarchy. The first level is the network address and the second level is the host address carried in the suffix. In many cases, these two levels of addressing is not enough. For instance, if we consider an organization with a B class address, then all the hosts appear to be organized into a single group, described by the network address. However, hosts within an organization are typically grouped together to form a number of different LANs. In order to distinguish the LANs the suffix of the IP address is subdivided into a subnet part and a host part. Each LAN is assigned a subnet address carried in the subnet part, and a host in the LAN is assigned an address which is carried in the host part. The actual parsing of the suffix in these two sub-fields is dictated by a subnet mask. The subnet mask is only known to the routers within the network since the subnets are not visible outside the network. This technique is known as *subnetting*.

Classless inter-domain routing (CIDR)

In the early 90's it became apparent that the rapid expansion of the Internet will cause a depletion of IP addresses and an explosion of the routing tables. The main cause for the address depletion was the wasteful usage of class B addresses. Typically, an organization may have a class B address but it may only have a small number of hosts, thus leaving the host address space largely unused. Also, the routing table explosion was due to the fact that a router is obliged to keep all the addresses of all the registered networks.

In order to alleviate these two problems the *classless inter-domain routing* (CIDR) scheme was proposed. This scheme permits the assignment of contiguous class C addresses and at the same time it reduces the number of entries required in a routing table.

The basic idea in CIDR is to allocate blocks of class C network addresses to each ISP. Organizations using the ISP are sub-allocated a block of 2^n contiguous addresses. For instance, if an organization requires 2000 addresses, then it will be allocated a block of 2048 or 2^8 contiguous class C addresses.

Hierarchical sub-allocation of addresses in this manner implies that clients with addresses allocated out of a given ISP will be routed via the ISP's network. This permits all these addresses to be advertised outside the ISP's network in an aggregate manner. As an example, let us assume that an ISP was allocated 131,072 class C network addresses starting at 194.0.0.0. That means that the lowest network address is 194.0.0.0 or 11000010 00000000 00000000 00000000 and the highest network address is 195.255.255.255 or 11000011 11111111 11111111 11111111. We observe that any address whose first seven bits are 1100001 belongs to the group of addresses allocated to the ISP. This *prefix* can be calculated by performing a bit-wise AND operation between the lowest address and the mask 254.0.0.0 or 11111110 00000000 00000000 00000000. Routers outside the ISP's network are provided, therefore, only with the base address 194.0.0.0 and the mask 254.0.0.0. This information suffices in order to identify whether an address of an IP packet has the same prefix as the ISP. The operation of calculating a prefix using a base network address and a mask can be seen as the opposite of subnetting and it is known as *supernetting*.

The above use of contiguous addresses gives rise to better usage of the address space. Also, by only advertising a base address and a mask, we minimize the amount of information that a router has to keep in its routing table. We note that there are network addresses that were allocated prior to the introduction of CIDR, and a router has to keep these addresses in its table as well.

In order to further simplify routing, blocks of addresses were also allocated according to geographic regions as shown in table 2.3.

Region	Lower address	Higher address
Europe	194.0.0.0	195.255.255.255
North America	198.0.0.0	199.255.255.255
Central/South America	200.0.0.0	201.255.255.255
Pacific Rim	202.0.0.0	203.255.255.255

Table 2.3: Allocation of addresses per region

Finally, we note that the class A, B, and C addresses are no longer used for routing. Instead CIDR is applied to all addresses, which explains why this scheme is called classless.

2.8.3 ARP, RARP and ICMP

The TCP/IP protocol suite includes other protocols such as the *address resolution protocol* (ARP), the *reverse address resolution protocol* (RARP) and the *internet control message protocol* (ICMP).

ARP is used to translate a host's IP address to its corresponding hardware address. This address translation is known as *address resolution*. The ARP standard defines two basic messages, namely a *request* and a *response*. A request message contains an IP address and requests the corresponding hardware address. A reply message contains the IP address sent in the request and the hardware address.

RARP does the opposite to ARP. It identifies the IP address of a host that corresponds to a known hardware address.

ICMP defines several error and information messages used in the Internet to report various types of errors or send various types of information. Some of the principal

messages are: *source quench*, *time exceeded*, *destination unreachable*, *redirect*, *fragmentation required*, *parameter problem*, *echo request/rep*ly, and *timestamp request/reply*.

A *source quench* message is sent by a router when it has run out of buffer space and it cannot accept more datagrams. A *time exceeded* message is sent by a router when the time to live field in a datagram is zero. The datagram is dropped by the router. The same message is also used by a host if the reassembly timer expires before all fragments from a given datagram have arrived. A *destination unreachable* message is sent by a router to a host that created a datagram, when it decides that the datagram cannot be delivered to its final destination. A *redirect message* is sent by a router to the host that originated a datagram, if the router believes that the datagram should have been sent to another router. A *fragmentation required* message is sent by a router to the host of a datagram, if it finds that the datagram is larger than the *maximum transfer unit* (MTU) of the network over which it must be sent. The datagram is rejected by the router. A *parameter problem* message is used to indicate that an illegal value has been discovered in the IP header of a datagram. *Echo reply* and *echo request* is used to test if a user destination is reachable and alive. *Timestamp request* and *timestamp reply* are similar to the echo request/reply messages except that the arrival time of the request message and the departure time of the reply message are also recorded.

2.8.4 IP version 6 (Ipv6)

Due to the rapid growth of the Internet, it was felt that the address space of the current IP will soon be inadequate to cope with the demand for new IP addresses. This consideration coupled with the need to provide new mechanisms for delivering real-time traffic, such as audio and video, led to the development of a new IP, known as IPv6.

IPv6 retains many of the basic concepts from IPv4. The new features are 128-bit addresses, new header format, extension headers, support for audio and video, and extensible protocol.

Problems

1. Explain why for bursty traffic, packet switching is preferred to circuit switching.

2. Consider the use of 1000-bit frames on a 1 Mbps satellite channel with a 270 msec delay (one-way propagation delay). What is the maximum link utilization for
   a. Stop-and-wait flow control?
   b. Flow control with a sliding-window of 7?
   c. Flow control with a sliding-window of 127?

3. Let us see if you remember polynomial or binary division! Calculate the FCS for the following two cases:
   a. P = 110011 and M = 11100011
   b. P = 110011 and M = 1110111100

4. Consider the following bit stream that has been bit-stuffed: 0111101111100111110100. What is the output stream? (The bit stream does not contain delimiters).

5. Stations A communicates with station B using the HDLC protocol over a 56 Kbps link. A transmits frames to B continuously back-to-back, and the transmission is error-free. The payload of each frame is exactly 1500 bytes. What percent of the bandwidth of the link is used for overheads (i.e., the HDLC header and trailer).

6. In the DS-1 format, what is the control signal data rate for each voice channel?

7. In X.25, why is the virtual-circuit number used by one DTE of two communicating DTEs different from the virtual-number used by the other DTE? After all, it is the same full-duplex virtual circuit!

8. Make-up an arbitrary IP header and calculate its checksum. Now, introduce errors in the bit stream (i.e. flip single bits) so that the checksum when calculated by the receiver will fail to detect that the IP header has been received in error.

9. Consider the IP address: 152.1.213.156.
   a. What class address is it?
   b. What is the net and host address in binary

10. A class B IP network has a subnet mask of 255.255.0.0.
    a. What is the maximum number of subnets that can be defined?
    b. What is the maximum number of hosts that can be defined per subnet?

CHAPTER 3
Frame Relay

Frame relay was originally defined by ITU-T as a network service for interconnecting *Narrowband ISDN* (N-ISDN) users. However, it very quickly became a stand-alone protocol when manufacturers of communications equipment realized its potential for transmitting data over wide area networks. It is very popular, and it is typically offered by public network operators, although it has also been implemented in private networks.

Frame relay has many common features with ATM networks, such as, layer 2 switching, no error or flow control between two adjacent nodes, and a feedback-based congestion control scheme which is similar in spirit to the *available bit rate* (ABR) scheme in ATM networks.

We first discuss the motivation behind the development of frame relay and its basic features. Then, we describe the *frame relay UNI*, and we conclude this Chapter with a discussion on congestion control. The topic of congestion control is examined in detail in Chapter 7.

3.1 Motivation and basic features

During the '80s, a number of significant changes took place in the environment in which communications networks operate. Transmission facilities were replaced by digital circuits based on optical fiber. The user equipment evolved from the dumb terminal to a powerful workstation which was capable of running large networking software and which could be directly attached to a LAN or a WAN. In parallel with these technological advances, there was an increasing demand for new bandwidth-thirsty applications, such

as moving large images and video conferencing. These new applications involved bursty traffic and required higher throughputs and faster response times.

Bringing all the above trends together, it became evident that in order to support the rapid transfer rates imposed by many new applications, and to provide the required response times, new high-speed WANs were needed. Due to the bursty nature of this traffic, these new networks should be based on packet switching rather than circuit switching. Frame relay and ATM networks are two such networks. Frame relay was originally defined to run over T1 and E1 links, and ATM networks were originally defined to run over OC-3 links, i.e., 155.52 Mbps. Both frame relay and ATM networks are connection-oriented packet switching networks. Frame relay was derived by removing and rearranging some of the functionality provided by some of the OSI protocol layers. ATM networks has many similarities to frame relay, but as will be seen in the next Chapter, they have a radically different architecture to frame relay and IP networks.

Older communication systems were designed based on the concept that the links between nodes were inherently unreliable. The ARQ mechanism in the data link layer was devised so that to guarantee an error-free transfer of packets over an unreliable link. The advent of fiber optics, however, made the data link layer ARQ mechanism redundant since fiber optics links introduce very few errors. Of course, there is no guarantee that the transmissions over a fiber optics link will be always error-free. Since it was not anticipated that there will be many retransmissions, it was felt that a considerable speed-up can be gained by removing the hop-by-hop ARQ scheme and simply rely on the end-devices, which had become quite intelligent, to recover packets that were either erroneously received or lost. Specifically, the recovery of these packets was left to the higher protocol layers, such as TCP, which run at the end-devices. This feature was implemented in both frame relay and ATM networks.

Another interesting development had to do with the speed of computers in relation to the speeds of the communication links in a WAN. In earlier computer networks, the

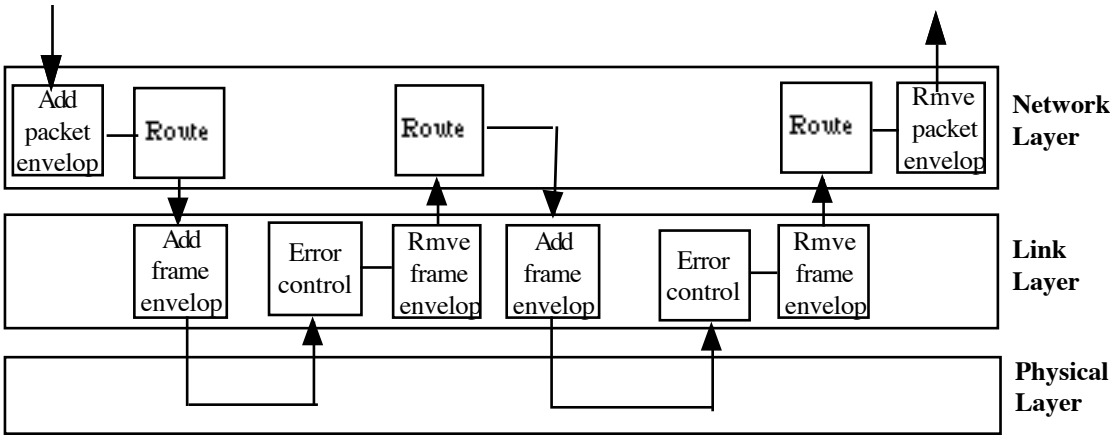


Figure 3.1: Typical flow in X.25

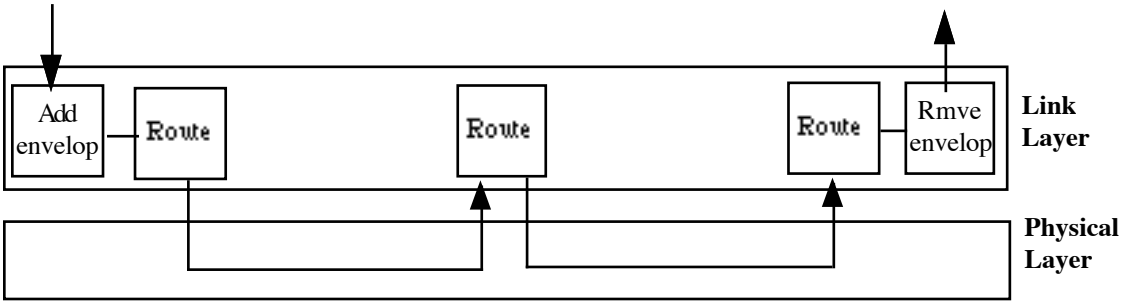


Figure 3.2 Frame relay

communications links were very slow. As a result, a node, which was basically a computer system, could run the necessary networking protocols fast enough so that it did not delay the transmission of packets on the network links. In view of this, the network links were the bottleneck in the network. With the advent of fiber-based fast transmission links, the bottleneck shifted from the communication links to computers. That is, the software that executed the necessary protocol layers could not run fast enough to keep up with the new transmission speeds. In frame relay, significant software speed-ups were achieved by moving some of the functionality provided by layer 3 to layer 2. Specifically, the routing decision as to which node a packet should be forwarded to, which in X.25 and in IP networks is typically carried out in layer 3, was moved down to layer 2.

In figure 3.1, we show the path through the first three protocol layers that a packet follows as it gets switched through the X.25 network. This is contrasted in figure 3.2, where we show how switching is accomplished in frame relay. As can be seen in figure

3.1, the switching decision is made at the network layer and the error control and recovery is done at the link layer. The two layers introduce their own encapsulation overheads, and passing a packet from one layer to another requires moving it from one buffer of the computer's memory to another, a time-consuming process.

In frame relay, as shown in figure 3.2, there is no ARQ at each hop and there is no processing at layer 3. Switching is carried out within the data link layer, thus avoiding additional overheads in memory transfers and encapsulation. A frame relay packet, known as a *frame*, is discarded by a frame relay node if its header is found in error. Also, a frame can be lost if it arrives at the input buffer of a frame relay node at a time when the buffer is full. We recall that in a network equipped with hop-by-hop ARQ, erroneously received packets or lost packets in a given hop, are recovered by the local data link layer. In frame

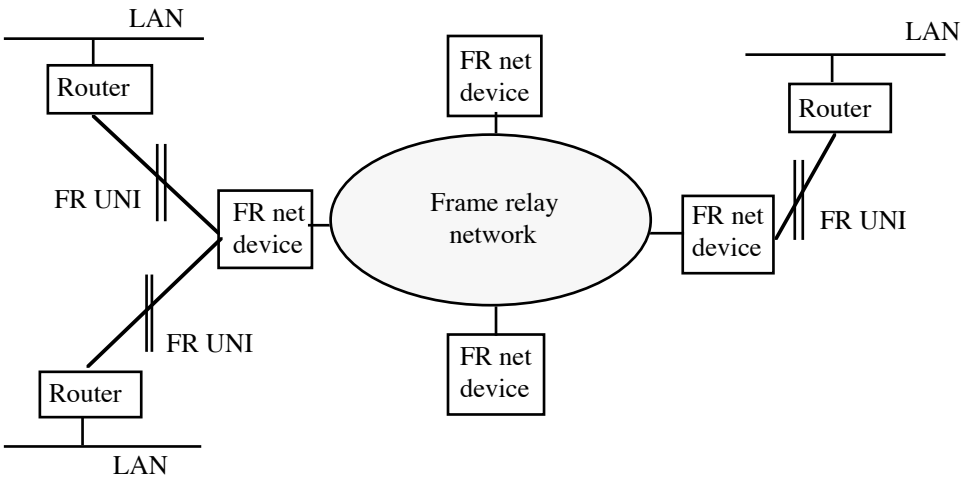


Figure 3.3: An example of the location of the frame relay UNIs

relay, discarded frames are only recovered by the end-user. As a result of these changes, frame relay provides a very efficient transport mechanisms.

As will be seen in the next Chapter, ATM networks have been built on similar principles.

3.2 The frame relay UNI

Frame relay is an extension of the *link access procedure for the D channel* (LAP-D) and it is based on the core functions of ITU-T's recommendation Q.9221 (Q.992 core). (LAP-D has a similar format to LAP-B, which is a subset of HDLC). LAP-D was chosen because virtually every data protocol, such as SNA, X.25, TCP/IP, and DECnet, can be easily made to conform to it. Thus, frame relay is compatible with a variety of higher layer protocols.

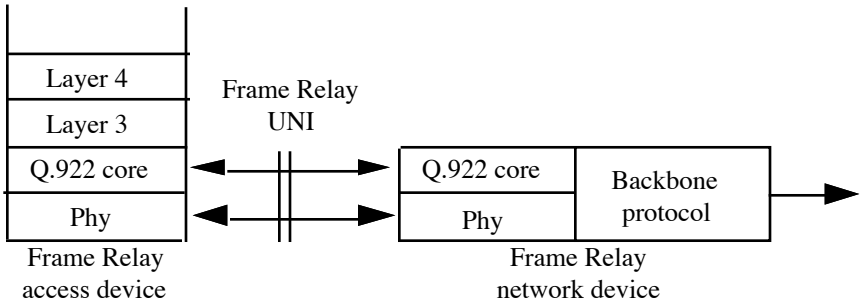


Figure 3.4: The protocol stack for the frame relay UNI

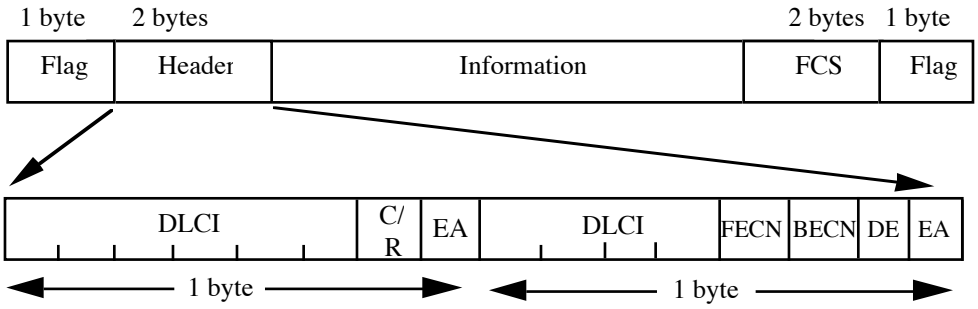


Figure 3.5: The structure of the frame relay frame

Frame relay is a standardized *user-network interface* (UNI) protocol which defines the interaction between *frame relay access devices* (FRAD), such as routers, bridges, and other control devices, and a nodal processor which is known as the *frame relay network device*. Frame relay does not define how the nodal processors are interconnected. An example of the location of frame relay UNIs is given in figure 3.3

The protocol stack is shown in figure 3.4. The structure of the frame is shown in figure 3.5. The following fields have been defined:

Flag: As in HDLC, a flag consisting of the bit pattern 01111110 is used as a delimiter. Bit stuffing is used to avoid duplication of the bit pattern inside the frame.

Data link connection identifier (DLCI): Frame relay is a connection-oriented protocol. That is, a virtual connection has to be established between the sender and the receiver before data can be transferred. The 10-bit DLCI field is used to identify a virtual connection, as explained below.

Command/response (C/R): 1-bit field used to carry application-specific information. It is carried transparently by the network.

Forward explicit congestion notification (FECN): 1-bit field used for congestion control, as explained in section 3.5.

Backward explicit congestion notification (BECN): 1-bit field used for congestion control, as explained in section 3.5.

Discard eligibility (DE): 1-bit field used for congestion control. The use of this bit is explained in section 3.5.

Address extension (EA): The basic header of frame relay is two bytes. It can be extended to 3 or 4 bytes so that to support DLCIs with more than 10 bits. The 1-bit EA field is used to indicate whether the current byte in the header is the last one. For instance, in a 2 byte header, EA will be set to 0 in the first byte, and to 1 in the second byte.

FCS: This field contains the frame check sequence obtained using the pattern $x^{16}+x^{12}+x^5+1$.

The length of the information field has to be an integer number of bytes before bit-stuffing, with a minimum size of 1 byte. The frame relay Forum recommends a defaulted maximum value of 1600 bytes, and ANSI recommends up to 4,096 bytes.

Frame relay is an extremely simple protocol. The receiver checks the integrity of each frame using the FCS. The frame is discarded if it is found to be in error. If no errors are found, the frame's DLCI is looked up in a table. The frame is discarded if its DLCI number is not found in the table. Otherwise, the frame is relayed towards its destination. Valid frames can also be discarded in order to alleviate congestion. When a frame is discarded, the end-devices are not notified. A discarded frame will eventually be noted by

the destination end-device, which will then request the sending end-device to retransmit the frame. This re-transmission procedure is not part of the frame relay protocol, and it should be done by a higher-level protocol.

DLCI: Data link connection identifier

All DLCIs have local significance. That is, they are only used to identify a virtual circuit connection between a frame relay access device and the nodal frame relay network device. The same number can be used for another connection at another UNI. This concept of using locally significant connection identifiers is also found in X.25 and in ATM networks.

In the example given in figure 3.6, there are two connections between end-devices A and B, and end-devices A and C, shown by a solid line. The connection between A and B is identified by DLCI=25 on A's UNI and by DLCI=20 on B's UNI. The connection between A and C is identified by DLCI=122 on A's UNI and by DLCI=134 on C's UNI. If A wants to send a frame to B, then it will populate the frame's DLCI field with the number 25. The frame relay network will deliver the frame to B with the DLCI field set to 20. Likewise, if B wants to transmit a frame to A, it will populates the frame's DLCI field with the number 20, which will be delivered to A with a DLCI equal to 25.

Frame relay connections can be permanent virtual connections (PVC) or switched virtual connections (SVC). PVCs are set-up administratively using network management procedures, whereas SVCs are set up on demand using the ITU-T Q.933 signalling protocol.

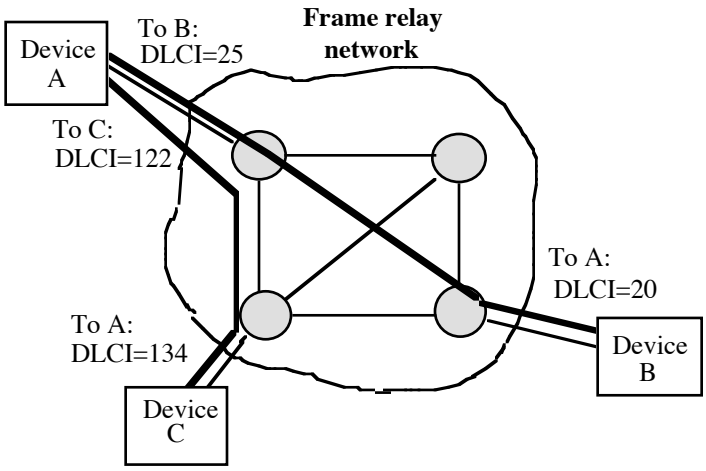


Figure 3.6: An example of DLCI addressing

3.3 Congestion control

We recall that in a frame relay network there is no ARQ mechanism in the data link layer. Because of this, a frames will get lost if it arrives at a node during the time when the node's input buffer is full. The problem, therefore, is how to ensure a minimal frame loss while at the same time utilize the network as much as possible. The same problem arises in ATM network, see Chapter 7.

The data rate, expressed in bits per second, of the user's physical access channel to the frame relay network is known as the *access rate*. This is a physical upper bound on the rate at which the user sends traffic to the network, and it may take values such as 64 Kbps, 128 Kbps, and 1.544 Mbps.

The traffic that a user sends to a frame relay network is described by the following three parameters: *committed burst size* (B_c), *excess burst size* (B_e), and *committed information rate* (CIR). The committed burst size B_c defines the maximum amount of bits a user may submit to the network during a pre-determined time-interval T_c. The number of bits submitted by the user in a time-interval T_c can be exceeded by up to a certain value defined by the excess burst size B_e. Finally, the committed information rate CIR is the data transmission rate, expressed in bits per second, that the frame relay network is committed to transfer, and it is calculated as the average transmission rate over the time interval T_c. These three traffic parameters are negotiated between the user

and the frame relay network at set-up time. The interval T_c is calculated from B_c and CIR as follows: $T_c = B_c/CIR$.

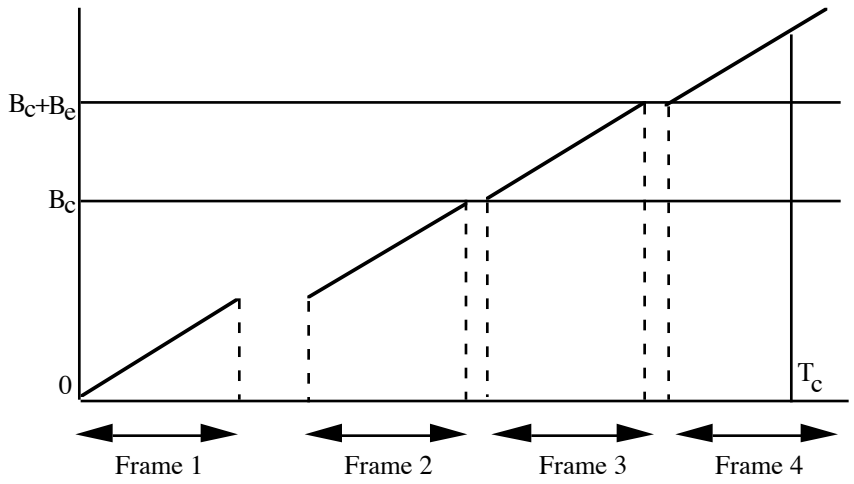


Figure 3.7: Policing the amount of traffic submitted by the user

The network polices continuously the amount of data submitted by the user during each interval T_c. At the beginning of each time-interval, the cumulative number of received bits is set to 0. When a frame arrives, the cumulative number of bits received is updated, and if it is less than or equal to B_c, the frame is allowed to enter the network and the network guarantees its delivery. However, if it exceeds B_e, the network will attempt to carry this frame without any guarantee of delivering it. Finally, if it exceeds B_c+B_e, then the frame is discarded. This policing is carried out on each successive time-interval, which are back-to-back and non-overlapping.

An important factor in this policing scheme is the DE (discard eligibility) bit in the frame. The DE bit is used to indicate whether a frame can be discarded in case when congestion arises in a node. This bit can be set by the user or by the network through the policing mechanism. In the latter case, it is used by the network to mark the excessive traffic, that is the traffic that exceeds the B_c value but it is still less or equal to B_c+B_e. The network marks the violating frames, by setting their DE to 1. If the network experiences congestion, it attempts to alleviate it by discarding all the frames whose DE=1. The DE bit is similar to the CLP bit used in ATM networks, see section 4.2.

An example of this policing mechanism is given in figure 3.7. Time is indicated on the horizontal axis, and the cumulative number of bits submitted to the network by the user in a single time-interval is indicated on the vertical axis. We note that on the horizontal axis, a single time-interval T_c is marked, and on the vertical axis there are two marks, one for B_c and another for B_c+B_e. The rate of transmission from the user to the network is equal to 1 bit per unit-time. The time it takes to transmit each frame and the cumulative number of

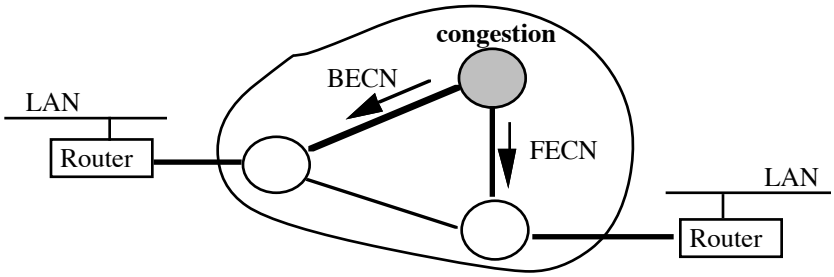


Figure 3.8: Backward and forward explicit congestion notification

received bits is indicated on the horizontal and vertical axes respectively. As can be seen the cumulative number of bits when frame 1 is received is less than B_c and consequently, the frame is allowed to enter the network and its delivery is guaranteed. Likewise, frame 2 will also enter the network and its delivery is guaranteed. Frame 3 also arrives within the same time-interval, but now the cumulative number of bits received is over B_c and exactly equal to B_c+B_e. In this case, frame 3 will be marked as a violating frame, that is, its DE bit will be set to 1, and will be allowed into the network, but its delivery is not guaranteed. Frame 4 also begins to arrive during the same time-interval, but since the total number of bits is over the limit, it will be simply rejected. This policing mechanism applies continuously to each successive time-interval. The time-intervals are back-to-back and they do not overlap. Also, we note that each connection is policed separately.

In addition to the above policing mechanism, the frame relay network employees a congestion control scheme inside the network, which is feedback-based. That is, when congestion occurs, the network requests the transmitting end-devices to reduce their transmission rate, or even to stop transmitting, for a period of time until the congestion

goes away. This scheme is known as the *explicit congestion notification*, and it could be either a *backward explicit congestion notification* (BECN), or a *forward explicit congestion notification* (FECN), as shown in figure 3.8.

In FECN, if congestion occurs at a node, as witnessed by the occupancy level in an output buffer, the node turns on the FECN bit of all the frames going out from this output buffer. (It is possible that the EFCN bit of these frames has already been turned on, if the frames have already experienced congestion at an upstream node, in this case, no further action is taken.) All downstream nodes along the paths followed by the frames will know that congestion has occurred in an upstream node. Finally, a receiving end-device will be notified that congestion has occurred in one or more upstream nodes since the FECN field of the frames that it will receive will be on. The receiving end-device can then slow down

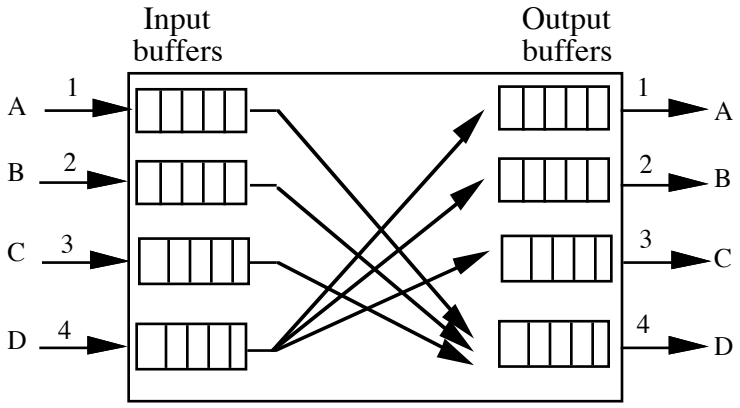


Figure 3.9: An unfolded generic switch

the transmitting end-device using a method, such as, reducing its window size, or delaying to send back acknowledgements. The action to slow down the transmitter, of course, will be taken by an upper layer protocol such as TCP. The FECN bit is the same as the EFCN bit in ATM networks, see section 4.2.

With some upper layer protocols, it is more efficient to notify directly the transmitting end-devices that congestion has occurred, rather than their receiving end-devices. This can be achieved using the BECN scheme. In this scheme, the node turns on the BECN bit of the frames going to the opposite direction, i.e., towards the transmitting

end-devices. All upstream nodes along the path followed by these frames will know that congestion has occurred in one or more nodes downstream, and eventually, a transmitting end-devices will be notified.

In order to clarify these two schemes, we consider a generic switch shown in figure 3.9, consisting of 4 input ports and 4 output ports, marked from 1 to 4. Each input port consists of an input link and an input buffer, where incoming frames are stored. Likewise, each output port consists of an output buffer and an output link. Incoming frames are processed by the switch, and then they are placed in the appropriate output buffer from where they get transmitted out. Processing of a frame involves the following two steps: a) carry out the CRC check to verify that the frame's header has been correctly received, and b) look-up the DLCI in a table to find out the new DLCI and the destination output port number from where the frame should be transmitted out. We note that figure 3.9 gives an *unfolded* view of the switch. That is, the input ports seems to be different from the output ports. In real-life, however, input port i and output port i are in fact a single duplex port.

Now. let us assume that end-devices A, B, C, and D are attached directly to this switch, as follows. A is attached to port 1, B to port 2, C to port 3, and D to port 4. That is, A transmits frames to the switch through input port 1, and receives frames from the switch from output port 1. Similarly, B transmits to the switch through input port 2, and receives frames from the switch through output port 2, and so on. We assume that A, B, and C have established separate connections to D. These connection are duplex. That is, user A sends frames to the input buffer 1, which are then transferred to the output buffer 4, from where they get transmitted out to user D. User D transmits frames to A in the reverse order. That is, it sends frames to the input buffer 4, from where they get switched to the output buffer 1 and then get transmitted out to user A. The same applies to the other connections.

We are now ready to explain the two congestion control schemes. Let us assume that the output buffer 4 gets congested. That is, the total number of frames in the buffer (received from A, B, and C) is over a pre-specified threshold, such as 70% of its total buffer capacity. (This threshold was selected arbitrarily for the sake of the example). In the FECN scheme, the switch will turn the FECN bit of all the frames going out from

output port 4. As a result, D will recognize that congestion has occurred upstream, and through its higher layer protocols will attempt to slow down A, B, and C.

In the BECN scheme, the switch will identify which connections utilize output port 4, and turn the BECN bit of the frames going to the opposite direction of these connections. That is, it will turn on the bit of the frames going from D to A, D to B, and D to C. End-devices A, B, and C will receive frames from D with their BECN bit turned on, they will deduce that congestion has occurred downstream from them, and they will reduce or stop their transmissions to D.

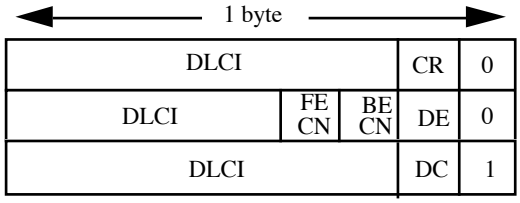
In order for these two schemes to work well, a number of parameters have to be tuned carefully, such as: a) the buffer threshold above which the switch should start turning on the FECN or BECN bit, b) how many frames with their FECN or BECN bit turned on should be received before some action should be taken, c) by how much a transmitting source should slow down, and d) a buffer threshold below which the switch stops turning on the FECN or BECN bit.

The consolidated link layer management

In order for the BECN scheme to work, there should be frames going back towards the transmitting end-devices. If there are no frames going in the reverse direction, a transmitting end-device cannot be informed that congestion has occurred somewhere in the network. In view of this, ANSI and ITU-T developed the *consolidated link layer management (*CLLM), an optional signalling mechanism for congestion control. A CLLM message contains a list of DLCIs that are likely to be causing congestion. Using such messages, the network can directly request that the owner of the DLCI temporarily suspends transmission. The CLLM message follows the same length restrictions as the regular frame relay frame, and, therefore, it may consist of more than one frame. CLLM uses DLCI 1023. The CLLM and BECN schemes may be used together or separately.

Problems

1. Describe the main features of frame relay.
2. Explain why in frame relay there is no processing at layer 3.
3. The structure of the frame relay 3-byte header is as follows:



Calculate the total number of possible DLCI values that can be defined with this header.

5. Give two reasons why it is preferable that the DLCI values have a local significance and not global.

6. A user is connected to a frame relay network over a 128 Kbps link. The user negotiates a CIR of 100 Kbps.
   a. How does the time-interval T_c changes as B_c changes from 12000 bits to 40,000 bits?
   b. Explain this behaviour of T_c.
   c. How is it possible for the user to submit to the network more than B_c for each time-interval?

7. Describe in your own words how the FECN and BECN schemes work.

PART TWO:
THE ATM ARCHITECTURE

Part Two deals with the main components of the architecture of ATM networks. Part Two consists of Chapters 4, 5, 6 and 7.

Chapter 4: Main Features of ATM Networks

In this Chapter, we describe some of the main features of ATM networks, such as the header of the ATM cell, the ATM stack, and the ATM interfaces. We also briefly examine the ATM physical layer and the various public and private physical layer interfaces.

Chapter 5: The ATM Adaptation Layer

The ATM adaptation layer is a protocol layer which is particular to ATM networks. Several different ATM adaptation layers have been standardized for different classes of traffic, and they are described in detail in this Chapter.

Chapter 6: ATM Switch Architectures

ATM switch architectures were designed specially to switch ATM cells at high speeds. Several different classes of ATM switch architectures are presented in this Chapter.

Chapter 7: Congestion Control in ATM Networks

In this Chapter, we discuss various congestion control schemes that are used in ATM networks to provide different quality of service to different types of traffic.

CHAPTER 4

Main Features of ATM Networks

In this Chapter, we present the main features of the ATM architecture. We start with a brief account of the considerations that lead to the standardization of the ATM packet, known as *cell*. Then, we describe the structure of the header of the ATM cell, the ATM protocol stack, and the various ATM interfaces that have been standardized. Finally, we describe the physical layer and various public and private physical layer interfaces.

4.1 Introduction

The *Asynchronous Transfer Mode* (ATM) is the preferred architecture for Broadband ISDN (B-ISDN). The term *Asynchronous Transfer Mode* is in contrast to *Synchronous Transfer Mode* (STM), which was proposed prior to the standardization of ATM and which is based on the PDH hierarchy. The term *transfer mode* means a telecommunication technique for transferring information.

The ATM architecture was designed with a view to transmitting voice, video, and data on the same network. These different types of traffic have different tolerance for packet loss and end-to-end delay, as shown in table 4.1. For instance, packets containing voice have to be delivered on time so that the play-out process at the destination does not run out of data. On the other hand, the loss of some data may not necessarily deteriorate the quality of the voice delivered at the destination. At the other extreme, when transporting a data file, loss of data cannot be tolerated since this will compromise the file's integrity, but there is no stringent requirement that the file should be delivered as fast as possible.

ATM was standardized by ITU-T in 1987. It is based on the packet-switching principle and it is connection-oriented. That is, in order for a sender to transmit data to a receiver, a connection has to be established first. The connection is established during the call set-up phase, and when the transfer of data is completed, it is torn down. ATM, unlike IP networks, has built-in mechanisms for providing different quality-of-service to different types of traffic.

	packet loss sensitive	Delay sensitive
voice	low	high
video	moderate	high
data	high	low

Table 4.1: Different tolerances for different types of services

As in frame relay, there is neither error control nor flow control between two adjacent ATM nodes. Error control is not necessary since the links in a network have a very low bit-error rate. In view of this, the payload of a packet is not protected against transmission errors. However, the header is protected in order to guard against forwarding a packet to the wrong destination! The recovery of a lost packet or a packet that is delivered to its destination with erroneous payload is left to the higher protocol layers. The lack of flow control requires congestion control schemes that permit the ATM network operator to carry as much traffic as possible without losing too many cells. These congestion control schemes are examined in Chapter 7.

The ATM packet is known as *cell*, and it has a fixed size of 53 octets. It consists of a payload of 48 octets and a header of 5 octets, as shown in figure 4.1. It was originally envisioned that the links in an ATM network will have a minimum speed of 155.52 Mbps, which corresponds to an OC-3 link. At such high speeds, an ATM switch does not have much time to process the information in the header. In view of this, the header provides a limited number of functions, such as: virtual connection identification, a limited number of congestion control functions, payload-type indication, and header error control.

Several consideration led ITU-T to decide on a packet size of 53 octets. Below, we examine some of them.

Delay through the ATM network

In the early days of the standardization process of ATM, which was around the mid-80s, it was assumed that the incoming and outgoing buffers of an ATM switch will be small, so

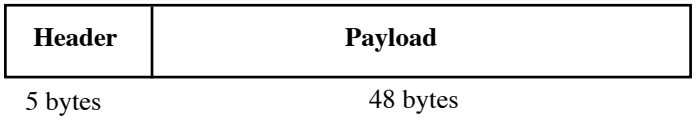


Figure 4.1: The ATM cell

that the queueing delay in a buffer will be also small. Telephone operators and equipment providers, who had a significant influence in the early standardization process of ATM, felt that small buffers were necessary in order to guarantee quick delivery of packets containing voice. (As will be seen in Chapter 6, this is no longer necessary and in fact currently ATM switches are equipped with very large buffers.) An argument, therefore, in favour of short packets had to do with the queueing delay of a packet in a buffer. If packets are short and buffers are small, then the queueing delay of a packet in a switch is also small.

Other arguments in favour of short packets were centered round the packetization delay in a real-time application. If a real-time application generates data at a rate which is very slow compared to the transmission speed of the network, then it will take a long time to fill up a packet, particularly if the packet has a large size. For instance, let us consider the transmission of 64 Kbps voice over ATM. We recall from section 2.5, that the voice signal is sampled 8,000 times per second, which gives rise to 8,000 bytes per second, or one byte every 125 μsec. If the packet size is 16 bytes, then it will take 16x125 μsec, or 2 msec to fill up a packet. If the packet size is 64 bytes, then it will take 64x125 μsec, or 8 msec to fill up a packet. So, the smaller the packet size, the less the delay to fill up a packet. Of course, the packetization delay can be kept low, if a packet is partially filled. This, however, may result in lower utilization of the network.

Echo cancellation

In order to avoid echo problems for voice, the end-to-end delay has to be kept small. ITU-T requires the use of echo cancellers if the delay is greater than 24 msec. This was another argument in favour of short packets.

Header conversion

The main function of an ATM switch is to transfer cells from its input ports to its output ports. When a cell arrives at an ATM switch on an input port, the switch has to find out its destination output port. This is done by using the connection identifier, carried in the cell's header, in a table in order to obtain the destination output port and a new connection identifier. This mechanism is known as header conversion, and it will be explained later on in this Chapter in section 4.2. Header conversion has to be done on the fly.

An argument in favour of longer packets was made based on the fact that the longer the packet, the more time the switch has to do the header conversion. For instance, let us assume that an ATM switch is equipped with links that transmit at OC-3, i.e., 155.52 Mbps. If the cell size is 53 bytes, then a maximum of about 365,566 cells can arrive per second. This translates to 2.7 μsec per cell. That is, assuming that cells arrive back-to-back, a new cell arrives approximately every 2.7 μsec. This means that the switch has 2.7 μsec available to carry out the header conversion. Now, let us assume a cell size of 10 bytes. Then, a maximum of about 1,937,500 cells can arrive per second. That is, if cells arrive back-to-back, a new cell arrives approximately every 0.5 μsec. In this case, the switch has only 0.5 μsec for the header conversion.

Fixed vs variable packet length

The question as to whether the ATM packet should be of variable or fixed size was also debated at ITU-T. An argument in favour of variable-size packets had to do with the overhead introduced by the header. A variable-size packet was anticipated to be much larger than the small fixed-size packet that was being considered by ITU-T. Therefore,

the overhead due to the header would have been much lower for variable-size packets than for fixed-size packets.

Despite this argument in favour of variable-size packets, fixed-size packets were deemed preferable because it was less complex to design ATM switches for fixed-size packets than for variable-size packets.

The compromise!

There was an agreement in ITU-T that the ATM packet should be of fixed size with a length between 32 and 64 bytes. The European position was in favour of 32 bytes since there will be no need for echo cancellers. The USA/Japan position was in favour of 64 bytes due to transmission efficiencies. In ITU-T SGXVIII meeting in 1989, the members agreed to compromise to 48 byte payload!

4.2 The structure of the header of the ATM cell

Two different formats for the cell header were adopted, one for the *user network interface* (UNI) and a slightly different one for the *network-network interface* (NNI). The UNI is concerned with the interface between an ATM end-device and the ATM switch to which it is attached. An ATM end-device is any device that can be attached directly to an ATM network and it can transmit and receive ATM cells. The NNI is used between two ATM switches belonging to the same network or to two different networks.

The format of the cell header for these two interfaces is shown in figure 4.2. As we can see, these two formats only differ in the first field. We now proceed to discuss in detail each field in the header. Understanding the meaning of these fields helps to better understand the underlying network architecture.

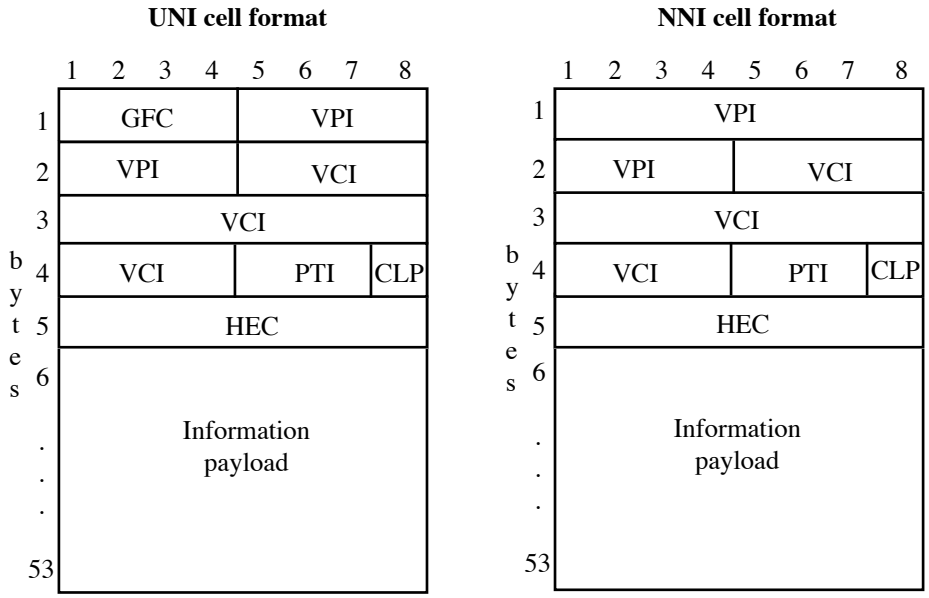


Figure 4.2: The structure of the cell header

4.2.1 Generic flow control (GFC)

This field permits multiplexing of transmissions from several terminals on the same user interface. It is used to control the traffic flow from the end-device to the network.

4.2.2 Virtual path identifier / virtual channel identifier (VPI/VCI)

As mentioned in the previous section, ATM is connection oriented. An ATM connection is identified by the combined *virtual path identifier* (VPI) and *virtual channel identifier* (VCI). Such a connection is referred to as a *virtual channel connection* (VCC). The VPI/VCI field is 24 bits in the UNI interface and 28 bits in the NNI interface. The VPI field is 8 bits in the UNI interface and 12 bits in the NNI interface. Therefore, in a UNI interface we can have a maximum of 256 virtual paths, and in a NNI interface we can have a maximum of 4,096 virtual paths. In each interface we can have a maximum of 65,536 VCIs. A VPI can be assigned to any value from 0 to 255. VCI values are assigned as follows: 0 to 15 are reserved by ITU-T, 16-31 are reserved by the ATM Forum, and 32 to 65535 are used for user VCCs.

The combined VPI and VCI allocated to a connection is known as the *connection identifier* (CI). That is, CI = {VPI,VCI}.

A virtual channel connection between two users consists of a path through a number of different ATM switches. For each point-to-point link that lies on this path, the

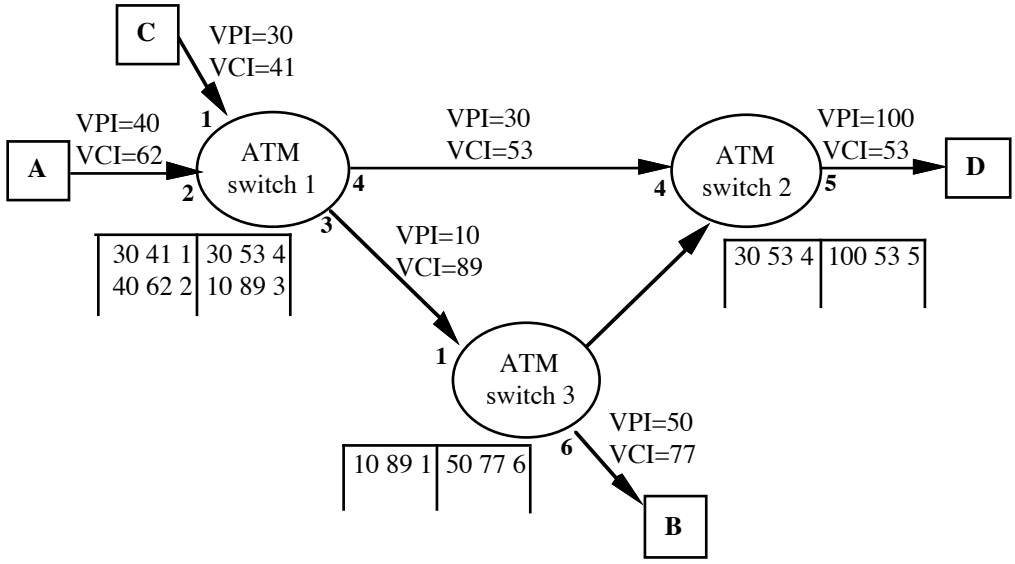


Figure 4.3: An example of label swapping

connection is identified by a different VPI/VCI. That is, each VPI/VCI has *local significance* and it is translated to a different VPI/VCI at each switch that the cell traverses. This operation is referred to as header conversion, or as *label swapping*, since the connection identifier is also known as a *label*. Label swapping (or header conversion) involves a look-up in the switching table of the ATM switch. The VPI/VCI of the incoming cell is indexed into the switching table and the new VPI/VCI that the cell will carry in its header on the outgoing link is obtained. At the same time, the output port number is also obtained, so that the switch knows to which output port to forward the cell. As mentioned above, labels have only local significance, that is they are only used on a single link. This simplifies the selection process of a new label for particular link.

An example of label swapping is given in figure 4.3. Each switch is represented by a circle, and the switching table is given immediately below the circle. We assume that the switching table is centralized and it contains information for all input ports. The first column in the switching table gives the VPI/VCI of each incoming connection and its input port. The second column gives the new label and the destination output port of

each connection. Let us follow the path from A to B that traverses through ATM switches 1 and 3. We see that on the incoming link to ATM switch 1, the connection has the label VPI=40, VCI=62. From the switching table we find that its new label is VPI=10, VCI=89 and it should be switched to output port 3 of ATM switch 1. At ATM switch 3, we see that the connection's new label on the outgoing link is VPI=50, VCI=77, and its destination output port is 6. Therefore, the path from A to B, consists of the following three different labels: VPI/VCI=40/62; VPI/VCI=10/89; and VPI/VCI=50/77. We can also identify the input and output ports of each switch, through which the connection is established. We see that it enters ATM switch 1 at input port 2, exits from the same switch from output port 3, enters ATM switch 3 at input port 1, and finally exits from output port 6. Similarly, we can follow the path from C to D.

ATM connections are point-to-point and point-to-multipoint. Point-to-point connections are bi-directional, and point-to-multipoint connections are unidirectional. An ATM connection, depending upon how it is set-up, maybe either a *permanent virtual connection* (PVC) or a *switched virtual connection (*SVC). A PVC is established manually by a network administrator using network management procedures. Typically, it remains in place for a long period of time. An SVC is established in real-time by the network using signalling procedures, and it remains up for an arbitrary amount of time. The signalling protocols used to establish and release a point-to-point and point-to-multipoint SVC are described in Chapters 10 and 11.

A point-to-point SVC is established when an end-device A sends a SETUP message to the switch to which it is attached, known as the *ingress* switch, requesting that a connection is established to a destination end-device B. The ingress switch calculates a path through the network to B, and then it forwards the set-up request to the next-hop switch on the path, which forwards it to its next-hop switch and so on until the set-up request reaches the switch to which B is attached, known as the *egress* switch. This last switch sends the set-up request to B, and if it is accepted, a confirmation message is sent back to A. At that time, A may begin to transmit data. Each switch on the path allocates some bandwidth to the new connection, selects a VPI/VCI label, and updates its switching table. When the connection is terminated, each switch removes the entry in the

switching table associated with the connection, the VPI/VCI label is returned to the pool of free labels, and the bandwidth that was allocated to the connection is released.

Each switch on the path of a new connection has to decide independently of the other switches whether it has enough bandwidth to provide the quality-of-service requested for this connection. This is done using a *call admission control* (CAC) algorithm. CAC algorithms are examined in detail in Chapter 7.

In addition to permanent and switched virtual connections, there is another type of connection known as a *soft PVC*. Part of this connection is permanent and part of it is switched. The connection is set-up using both network management procedures and signalling procedures.

PTI	Meaning
000	User data cell, congestion not experienced, SDU type=0
001	User data cell, congestion not experienced, SDU type=1
010	User data cell, congestion experienced, SDU type=0
011	User data cell, congestion experienced, SDU type=1
100	Segment OAM flow-related cell
101	End-to-end OAM flow-related cell
110	RM cell
111	Reserved

Table 4.2: The payload type indicator (PTI) values

4.2.3 Payload type indicator (PTI)

The field for the payload type indicator is 3 bits. It is used to indicate different types of payloads, such as user data and OAM. It also contains a bit used for explicit congestion control notification (EFCN) and a bit used in conjunction with the *ATM adaptation layer 5*. The explicit congestion control notification mechanism is similar to the FECN scheme for frame relay, discussed in section 3.3, and it is used in the ABR scheme described in section 7.8.1. Also, the ATM adaptation layer 5 is described in detail in section 5.5. Table 4.2 summarizes the PTI values.

Bit 3, which is the leftmost and most significant bit, is used to indicate if the cell is a user data cell (bit set to 0), or an *operations, administration, maintenance* (OAM) data cell (bit set to 1). For a user data cell, bit 2 carries the explicit forward congestion

indicator. It is set to 0 if no congestion has been experienced, and to 1 if congestion has been experienced. Also, for a user data cell, bit 1 is used by the ATM adaptation layer 5. It is set to 0 if the *service data unit* (SDU) type is 0, and to 1 if the SDU type is 1.

For OAM data cells, two types are defined. In addition, a *resource management* (RM) cell is defined which is used in conjunction with the *available bit rate* (ABR) mechanism, a feedback-based congestion control mechanism described in Chapter 7.

4.2.4 Cell loss priority bit (CLP)

The *cell loss priority* (CLP) bit is used to indicate whether a cell can be discarded when congestion arises inside the network. If a cell's CLP bit is 1, then the cell can be discarded. On the other hand, if the cell's CLP bit is 0, then the cell cannot not be discarded.

The CLP bit is similar to the DE bit in frame relay, and its use for congestion control in ATM networks is discussed in Chapter 7.

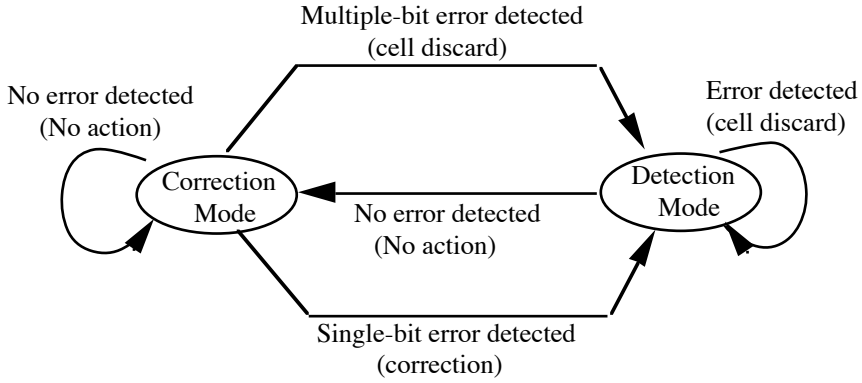


Figure 4.4: The header error control state machine

4.2.5 Header error control (HEC)

The header error control (HEC) field is used to detect single-bit or multiple-bit transmission errors in the header. CRC is used with a 9-bit pattern given by the polynomial x^8+x^2+x+1. The HEC field contains the 8-bit FCS obtained by dividing (the first 32 bits of the header)x2^8 by the above pattern.

The state machine that controls the head error correction scheme is shown in figure 4.4. It is implemented in the physical layer, as described in section 4.5.

At initialization, the receiver's state machine is set to the *correction mode*. Each time a cell arrives, the CRC is carried out. If no errors are found, the cell is allowed to proceed to the ATM layer and the state machine remains in the correction mode. If a single-bit error is detected, then the error is corrected and the cell is allowed to proceed to the ATM layer, but the state machine switches to the *detection mode*. If a multi-bit error is detected, the cell is discarded and the state machine switches to the detection mode. In detection mode, each time a cell comes in, the CRC is carried out, and if a single-bit or a multi-bit error is detected, the cell is discarded and the state machine remains in the detection mode. If no errors are detected, then the cell is allowed to proceed to the ATM layer and the state machine shifts back to the correction mode.

4.3 The ATM protocol stack

The ATM protocol stack is shown in figure 4.5. It consists of the physical layer, the ATM layer, the ATM adaptation layer, and higher layers that permit various applications to run on top of ATM. It is important to note that the ATM layer and the ATM adaptation layer

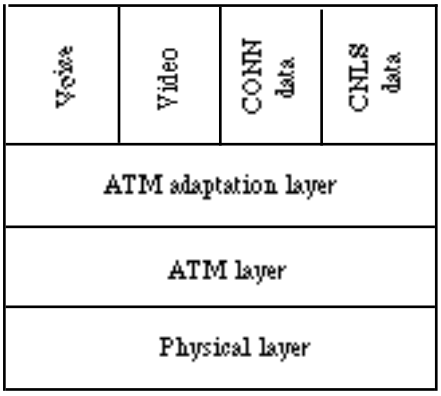


Figure 4.5: The ATM protocol stack

do not correspond to any specific layers of the OSI reference model, and it is erroneous to refer to the ATM layer as the data link layer.

The ATM stack shown in figure 4.5 is for the delivery of data. A similar stack is given in Chapter 10 for the ATM signalling protocols.

4.3.1 The physical layer

The physical layer transports ATM cells between two adjacent ATM layers. It is subdivided into the *transmission convergence* (TC) sublayer and the *physical medium-dependent* (PMD) sublayer.

The TC sublayer interacts with the ATM layer and the PMD sublayer. On the transmitter's side, it receives ATM cells and generates a stream of bits that passes on to the PMD sublayer. On the receiver's side, it recovers ATM cells from the stream of bits and passes them on to the ATM sublayer. The PMD sublayer on the transmitter's side is concerned with the transmission and the transport across a link of a bit stream received from the TC sublayer. On the receiving side, it detects and recovers the arriving bit stream and passes it to the TC sublayer.

The physical layer and various public and private ATM physical layer interfaces are described below in section 4.5.

4.3.2 The ATM layer

The ATM layer is concerned with the end-to-end transfer of information, i.e., from the transmitting end-device to the receiving end-device. Below, we summarize its main features.

Connection-oriented packet switching

The ATM layer is a connection-oriented packet-switched network. Unlike the IP network, an ATM end-device cannot transmit cells to a destination ATM end-device over an ATM network, without first establishing a virtual channel connection. Cells are delivered to the destination in the order in which they were transmitted.

A connection is identified by a series of VPI/VCI labels, as explained in section 4.2, and it may be point-to-point or point-to-multipoint. Point-to-point connections are bi-directional, whereas point-to-multipoint connections are unidirectional. Connections may

be either permanent virtual circuits (PVC) or switched virtual circuits (SVC). PVCs are set-up using network management procedures, whereas SVCs are set-up on demand using ATM signalling protocol procedures.

Fixed size cells:

In the ATM layer, packets are fixed-size cells of 53 bytes long, with a 48-byte payload and 5-byte header. The structure of the header was described in detail in section 4.2.

Cell switching

Switching of cells in an ATM network is done at the ATM layer. An example of the ATM stacks used when two end-devices communicate with each other is given in figure 4.6. Both end-devices run the complete ATM stack, that is, the physical layer, the ATM layer, the ATM adaptation layer (AAL), and the application layer. The ATM switches only need the physical layer and the ATM layer in order to switch cells. Different types of ATM switch architectures are described in Chapter 6.

No error and flow control

We recall that in the OSI model the data link layer provides error and flow control on each hop using the ARQ mechanism. In ATM networks, there is neither error control nor flow control between two adjacent ATM switches which are connected with a point-to-point link. A cell simply gets lost if it arrives at an ATM switch at a time when the switch experiences congestion. Also, it is possible that the ATM network may deliver a cell to a destination end-device with an erroneous payload.

The probability that a cell is delivered to the destination end-device with an erroneous payload is extremely small because of the high reliability of fiber-based transmission links. Typically, the probability that a bit will be received wrongly is over 10^{-8}. Now, if we

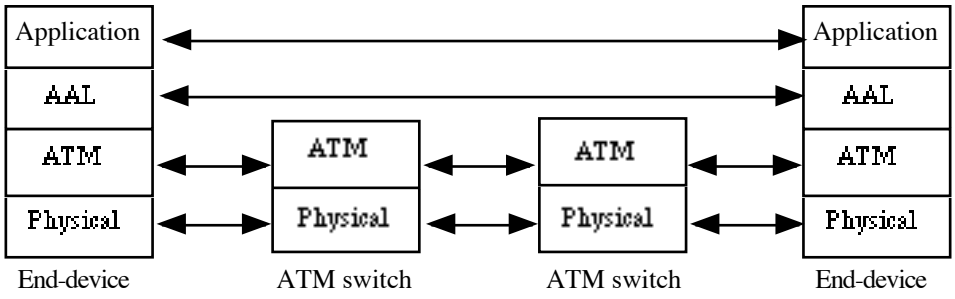


Figure 4.6: Cell switching in an ATM network

assume that bit errors occur independently of each other, then the probability that the payload of an ATM cell, which consists of 48 bytes or 384 bits, will not contain errors is $(1-10^{-8})^{384}$. Therefore, the probability that it will contain one or more erroneous bits is $1-(1-10^{-8})^{384}$,which is very low.

The cell loss rate is a quality-of-service parameter that can be negotiated between the end-device and the ATM network at set-up time. Different applications tolerate different cell loss rates. For instance, video and voice are less sensitive to cell loss than a file transfer. Cell loss rates typically vary from 10^{-3} to 10^{-6}. The ATM network guarantees the negotiated cell loss rate for each connection.

In the ATM standards there is a mechanism for recovering lost cells or cells delivered with erroneous payload, but this mechanism is only used to support the ATM signalling protocols, see SSCOP in Chapter 10. The recovery of the data carried by lost or corrupted cells is expected to be carried out by a higher-level protocol, such as TCP. We note that depending upon the application that created the data, it may not be necessary or there may not be enough time to recover such cells. For instance, it may be deemed unnecessary to recover lost cells when transmitting video over ATM.

When TCP/IP runs over ATM, the loss or corruption of the payload of a single cell results in the retransmission of an entire TCP PDU. In order to clarify this point, let us assume that we want to send a single TCP PDU over an ATM network. This PDU will be encapsulated by IP and it will be passed on to the ATM network. (For simplicity, we assume no fragmentation of the IP PDU.) As will be seen in the next Chapter, the ATM adaptation layer will break the IP PDU to small segments and each segment will be placed in the payload of an ATM cell. Let us assume that the IP PDU will be carried in n

ATM cells. When these n cells arrive at the destination, their payloads will be extracted and the original IP PDU will be reconstructed, from which the TCP PDU will be extracted.

Now, let us assume that one of these n cells is either lost or its payload is corrupted. If this causes the IP header to get corrupted, then IP will drop the PDU. TCP will eventually detect that the PDU is missing and it will request its retransmission. On the other hand, if the cell in question causes the TCP PDU to get corrupted, then TCP will again detect it and it will request its retransmission. In either case, the loss of a cell or the corruption of the payload of a cell will cause the entire PDU to be retransmitted. Since this is not expected to happen very often, it should not affect the performance of the network.

Addressing

Each ATM end-device and ATM switch has a unique ATM address. Private and public networks use different ATM addresses. Public networks use E.164 addresses and private networks use the OSI NSAP format. Details on ATM addresses are given in section 10.5.

We note that ATM addresses are different to IP addresses. In view of this, when running IP over ATM, it is necessary to translate IP addresses to ATM addresses and vice versa. Address resolution protocols are discussed in Chapter 8.

Quality of service

Each ATM connection is associated with a quality-of-service category. Six different categories are provided by the ATM layer, namely, *constant bit rate* (CBR), *real-time variable bit rate* (RT-VBR), *non-real-time variable bit rate* (NRT-VBR), *available bit rate* (ABR), *unspecified bit rate* (UBR), and *guaranteed frame rate* (GFR). The CBR category is intended for real-time applications that transmit at a constant rate, such as circuit emulation. The RT-VBR category is intended for real-time applications that transmit at a variable rate, such as encoded video and voice. The NRT-VBR category is for delay-sensitive applications that transmit at a variable rate but do not have real-time constraints. This category can be used by frame relay, when it is carried over an ATM network. The UBR category is intended for delay tolerant applications such as those running on top of TCP/IP. The ABR category is intended for applications which can vary

their transmission rate according to how much slack capacity there is in the network. Finally, the GFR category is intended to support non-real-time applications that may require a minimum guaranteed rate.

Each quality-of-service category is associated with a set of traffic parameters and a set of quality-of-service parameters. The traffic parameters are used to characterize the traffic transmitted over a connection, and the quality-of-service parameters are used to specify the cell loss rate and the end-to-end delay required by a connection. The ATM network guarantees the negotiated quality-of-service for each connection.

The topic of quality of service in ATM networks is discussed in Chapter 7.

Congestion control

In ATM networks, congestion control permits the network operator to carry as much traffic as possible without affecting the quality of service requested by the users. Congestion control may be either *preventive* or *reactive*. In preventive congestion control, one prevents the occurrence of congestion in the network using a *call admission algorithm* (CAC) to decide whether to accept a new connection, and subsequently policing the amount of data transmitted on that connection. In reactive congestion control, one controls the level of congestion in the network by regulating how much the end-devices transmit through feedback messages. These two scheme are described in detail in Chapter 7.

4.3.3 The ATM adaptation layer

On top of the ATM layer, there is a specialized layer known as the *ATM adaptation laye*r (AAL). The purpose of AAL is to isolate higher layers from the specific characteristics of the ATM layer. AAL consists of the *convergence* sublayer and the *segmentation-and-reassembly* sublayer. The convergence sublayer provides functions which are specific to the higher layer utilizing the ATM network. The segmentation-and-reassembly sublayer, at the transmitting side, is responsible for the segmentation of higher layer PDUs into suitable size for the information field of an ATM cell. At the receiving side, it reassembles the information fields of ATM cells into higher layer PDUs. Four different ATM adaptation layers have been standardized for the transfer of data, and they are

described in Chapter 5. Also, a signalling ATM adaptation layer used by the ATM signalling protocols is described in Chapter 10.

4.3.4 Higher level layers

Various applications may run on top of AAL, such as, voice, circuit emulation, and video. Also, connection-oriented protocols (CONN) such as frame relay, connectionless protocols (CNLS) such as TCP/IP, signalling protocols, and network management protocols run on top of AAL.

4.4 ATM Interfaces

A number of interfaces have been standardized as shown in figure 4.7. These interfaces allow the interconnection and inter-operability of ATM equipment supplied by different

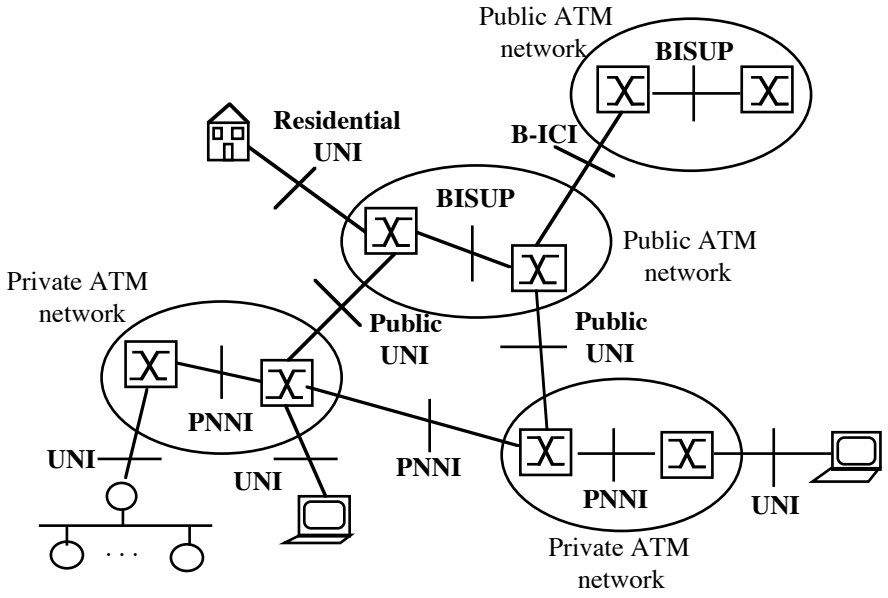


Figure 4.7: ATM interfaces

vendors, such as ATM-ready workstations, ATM switches, ATM-connected routers and interworking units.

The most well-known interface is the *private user network interface* (UNI) which was one of the earliest interfaces that was standardized. This interface is commonly referred to as the UNI, rather than the private UNI, and it is concerned with the interface between an end-device and a private ATM network. An ATM end-device may be any device that transmits and receives ATM cells to/from an ATM network. Typically, an ATM end-device is an ATM-ready workstation or an interworking unit that encapsulates data into ATM cells. The UNI marks the demarcation point between an ATM end-device and a private ATM network. Various aspects of the UNI are described in this book. For instance, the physical layer is described in section 4.5, the cell structure is described in section 4.2, traffic management is described in Chapter 7, and signalling over the UNI is described in Chapter 10.

The *private network-network interface* or *private network node interface* (PNNI) is used between two ATM switches that either belong to the same private ATM network or to two different private ATM networks. This interface consists of the *PNNI routing protocol* and the *PNNI signalling protocol*. The PNNI routing protocol is responsible for routing a new call, that is a new connection, from the *calling user* to the *called user*. The calling user is the end-device that issues the set-up request, and the called user is the destination end-device. This connection may have to traverse several ATM switches that belong to the same network as the calling user, and it may also have to traverse several other ATM networks before it reaches the called user. The PNNI routing protocol distributes topology information between switches and networks of switches, which is used to construct paths through the network. A hierarchical mechanism enables PNNI to scale up well in large wide-area networks. The PNNI signalling protocol is used to relay signalling messages from the originating UNI to the destination UNI. PNNI is described in Chapter 11.

The public UNI is the interface used between a private ATM network or an end-device and a public ATM network. This interface has been aligned with the private UNI. The interface used between two ATM switches in a public network is B-ISUP, and the interface between two public networks is the *BISDN Inter Carrier Interface* (B-ICI). A B-ICI supports an interface between carriers that allows different services, such as cell relay, circuit emulation, frame relay, and SMDS.

Finally, some interesting new residential access interfaces have emerged which can be used to provide basic telephone services, known as POTS (plain old telephone service), access to the Internet, and TV channel distribution over the same wire. Typically, in the USA each home has a telephone line and TV cable. The telephone line is a twisted pair that connects the home to the nearest telephone switch, commonly referred to as the *central office* (CO), and the TV cable is a coax cable that connects the home to the cable distribution network. The telephone network and the cable distribution network are two separate systems. One uses the telephone for POTS and also to connect to the Internet over a modem, and receives TV channels over the cable distribution network.

In recent years, cable modems have been developed that can be used to provide access to the Internet over the TV cable distribution network. This is in addition to the TV channel distribution. Also, a family of technologies has been developed that can be used to provide access to the Internet, in addition to POTS, over the twisted pair. This family of technologies is known as the *x-type digital subscriber line* (xDSL), where x takes different letters of the alphabet to indicate different techniques. In Chapter 9, we describe in detail one of these technologies, namely the *asynchronous digital subscriber line* (ADSL). This is a very popular technology and it currently enables speeds from the network to the user of up to 8 Mbps, and from the user to the network of up to 800 Kbps.

Cable modems and xDSL are two competing technologies offered by different groups of operators. Cable modems are offered by cable operators and xDSL is offered by telephone operators.

4.5 The physical layer

The physical layer transports ATM cells between two adjacent ATM layers. The ATM layer is independent of the physical layer, and it operates over a wide variety of physical link types. The physical layer is subdivided into the *transmission convergence* (TC) sublayer and the *physical medium dependent* (PMD) sublayer.

The PMD sublayer on the transmitter's side is concerned with the transmission and transport across a link of a stream of bits that it receives from the TC sublayer. At the receiver's side, it recovers the stream of bits and passes it on to the TC sublayer.

The TC sublayer interacts between the ATM layer and the PMD sublayer. On the transmitter's side, it receives ATM cells from the ATM layer and creates a bit stream that passes on to the PMD sublayer. On the receiver's side, it reconstructs the ATM cells from the bit stream that it receives from the PMD sublayer and passes them on to the ATM layer.

4.5.1 The transmission convergence (TC) sublayer

The following are the main functions performed by this sublayer:

HEC cell generation and verification

The ATM layer passes to the physical layer ATM cells for transmission over the link. Each ATM cell is complete, except for the HEC byte. This byte is computed and inserted into the HEC field in the TC sublayer. At the receiving side of the link, the HEC state machine is implemented in the TC sublayer. TC will drop any cell whose headers was found to be in error.

Decoupling of cell rate

The PMD sublayer expects to receive a continuous stream of bits. During the time that ATM cells are not passed down from the ATM layer, TC inserts idle cells in-between the cells received from the ATM layer so that to maintain the continuous bit stream expected from PMD. These idle cells are discarded at the receiver's TC sublayer. They are identified uniquely since their header is marked as: VPI=0, VCI=0, PTI=0, and CLP=0.

Cell delineation

Cell delineation is the extraction of cells from the bit stream received from the PMD sublayer. The following procedure for cell delineation is based on the HEC field.

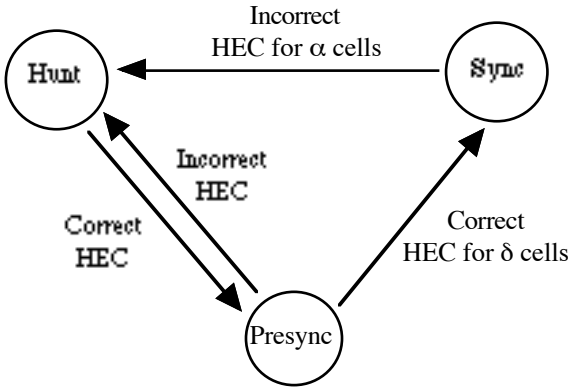


Figure 4.8: The cell delineation state machine

Let us consider a bit stream, and let us assume that we have guessed correctly the first bit of a new cell. This means that this bit and the following 39 bits make up the header of the cell. Consequently, if we carry out the CRC operation on these 40 bits, the resulting FCS should be zero. If it is not zero, then that means that the bit we identified as the beginning of the cell is not really the first bit. We repeat this process starting with the next bit, and so on, until we get a match. At that moment, we have correctly identified in the bit stream the beginning of a cell.

This simple idea is used in the state machine for cell delineation shown in figure 4.8. The state machine consists of the *hunt* state, the *presync* state and the *sync* state. At the beginning, the state machine is in the *hunt* state. In this state, the incoming bit stream is continuously monitored in order to detect the beginning of a cell using the above procedure. When a match occurs, the state machine moves to the *presync* state. In this state, it checks that the FCS of δ consecutive cells is zero. If a mismatch is found, that is the FCS of one of these cells is not zero, then the state machine goes back to the hunt state. Otherwise, synchronization with the bit stream has been achieved, and the state machine moves to the *sync* state. Synchronization is assumed to be lost if α consecutive mismatches occur. In this case, the state machine shifts to the hunt state. ITU-T recommends that $\delta=6$ and $\alpha=7$.

While in the sync state, the HEC state machine described in section 4.2, is used to detect errors in the header of the incoming cells. We recall that when the state machine is in the correction mode, a cell is dropped if more than one erroneous bits are detected in its header. If only one erroneous bit is detected, then it is corrected and the cell is delivered to the ATM layer. When the state machine is in the detection mode, a cell is

dropped if one or more erroneous bits are detected in its header. When the state machine is in either state, a cell is delivered to the ATM layer if no errors are detected in its header.

Transmission frame generation and recovery

In frame oriented transmission systems, such as SONET, at the sender's side TC generates frames by placing frame-related information and ATM cells into a well-defined frame structure. At the receiver's side, it recovers the frames and subsequently the ATM cells from the bit stream.

4.5.2 The physical medium-dependent (PMD) sublayer

The following are the main functions performed by this sublayer:

Timing function

This is used to synchronize the transmitting and receiving PMD sublayers. It generates timing for transmitted signals and it derives correct timing for received signals.

Encoding/decoding

PMD may operate on a bit-by-bit basis or with a group of bits as in the 4B/5B and 8B/10B schemes. In the 4B/5B encoding scheme, each group of 4 bits is coded by a 5-bit code, and in 8B/10B each group of 8 bits is coded by a 10-bit code. Coding groups of bits is known a*s block-coding*. Block coding requires more bandwidth than it effectively provides. For instance, FDDI uses 4B/5B block coding and it runs at 125 Mbps which gives an effective bandwidth of 100 Mbps. There are several benefits with block coding, such as, bit boundary detection and exchange of control information. In addition, a speed-up in the execution of the protocol is achieved, since it operates in chunks of bits. For instance, since FDDI uses 4B/5B coding, it operates in chunks of 4 bits.

4.5.3 ATM physical layer interfaces

ATM was originally developed for high-speed fiber-based links, such as OC-3. Currently, a large variety of physical layer interfaces are in use. A list of public and private interfaces is given in tables 4.2 and 4.3.

Standard	Bit rate	Physical media
SONET STS-48	2.4 Gbps	single-mode fiber
SONET STS-12	622.080 Mbps	single-mode fiber
SONET STS-3c	155.520 Mbps	single-mode fiber
SONET STS-1	51.840 Mbps	Single-mode fiber
PDH E4	139.264 Mbps	Coaxial pair
PDH DS3	44.736 Mbps	Coaxial pair
PDH E3	34.368 Mbps	Coaxial pair
PDH E2	8.448 Mbps	Coaxial pair
PDH J2	6.312 Mbps	Twisted pair/Coax
PDH E1	2.048 Mbps	Twisted pair/Coax
PDH DS1	1.544 Mbps	Twisted pair
inverse mux	nx1.544 Mbps	Twisted pair
inverse mux	nx2.048 Mbps	Twisted pair/Coax

Table 4.2: Public interfaces

SONET/SDH

SONET stands for *synchronous optical network*, and it was designed originally by Bellcore (now Telcordia) for the optical fiber backbone of the telephone network. An outgrowth of SONET was standardized by ITU-T, and it is known as the *synchronous digital hierarchy* (SDH).

The basic structure of SONET is an 810-byte frame, which is transmitted every 125 μsec. This corresponds to 51.84 Mbps, and each byte provides a 64-Kbps channel. This basic structure is referred to as the *synchronous transport signal level 1* (STS-1).

Faster speeds are obtained by multiplexing several STS-1 frames, such as, STS-3, STS-12, STS-24 and STS-48. In general an STS-N signal operates at Nx51.84 Mbps.

.

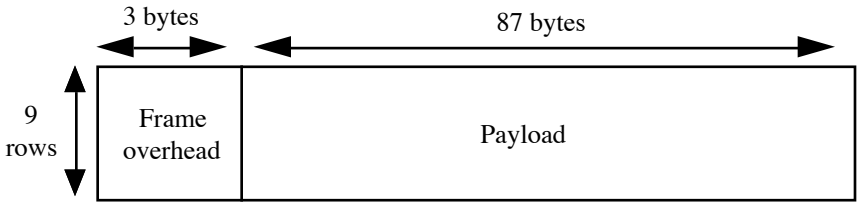


Figure 4.10: The STS-1 frame structure

Standard	Bit rate	Physical media
SONET STS-48	2.4 Gbps	Single-mode fiber
SONET STS-12	622.080 Mbps	Multi-mode fiber
SONET STS-12	622.080 Mbps	Single mode fiber
SONET STS-3c	155.520 Mbps	UTP 5
SONET STS-3c	155.520 Mbps	UTP 3
SONET STS-3c	155.520 Mbps	SM, MM fiber/ Coax
Fiber channel	155.520 Mbps	Multi-mode fiber
TAXI (FDDI)	100.000 Mbps	Multi-mode fiber
SONET STS1	51.840 Mbps	SM, MM fiber/Coax
SONET STS1	51.840 Mbps	UTP 3
ATM 25	25.600 Mbps	UTP 3
Cell stream	25.600 Mbps	UTP 3
Cell stream	155.52 Mbps	Multi-mode fiber/STP

Table 4.3: Private interfaces

SONET signals may be transported by either electrical or optical means. The STS electrical signal, when transmitted over fiber, are converted to a corresponding optical signal called the *optical carrier* (OC). The OC hierarchy starts at 51.84 Mbps (OC-1).

Higher rates are obtained by multiplexing several OC-1s, such as OC-3, OC-12, OC-24, and OC-48.

The basic frame of SDH is known as the *synchronous transport module level 1* (STM-1), and it corresponds to 155.52 Mbps. This structure is compatible with STS-3c, which is obtained by concatenating 3 STS-1 signals. Faster speeds are obtained by multiplexing STM-1 frames into a larger frame. STM-N carries N times the payload of an STM-1 frame. Table 4.3 gives the SONET/SDH hierarchy.

The SONET/SDH frame structure is arranged in rows and columns as shown in figure 4.9. Each frame is transmitted row by row every 125 μsec. The SONET/SDH overheads occupy the first few columns of the frame, and the remaining columns are used to carry the payload. Frames consist of 9 rows, and in the case of STS-1, it consists of 90 columns, of which 3 are used for overhead. The STM-1 frame consist of 270 columns of which 9 are used for overhead.

Optical carrier level	SDH level (ITU-T)	SONET level (electrical)	Data rates
OC-1		STS-1	51.84 Mbps
OC-3	STM-1	STS-3	155.52 Mbps
OC-12	STM-4	STS-12	622.08 Mbps
OC-24	STM-8	STS-24	1.244 Gbps
OC-48	STM-16	STS-48	2.488 Gbps
OC-N	STM-N/3	STS-N	N*51.84 Mbps

Table 4.4: The SONET/SDH hierarchy

ATM cells are directly mapped onto the payload of the STM-1 frame. It is possible that an ATM cell will straddle over two frames. The start of the first cell is identified by a pointer in the overhead section. An alternative scheme for identifying the beginning of the first cell uses the cell delineation procedure described in section 4.5.1.

Plesiochronous digital hierarchy (PDH)

PDH is the older digital hierarchy of signals, based on time-division multiplexing (see section 2.5). ATM was originally defined for SONET/SDH data rates. However, it was also defined over PDH in order to expedite initial deployment of ATM, and also because many corporations did not require 155 Mbps or more bandwidth. ATM over PDH is

deployed at both DS1/E1 and DS3/E3 data rates. Higher data rates have also been defined.

Nx64 Kbps

At data rates below DS1/E1, the ATM cell interface suffers from an unacceptable level of overheads due to the ATM cell header and also due to the AAL encapsulation. For fractional T1 speeds, i.e., for speeds which are integer multiples of 64 Kbps, a frame-based ATM interface known as *frame UNI* (FUNI) is deployed. It is based on an older interface known as ATM-DX1, and it supports signalling, quality of service, and network management. FUNI is used for data traffic.

Inverse mulitplexing for ATM (IMA)

Inverse multiplexing for ATM provides a middle ground for users who require multiple T1/E1 links across a wide area, but not as much as a T3/E3 link.

We recall that a multiplexer combines the traffic streams from N incoming links into a single data stream, which is transmitted to a demultiplexer over a link. The demultiplexer

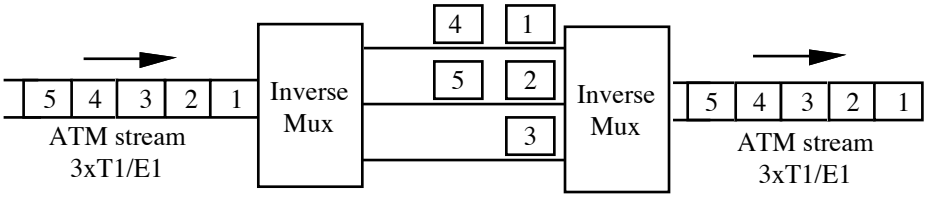


Figure 4.10: Inverse multiplexing for ATM

breaks out the data stream into the N individual data streams and transmits them out over their corresponding output links. An example of a multiplexer/demultiplexer is shown in figure 2.10.

IMA does the exact opposite. As shown in figure 4.10, a demultiplexer is connected to a multiplexer with several T1/E1 links. The demultiplexer breaks up the incoming stream and transmits the ATM cells over the T1/E1 links. At the other side of

these links, the multiplexer combines the individual cell streams into the original cell stream and in the order in which they arrived. In the example in figure 4.10, we see that incoming cells 1, 2, 3, 4, and 5, are transmitted over the three links, and at the other side they are multiplexed back into a single stream with the cell order preserved.

The IMA protocol can also add/delete T1/E1 links on demand, compensate for different link delays, handle link failure and automatic link recovery, and support all the ATM quality-of-service categories. IMA is available for private and public interfaces, and it can also support the use of leased lines and dial ISDN lines.

TAXI (FDDI)

This interface is referred to as TAXI because of the chipset used. It was the first ATM interface to be deployed in the local area. It is equivalent to the FDDI interface and it uses 4B/5B block coding. As mentioned in section 4.5.2, 4B/5B block coding is based on a scheme where chunks of 4 bits are coded as 5-bit chunks. The overall data rate is 125 Mbps, of which 100 Mbps is effectively used. Unlike the single-mode fiber used by FDDI, this interface is based on multi-mode fiber, and it is limited to a distance of 2 kilometers.

ATM 25

Developed to bring affordable ATM to the desktop. It was proposed originally by IBM and Chipcom. It uses IBM's token ring physical layer but with 4B/5B encoding.

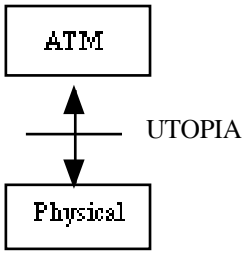


Figure 4.11: UTOPIA

Fiber channel

The fiber channel standard was adopted by the ATM Forum as a method for transporting data in a local-area environment. Though standardized early, it has not seen wide support. Distances range from 2 meters to 2 kilometers at data rates from 100 Mbps to 4 Gbps. Block coding 8B/10B is used.

Cell-based clear channel

The clear channel interface can support multiple physical media and data rates in a local area environment. An ATM cell stream is directly transported over a number of different interfaces. Cell rate decoupling may be based on either idle or unassigned cells. Cell delineation is performed using the cell delineation state machine described in section 4.5.1.

Other interfaces

Interfaces for residential access networks, such as the *asymmetric digital subscriber line* (ADSL), wireless ATM, and ATM over satellite have also been developed.

ADSL is one of the access technologies that can be used to convert the telephone line into a high-speed digital link. It is part of a family of technologies, known as xDSL, where x takes different letters of the alphabet to indicate different transmission techniques. ADSL can support basic telephone service and access to the Internet. Currently, it can provide data rates of up to 8 Mbps from the network to the home and up to 800 Kbps from the home to the network. ADSL is described in Chapter 9.

4.6 UTOPIA and WIRE

The *universal test and operations physical interface for ATM* (UTOPIA) is an interface between the physical layer and the ATM layer, as shown in figure 4.11. It was defined by the ATM Forum in order to enable interoperability between ATM physical layer devices and ATM layer devices made by different vendors.

Various specifications for UTOPIA have been defined: UTOPIA levels 1 and 2 describe an 8-bit and 16-bit data path with a maximum data rate of 800 Mbps; UTOPIA level 3 defines an 8-bit, 16-bit and 32-bit data path with a maximum data rates of up to

3.2 Gbps; and UTOPIA level 4 defines an 8-bit, 16-bit and 32-bit data path with a maximum speed of up to 10 Gbps. Various configurations can be supported that enable the interconnectivity of one more ATM layer devices with one or more physical layer devices.

The *workable interface requirements example* (WIRE) describes an interface between the TC sublayer device and the PMD sublayer device. This is a natural interface as this is the point of change of technology from digital technology supporting moderate clock rates on the TC side to high-speed mixed mode technology on the PMD side.

Problems

1. Why is there error control for the header and not for the payload of an ATM cell?

2. Why is there no data link layer flow control in ATM networks?

3. Give one argument in favour of fixed-size ATM cells.

4. How long does it take to transmit an ATM cell over a link, when the link is:
   a. a T1 line
   b. an OC-3
   c. an OC-12
   d. an OC-24
   e. an OC-48

5. Consider the HEC mechanism. Let p be the probability that a bit is received in error.
   a. With what probability a cell is rejected when the HEC state machine is in the correction mode?
   b. With what probability a cell is rejected when the HEC state machine is in the detection mode?
   c. Assume that the HEC state machine is in the correction mode. What is the probability that n successive cells, where $n \geq 1$, will be rejected.
   d. Assume that the HEC state machine is in the correction mode. What is the probability p(n) that n successive cells will be accepted, where $n \geq 1$. (Hint: Write down the expression for p(1) and p(2), and express p(3) as a function of p(1) and p(2). Then, write down the general expression for p(n) for any n as a function of p(n-1) and p(n-2).)

6. Let us a consider a large ATM switch, and let us assume that the routing table is centralized and that it contains information for all input ports. In order to reduce the time it takes to look-up an entry in the routing table, a hardware mechanism known as *associative registers* is used. Each register contains a *key* and a *value*, where the key is the label and port number of an incoming cell, and the value is the new label and the output port of the cell. When the associative registers are presented with an item, it is compared with all the keys simultaneously. This makes the search very fast. We assume that only part of the routing table is in the associative registers, since the hardware can be expensive. The entire routing table is kept in the main memory. If an item offered to the associative registers is found, then we have a *hit*. Otherwise, we have a *miss*, and the appropriate entry from the routing table in the main memory is fetched and loaded into the associative registers by over-writing another entry.

Let A be the number of the associative registers, T the maximum number of entries in the routing table, S the time it takes to search the associative registers, and R the average time to search the routing table and load the correct entry into the associative registers.

a. Write down the expression for the time W that it takes to find the outgoing label and port number, as a function of the above parameters, assuming that both the associative registers and the routing table are full.
b. What is the region of feasible values of A and R for which W is less than 100 msec, assuming that T = 2000 and S = 0.

7. Consider the case of an application running over an ATM network. Assume that each packet generated by the application is carried by n ATM cells which are transmitted back-to-back. The time to transmit a cell is T and the average time it takes for a cell to traverse the ATM network and reach the destination is D. When all the ATM cells belonging to the same packet are received by the destination, their payloads are extracted from the cells and they are assembled to the original packet. Subsequently, the CRC operation is carried out on the reassembled packet. If the CRC check is correct, the packet is released to the application. Otherwise, a negative acknowledgment is sent back to the source requesting the retransmission of the entire packet. The time it takes to carry out the CRC check is F, and the time it takes for the negative acknowledgment to reach the source is D. Let p be the probability that the ATM cell is received with erroneous payload.
   a. What is the probability that all n cells are received correctly?
   b. What is the probability that exactly m cells are received erroneously, where m<n.
   c. Write down the expression for the average time, call it W, required to transfer correctly an application packet to the destination.
   d. Plot W against the probability p, assuming that n = 30, D = 20 msec, T = 3 μsec, and F = 0. Vary p from 0.1 to 0.00001.
   e. Discuss the results of the above plot in the light of the fact that there is no data link layer in ATM.

CHAPTER 5
The ATM Adaptation Layer

The purpose of the *ATM adaptation layer* (AAL) is to isolate a higher layer from the specific characteristics of the ATM layer. In this Chapter, we describe the four ATM adaptation layers that have been standardized, namely, *ATM adaptation layer 1* (AAL 1), *ATM adaptation layer 2* (AAL 2), *ATM adaptation layer 3/4* (AAL 3/4), and *ATM adaptation layer 5* (AAL 5). These ATM adaptation layers are used for the transfer of user data. A fifth adaptation layer used by the ATM signalling protocols, *the signalling ATM adaptation layer* (SAAL), has also been standardized and it is described in Chapter 10.

5.1 Introduction

The *ATM adaptation layer* (AAL) is sandwiched between the ATM layer and the higher-level layers, as shown in figure 5.1. AAL consists of two sublayers, namely, the *convergence sublayer* (CS), and the *segmentation and reassembly sublayer* (SAR), as shown in figure 5.2. The convergence sublayer (CS) provides service-specific functions, and it is further sub-divided into the *service specific convergence sublayer* (SSCS) and the *common part sublayer* (CPS). The segmentation and re-assembly sublayer (SAR), at the transmitting side, provides segmentation of higher layer PDUs into suitable size for the

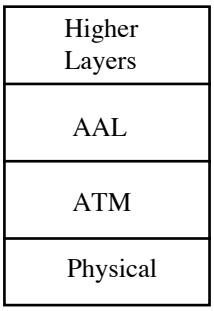


Figure 5.1 The ATM adaptation layer

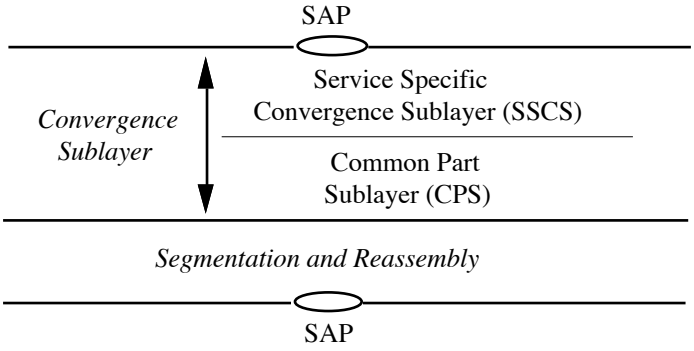


Figure 5.2: The ATM Adaptation Sublayers

information field of an ATM cell. At the receiving side, it reassembles the information fields of ATM cells into higher layer PDUs.

The various types of services provided by AAL to higher-level applications were classified to four different classes, namely A, B, C, and D. These classification scheme was based on the following three attributes: a) the need to transmit timing information between source and destination; b) whether the transmission is constant bit rate or variable bit rate; and c) whether it is connection-oriented or connectionless. Table 5.1 summarizes the values of these three attributes for each class.

In certain cases, it is necessary for the transmitting application to be synchronized with the receiving application. In order to achieve such a synchronization, it may be necessary that timing information is exchanged between the two applications. The first attribute, therefore, indicates whether such an exchange of timing information is necessary. The second attribute has to do with the rate at which the source transmits, and specifically, whether it transmits at a constant bit rate or variable bit rate. A constant bit rate source produces a fixed number of bits each unit of time, whereas a variable bit rate source transmits at a rate that varies over time. For instance, a 64 Kbps voice always

produces 8 bits every 125 μsec, whereas the transmission rate for encoded video is variable since the amount of information that has to be transmitted for each image varies. Another example of a variable bit rate application is the transfer of a data file. In this case, the application does not transmit continuously. Typically, it goes through an active period of time during which it transmits data, followed by a silent period of time during which it does not transmit at all. The silent periods are due to a variety of reasons, such as, the application is waiting for more data to become available, or it is waiting for an acknowledgement before it can send more data. This cycle of an active period followed by a silent period repeats until the entire file is transmitted. The application, therefore, is either transmitting or it is silent, which

Attributes	Class A	Class B	Class C	Class D
Timing between source and dest.	Yes	Yes	No	No
Bit Rate	Constant	Variable	Variable	Variable
Connection mode	CO	CO	CO	CL

Table 5.1: The attributes of the four AAL classes

means that its rate of transmission varies over time. Finally, the classification between connection-oriented and connectionless mode refers to whether a source requires a connection-oriented service or a connectionless service. The implication being, that the connection-oriented service has a better quality of serviced than the connectionless service.

A good example of class A service is the *circuit emulation service* (CES). This service emulates a point-to-point T1 or E1 circuit and a point-to-point fractional T1 or E1 circuit, over an ATM network. The fractional T1 or E1 service provides a Nx64 Kbps circuit, where N can be as small as 1 and as high as 24 for T1 and 31 for E1. CES is connection oriented and the user equipment attached at each end of the emulated circuit transmit at a constant bit rate. Clock synchronization of the user equipment can be maintained by passing timing information over the ATM network.

Examples of class B service are audio and video services transmitted at a variable bit rate. These are connection-oriented services, where the user equipment transmit at a

variable bit rate and require clock synchronization. As in class A, clock synchronization can be achieved by passing timing information from the sender to the receiver over the ATM network.

Service classes C and D do not provide clock synchronization between sender and receiver. A frame relay service provides connection-oriented data transfers at a variable transmission rate and it is a good candidate for Class C. On the other hand, an application running on top of TCP/IP can use class D.

In addition to these four classes, class X was defined as the *raw cell* service. It allows the use of a proprietary AAL with terminal equipment that supports a vendor-defined AAL. That is, the AAL, traffic type (i.e. variable or constant bit rate) and timing requirements are user defined.

Corresponding to these four service classes are four ATM adaptation layers, referred to as *AAL 1*, *AAL 2*, *AAL 3/4*, and *AAL 5*. These adaptation layers are used for the transfer of user data, and they are described in detail in this Chapter. A fifth adaptation layer used by the ATM signalling protocols, the signalling ATM adaptation layer (SAAL), is described in Chapter 10.

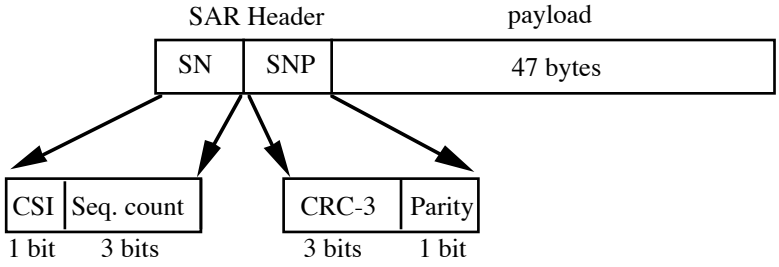


Figure 5.3: The SAR encapsulation for AAL 1

5.2 ATM adaptation layer 1 (AAL 1)

This AAL was proposed for class A services, and it can be used for applications such as circuit emulation services, constant-bit rate video, and high-quality constant-bit rate audio. It provides transfer of constant bit rate data, delivery at the same bit rate, and transfer of timing information between the sending and the receiving application. Also, it can handle cell delay variation, and it can detect lost or misrouted cells.

AAL 1 consists of a SAR sublayer and a convergence sublayer (CS). The SAR sublayer is responsible for the transport and bit error detection, and possibly correction,

of blocks of data received from CS. The CS sublayer performs a variety of functions, such as handling of the cell delay variation, processing of the sequence count, structured and unstructured data transfers, and transfer of timing information.

5.2.1 The AAL 1 SAR sublayer

The SAR sublayer accepts blocks of 47 bytes from the convergence sublayer, and adds a 1-byte header to form the SAR-PDU. The SAR-PDU is then passed on to the ATM layer, where it gets encapsulated with a five-byte ATM header. The ATM cell is then passed on to the physical layer which transmits it out. At the receiving SAR sublayer, the 1-byte header is stripped and the payload of the SAR-PDU is delivered to the receiving CS.

The encapsulation of the SAR PDU is shown in figure 5.3 . The header consists of two fields, the sequence number (SN) field and the sequence number protection field (SNP). Both fields are 4 bits long.

The SN field contains the sub-fields:

Convergence sublayer indication (CSI): It carries an indication which is provided by the CS. The default value of the CSI bit is “0”.

Sequence count: This sequence count is provided by the transmitting CS and it is associated with the block of data in the SAR-PDU. The count starts at 0 and it is increased sequentially modulo 8. It is used by the receiving CS to detect lost or misinserted cells.

The SNP field contains the following two sub-fields:

CRC-3: It is computed over the CSI and sequence count fields

Parity: Even parity bit used calculated over the CSI, sequence count, and CRC-3 fields.

The transmitting SAR computes the FCS for the first 4 bits of the header and inserts it in the CRC-3 field. The pattern used to compute the FCS is given by the polynomial: x^3+x+1. After completing the CRC operation, the transmitting AAL

calculates the even parity bit on the first 7 bits of the header and inserts the result in the parity field.

The receiving SAR examines each SAR-PDU header by checking the FCS and the even parity bit. The state machine that controls the receiver's error detection and correction scheme is the same as the header error control scheme used for the ATM header, described in section 4.2 and shown in figure 4.4. At initialization, the state machine is set to the correction mode. Each time a SAR-PDU comes in, the FCS and the parity bit are checked, and if no errors are found, the SN field is declared to be valid and the state machine remains in the correction mode. If a single-bit error is detected, it is corrected and the SN field is declared to be valid, but the state machine switches to detection mode. If a multi-bit error is detected, the SN field is declared to be invalid and the state machine switches to detection mode. In detection mode, each time a SAR-PDU comes in, the FCS and the parity bit are checked and if a single-bit or a multi-bit error is detected, the SN field is declared to be invalid and the state machine remains in detection mode. If no errors are detected, then the SN field is declared to be valid and the state machine shifts back to the correction mode.

The receiving SAR sublayer conveys to the receiving CS, the sequence count, the CS indication, and the status of the SN field, i.e., valid or invalid.

This error detection and correction scheme runs in addition to the error detection and correction scheme for the ATM header. However, these two mechanisms apply to two different fields of the ATM cell. The header error control mechanism applies to the first four bytes of cell's header, whereas the above scheme applies to the SN field.

5.2.2 The AAL 1 CS sublayer

The convergence sublayer performs various functions, such as handling of the cell delay variation, processing of the sequence count, forward error correction, performance monitoring, structured and unstructured data transfers, and transfer of timing information. Bellow, we describe each of these functions.

Handling of the cell delay variation

In order for AAL 1 to support constant bit rate applications, it has to deliver the data stream to the receiving application at the same bit rate at which it was transmitted. The data stream, as we have seen, is carried by ATM cells which may have to traverse a number of ATM switches before they reach their destination. In view of this, the arrival of these cells may be occasionally delayed because of congestion in the network. Also, the opposite may occur. That is, a group of cells may arrive closer to each other than they were originally transmitted.

To compensate for this variability in the arrival of the cells, CS writes the incoming SAR-PDUs into a buffer, from where it delivers the data stream to the receiving AAL application at a constant bit rate. (A similar method is used for instance when we listen to a radio station over the Internet.) In the event of buffer underflow, it may be necessary for the CS to insert dummy bits in order to maintain bit count integrity. Also, in the event of buffer overflow, CS drops the appropriate number of bits.

Processing of the sequence count

The sequence count values are processed by CS in order to detect lost or misinserted cells. Detected misinserted cells are discarded. In order to maintain bit count integrity of the AAL user information, it may be necessary to compensate for lost cells by inserting dummy SAR-PDU payloads.

Forward error correction

For video and high quality audio forward error correction may be performed in order to protect against bit errors. This may be combined with interleaving of AAL user bits to give a more secure protection against errors.

Performance monitoring

The CS may generate reports giving the status of end-to-end performance, as deduced by the AAL. The performance measures in these reports may be based on events of lost or misinserted cells, buffer underflows and overflows, and bit error events.

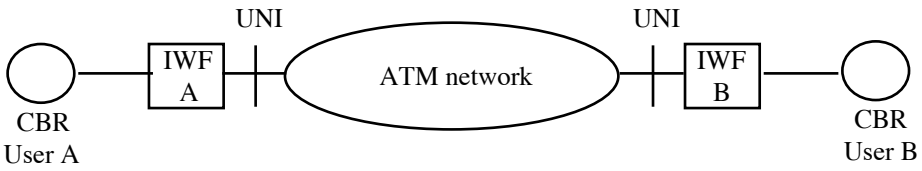


Figure 5.4: The interworking function in CES

Structured and unstructured data transfers

We first note that two CS-PDU formats have been defined to support structured and unstructured data transfers. These two formats are: the *CS-PDU non-P format* and the *CS-PDU P format*. The CS-PDU non-P format is constructed from 47 bytes of information supplied by an AAL user. The CS-PDU P format is constructed from a 1-byte header and 46 bytes of information supplied by the AAL user. The 1-byte header contains a single field, the *structured pointer* (SP), used to point to a block of data. The CS-PDU P format or non-P format becomes the payload of the SAR-PDU.

Structured and unstructured data transfers are used in circuit emulation services (CES), which emulate a T1/E1 link over an ATM network using AAL 1. In order to implement CES an *interworking function* (IWF) is required, as shown in figure 5.4. The two IWFs are connected over the ATM network using AAL 1 via a bi-directional point-to-point virtual circuit connection. CES provides two different services, namely *DS1/E1 unstructured service* and *DS1/E1 Nx64 Kbps structured service*.

In the unstructured service, CES simply carries an entire DS1/E1 signal over an ATM network. Specifically, IWF A receives the DS-1 signal from user A which it packs bit-by-bit into the 47-byte non-P format CS-PDU, which in turn becomes the payload of a SAR-PDU. The SAR-PDUs then become the payload of ATM cells which are transferred to IWF B, from where the bits get delivered to B as a DS-1 signal.

In the structured service, CES provides a fractional DS1/E1 service, where the user only requires an Nx64 Kbps connection. N can be as small as 1 and as high as 24 for T1 and 30 for E1. An Nx64 Kbps service generates blocks of N bytes, which are carried in P and non-P format CS-PDUs. The beginning of the block is pointed to by the pointer in the 1-byte header of the CS-PDU P format.

Transfer of timing information

Some AAL users require that the clock frequency at the source is transferred to the destination. CS provides mechanisms for transferring such timing information.

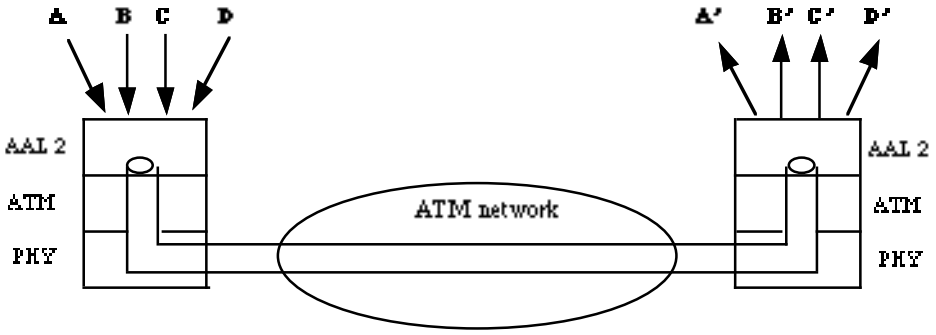


Figure 5.5: AAL 2 can multiplex several data streams

If the sender's clock and the receiver's clock are phase-locked to a network's clock, then there is no need to transfer the source's clock frequency to the receiver, and AAL 1 is not required to transfer any timing information. However, if the two clocks are not phase-locked to a network's clock, then AAL 1 is required to transfer the source clock frequency to the receiver. This can be done using the *synchronous residual time stamp* (SRTS) method, where AAL 1 conveys to the receiver the difference between a common reference clock derived from the network and the sender's service clock. This information is transported in the CSI bit of the SAR-PDU header over successive cells. The common reference clock has to be available to both the sender and receiver. This is the case, for instance, when they are both attached to a synchronous network like SONET.

When a common reference clock is not available, the *adaptive clock method* can be used. In this method, the receiver writes the received information into a buffer and then reads out from the buffer with a local clock. The fill level of the buffer is used to control the frequency of the local clock as follows. The buffer fill is continuously measured, and if it is greater than the median, then the local clock is assumed to be slow and its speed is increased. If the fill level is lower than the median, then the clock is assumed to be fast and its speed is decreased.

5.3 ATM Adaptation Layer 2 (AAL 2)

This adaptation layer was originally defined for class B services. It was later on redefined to provide an efficient transport over ATM for multiple applications which are delay-sensitive and have a low variable bit rate, such as voice, fax, and voiceband data traffic. Specifically, AAL 2 was designed to multiplex a number of such low variable bit rate data streams on to a single ATM connection. At the receiving side, it demultiplexes them back to

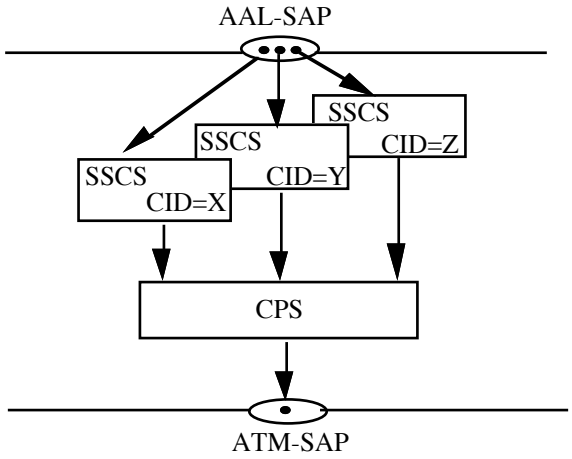


Figure 5.6: Functional model of AAL 2 (sender side)

the individual data streams. An example of AAL 2 is given in figure 5.5. In this example,the transmitting AAL 2 multiplexes the data streams from users A, B, C, and D, onto the same ATM connection. The receiving AAL 2 demultiplexes the data stream into individual streams and delivers each stream to the peer user A', B', C', and D'.

Target networks for AAL2 are narrowband and wideband cellular networks and private narrowband networks (PBX).

The AAL 2 services are provided by the convergence sublayer, which is subdivided into the service specific convergence sublayer (SSCS) and the common part sublayer (CPS). There is no SAR layer in AAL 2.The multiplexing of the different user data streams is achieved by associating each user with a different SSCS. Different SSCS protocols may be defined to support different types of service. Also, the SSCS may be null. Each SSCS receives data from its user and passes this data to the CPS in the form of

short packets. The CPS provides a multiplexing function, whereby the packets received from different SSCS are all multiplexed onto a single ATM connection. At the receiving side of the ATM connection, these packets are retrieved from the incoming ATM cells by CPS and delivered to their corresponding SSCS receivers. Finally, each SSCS delivers its data to its user. The functional model of AAL 2 at the sender's side is shown in figure 5.6

A transmitting SSCS typically uses a timer to decide when to pass on data to CPS. When the timer expires, it passes the data that it has received from its higher-layer application to CPS in the form of a packet, known as the *CPS-packet*. Since the applications that use AAL 2 are low variable bit rate, the CPS-packets are very short and they may have a variable length. Each CPS-packet is encapsulated by CPS and then it is packed into a *CPS-PDU*. As mentioned above, AAL 2 has been designed to multiplex

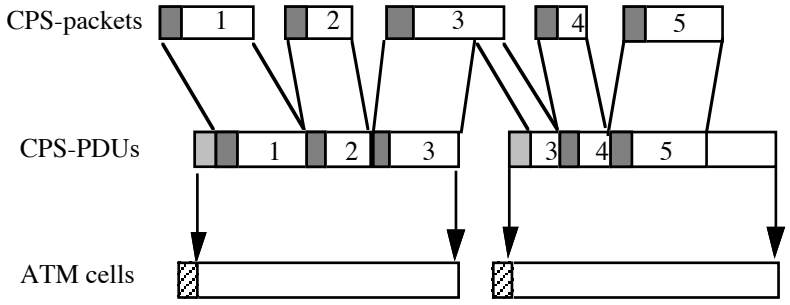


Figure 5.7: Packing CPS-packets into CPS-PDUs

several SSCS streams onto a single ATM connection. This is done by packing several CPS-packets into a single CPS-PDU, where each CPS-packet belongs to a different SSCS stream. It is possible for a CPS-packet to straddle over two successive CPS-PDUs, as shown in figure 5.7. In this figure, we see that CPS-packets 1 and 2 fit entirely in a CPS-PDU, whereas CPS-packet 3 has to be split between the first and the second CPS-PDU. The point where a CPS-packet is split may occur anywhere in the CPS-packet, including the CPS-packet header. The unused payload in a CPS-PDU is padded with zero bytes.

CPS-packets are encapsulated with a three-byte header, and CPS-PDUs with a one-byte header. The CPS-PDU, after encapsulation, is exactly 48 bytes long, and it becomes the payload of an ATM cell. The structure of the CPS-packet and the CPS-PDU is shown in figure 5.8.

The header of the CPS-packet contains the following fields:

Channel identifier (CID): CPS can multiplex several streams, referred to as *channels*, onto a single ATM connection. Each channel is identified by the channel identifier, given in the 8-bit field CID. A channel is bi-directional and the same CID value is used in both

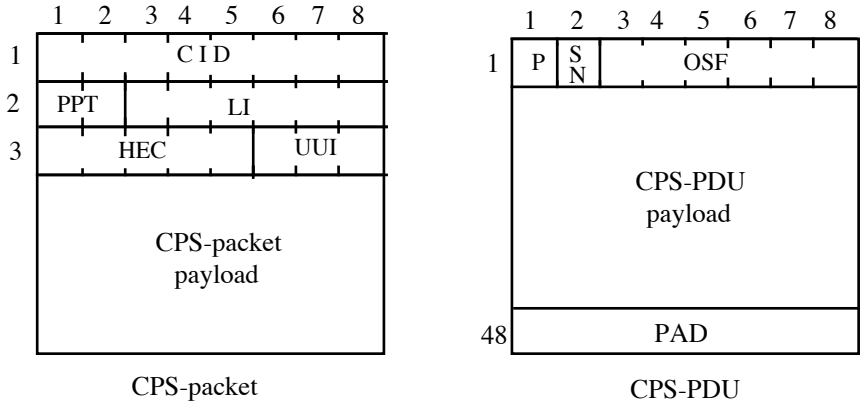


Figure 5.8: The structure of the CPS-packet and CPS-PDU

directions. CID values are allocated as follows: 0 is not used as a channel identification because it is used as padding; 1 is used by *AAL 2 negotiation procedure* (ANP) packets, described below; 2 to 7 are reserved; and 8 to 255 are valid CID values used to identify channels.

Packet payload type (PPT): The packet payload type is a 2-bit field and it serves two types of functions. When PPT≠3, the CPS-packet is serving a specific application, such as carrying voice data, or carrying an ANP packet. When PPT=3, the CPS-packet is serving an AAL network management function associated with the management of the channel identified in the CID field.

Length indicator (LI): The total number of bytes in the payload of the CPS-packet is indicated in this 6-bit field. Its value is one less than the number of bytes in the CPS-packet payload. The default maximum length of the CPS-packet payload is 45 bytes. A maximum length of 64 bytes can be negotiated by ANP or by the network management protocol.

Header error control (HEC): This 5-bit field carries the FCS obtained from the CRC operation carried over the CID, PPT, LI, and UUI fields using the pattern x^5+x^2+1. The receiver uses the contents of the HEC to detect errors in the header.

User-to-user-indication (UUI): This is a 3-bit field and it is used to transfer information between the peer CPS users. This information is transported by the CPS transparently.

The header of the CPS-PDU is referred to as the *start field* (STF), and it contains the following fields:

Parity (P): A 1-bit field used to detect errors in the STF. The transmitter sets this field so that the parity over the 8-bit STF is odd.

Sequence numbers (SN): A 1-bit field used to number modulo 2 the successive CPS-PDUs.

Offset field (OSF): The CPS-PDU payload may carry CPS packets in a variety of different arrangements. For instance, in the example given in figure 5.7, the first CPS-PDU contains two complete CPS-packets, CPS-packet 1 and 2, followed by a partial CPS-packet, CPS-packet 3. The second CPS-PDU in the same figure, consists of the remaining of CPS-packet 3, two complete packets, CPS-packets 4 and 5, and padding.

In order to aid the receiving CPS extract the CPS-packets from the payload of a CPS-PDU, a 6-bit offset field (OSF) is used to indicate the start of a new CPS-packet in the CPS-PDU payload. Specifically, OSF gives the number of bytes between the end of the STF, i.e., the header of the CPS-PDU, and the start of the first CPS-packet in the CPS-PDU payload. In the absence of a start of a CPS-packet, it gives the start of the PAD field. The value 47 indicates that there is no beginning of a CPS-packet in the CPS-PDU.

We note that the AAL 2 CPS transfers the CPS-PDUs in a non-assured manner. That is, a CPS-PDU may be delivered or it may be lost. Lost CPS-PDUs are not recovered by retransmission.

The function that provides the dynamic allocation of AAL 2 channels on demand , is called the *AAL negotiation procedures* (ANP). This function is carried out by an AAL

2 layer management entity at each side of an AAL 2 link. This layer management entity uses the services provided by AAL 2 through a SAP for the purpose of transmitting and receiving ANP messages. These messages are carried on a dedicated AAL 2 channel with CID=1, and they control the assignment, removal and status of an AAL 2 channel. The following types of messages have been defined: assignment request, assignment confirm, assignment denied, removal request, removal confirm, status poll, and status response.

5.4 ATM adaptation layer 3/4 (AAL 3/4)

This adaptation layer was first designed to be used in SMDS (Switched Multi-Megabit Data Service), and later on it was standardized for ATM networks. It was originally proposed for class C services, but very quickly it became obvious that it can also be used for class D services. In view of this, it was named AAL 3/4. This adaptation layer achieves per-cell integrity, as it can detect a variety of different errors related to the segmentation and reassembly of packets. However, it is quite complex and it has a lot of overhead. AAL 3/4 was originally favoured by the telecommunications industry, but due to its complexity, it was not very popular within the computer industry. In reaction to AAL 3/4, a simpler adaptation layer, known as AAL 5, was proposed. AAL 5 is a very popular ATM adaptation layer and it is presented in section 5.5.

The AAL 3/4 functions are provided by the convergence sublayer and the SAR sublayer. The convergence sublayer is subdivided into the service specific convergence sublayer (SSCS) and the common part sublayer (CPS). We note that in the standards CPS is referred to as the *common part convergence sublayer* (CPCS).

Different SSCS protocols may be defined in order to support specific AAL 3/4 applications. SSCS may also be null, which is the assumption made in this section. AAL 3/4 provides a *message mode* service and a *streaming mode* service. In message mode, a

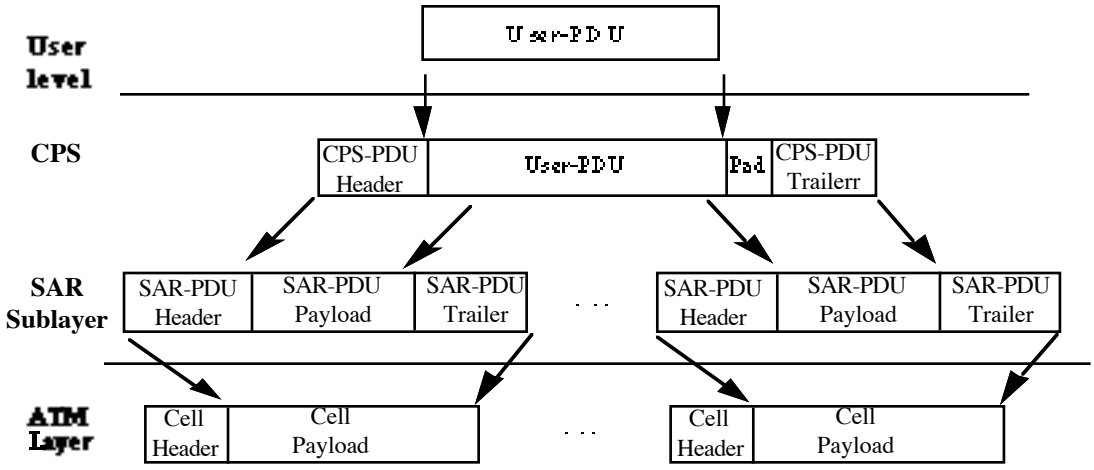


Figure 5.9: The CPS and SAR headers and trailers (sender side)

user-PDU is transported in a single CPS-PDU. It may also be segmented, and each segment is transported in a single CPS-PDU. In streaming mode, one or more user-PDUs are blocked together and transported in a single CPS-PDU.

Both message mode and streaming mode may offer an *assured* or a *non-assured* peer-to-peer operation. In the assured operation, every CPS-PDU is delivered with exactly the data content that the user sent. The assured service is provided by retransmiting missing or corrupted CPS-PDUs. Flow control may be used optionally. The assured operation may be restricted to point-to-point ATM connections. In the non-assured operation, lost or corrupted PDUs are not corrected by retransmission. An optional feature may be provided to allow corrupted PDUs to be delivered to the user. Flow control may be provided as an option for point-to-point ATM connections.

The encapsulation at the transmitting CPS and SAR is shown in figure 5.9. CPS adds a CPS-PDU header and a CPS-PDU trailer for each user-PDU, to form a CPS-PDU. The CPS-PDU is then segmented and each segment is encapsulated with a SAR-PDU header and a SAR-PDU trailer to form a SAR-PDU. Each SAR-PDU becomes the payload of an ATM cell.

The SAR layer introduces a 2-byte header and a 2-byte trailer as shown in figure 5.10. The following fields have been defined:

Segment type (ST): Four different segment types are defined, namely, *beginning of message* (BOM), *continuation of message* (COM), *end of message* (EOM), and *single*

segment message (SSM). The coding of the message type is as follows: 10 (BOM), 00 (COM), 01 (EOM), 11 (SSM). The segment type is used to indicate the position of a segment within a CPS-PDU.

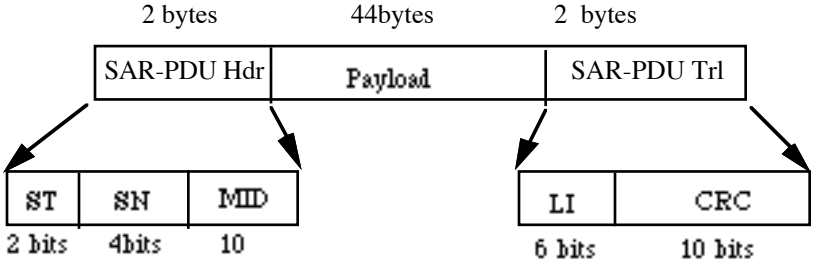


Figure 5.10: SAR encapsulation

Sequence number (SN): All the SAR-PDUs associated with the same CPS-PDU are numbered sequentially. This number is carried in the sequence number field. The sender can select any value between 0 and 15 as the sequence number of the first SAR-PDU of a CPS-PDU. These sequence numbers do not continue on to the next CPS-PDU.

Multiplexing identifier (MID): Multiple connections may be multiplexed over the same ATM connection. Each of these multiplexed connections is identified by a different MID number. If no multiplexing is used, this field is set to zero.

Length identification (LI): It gives the number of bytes of CPS-PDU that are included in the SAR-PDU (value ≤ 44).

CRC: This field contains the FCS obtained by carrying out a cyclic redundancy check over the entire contents of the SAR-PDU using the pattern $x^{10}+x^9+x^5+x^4+x+1$.

CPS introduces a 4-byte header and a 4-byte trailer as shown in figure 5.11. The following fields have been defined:

Common part indicator (CPI): It identifies the convergence service to be provided.

Beginning/End tag (Btag/Etag): It has the same value in the header Btag and trailer Etag. This field permits to check the association of the CPS-PDU header and trailer, and it changes for each successive CPS-PDU.

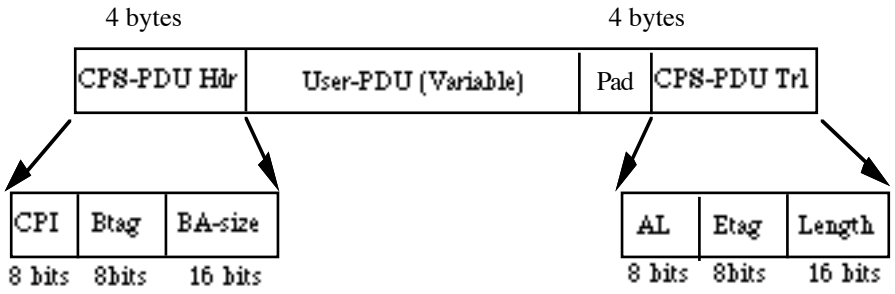


Figure 5.11: CPS encapsulation

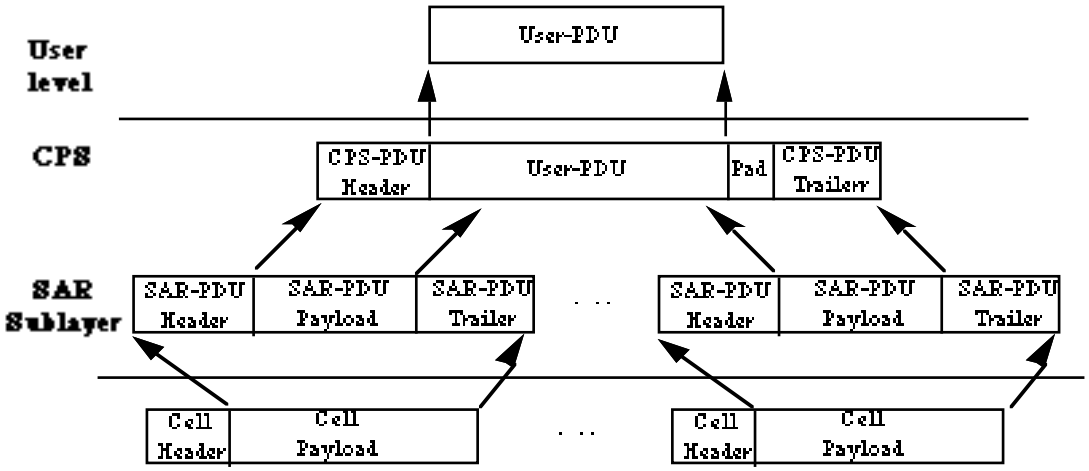


Figure 5.12: Re-assembly at the receiver

Buffer allocation size (BA-size): It is used to indicate to the receiver the maximum buffer requirements necessary to receive the CPS-PDU.

Pad: This field is used so that the CPS payload is an integer multiple of 32 bits. The pad maybe from 0 to 3 bytes, and it does not convey any information.

Alignment field (AL): This field is used in order to maintain a 32 bit alignment of the CPS PDU trailer. It is an unused byte, and it is strictly used as filler.

Length: It specifies in bytes the length of the user-PDU.

The re-assembly of the ATM cells into the original user-PDUs is shown in figure 5.12. The SAR-PDUs are extracted from the ATM cells, stripped off their headers and trailers, and are re-assembled into a CPS-PDU. The CS-PDU is stripped off its header and trailer and it is delivered to the higher layer.

5.5 ATM Adaptation Layer 5 (AAL 5)

This adaptation layer is suitable for connectionless services (class D). It is also used in the signalling AAL (SAAL) described in Chapter 10. It was proposed in reaction to AAL 3/4 which is unnecessarily complex.

The AAL 5 services are provided by the convergence sublayer (CS) and the SAR sublayer. CS is further sub-divided into the SSCS and CPS. We note that in the standards CPS is referred to as the *common part convergence sublayer* (CPCS)

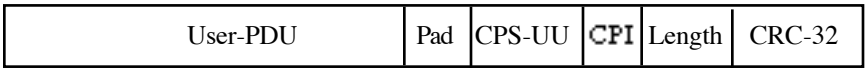


Figure 5.13: Encapsulation of a user-PDU

Different SSCS protocols may be defined in order to support specific AAL users. The SSCS may be null, which is the assumption made in this section.

CPS provides a non-assured transfer of user-PDUs with any length that can vary from 1 byte to 65,535 bytes. At the receiving side, CPS can detect erroneous CPS-PDUs and it can indicate that to the higher-level application. However, since it provides a non-assured service, it does not recover erroneous CPS-PDUs by retransmission. This is left to a higher-level protocol, such as TCP. It also delivers user-PDUs in the order in which it received them.

CPS provides both message mode service and streaming mode service. In message mode, it is passed a user-PDU which it transfers in a single CPS-PDU. In streaming mode, it is passed several user-PDUs over a period of time which it blocks together and transports to the destination in a single CPS-PDU.

A user-PDU is encapsulated by CPS into a CPS-PDU by adding a trailer, as shown in figure 5.13. The following fields in the trailer have been defined:

Padding (Pad)*:* It can be between 0 and 47 bytes long, and it is added so that the entire CPS-PDU including the padding and the remaining fields in the trailer becomes an integer multiple of 48 bytes. The padding is made up of unused bytes which do not convey any information.

CPS user-to-user indication (CPS-UU): A 1-byte field used to transfer transparently CPS user-to-user information.

Common part indicator (CPI): A 1-byte field to support future AAL 5 functions.

Length: A 2-byte field used to indicate the length in bytes of the CPS-PDU payload, i.e. the user-PDU. This is used by the receiver to detect if there has been loss or gain of information.

CRC-32: This 4-byte field contains the FCS calculated by the transmitting CPS over the entire contents of the CPS-PDU, that is, the user-PDU, pad, CPS-UU, CPI, and length. The pattern used is: $x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$.

The SAR sublayer fragments a CPS-PDU into a sequence of 48-byte segments, and each segment is carried in the payload of an ATM cell. There is no encapsulation at the SAR sublayer, as in AAL 3/4. However, the ATM cell that carries the last segment of a CPS-PDU is marked by setting the SDU type indication in its PTI field to 1 (see table 4.2). Specifically, each ATM cell that carries a segment of a CPS-PDU has its SDU type indication set to zero, except the ATM cell that carries the last segment of the CPS-PDU whose PTI field contains the indication SDU type=1.

The receiving SAR appends the payloads of the ATM cells into a buffer, associated with the VCC over which the ATM cells are being transmitted, until it either encounters an ATM cell with an indication SDU-type=1 in its PTI field, or the CPS-PDU exceeds the buffer. Upon occurrence of either event, the buffer is passed on to a higher-level application with an indication as to whether the indication SDU-type=1 was received or not, and in the case it was received, whether the CRC check was correct or not.

Problems

1. Consider the case where a DS-1 signal is transported over ATM via AAL 1 using the unstructured data transfer mode. How long does it take to fill-up a SAR-PDU?

2. In AAL 2, the receiving CPS retrieves the CPS-packets carried in the payload of a CPS-PDU using the offset field (OSF) in the header of the CPS-PDU and the LI field in the header of each CPS-packet carried in the CPS-PDU. The use of the OSF may appear redundant! Construct an example where the receiving CPS cannot retrieve the CPS-packets carried in a CPS-PDU by using only the LI field in each CPS-packet. Assume no cell loss or corrupted ATM payloads.

3. In AAL 2, why does the value 0 cannot be used to indicate a CID?

4. The following CPS-packets have to be transmitted over the same AAL2 connection: CPS-packet 1 (20 bytes), CPS-packet 2 (48 bytes), CPS-packet 3 (35 bytes), and CPS-packet 4 (20 bytes). For simplicity, assume that the length of each of these CPS-packets includes the three-byte CPS-packet header.
   a. How many CPS-PDUs are required to carry these four CPS-packets?
   b. What is the value of the OSF field in each CPS-PDU?

5. A voice source is active (talkspurt) for 400 msec and silent for 600 msec. Let us assume that a voice call is transported over an ATM network via AAL 2. The voice is coded to 32 Kbps and silent periods are suppressed. We assume that the SSCS has a timer set to 5 msec. That is, each time the timer expires, it sends the data it has gathered to CPS as a CPS-packet. Assume that the timer begins at the beginning of the busy period.
   a. How long (in bytes) is each CPS-packet?
   b. How many CPS-packets are produced in each active period?

6. A 1500-byte user-PDU is transported over AAL3/4 in a single CPS-PDU.
   a. How many additional bytes are introduced by CPS?
   b. How many SAR-PDUs are required to carry the resulting CPS-PDU?
   c. How many additional bytes are introduced by the SAR in each SAR-PDU?
   d. What is the maximum and minimum useful payload in each ATM cell that carry the resulting SAR-PDUs (i.e., the maximum and minimum of the number of bytes belonging to the original user-PDU)?

7. A 1500-byte user-PDU is transported over AAL5.
   a. How many bytes of padding will be added?
   b. How many cells are required to carry the resulting CPS-PDU?
   c. What is the total overhead (i.e. additional bytes) associated with this user-PDU?

CHAPTER 6

ATM Switch Architectures

The advent of ATM gave rise to a new generation of switches capable of switching cells at high speeds. These ATM switches can be grouped into three classes, namely, *space-division switches*, *shared memory switches*, and *shared medium switches*. In this Chapter, we describe in detail these three types of architectures. Also, we present various scheduling policies that can be used to schedule the transmission of cells out of a switch. Finally, we describe in some detail an existing switch.

6.1 Introduction

The main function of an ATM switch is to transfer cells from its incoming links to its outgoing links. This is known as the *switching* function. In addition, a switch performs several other functions, such as signalling and network management. A generic model of an ATM switch consisting of N input ports and N output ports is shown in figure 6.1. Each input port may have a finite capacity buffer where cells wait until they are transferred to their destination output ports. The input ports are connected to the output ports via the *switch fabric*. Each output port may also be associated with a finite capacity buffer, where

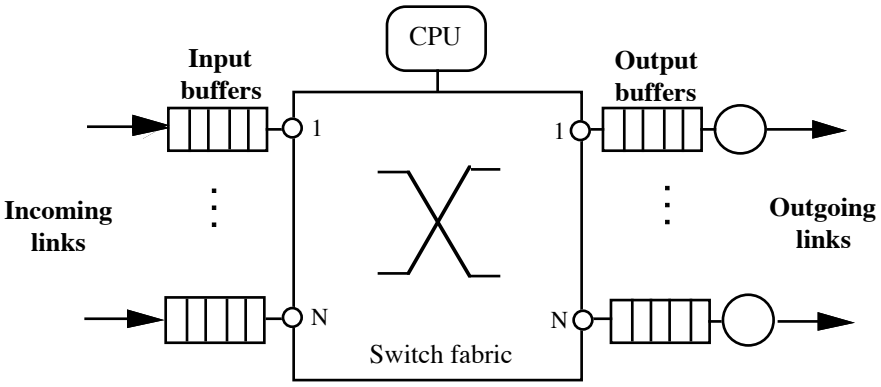


Figure 6.1: A generic model of an ATM switch

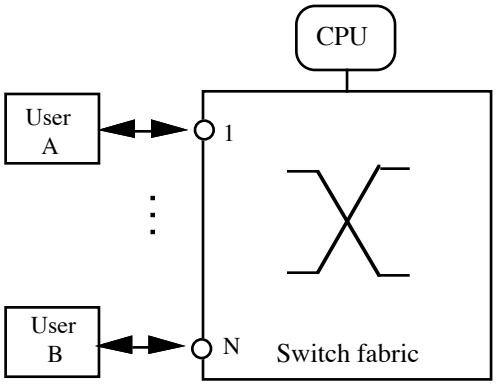


Figure 6.2: A folded ATM switch

cells can wait until they are transmitted out. Depending upon the structure of the switch fabric, there may be additional buffering inside the fabric. An ATM switch whose input ports are equipped with buffers, irrespective of whether its output ports have buffers or not, is referred to as an *input buffering* switch. An ATM switch is referred to as an *output buffering* switch if only its output ports are equipped with buffers. Depending on the switch architecture, cell loss may occur at the input ports, within the switch fabric, and at the output ports.

An ATM switch is equipped with a CPU, which is used to carry out signalling and management functions.

We note that the switch in figure 6.1 was drawn as an *unfolded* switch. That is, the input and output ports were drawn separately with the switch fabric in-between. As a result, the flow of cells from the input ports to the output ports is from left to right. In a real-life switch, each port is typically *dual*, that is, it is both an input and an output port. The link connecting to a port may be either duplex, i.e. it can transmit in both directions

at the same time, or it may consist of two separate cables, one for transmitting cells to the switch and the other for transmitting cells out of the switch. In figure 6.2, we show an ATM switch drawn as a *folded* switch, that is, each port is shown both as an input and an output port. For instance, user A is attached to port 1 which means that it transmits cells to the switch and receives cells from the switch via port 1. User B is attached to port N and, therefore, it transmits cells to the switch and receives cells from the switch via port N. For presentation purposes, switches are typically drawn unfolded, and in this book we follow the same convention. However, it is important to remember that when referring to a diagram of an unfolded switch, the device that transmits to the switch on input port i, also receives cells from the switch from output port i.

VPI	VCI	Input port	VPI	VCI	Output port
10	20	1	12	43	12
11	21	16	13	44	10
12	20	10	14	44	8
43	12	12	14	43	2

Table 6.1: A header conversion table

Header conversion typically takes place before a cell is transferred to its destination output port. The value of the VPI/VCI fields of an arriving cell is looked up in a table, which provides the new VPI/VCI values and the destination output port number. Typically, this table is not very large. In view of this, the table look-up function is not a time-consuming process. Depending upon the type of switch architecture and the way that it has been implemented, there may be a single header conversion table for the entire switch serving all the input ports or one header conversion table per input port. An example of such a table assuming a 16x16 switch is shown in table 6.1.

An ATM switch can be deployed at the edge of an ATM network or inside an ATM network. A switch deployed at the edge of an ATM network may be equipped with various interfaces, such as ADSL, FDDI, Ethernet and token ring, so that it can receive and transmit traffic from/to residential access networks, LANs and MANs. A switch deployed inside an ATM network has only ATM interfaces.

ATM switch architectures can be grouped into the following three classes: *space-division switch, memory sharing switch,* and *medium sharing switch.*

Space-division switch architectures are based on *multi-stage interconnection networks* (MIN). A MIN consists of a network of interconnected switching elements, arranged in rows and columns. MINs were originally used to construct telephone switches. Later on, they were used in tightly coupled multi-processor systems to interconnect processors to memory modules, and in the early 90s, they were used extensively to construct ATM switches.

A shared memory switch architecture utilizes a single memory which is used to store all the cells arriving from the input ports. Cells stored in the memory are organized into linked lists, one per output port. The cells in each linked list are transmitted out of the switch by its associated output port. ATM shared memory switches are currently very popular.

Finally, in a medium sharing switch, all arriving cells at the switch are synchronously transmitted onto a bus. Each output port i can see all the cells transmitted on

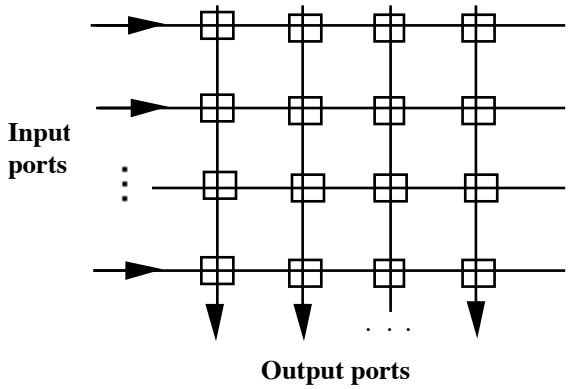


Figure 6.3: A cross-bar switch

the bus, and it receives those cells whose destination is output port i. There is a buffer in front of each output port, where the cells can wait until they are transmitted out.

In the following sections of this Chapter, we describe in detail each of these three types of switch architectures.

6.2 Space-division switch architectures

Space-division ATM switch architectures have been extensively studied. The main features of these architectures is that the control of the switching function is not centralized. Also, multiple concurrent paths can be established from input ports to output ports, each with the same data rate as an incoming link. We will examine the following switching fabrics: *cross-bar*, *banyan*, *Clos* and N^2*disjoint paths*.

Depending on the structure of the switching fabric, it may not be possible to set-up the required paths from the input ports to the output ports without any conflicts, i.e. two (or more) paths requiring the same link at the same slot. Blocking the transfer of a cell from its input port to its output port because a link along its path is used by another cell following a different path, is known as *internal blocking*. Another type of blocking, which is referred to as *external blocking*, occurs when two or more cells seek the same output port at the same time.

6.2.1 The cross-bar switch

This is a very popular fabric and it has been used extensively in circuit switching and packet switching. It consists of a square array of NxN crosspoint switches as shown in figure 6.3. Each crosspoint can assume a cross state or a bar state, as shown in figure 6.4.

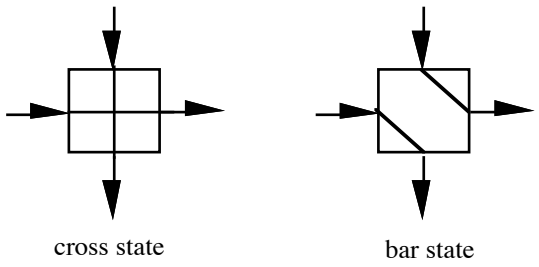


Figure 6.4: The cross state and bar state of a crosspoint

In the cross state, the horizontal input is connected to the horizontal output, and the vertical input is connected to the vertical output. In the bar state, the horizontal input is connected to the vertical output, and the vertical output is connected to the horizontal output. All crosspoints are originally in the cross state.

Let us assume that two cells arrive at a crosspoint at the same time, one on the horizontal input and the other on the vertical input. Let us also assume that the crosspoint is in the cross state. Then, both cells can successfully go through the crosspoint if the destination output of the cell on the horizontal input is the horizontal output, and the destination output of the cell on the vertical input is the vertical output. However, if both cells want to go out of the same output, say the vertical output, then only the cell coming in on the vertical input will go through. A similar operation takes place when the crosspoint is in the bar state.

We shall refer to a crosspoint by its row and column number. For instance, crosspoint (i,j) lies at the intersection of the ith row and the jth column, where the ith row is associated with the ith input port and the jth column is associated with the jth output port.

The switch fabric is self-routing. A cell at input port i destined for output port j, is first propagated horizontally from input port i to the crosspoint (i,j) by setting all the crosspoints on its way to the cross state. When it gets to crosspoint (i,j), it sets it to the bar state and it then proceeds to traverse vertically downwards to the output port j by also setting the crosspoints on its way to the cross state. This is the only path that a cell can follow from input port i to output port j. No other paths through the switch fabric are feasible. For instance, it is not possible for the cell to be propagated horizontally across a few crosspoints, then vertically across a few more crosspoints, then again horizontally across a few more crosspoints and so on until it gets to its destination output port.

Only one cell per input port can be launched into the switch fabric at a time. Thus, in an NxN switch fabric, at most N cells can be transferred to their output ports at the same time. This can only occur when each cell has a different destination output port. For

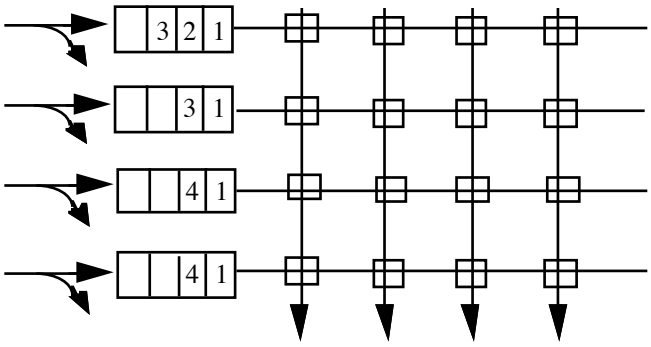


Figure 6.5: A 4x4 cross-bar with input buffering

example, let us assume that a cell from input port 1 is destined to output port 5, and a cell from input port 2 is destined to output port 3. Then, both cells can be transmitted at the same time without any conflict, since they use different links of the cross-bar switch fabric. Specifically, the cell from input port 1 will be propagated horizontally until it reaches crosspoint (1,5), it will set it to the bar state, and then it will traverse vertically down to output port 5. Likewise, the cell from input port 2 will traverse horizontally until it reaches crosspoint (2,3). It will set it to the bar state and then traverse vertically to output port 3. Now, let us assume that both cells have the same destination output port. In this case, conflict will arise as the two cells eventually will have to use the same vertical links. This is an example of internal blocking. It can also be seen as external blocking, since the two cells compete for the same output port. In fact, the distinction between internal and external blocking is not very clear in this switch fabric.

The cross-bar switch is typically operated in a slotted fashion. The length of the slot is such that a cell can be transmitted and propagated from an input port to an output port. At the beginning of a slot, all input ports that have a cell to send out start transmitting. At the end of the slot, the transmitted cells have all been switched to their output ports, assuming that no blocking occurred.

In order to alleviate the blocking problem, several solutions have been proposed. The simplest solution is to block the transmission of a cell that cannot proceed through a crosspoint. This occurs when a cell, while moving vertically towards its output port, arrives at a crosspoint which has already been set to the bar state by another cell. In this case, a blocking signal is returned on a reverse path to its input port. The input port stops

transmitting, and it retries in the next slot. An alternative solution is to provide an arbiter who decides when each input port transmits so that there is no contention. The arbiter can be centralized, or each output port can have its own arbiter. In both solutions, it is necessary that each input port is equipped with a buffer, where cells can wait until they are successfully transmitted out. This makes the cross-bar into an input buffering switch.

The main problem with an input buffering switch is the lost throughput due to a situation that may occur referred to as the *head-of-line blocking*. This happens when a cell at the top of an input buffer is blocked. The cells behind it cannot proceed to their destination output port, even if these destination output ports are free. We demonstrate this using the 4x4 cross-bar shown in figure 6.5. The number shown in each position of an input buffer indicates the destination output port of the cell held in that position. We see that the destination output port of the cell at the top of each input buffer is the output port number 1. In this case, only one of the four cells will be transmitted. The remaining cells will be blocked, which means that the cells behind them will also be blocked even if they are not going to the same output port. It also means that only one path through the fabric will be used, while the other three will remain unused.

Simulation studies have shown that the maximum throughput of an input buffering cross-bar switch is 0.586 per slot per output port. This result was obtained assuming that the input buffers always have at least one cell to transmit, and that the probability a cell chooses a destination output port is 1/N, where N is the number of ports.

6.2.2 Banyan networks

A banyan network is constructed by interconnecting 2x2 switching elements. The switching elements are arranged in rows and columns, known as *stages*. Stages are numbered in an ascending order from left to right. Rows are also numbered in an ascending order from the top down. An example of a banyan network is shown in figure 6.6. Banyan networks are modular and they can be implemented in VLSI.

A banyan network is an example of a *multi-stage interconnection network*. These networks may take different forms, depending on how the switching elements are interconnected, and on the number of input and output ports of the switching element. In

section 6.2.3, we will examine another multi-stage interconnection network, known as the *Clos network*.

In a banyan network, there exists only a single path between an input port and an output port. Up to N paths can be established between input and output ports, where N is the number of ports. The output ports of the banyan network are addressed in binary in an ascending order. For instance, in the 8x8 banyan network shown in figure 6.6, there are 8 output ports and they are numbered in binary as follows: 000, 001, 010, 011, 100, 101, 110, and 111.

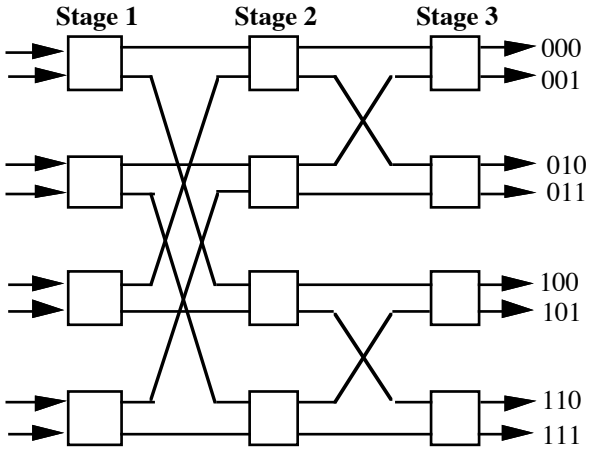


Figure 6.6: An 8x8 banyan network

A 2x2 switching element consists of two inlets, referred to as the upper and the lower inlet, and two outlets, referred to as the upper and the lower outlet, and it is *bufferless*. That is, it does not have an internal buffer for holding cells. When two cells are offered to a switching element, one through the upper inlet and the other through the lower inlet, both cells can be switched if one is going to the upper outlet and the other to the lower outlet. A collision will occur if both incoming cells seek the same outlet, either the upper or the lower one. A banyan network consisting of bufferless switching elements is known as a *bufferless banyan network*.

Each cell is self-routed through the banyan network as follows. Let us consider a cell in an input port, which is about to be launched into the banyan network. The binary address of its destination output port is attached in front of the cell in the reverse order. At each switching element, a cell coming in from the upper or lower inlet, is routed to the

upper outlet if its leading bit is 0 and to the lower outlet if it is 1. The leading bit is then dropped. For example, let us assume that a cell in the first input port is about to be launched into the banyan network shown in figure 6.6. Its destination is output port 011. The bit string 110 is attached in front of the cell, and it is used for the self-routing function. At the switching element in the first stage, the cell is routed to the upper outlet, and the leading bit is dropped. The attached bits are now 11. At the switching element in the second stage, the cell is routed to the lower outlet and the leading bit is dropped. At this moment, only one attached bit is left which is 1.At the last switching element in stage three, the cell is routed to the lower outlet and the leading bit is dropped. The cell has now established a path from its input port to the destination output port. The cell holds the path until the

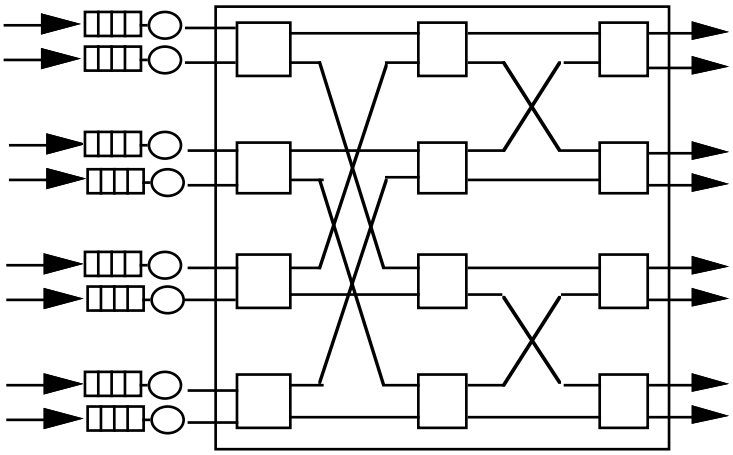


Figure 6.7: Input buffering banyan switch with cell deflection

entire cell is transmitted and propagated through the banyan network. Typically, only a few bits of the cell occupy each link along the established path.

In this banyan network, two different paths may have to share a common link. For instance, in figure 6.6, the path from the first input port to output port 001 and the path from the fifth input port to output port 000, share the same link between the second and the third stage. This is an example of internal blocking. External blocking can also occur when two different paths have the same destination output port. Due to internal and external blocking, the throughput of this network is worse than that of a cross-bar,

assuming uniformly distributed destinations. In view of this, all switch architectures which are based on banyan networks use various mechanisms for overcoming either internal blocking or external blocking or both. Such mechanisms are: input buffering with cell deflection, buffered switching elements, multiple copies of a banyan network, and removing internal and external blocking through a sorter. Below, we examine each of these solutions.

Input buffering with cell deflection

An example of this architecture is shown in figure 6.7. Each input port is equipped with a buffer, which can accommodate a finite number of cells. Cells are stored in the input buffer upon arrival from the incoming link. A cell is lost if it arrives at a time when the input buffer is full. Each input port is managed by a cell processor, indicated in figure 6.7 with a circle. The cell processor carries out the header conversion and attaches the self-routing bits. It is also responsible for transmitting the cell into the fabric. The banyan network is bufferless and it is operated in a slotted fashion. A slot is long enough so that a cell can be completely transmitted and propagated to its destination output port. At the beginning of

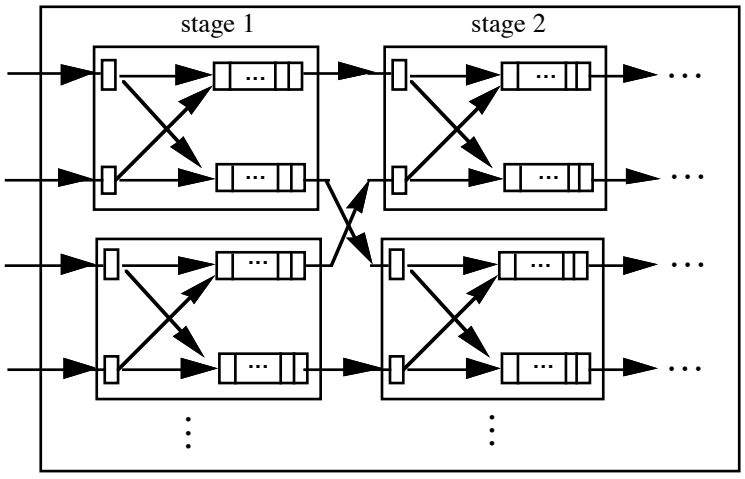


Figure 6.8: The NW corner of a buffered banyan network

each slot, each cell processor launches a cell into the fabric. If two cells collide onto the same output port of a switching element, then one of the two cells will go through and the

other will be blocked. This is done by sending back to the appropriate cell processor a blocking signal. The cell processor stops transmitting the cell, and it waits until the next slot to launch its cell again. In the worst case, a cell processor may have to try for several slots before it succeeds to launch a cell through the switch fabric successfully. This scheme has similar problems as the cross-bar with input buffering, that is, its throughput is reduced to head-of-line blocking.

Buffered switching elements

In this architecture, each switching element is equipped with an internal buffer for holding entire cells. Dedicated buffers can be placed at the input ports, or at the output ports, or at both input and output ports of the switching element. Also, instead of dedicated buffers, a memory can be used which can be shared by all input and output ports of the switching element. A banyan network consisting of buffered switching elements is known as a *buffered banyan network.*

Below, we discuss the operation of a buffered banyan switch consisting of buffered switching elements with input and output buffers. The north-west corner of the switch is shown in figure 6.8. Each input port of the switching element has a buffer that can accommodate one cell, and each output port has a buffer that can accommodate m cells, where m is typically greater than one. The switch is operated in a slotted manner, and the duration of the slot is long enough so that a cell can be completely transferred from one

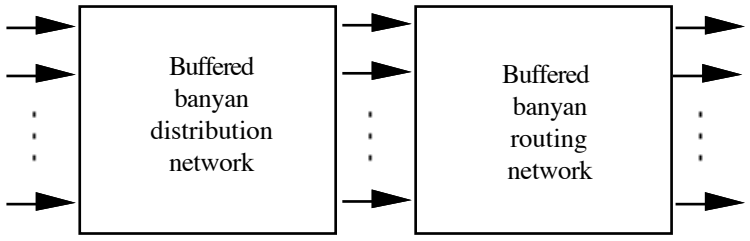


Figure 6.9: A buffered banyan switch with a distribution network

buffer to another. All transfers are synchronized, so that they all start at the beginning of a slot and they all end at the end of the slot. Also, all the buffers function so that a full

buffer will accept another cell during a slot, if the cell at the head of the buffer will depart during the same slot.

A cell is switched through the fabric in a store-and-forward fashion. Upon arrival at the switch, a cell is delayed until the beginning of the next slot. At that time it is forwarded to the corresponding input buffer of the switching element in the first stage, if the buffer is empty. The cell is lost if the buffer is full. This is the only place where cell loss can occur. A cell in an input buffer of a switching element is forwarded to its destination output buffer of the same switching element if there is a free space. If there is no free space, the cell is forced to wait in its input buffer until the end of the slot. At the beginning of the next slot it will again attempt to move to its destination output buffer. If both input buffers contain a cell destined for the same output buffer in the switching element, and this output buffer has only one empty space, then one cell chosen randomly from the two input buffers will be forwarded. The transfer of a cell at the head of the output buffer of a switching element to the input buffer of the switching element in the next stage is controlled by a backpressure mechanism. If the input buffer of the next switching element is free, then the cell is forwarded to the switching element. Otherwise, the cell waits until the next slot and it tries again. Due to the backpressure mechanism, no cell loss can occur within the buffered banyan switch.

The buffered banyan switch alleviates internal and external blocking. However, it can only accommodate a fixed number of cells in its switching elements. If an output port becomes *hot*, that is, it receives a lot of traffic, then a bottleneck will start building up in the switching element associated with this output port. Cell loss can eventually occur, as the bottleneck will extend backwards to the input ports of the switch.

In order to minimize queueing at the beginning stages of the buffered banyan switch, another buffered banyan switch is added in front of it, referred to as the *distribution network*, as shown in figure 6.9. The distribution network is an identical

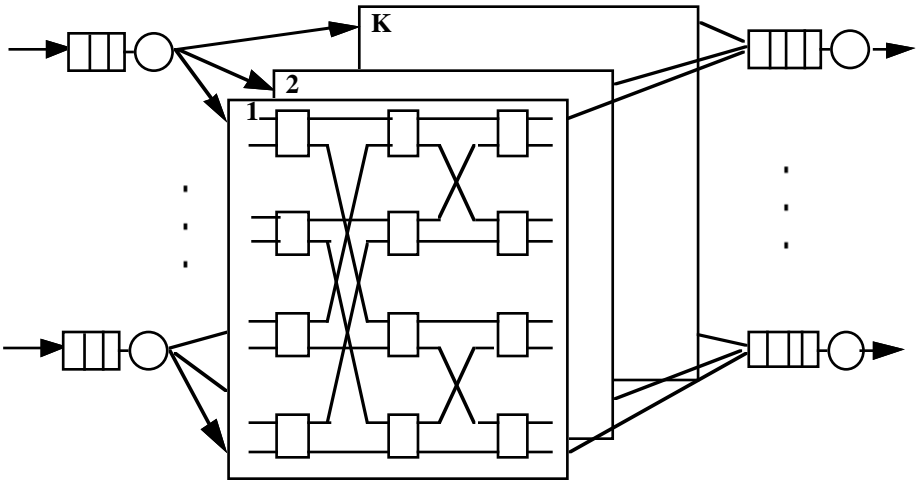


Figure 6.10: Multiple copies of banyan networks

copy of the buffered banyan switch, which is now referred to as the *routing network*, with the exception that cells in each switching element are routed alternatively to each of the two outputs. This mechanism distributes the traffic offered to the routing network randomly over its input ports, which has the effect of minimizing queueing in the switching elements at the beginning stages of the routing network.

Multiple copies of a banyan network

An alternative way to minimize internal blocking is to use multiple identical copies, say K copies, of a bufferless banyan network, as shown in figure 6.10. Typically, K is smaller than the total number of input/output ports of the switch. This scheme can be seen as an extension of the bufferless banyan switch with input buffering and cell deflection, described above.

A packet processor can launch a cell on any of the K banyan networks. The packet processor is busy during the initial time when the leading bits of a cell are propagated through the fabric. Once the cell has established its path to the output port, the packet processor becomes free and it can launch another cell on a different banyan network. Deflection is still possible within each banyan network, and, therefore, a small queue at each input port is necessary. When a cell is deflected on a banyan network, the packet processor will attempt to launch it on a different banyan network. A packet processor can launch up to K cells, one per banyan network. These K cells may be all

directed to the same output port or to different output ports, depending upon their destination. Also, a maximum of K cells may be in the process of arriving at any of the output ports, and therefore, output port buffers are required.

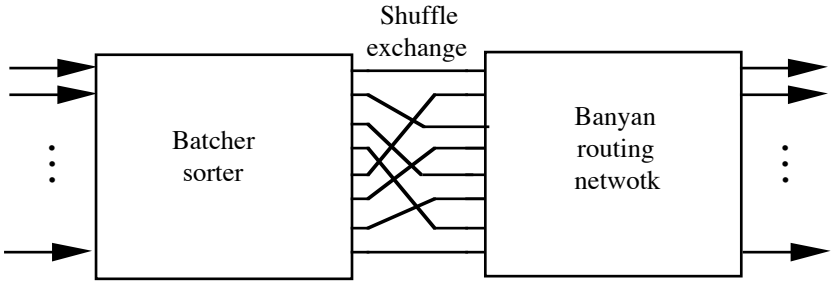


Figure 6.11: A Batcher-banyan switch

The Batcher-banyan switch

This type of switch consists of a *Batcher sorter* linked to a banyan network via a shuffle-exchange, as shown in figure 6.11. The Batcher sorter is a multi-stage interconnection network that orders the cells offered to it according to their destination output port address. That is, if N cells are offered to the Batcher sorter, one per input port, then these N cells will appear at the output ports of the Batcher sorter in an ascending order, as follows. The cell with the lowest destination output port address will appear at the top output port, then the cell with the next lowest destination output port address will appear at the second output port, and so on. Two cells with the same destination output port address will appear at adjacent output ports.

The Batcher sorter consists of 2x2 switching elements, as shown in figure 6.12. Each switching element compares the addresses of the two incoming cells and it switches the one with the largest address to the output port indicated by the arrow. The other cell is switched to the other output port. If both cells have the same address, then the incoming cell on the upper (lower) input port is switched to the upper (lower) output port. If only a single cell is present, then it is switched as if it had the lowest address. That is, it is switched to the output port not indicated by the arrow.

An example of a Batcher sorter is shown in figure 6.13. Three cells with switch destination output ports 8, 3 and 1 are presented to the network. (The switch destination output ports refer to the output ports of the banyan network linked to the Batcher sorter).

Using the above routing rules, we see that these three cells will eventually appear at the three top output ports of the Batcher sorter sorted in an ascending order according to their

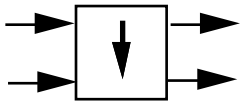


Figure 6.12: A Batcher switching element

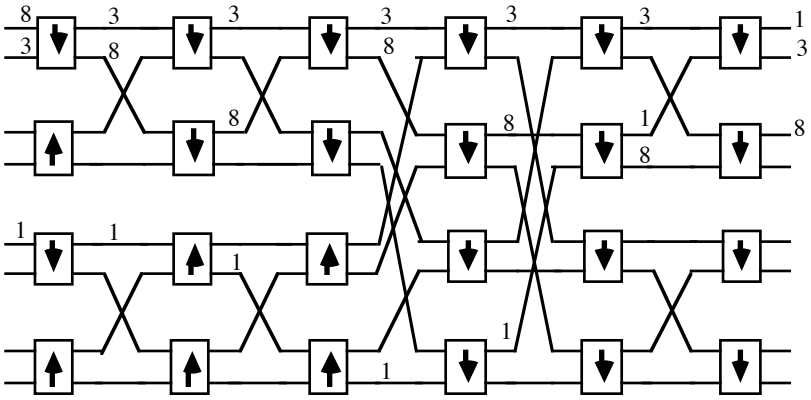


Figure 6.13: A Batcher sorter

destination output ports. It has been shown that the Batcher sorter produces *realizable* patterns. That is, the order in which the cells appear at the output ports of the Batcher sorter forms a pattern that can be switched by the banyan network without any internal blocking. This is true if the cells presented to the Batcher network have different switch output port destinations.

The *starlite* switch, shown in figure 6.14, is based on the Batcher-banyan scheme, but it has an additional mechanism for eliminating external conflicts. This mechanism makes use of the fact that cells with the same switch destination output port appear at consecutive output ports of the Batcher sorter. Consequently, one of them can be forwarded to the banyan network while the remaining cells are trapped. This is done in the trap network. All the trapped cells are concentrated in the top M lines, and the selected cells are concentrated in the remaining N lines which are connected to the banyan routing network. The trapped cells are re-circulated into the switch fabric. Buffering is provided for

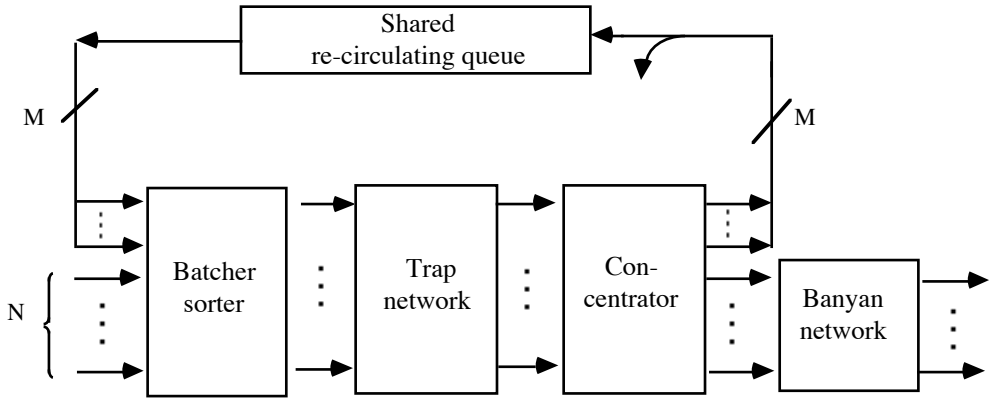


Figure 6.14: The starlite switch

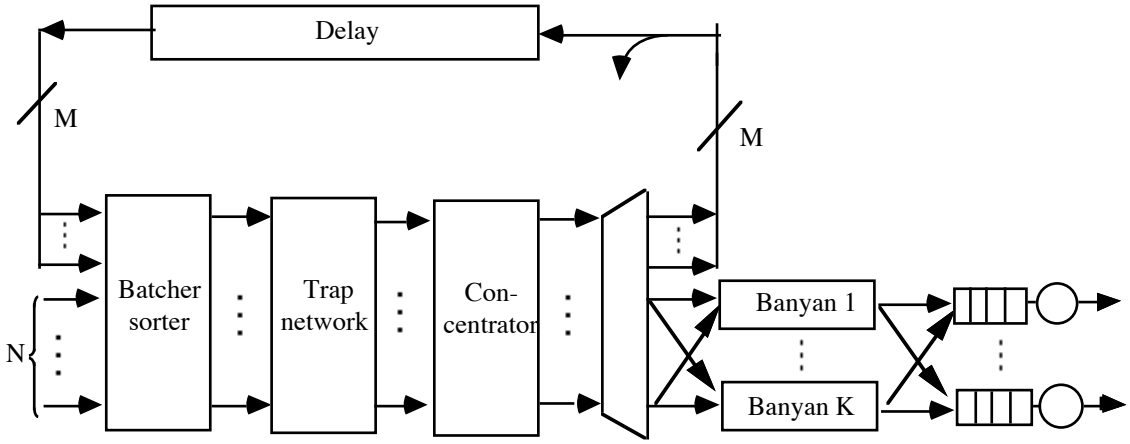


Figure 6.15: The sunshine switch

re-circulating cells. Re-circulated cells have priority over new cells in order to maintain the order in which cell were transmitted. Cell loss can take place within the re-circulating queue. Also, cell loss can take place at any of the N incoming links of the Batcher sorter.

A variation of the starlite switch is the *sunshine* switch shown in figure 6.15. This switch consists of K multiple banyan networks instead of a single network employed in the starlite switch. In view of this, up to K cells with the same switch output address can be selected by the network and forwarded to the banyan networks. Due to the multiple banyan networks, a buffer is required at each output port with a size of at least K cells. This switch has a lower re-circulating rate than the starlite switch.

6.2.3 Clos networks

A Clos network is another example of a multi-stage interconnection network. Unlike banyan networks, Clos networks allow multiple paths between an input port and an output port. A three-stage Clos network is constructed as follows. The first stage consist of k nxm switching elements, that it, each switching element has n input ports and m output ports, where m>n. The second stage consists of m kxk switching elements, and the third stage consists of k mxn switching elements. The switching elements are interconnected as follows. The output port i of the jth switching element in stage s, is connected to the jth input port of the ith switching element in the (s+1)st stage, s=1,2. This Clos network provides m different paths between an input port of a switching element in the first stage and an output port of a switching element in the third stage.

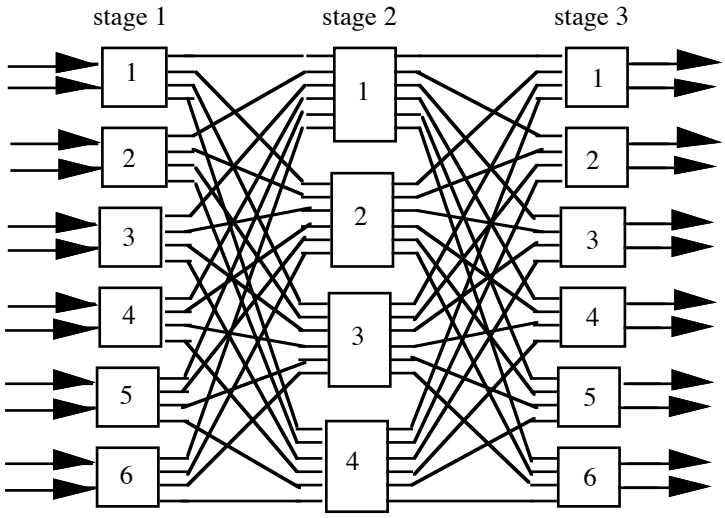


Figure 6.16: A three-stage Clos network

An example of a three-stage Clos network, consisting of 6 2x4 switching elements in the first stage, 4 6x6 switching elements in the second stage, and 6 4x2 switching elements in the third stage, is shown in figure 6.16. We observe, that the number of ports of a switching element in the second stage is equal to the number of switching elements in the first stage, and, the number of switching elements in the second stage is equal to the number of output ports of a switching element in the first stage. Also, the third stage is symmetric to the first stage. Finally, we note that for each input port of a switching element in the first stage, there are four different paths to the same output port of a

switching element in the third stage, and the number of paths is equal to the number of output ports of a switching element in the first stage.

When a new connection is being set-up through a Clos network switch, a routing algorithm is used in order to calculate a path through its switch fabric. This is necessary since there are multiple paths between an input and an output port . All the cells belonging to this connection follow the same path when traversing the Clos network.

6.2.4 Switch architectures with N^2 disjoint paths

So far, we have examined different schemes designed to alleviate the problem of internal and external blocking. An alternative switch architecture which does not suffer from internal or external blocking can be obtained by allowing each input port of the switch to have its own dedicated link to each output port of the switch. Therefore, the total number of independent paths that can be set-up between the input and output ports is N^2, where N is the number of ports. The most-well known architecture of this class is the *knock-out*

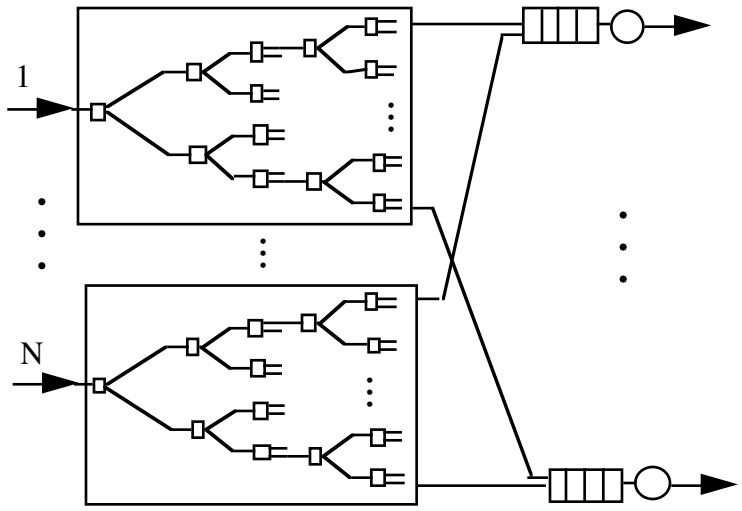


Figure 6.17: A cross-bar tree architecture

switch. This was an experimental switch and it was quite complicated. A considerably simpler switch architecture of the same class is the *cross-bar tree* architecture shown in figure 6.17. As can be seen, it consists of N planes, each interconnecting an input port of the switch to all the N output ports using 1x2 switching elements. The switch is operated

in a slotted manner, and each cell is self-routed to its destination. Output buffers are placed at each output port, since more than one cell may arrive in the same slot.

6.3 Shared memory ATM switch architectures

This is a very popular ATM switch architecture and its main feature is a shared memory that is used to store all the cells coming in from the input ports. The cells in the shared memory are organized into linked lists, one per output port, as shown in figure 6.18. The shared memory is dual-ported. That is, it can read and write at the same time. Currently memories can handle up to 5 Gbps. At the beginning of a slot, all input ports that have a cell, write it into the shared memory. At the same time, all output ports with a cell to transmit read the cell from the top of their linked list and transmit it out. If N is the number of input/output ports, then in one slot, up to N cells can be written into the shared memory and up to N cells can be transmitted out of the shared memory. If the speed of transmission on each incoming and outgoing link is V, then the switch can keep up at maximum arrival rate, if the memory's bandwidth is at least 2NV.

The total number of cells that can be stored in the memory is bounded by the memory's capacity B, expressed in cells. Modern shared memory switches have a large

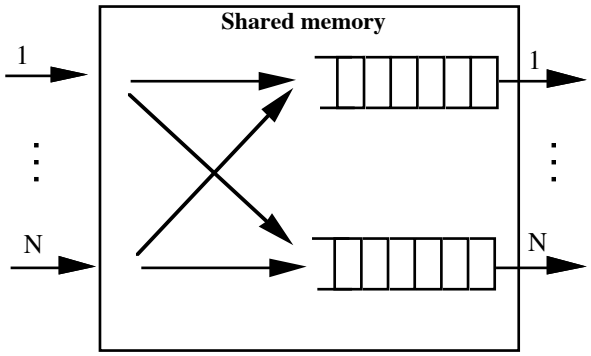


Figure 6.18: A shared memory switch

shared memory and they can hold hundred of thousands of cells. The total number of cells allowed to queue for each output port i is limited to B_i, where $B_i<B$. That is, the linked list associated with output port i cannot exceed B_i. This constraint is necessary in order to avoid starvation of other output ports when output port i gets *hot*. An output port

gets hot when a lot of the incoming traffic goes to that particular port. When this happens, it is possible that the linked list associated with the hot output port may grow to the point that it takes over most of the shared memory. In this case, there will be little space left for cells destined to other output ports. Typically, the sum of the B_i capacities of all linked lists is greater than B. More complicated constraints can also be used. For instance, each linked list i may be associated with a minimum capacity LB_i in addition to its maximum capacity B_i, where $LB_i<B$. LB_i is a dedicated buffer for output port i and it is never shared with the other output ports. The sum of the LBi capacities of all linked lists is less than B.

Cell loss occurs when a cell arrives at a time when the shared memory is full , that is it, contains B cells. Cell loss also occurs when a cell with destination output port i arrives at a time when the total number of cells queued for this output port is B_i cells. In this case, the cell is lost, even if the total number of cells in the shared memory is less than B.

A large switch can be constructing by interconnecting several shared memory switches. That is, the shared memory switch described above is used as a switching element, and all the switching elements are organized into a multistage interconnection network.

An example of a shared memory switch is the one shown in figure 6.18, and it was proposed by Hitachi. First, cells are converted from serial to parallel (S/P), and header conversion (HD CNV) takes place. Subsequently, cells from all input ports are multiplexed and written into the shared memory. For each linked list, there is a pair of address registers (one to write, WA, and one to read, RA). The WA register for linked list i contains the

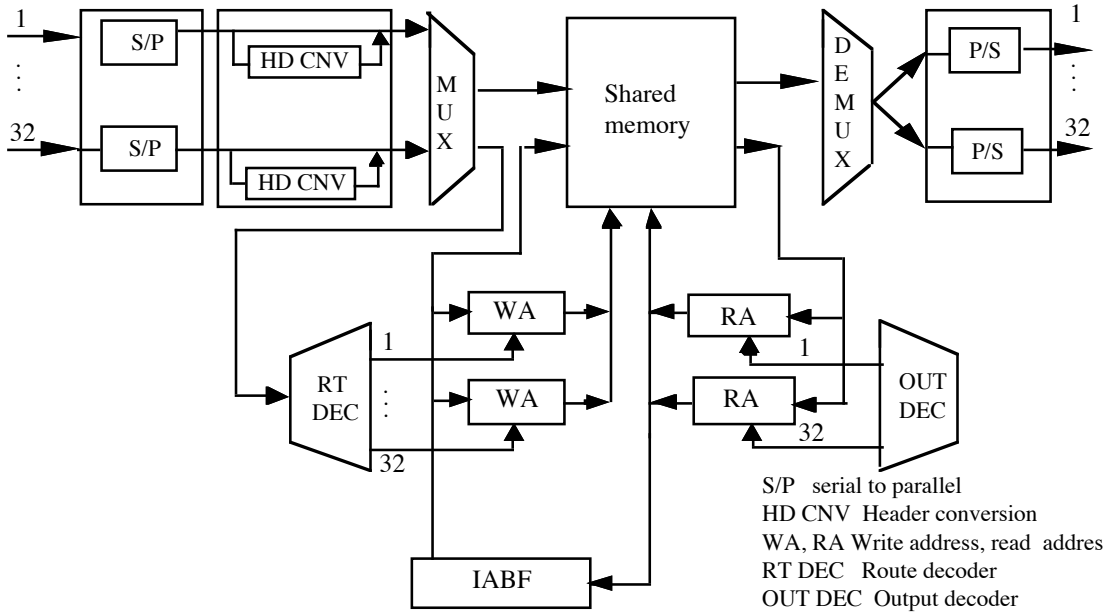


Figure 6.19: The Hitachi shared memory switch

address of the last cell of list i, which is always empty. The incoming cell is written in that address. At the same time, an address of a new empty buffer is read from the IABF chip, which keeps a pool of empty buffer locations, to update WA. Similarly, at each slot a packet from each linked list is identified through the content of the RA register, retrieved, demultiplexed and transmitted. The empty buffer is returned to the pool, and RA is updated with the next cell address of the linked list. Priorities may be implemented, by maintaining multiple linked lists, one for each priority, for the same output port.

One of the problems associated with the construction of early shared memory switches was that memories did not have the necessary bandwidth to support many input ports at a high speed. As a way of getting round this technical problem, a scheme known as bit-slicing was used in the Hitachi switch, in which K shared memories were employed instead of a single one. An arriving cell is divided into K sub-cells, and each sub-cell is written into a different shared memory at the same time in the same location. As a result, a cell in a linked list is stored in K fragments over the K shared memories. All the pointers for the linked list are the same in the K memories. Transmitting a cell out of the switch requires reading these K fragments from the K memories. Now if we assume that we have N links, and that the speed of each incoming and each outgoing link is V, then

the bandwidth that each shared memory is required to have is 2NV/K, rather than 2NV as in the case of a single shared memory.

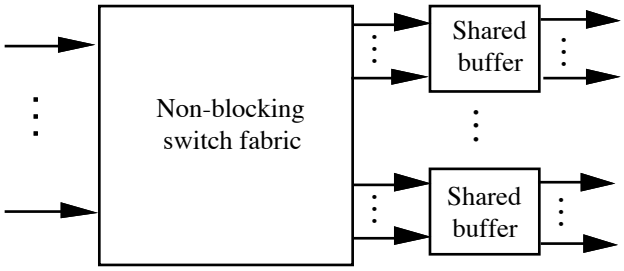


Figure 6.20: Shared memory used for a group of output ports

The shared memory switch architecture has also been used in a non-blocking switch with output buffering, as shown in figure 6.20. Instead of using a dedicated buffer for each output port, a shared memory switch is used to serve a number of output ports. The advantage of this scheme is the following. When using a dedicated buffer for each output port, free space in the buffer of one output port cannot be used to store cells of another output port. This may result in poor utilization of the buffer space. This problem is alleviated with multiple output ports sharing the same memory.

6.4 Shared medium ATM switch architectures

In this architecture, the input ports are connected to the output ports via a high-speed parallel bus, as shown in figure 6.21. Each output port, indicated in figure 6.21 by a circle, is connected to the bus via an interface and an output buffer. The interface is capable of receiving all cells transmitted on the bus, and through an address filter (A/F) it can determine whether each cell transmitted on the bus is destined for its output port. If a cell is destined for its output port, the cell is written in the output buffer, from where it is eventually transmitted out by the output port. Cell loss can occur when a cell arrives at an output buffer at a moment when the buffer is full.

It is possible that the input ports may be equipped with small buffers where the incoming cells can wait until they are successfully forwarded to their destination output port. Backpressure may be employed between the output and the input buffers, to avoid

losing cells at the output buffers. In this case, cell loss can only occur at the input ports of the switch, when a cell arrives at an input port at a time when its input buffer is full.

The bus is slotted and it has a bandwidth equal to NV, where N is the number of ports and V is in the speed of an incoming or an outgoing link. There are as many bus slots as the number of input ports, and the order in which the input ports are served by the bus is determined by a scheduler. The simplest scheduler is based on time-division multiplexing.

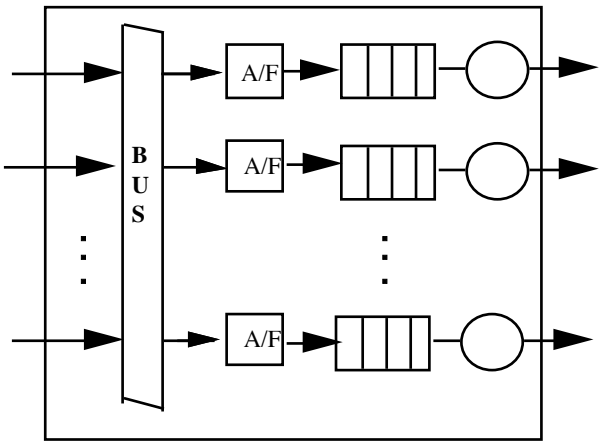


Figure 6.21: A shared medium switch architecture

The N bus slots are organized into a frame, and the frame repeats for ever. Each input port owns a specific bus slot within the frame, and it can transmit one cell per frame when its bus slot comes up. If the input port has no cell to transmit, its bus slot goes by unused. A slightly more sophisticated scheduler that avoids the problem of unused bus slots is the *modified time-division multiplexing algorithm*. In this algorithm, the bus slots are not organized into a frame and each input port does not own a bus slot within the frame. The scheduler only serves the input ports that have a cell to transmit in a cyclic manner. Input ports that do not have a cell to transmit are skipped. The algorithm works as follows. Assuming that it has just served input port i, it attempts to serve the next input port i+1 (mod N). If this input port has a cell to transmit, then the cell is transmitted in the next bus slot. If the input port has no cell to transmit, then the bus slot is not wasted. The scheduler searches the input ports after input port i+1 (mod N) sequentially and allocates

the bus slot to the first input port that it will find that has a cell to transmit. Other more complex algorithms can also be defined.

An example of a shared medium switch is the ATOM switch shown in figure 6.22. In order to achieve the required speed on the shared bus bit-sliced organization is employed. An incoming cell is converted into a bit-serial stream which is then divided into P parallel streams, each feeding one of the P parallel subswitches. Routing of cells to output queues is performed in an identical manner over the P subswitches. This is achieved using a centralized address controller which processes the headers of the incoming cells and instructs the subswitches as to which output buffer each cell is destined for. Specifically, the headers are extracted and routed to the address controller. The address controller

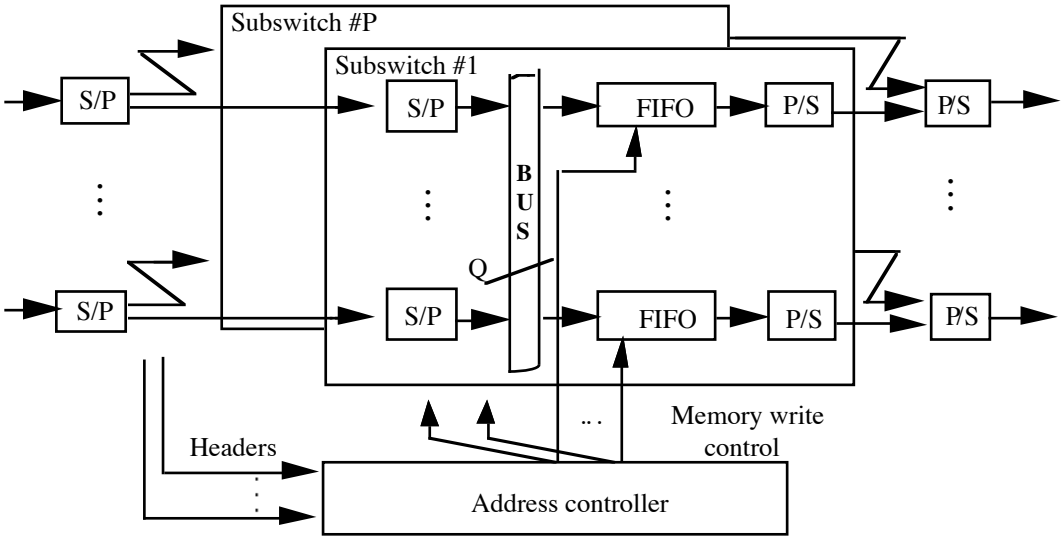


Figure 6.22: The ATOM switch architecture

multiplexes the headers and broadcasts them to the address filters of each output port. Each address filter determines which of the cells are to be written into its respective input buffer, and send the appropriate write control signals to the subswitches.

Large ATOM switches can be constructed by interconnecting shared medium switching elements so that to form a multi-stage interconnection network.

6.5 Non-blocking switches with output buffering

As mentioned earlier on in this Chapter, an ATM switch whose input ports are equipped with buffers, irrespective of whether its output ports have buffers or not, is referred to as an input buffering switch. It is referred to as an output buffering switch if only its output ports have buffers. In section 6.2.1, we discussed how a cross-bar switch with input buffers suffers from head-of-line blocking. This type of blocking occurs in all switches with input buffering, and it may cause the time it takes for a cell to traverse the switch to increase, which has the effect of decreasing the throughput of the switch. Output buffering switches do not suffer from head-of-line blocking, and in view of this, they are preferable to input buffering switches.

ATM switches are also classified to *blocking* and *non-blocking* switches. In a non-blocking switch, its switching fabric does not cause internal or external blocking. On the other hand, in a blocking switch, its switching fabric may cause internal or external blocking, and therefore, cells may collide while they are traversing the switch. In general, non-blocking switches are preferable to blocking switches.

A good example of a non-blocking switch with output buffering is the switch architecture with N^2 disjoint paths, discussed in section 6.2.4. In this switch, each input port has its own dedicated path to each output port, thus eliminating the possibility of internal and external blocking. Also, there are no buffers at the input ports, which eliminates head-of-line blocking.

The shared memory switch, discussed in section 6.3, is also a non-blocking switch with output buffering. In this case, the output buffers are the linked lists formed in the shared memory.

Finally, the shared medium ATM switch architecture without input buffers, discussed in section 6.4, is also a non-blocking ATM switch with output buffering.

6.6 Multicasting in an ATM switch

ATM connections may be point-to-point or point-to-multipoint. A point-to-point connection is used by two end-devices in order to communicate over an ATM network, and a point-to-multipoint connection is used for multicasting and broadcasting. Multicasting is different to broadcasting. In a multicast, an end-device, called the *root*,

transmits the same information to a specific group of end-devices, called the *leaves*. The root and the leaves are members of a particular multicast group, which is a subset of the set of all end-devices attached to the network. In a broadcast, the root transmits the same information to all the end-devices attached to the network.

An ATM switch should be able to carry both point-to-point and point-to-multipoint connections. In a point-to-point connection, an ATM switch simply transfers incoming cells belonging to the connection to a specific output port of the switch. In a point-to-multipoint connection, an ATM switch transfers incoming cells belonging to the connection to a number of different output ports of the switch. The number of leaves in the multicast may change during the time that the connection is up. New leaves may be added and existing ones may be dropped. These changes may result to changes in the set of destination output ports of an ATM switch to which the incoming cells should be delivered.

Various solutions have been proposed to introduce multicasting in the switch architectures examined in the previous sections. In the case of multi-stage interconnection switch architectures, two different approaches can be used to implement multicasting, namely, the *time* and *space* approach. In the time approach, each packet processor transmits each multicast cell to all the destination output ports of the switch, one at a time. That is, if a multicast cell should be distributed to four different output ports, then the packet processor will transmit it four times, each time to a different target output port. This is an acceptable solution when the size of the multicast group is small, but it introduces large delays when the multicast group is large. In the space approach the cell is duplicated using a copy network. For instance, let us consider the buffered banyan switch shown in figure 6.9. It consists of a distribution network and the routing network. This switch can be seen as a point-to-point switch. That is, it can deliver incoming cells belonging to a point-to-point connection to a specific output port of the switch. Multicasting can be introduced using a copy network in front of the distribution network. The structure of the copy network is the same as that of the distribution and routing networks. That is, it consists of 2x2 buffered switching elements which are interconnected in a banyan network. Each switching element in the copy network can generate up to two copies of each passing cell The copy network performs cell

duplication according to the size of the multicast group. Each duplicated cell goes through the distribution network and then it is switched to its destination output port by the routing network. The copy network can modify the number of copies and their destination output port as leaves are added or dropped.

The shared memory switch architecture is more amenable to supporting multicasting than a multi-stage interconnection switch architecture. Various schemes for transmitting a multicast cell out of a number of output ports have been proposed. The simplest scheme is to copy a multicast cell to all the linked lists associated with the destination output ports of its multicast group. This method may result to higher memory usage, particularly when the multicast group is very large. Another method is to keep a separate linked list for multicast cells. In an NxN shared memory switch, the N linked lists, one per output port, are used to hold cells belonging only to point-to-point connections. In addition to these N linked list, a new linked list is introduced which holds all the multicast cells. This mulitcast linked list is not served at the same time as the N linked lists. Also, during the time that cells are transmitted out of the multicast linked list, no cells are transmitted out of the N point-to-point linked lists. Transmission of a multicast cell is organized in such a way so that it is transmitted out of all its destination output ports simultaneously. Various policies can be used to decide how the switch distributes its time between serving the N linked lists and serving the multicast linked list..

The shared medium switch architecture is probably the most suitable architecture for transmitting multicasting traffic. This is because a cell transmitted over the bus can be received by all output ports. The address filter of an output port will accept a multicast cell if it recognizes its address. The problem associated with this switch is that an output buffer associated with a multicast may become full. If backpressure is used, the transmitter will not be able to complete its multicast to all the target output ports. This problem can be

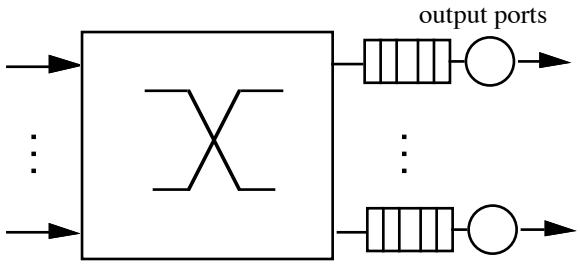


Figure 6.23: A non-blocking switch with output buffering

resolved by keeping a list of all the output ports that did not receive the multicast, and keep multicasting the cell only to the output ports in the list.

6.7 Scheduling algorithms

Early ATM switches were equipped with very small buffers. For instance, in an output buffering switch, each output buffer may have had a capacity of less than 100 cells. The cells were served, i.e. transmitted out of the switch, in the order in which they came, that is in a first-in-first-out (FIFO) fashion. The FIFO scheduling algorithm does not take into account priorities among the cells, and the fact that cells may belong to different connections with different quality-of-service parameters. Therefore, a cell belonging to a connection used for a file transfer will be served first if it arrives before a cell belonging to a connection which is carrying data from a real-time application, such as voice or video. The justification for the FIFO algorithm was based on the fact that the queueing delay in a buffer was small, since the buffer size was small. Therefore, there was no need for a complex scheduling algorithm that would serve cells from different connections with different priorities. Another consideration in support of the FIFO algorithm was that it was easy to implement.

It became apparent, however, that it is possible to implement effectively complex scheduling algorithms that can manage large buffers with many queues, so that different connections could be served according to their requested quality of service. A number of different schedulers have been proposed and implemented. Below, we discuss some of these scheduling algorithms.

Static priorities

Let us consider a non-blocking switch with output buffering, as shown in figure 6.23. Each output buffer holds cells that belong to different connections that pass through this buffer. Each of these connections is associated with a quality-of-service category signalled to the switch at call set-up time. The cells belonging to these connections can be grouped into queues, one per quality-of-service category, and these queues can be associated with different scheduling priorities.

As will be seen in the following Chapter, several quality-of-service categories have been defined. Let us consider the following four categories: *constant bit rate* (CBR), *real-time variable bit rate* (RT-VBR), *non-real-time variable bit rate* (NRT-VBR), and *unspecified bit rate* (UBR). The CBR service category is intended for real-time applications which transmit at a constant bit rate, such as, unencoded video and circuit emulation. The RT-VBR service category is intended for real-time applications which transmit at a variable bit rate, such as encoded voice and video. The NRT-VBR service category is for applications that transmit at variable bit rate and do not have real-time requirements. The UBR service category is intended for delay-tolerant applications that do not require any guarantees, such as data transfers.

Using these four quality-of-service categories, each output buffer can be organized into four different queues, as shown in figure 6.24. An arriving cell at an output buffer joins one of the four queues according to the quality-of-service category of the connection that it belongs to.

These queues can be assigned static priorities, which dictate the order in which they are served. These priorities are called static because they do not change over time and they are not affected by the occupancy levels of the queues. For instance, the CBR queue has the highest priority, the RT-VBR the second highest priority, and so on, with the UBR queue having the lowest priority. These queues can be served as follows. Upon completion of the transmission of a cell, the next cell for transmission is selected from the CBR queue. If the CBR queue is empty, the next cell for transmission is selected from the RT-VBR queue. If this queue is empty, then the next cell is selected from the NRT-VBR queue, and so on. If all queues are empty, then no cell will be selected for transmission. Thus, in essence, the CBR queue is served until it becomes empty. Then, the next priority queue is

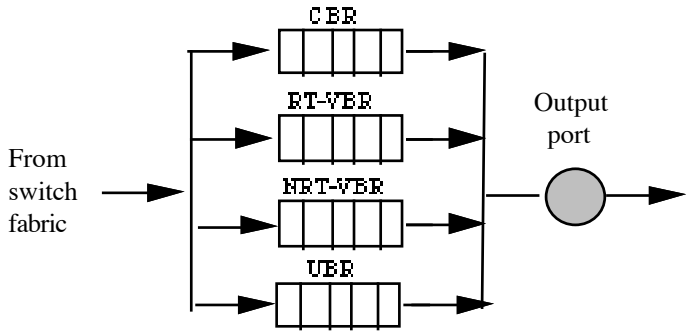


Figure 6.24: Logical queues for an output port

served until it becomes empty, and so on. If during the time that a cell from, say, the UBR queue is being transmitted out, a cell arrives in one of the higher-priority queues, i.e. the CBR queue, or the RT-VBR queue, or the NRT-VBR queue, then this high priority cell will be transmitted out next after the transmission of the cell from the UBR queue is completed.

Additional scheduling rules can be introduced which take into account the current status of the queues. A typical example of such a rule is the "aging factor". If a queue, typically a low priority queue, has not been served for a period of time which is longer that a pre-specified threshold, then the queue's priority is momentarily raised so that some of its cells can be transmitted out.

Early deadline first (EDF) algorithm

In this algorithm, each cell is assigned a deadline upon arrival at the buffer. This deadline indicates the time by which the cell should depart from the buffer. It is calculated by adding a fixed delay to the arrival time of the cell. This delay may vary according to the quality-of-service category of the cell. The scheduler serves the cells according to their deadlines, so that the one with the earliest deadline gets served first. A cell that is assigned a deadline closer to its arrival time, will suffer a low delay in the buffer. On the other hand, a cell that is assigned a deadline far away from the time that it arrived at the buffer, may suffer a longer delay before it gets transmitted out.

Using this scheme, cells belonging to delay-sensitive applications, such as voice or video, can be served first by assigning them deadlines closer to their arrival times.

The weighted round-robin scheduler

Each output buffer is organized into a number of queues. For instance, there could be one queue for each connection that passes through the particular output port. There could also

be fewer queues, such as one queue per quality-of-service category. The scheduler serves one cell from each queue in a round robin fashion. The queues are numbered from 1 to M, and they are served sequentially. That is, if a cell from queue 1 was just served, then the next queue to serve is queue 2. This sequential servicing of the queues continues until the Mth queue is served, whereupon, it goes back to queue 1. If the next queue to be served, say queue i, is empty the scheduler skips it and goes on to queue i+1. (This algorithm is similar to the modified time-division multiplexing algorithm described in section 6.4.)

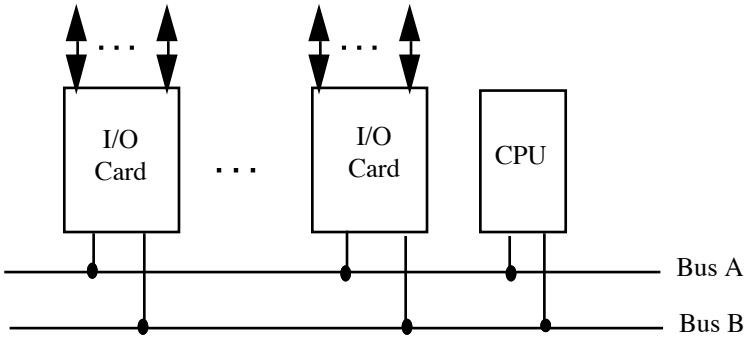


Figure 6.25: The architecture of the LDR200

Weighted round robin scheduling can be used to serve a different number of cells from each queue. For instance, let us assume that there are 5 connections with weights: 0.1, 0.2, 0.4, 0.7 and 1. Multiplying each weight by a common number so that all the weights become integer, in this case by 10, gives: 1, 2, 4, 7, and 10. The scheduler will serve one cell from the first queue, 2 from the second queue, 4 from the third queue, seven from the fourth queue, and ten from the fifth queue.

Now, let us consider the case where one of the queues, say queue 5, becomes idle for a period of time. Then, queues 1 to 4 will be served as before and queue 5 will be skipped each time its turn comes up. The ten slots that would have been used for queue 5, are now used for queues 1 to 4 proportionally to their weights.

6.8 The Lucent AC120 Switch

In this section, we describe the basic architecture of an existing product, the Lucent AC120 switch. The switch was designed to operate at the edge of an ATM network and it is equipped with interfaces for Ethernet, DS-1, DS-3, E1, E2 and OC-3. The switch combines features from both the shared memory switch architecture and the medium shared switch architecture. As shown in figure 6.25, the switch consists of a number of

I/O cards which are attached to two buses. Each bus runs at 600 Mbps. There is also a CPU attached to the bus, which is used for call management. Each I/O card can receive cells from both buses, but it can only transmit on one bus. Half of the I/O cards transmit onto the same bus, and the other half onto the second bus. If one bus develops an error, all the I/O cards switch to the other bus. The buses are slotted and each bus slot carries an ATM cell with an added proprietary header of 3 bytes. Transmission on each bus takes place using the modified time-division multiplexing algorithm described in section 6.4. That is, transmission is done in a round robin fashion among those I/O cards that have a cell to transmit. Each I/O card transmits for one time slot.

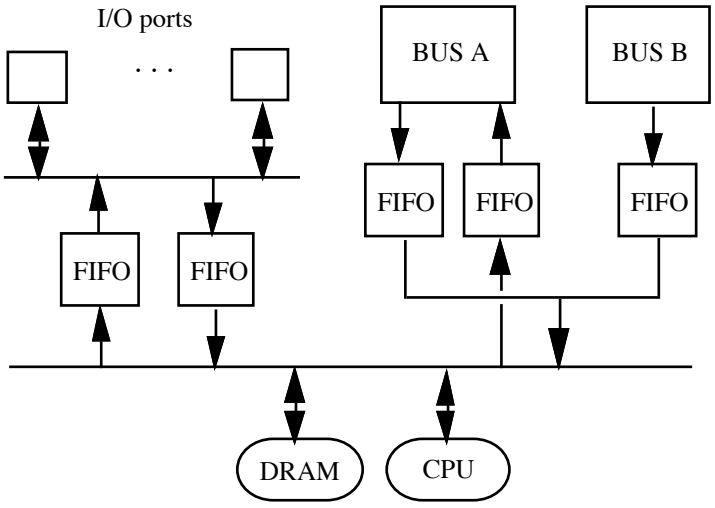


Figure 6.26: The architecture of an I/O card

Some of the relevant components of an I/O card are shown in figure 6.26. Each I/O card has I/O devices, a DRAM main memory and a CPU which controls all memory read/write functions, cell queueing, and management of the I/O card. An I/O card can receive cells from both buses, and under normal conditions it transmits only to one bus. It can also receive and transmit cells from its dual input/output ports, which are the actual input/output ports of the switch. The FIFO queues associated with the two buses are used to receive/transmit cells from/to the bus at the rate of 600 Mbps. Therefore, the total rate at which cells may arrive at both input FIFOs (assuming that no cells are being transmitted out) can be as high as 1.2 Gbps.

Cells are transferred from/to the input FIFOs to/from the shared memory using a direct memory access (DMA) scheme. It is possible that an input FIFO may become full. Backpressure is then used to protect against cell loss. This is done by instructing the other I/O cards not to transmit to the card experiencing congestion.

An input and output FIFO also serve all the duplex ports of the I/O card. Cells are written from/to the input FIFO to/from the shared memory at a rate matching the speed of the I/O ports. The switch supports the following interfaces: 1 OC-3, 2 DS-3, 6 DS-1, 4 E1, 6 E2, 5 Ethernet ports. Of the 5 Ethernet ports, four are 10 Mbps ports, and the fifth one can be configured either as 100 Mbps or as a 10 Mbps port.

We now proceed to examine briefly the set of queues maintained in the shared memory, which has a configurable capacity of up to one million cells. There are three queues for all the input ports, and 10 queues per output port, as shown in figure 6.27. An

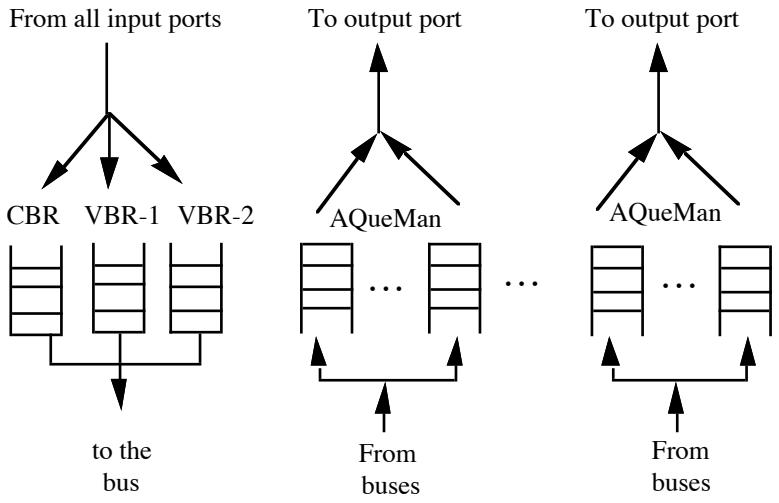


Figure 6.27: Queueing in the shared memory.

incoming cell from any of the input ports is queued into one of three queues, namely CBR, VBR-1, and VBR-2. These three queues are served to completion on a priority basis, with the CBR queue having the highest priority and the VBR-2 queue having the lowest priority. Traffic coming in from the two buses is queued into one of the 10 queues of the destination output port. Four of these queues are used for CBR traffic, namely, CBR-1, CBR-2, CBR-3, and CBR-4. The next five queues are used for VBR traffic,

namely VBR-1, VBR-2, VBR-3, VBR-4, and VBR-5. Finally, the last queue is used for UBR traffic. A proprietary scheduling algorithm, known as *AqueMan,* is used to schedule the order in which the cells are transmitted out of these 10 queues. The algorithm utilizes static priorities with additional scheduling rules based on the status of the queues. Specifically, queues CBR-1, CBR-2, CBR-3, CBR-4, and VBR-1 have top priority. That is, they are served to completion before the other five queues can be served. CBR-1 has the highest priority, CBR-2 has the next highest priority, and so on, with VBR-1 having the lowest priority among these five queues. When these five queues are empty, the scheduler will transmit a cell from the remaining five queues, i.e. VBR-2, VBR-3, VBR-4, VBR-5 and UBR, using an algorithm that takes into account the time a queue was served last (aging factor) and the number of cells in a queue (queue depth factor).

An I/O card is considered congested when the number of cells in its shared memory reaches a high water mark. At this point, entire queues are purged until the number of cells drops below a low water mark.

6.9 Performance evaluation of an ATM switch

The evaluation of the performance of an ATM switch is an important step prior to building a new switch or deploying a switch in the field. Typically, one is interested in quantifying the cell loss probability as a function of the load offered to the input ports. Other performance measures such as jitter and delay are also be of interest. It is relatively difficult to quantify jitter and delay, and in view of this, most of the performance studies of switch architectures have focused on calculating the cell loss probability.

A performance evaluation of a switch can be done by experimentation or by using modelling techniques. Experimentation involves carrying out real traffic measurements on a switch, whereas modelling involves building a simulation or a queueing model, which is then manipulated in order to obtain the desired performance measures. In order to carry out traffic measurements on a switch, one should able to reproduce realistic traffic loads and also the switch has to *exist*. Modelling is used when the switch does not exist, or when the specific configuration that will be employed in the field is not available.

Simulation techniques are easy to apply and one does not need to have good knowledge of queueing theory, though some knowledge of queueing theory may help design better simulation models!. Typically, a switch is dimensioned so that it will have a very low cell loss probability. This probability could be as low as 10^{-8}, that is, on the average there is one lost cell for every 100,000,000 arriving cells. One needs extremely long simulation runs in order to obtain a good statistical estimate of such a low probability. For instance, if no cell loss is observed after simulating a switch for 100,000,000 arriving cells, then that does not mean that the average cell loss probability is zero! One needs to simulate the switch for a number of arriving cells which is considerably larger than 100,000,000 in order to obtain a reliable statistical estimate. In view of this, estimating very low cell loss probabilities by simulation is not practical since it may be extremely time-consuming. Specialized techniques, such as *rare event simulation*, can be used to reduce the length of the required simulation run.

Queueing-based models, on the other hand, can run very fast on a workstation and they can be used to estimate very low cell loss probabilities. However, they are more difficult to develop, and they require good knowledge of queueing theory.

The way that one depicts the cell arrival pattern in a model, affects the performance of the switch. In many simulation and queueing-based models of ATM switch architectures it was assumed that the arrival of cells at an input port was Poisson or Bernoulli distributed. Also, it was assumed a uniform distribution of cells over the output ports. That is, if there are N output ports, then an arriving cell seeks each of these N output ports with probability 1/N. These assumptions make a switch architecture look good! That is, it can carry traffic loads at high link utilization without loosing too many cells. However, these arrival patterns are unrealistic and they should not be used in performance evaluation studies, since they give optimistic results. In general, ATM traffic tends to be bursty and correlated (see section 7.1) and it should not be modelled by a Poisson or a Bernoulli arrival process. As far as the output port destination probabilities are concerned, it makes sense to assume that a cell from input port i will go to output port j with some probability p_{ij}. This permits to test different destination patterns and cases where one or more output ports become *hot*. Also, one can assume that an entire train of cells goes to the same output port.

Problems

1. Draw an 8x8 banyan network, number the output ports, and give an example of how a cell can be self-routed from an input port to an output port.

2. Why are output buffering switches preferable to switches with input buffering?

3. In the buffered banyan switch described in section 6.2.2, is it possible to loose cells inside the switch fabric? Why?

4. Does the addition of a buffered banyan distribution network in front of a buffered banyan network eliminates external blocking? Why?

5. Consider the Batcher sorter network shown in figure 6.13. The following 8 cells with destinations 1, 4, 5, 3, 2, 1, 5, 8 are offered to the sorter at the same time. The first cell with destination 1 is offered to the top input port of the sorter, the second cell with destination 4 is offered to the second input port of the sorter, and so on. In what order they will appear at the output ports of the sorter?

6. Consider the switch architecture shown in figure 6.17.
   a. Explain why it is non-blocking
   b. Why is it necessary to have buffers at the output port?

7. Consider a shared-memory switch architecture and assume that the total number of cells that can be accommodated in the switch is B. Why is there an upper bound, less than B, on the size of each linked list?

8. Consider an NxN shared-memory switch. The speed of each incoming link and each outgoing link is V Mbps. An incoming cell is first processed (i.e. header conversion, addition of a tag in front of the cell used for internal purposes), then it is written in the memory. All incoming cells from the input ports are processed by the same CPU. How much time the CPU has to process a cell in the worst case, i.e., when cells arrive back-to-back on all input ports? (Assume that the CPU is not busy when a read or write to the shared memory is performed.

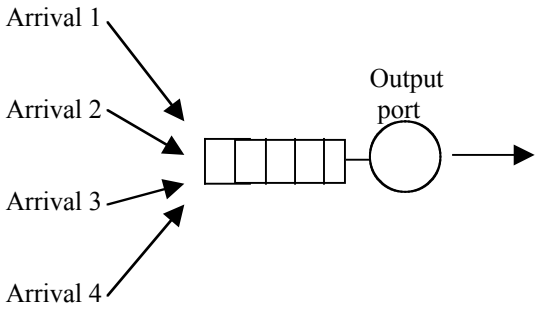
9. Consider an NxN shared memory switch. The speed of each incoming link and each outgoing link is V Mbps. What is the minimum speed of the memory in order to switch all the incoming traffic?

10. Consider an NxN shared medium switch. The speed of each incoming link and each outgoing link is V Mbps. What should be the speed of the bus in order to keep up with its incoming links?

A simulation model of an ATM multiplexer – Part 1

The objective of this project is to simulate the queueing model shown in the figure below that represents an output buffer of a non-blocking 4x4 ATM switch. This queueing model is known as the *ATM multiplexer*. For simplicity, we will assume that all the arrival processes are Bernoulli. In a follow-up simulation project described in the next Chapter, you will develop this simulation model further by introducing more realistic traffic models.

You can either write your own simulation program following the instructions given below, or use a simulation language.



The ATM multiplexer

Project description

The buffer has a finite capacity of B cells, and it receives cells from four different arrival streams, numbered 1 to 4. Each arrival stream is associated with an input port, and represents the stream of traffic from that input port destined for the output port under study. The buffer is served by a server which represents the output port.

Each arrival stream is slotted. That is, time is divided into equal time periods, known as *slots*. A slot is long enough so that a cell can completely arrive at the queue. Also, the time it takes to serve a cell from a queue by the output port, i.e. transmitted out, is also equal to one slot. We will assume that each arrival stream is a Bernoulli process. That is, a slot carries a cell with probability p or it is empty with probability 1-p.

The slots of each arrival stream and the service slot are all synchronized. That is, by the end of a slot, up to four new cells (one per arrival stream) may arrive at the buffer and one cell may depart from the buffer. An arriving cell is first buffered and then it is transmitted out. If a cell arrives at slot i to find the buffer empty, it will not depart from the buffer in the same slot. Rather, it will depart in slot i+1.

Task 1:

Write a simulation model to simulate the above system, with a view to estimating the cell loss probability as a function of p. A cell is lost if it arrives at a time when the buffer is full.

Plot several curves, each corresponding to a different value of B, in order to show how the cell loss probability varies as a function of p.

Task 2:

Augment your simulation to allow for different quality-of-service (QoS) queues in the output buffer. Specifically, we will assume that an arriving cell joins one the following four queues: CBR queue, RT-VBR queue, NRT-VBR queue, and UBR queue. A cell joins the CBR queue with probability p_{CBR}, the RT-VBR with probability $p_{RT\text{-}VBR}$, the NRT-VBR queue with probability $p_{NRT\text{-}VBR}$, and the UBR queue with probability p_{UBR}.

The CBR queue has the highest priority and the UBR queue has the lowest. These queues are served by the output port using the static priority rule described in section 6.7. That is, at the beginning of each slot, the cell to be transmitted out is selected from the CBR queue using the FIFO policy. If it is empty, then it is selected from the RT-VBR queue using the FIFO policy, and so on. If all queues are empty, then no cell is transmitted out.

The four queues share the total buffer space B associated with the output port. In order to make sure that each queue i does not grow too big so that it takes up most of the buffer space B, we impose a low bound L_i and an upper bound U_i. The upper bound is used to limit how big the ith queue can grow, and typically, it is selected so that the sum of the upper bounds of the four queues is larger than B. The lower bound L_i can be seen as a dedicated buffer permanently allocated to the ith queue. That is, the L_i buffer spaces are only used to store cells belonging to the ith queue. If L is the sum of the lower bounds of the

four queues, then we have that L<B. Therefore, the total number of buffer spaces that can be shared by the four queues is B-L.

In order to clarify the use of the upper and lower bound, let us consider the ith queue and let us assume that it is empty. The first cell that arrives to the ith queue, will be stored in the buffer (i.e., it will take up one of the L_i spaces) and the total number of dedicated buffers associated with the ith queue will be reduced by one. This will continue until all dedicated buffer spaces have been used up. When a cell arrives to the ith queue at a time when all dedicated buffer spaces have been used up, the cell will be accepted if the following two conditions are met:

a. The total number of cells in the ith queue is less than its upper bound U_i.
b. For each queue i, calculate the number of cells X_i which is over its lower bound L_i. (If the number of cells in the ith queue is less or equal to L_i, then X_i=0). Then, the sum $X_1+X_2+X_3+X_4$ has to be less than B-L.

Calculate the cell loss probability for all the cells arriving at the ATM multiplexer, and for each QoS queue, as a function of p for given values of the probabilities p_{CBR}, $p_{RT\text{-}VBR}$, $p_{NRT\text{-}VBR}$, and p_{UBR}, and for given L_i and U_i, i=1,2,3,4.

Plot several curves, each corresponding to a different value of B, p_{CBR}, $p_{RT\text{-}VBR}$, $p_{NRT\text{-}VBR}$, p_{UBR}, and L_i and U_i, i=1,2,3,4, in order to show how the cell loss probability varies as a function of p.

Structure of the simulation model

The simplest way to construct the simulation model is to use the unit-time design, with the unit time equal to a slot.

For task 1, do the following for each slot:

a If the buffer has at least one cell, then one cell departs. Update the total number of cells in the buffer.
b. Draw a pseudo-random number r, where 0<r<1. If r<p, then arrival stream 1 has a cell. Else, no arrival from stream 1.
c. Repeat above step for arrival streams 2, 3, and 4.
d. Calculate how many of these arriving cells will be admitted to the buffer, based on the current number of cells in the buffer. Keep a counter for the total number of arrivals and another counter for the total number of lost cells. Update the number of cells in the buffer.
e. Simulate for a total number of arrived cells, N, and print out, the ratio: number of lost cells/N.

For task 2, follow the same steps, but modify the rules for admiting a cell to the buffer and serving a cell from the buffer as explained in the previous section. Use the following procedure to determine which queue an admitted cell will join:

a. Draw a pseudo-random number r, where 0<r<1.
b. If $r< p_{CBR}$, then the cell will join the CBR queue.
c. If $p_{CBR} <r< p_{CBR} + p_{RT\text{-}VBR}$, then the cell will join the RT-VBR queue.
d. If $p_{CBR} + p_{RT\text{-}VBR} <r< p_{CBR} + p_{RT\text{-}VBR} + p_{NRT\text{-}VBR}$, then the cell will join the VRT-VBR queue.
e. If $r > p_{CBR} + p_{RT\text{-}VBR} + p_{NRT\text{-}VBR}$, then it will join the UBR queue.

(Remember to draw a new random number each time you need one!)