Part I

Foundations of LAN Switches

In This Part

CHAPTER 1

Laying the Foundation

Before we delve into the details of Local Area Network (LAN) switch operation, you need to consider the foundation on which LAN switches are built. This chapter examines four important building blocks that will be indispensable to your understanding of LAN switches in later chapters:

- Network architecture
- Device addressing
- LAN technology
- LAN standards

Each is considered specifically in the context of Local Area Networks and its relevance to LAN switching.

In addition, this chapter introduces the terminology that will be used consistently throughout the book. Very often, speakers, writers, equipment vendors, and network operations personnel use different sets of terms to describe the elements and behavior of computer networks: Is it an Ethernet frame or an Ethernet packet that is sent by a station?[1] While a name in itself is never inherently wrong — speakers and writers can define their own terminology any way they want — we need to agree on the meaning of a number of key words and phrases so that we can unambiguously describe and understand the behavior of network protocols and devices. We have tried throughout this book to use terminology in a way that both reflects common industry usage and is technically accurate. When there is a conflict between these points of view, we have opted for technical correctness. In any case, we have tried to be consistent and unambiguous.

[1]See section 1.5.2 for the answer.

It is not possible to provide a novice-level tutorial on every facet of networking that may be relevant to LAN switches. This book is not intended to be an introduction to computer networks; it is a comprehensive treatise on the design, operation, and application of switch technology in LANs. Most of the discussions here and in later chapters presume that the reader has some experience with networks and LAN technology. While this first chapter does provide background information, it is not intended as a primer, but as a reminder of the technologies and concepts on which later chapters build.

1.1 Network Architecture

The art of networking comprises a wide range of operations and technologies. Casual end users may think that ''the network'' is the browser or e-mail screen interface; this is all that they know (and from their perspective, probably all that they need to know) about networking. Programmers writing application code that must communicate among multiple machines may need to know about the programming interfaces and network facilities provided by the local operating system, but are generally unconcerned about the actual mechanisms used to deliver messages. Designers of high-speed optical fiber links used to interconnect network routers and servers should not have to worry about the data structures in the e-mail messages that may traverse a link.

In addition, the applications, functions, and technologies of networking are constantly changing. Every year, new ways of increasing the data rate of the communications channels in which our networks operate are introduced. New applications are constantly being written that use existing network facilities to provide improved or even revolutionary new services for users. You need to make sure that advances in one area of network technology are not constrained by limitations in other areas. For example, you want to be able to install a higher-speed communications link without having to wait for a new application or protocol to be designed that can take advantage of that link. Similarly, you want to ensure that the new communications link does not cause previously working applications to fail because those applications depend on some idiosyncrasy endemic to the older technology.

The key to achieving these goals is to separate the totality of network functions into discrete partitions called layers. Layering allows the appropriate technology to be applied to each function and to be changed without unduly affecting other layers. The number of layers is rather arbitrary; the issue is separation of functions. Architectural layers are defined such that each layer provides a set of distinct, related functions. Ideally, these functions are grouped such that layers can be as independent of each other as possible; only a minimum of information should have to pass between layer entities.

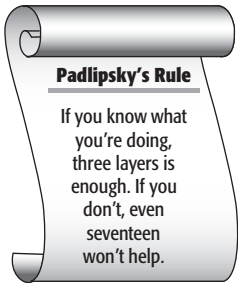


Figure 1-1 depicts the Open Systems Interconnect (OSI) model of network layering developed during the late 1970s and formally standardized in [ISO94]. It comprises seven layers of network system functions.

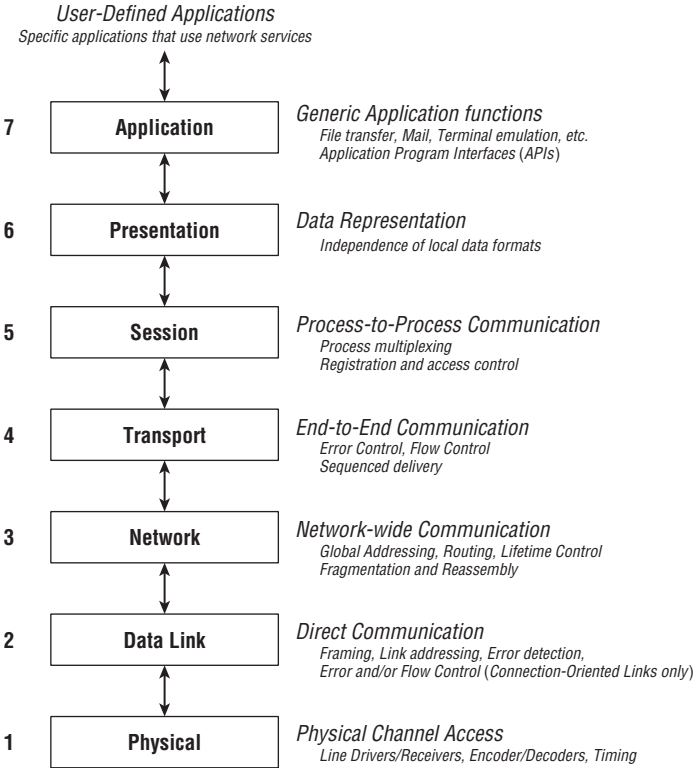


Figure 1-1 OSI reference model for network communications

In the sections that follow, we will take a look at the functions provided by each of these layers, with particular concern for their relevance to LANs and LAN switches.

1.1.1 Physical Layer

The Physical layer serves requests sent down from the Data Link layer (described in the following section), and comprises those elements involved with the actual transmission and reception of signals from the communications medium. The functions provided typically include line drivers and receivers, signal encoders and decoders, clock synchronization circuits, and so on. The exact nature of the device(s) implementing the Physical layer is a function of the design of the communications channel and the physical medium itself.

Examples of Physical layer interfaces are Token Ring, Ethernet, and FDDI. The Physical layer is also concerned with the actual transmission medium, such as network connectors, cabling types, cabling distance factors, and other mechanical considerations.

While a given networking device (for example, a LAN switch) must obviously include the circuitry needed to connect to the communications channel on which it is to be used, the nature of that channel has little impact on the higher-level operation of the device. For example, a LAN switch performs the same functions regardless of whether it is connected to an optical fiber channel operating at 1,000 Mb/s or a twisted pair copper wire channel operating at 10 Mb/s.

1.1.2 Data Link Layer

The Data Link layer provides services that allow direct communication between devices across the underlying physical channel. The communication can be point-to-point in nature (exactly two communicating stations) or point-to-multipoint (one-to-many), depending on the nature and configuration of the underlying channel.

In general, the Data Link layer must provide mechanisms for:

- **Framing:** The Data Link typically must provide a way to separate (delimit) discrete message transmissions (frames) in the Physical layer symbol stream.
- **Addressing:** Particularly when communicating among multiple stations on a common communications channel (as is typical of LANs), there must be a means to identify both the sender and target destination(s).
- **Error detection:** It is theoretically impossible for the underlying communications channel to be totally error free. While we hope that most transmissions will be received intact, there is always some residual rate of data errors, regardless of the technology employed within the Physical layer.[2] It is important that corrupted data not be delivered to higher-layer clients of the Data Link. At a minimum, the Data Link layer must detect virtually all errors. Depending on the design of the Data Link, it may either discard corrupted data (leaving error recovery to higher-layer entities) or take explicit action to correct or recover from the

[2]Ultimately, quantum (thermal) noise will introduce random errors into any communications channel, regardless of the quality of the components used or the lack of external sources of interference.

data corruption. These two modes of operation are explored in detail in section 1.1.8.1.

In general, LAN technology exists primarily at the Data Link and Physical layers of the architecture. Likewise, the functions performed by a LAN switch occur mainly at the Data Link layer.[3] As a result, this book focuses heavily on Data Link operation and behavior. Throughout the book, we show you how LAN switches significantly enhance the power and capabilities provided by the Data Link layer. As part of the design of these new features and the devices that implement them, you must often consider the impact of such Data Link modifications on the operation of higher-layer protocols.

Because it is so crucial to your understanding of LANs and LAN switching, section 1.1.8 provides an in-depth look at Data Link layer operation.

1.1.3 Network Layer

While the Data Link is concerned with the direct exchange of frames among stations on a single communications channel, the Network layer is responsible for station-to-station data delivery across multiple Data Links. As such, this layer must often accommodate a wide variety of Data Link technologies (both local and wide area) and arbitrary topologies, including partially complete meshes with multiple paths between endpoints. The Network layer is responsible for routing packets across the internetwork, usually through the action of intermediate relay stations known as routers (see section 1.5.3).[4]

Examples of Network-layer protocols include the Internet Protocol (IP) used in the TCP/IP suite, the Internetwork Packet Exchange protocol (IPX) used in NetWare, and the Datagram Delivery Protocol (DDP) used in AppleTalk.

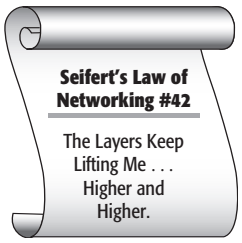
1.1.4 Transport Layer

In most network architectures, Transport is where the buck stops. While the underlying communications facilities may cause packets to be dropped, delivered out of sequence, or corrupted by errors, the Transport layer shields higher-layer applications from having to deal with these nasty details of network behavior. Transport provides its clients with a perfect pipe: an error-free, sequenced, guaranteed-delivery message service that allows process-to-process communications between stations across an internetwork, as long as a functioning communications path is available.

[3]Chapter 4 discusses the operation of so-called multilayer switches (MLS), which implement functionality at other layers.

[4]Pronunciation guide:
rou-ter (rō ō ′ter) noun
A device that forwards traffic between networks.
rou-ter (rou′ter) noun
A machine tool that mills out the surface of metal or wood.

To provide this end-to-end reliable delivery service, Transport often needs to include mechanisms for connection establishment, error recovery, traffic pacing (flow control), message sequencing, and segmentation/reassembly of large application data blocks. Examples of Transport protocols include the Transmission Control Protocol (TCP) of the TCP/IP suite, the Sequenced Packet Exchange (SPX) protocol of NetWare, and the AppleTalk Transaction Protocol (ATP).



1.1.5 Session Layer

The Session layer provides for the establishment of communications sessions between applications. It may deal with user authentication and access control (for example, passwords), synchronization and checkpointing of data transfers, and so on. The Session layer serves requests from the Presentation layer and sends requests to the Transport layer.

The Session layer sets up, manages, and ultimately terminates communication between end users and end user applications. It is able to combine different data stream types coming from various sources and synchronize the data so the end users can all be on the same page (so to speak).

Examples of some of the more well-known protocols that provide services related to this layer are: Network Basic Output System (NetBIOS), Network File Systems (NFS), Secure Shell (SSH), Structured Query Language (SQL), Real-time Transport Protocols, and . . .well, you get the drift.

1.1.6 Presentation Layer

The Presentation layer is responsible for the problems associated with communication between networked systems that use different methods of local data representation. When implemented, this layer allows data to be exchanged between machines that store information in different formats while maintaining consistent semantics (the meaning and interpretation of the data). The Presentation layer serves requests from the Application layer and sends requests to the Session layer.

Some of the services performed at this layer are the compression, delivery, and formatting of data. Data encryption is also normally performed at the Presentation layer. Protocols providing services at this layer include AppleShare File Protocol (AFP), Motion Pictures Experts Group (MPEG), and Tagged Image File Format (TIFF).

1.1.7 Application Layer

The Application layer provides generic application functions, such as electronic mail utilities, file transfer capability, and the like. It also provides the Application Program Interfaces (APIs) that allow user applications to communicate across the network. Note that, contrary to popular belief, the QSI Application layer does not include the user's networking applications; from an architectural perspective, end user applications reside above the OSI reference model altogether. The Application layer provides the facilities that allow user applications to easily use the network protocol stack — that is, generic application services and programming interfaces.

From the perspective of a LAN switch operation, you rarely need to consider the operation of protocols above Transport. A well-designed and functioning Transport implementation effectively shields the higher layers from all of the vagaries of networking.

1.1.8 Layering Makes a Good Servant but a Bad Master

Many people in the networking industry forget that the industry-standard layered architecture is not the OSI reverence model, to be worshipped, but a reference model, to be used as a basis for discussion. They believe that the standard model is like the Seven Commandments passed down from a network deity, and that any system that does not conform to the structure of the model is evil, or at least fundamentally flawed. This is complete and utter nonsense. The model is just that: a model for describing the operation of networks. It is not a standard to which networking protocols must adhere, or an engineering specification to which network components or devices must conform. The OSI reference model provides you with a common framework to discuss and describe the complete set of functions that may be performed in a practical network implementation. It should not, however, constrain any implementation from doing what is appropriate and right for its target application environment. Architectural purity may look nice on paper, but it doesn't make a system work properly.

In particular, our understanding of layered architecture should always be tempered by the following:

- Not all layers are required at any given time. In many environments, the functions provided at some layers of the OSI model are simply not needed. For example, when transferring ASCII e-mail files between

machines, there is no real need for a Presentation layer because ASCII is universally understood. The layer can be eliminated completely with no loss of functionality. The standard TCP/IP protocol suite eliminates both the Session and Presentation layers, yet it works quite well.[5]

- Any function not performed at one layer can be pushed up to a higher layer. Just because a network system does not implement some OSI-prescribed function in an exposed module using the OSI name for that layer does not mean that the system must live without the use of that function. For example, if a protocol suite does not include a Presentation layer, this does not imply that all communicating systems must use the same method of local data representation.[6] Lacking a Presentation layer, the burden of data format conversion between dissimilar systems just becomes the responsibility of the application that is providing the data transfer. This is, in fact, common practice.
- Don't confuse architecture with implementation. Even if the architecture of a network device can be presented in a layered fashion according to the OSI model, this does not mean that the implementation of that device must necessarily be partitioned according to the architectural layering. Architecture defines the functional boundaries of a system. Implementation should follow technology boundaries. In many cases, it is perfectly acceptable for software modules to cross layer boundaries. A single segment of code may implement the functionality described by multiple layers; there may be no exposed interfaces between certain layer entities. The tradeoff here is modularity versus performance. In a system highly constrained by processing power and/or memory, it may even be necessary and appropriate to write an entire protocol stack in one software module.[7]

This dichotomy between architecture and implementation is true for the hardware as well as the software components of a system. For example, many manufacturers produce integrated circuits designed to provide an interface to a local area network (LAN chip sets). Rarely does it make sense to build a ''Data Link chip'' and a ''Physical layer chip.'' The partitioning of functions between

[5]To be fair, TCP subsumes some of the OSI Session layer functions into the Transport layer.

[6]This is a good example because the Presentation layer is rarely implemented as shown in the OSI reference model.

[7]A good example is the highly unlayered implementation of DECnet for DOS-based systems. In an attempt to allow PCs to communicate with VAXen in the 1980s, Digital Equipment Corp. developed a DECnet protocol suite implementation that could run on an 8088-based computer with no more than 640 KB of memory. In order to provide even marginally acceptable performance, the implementation had few exposed interlayer interfaces. This avoided a lot of the context switching required with strictly layered implementations, at the expense of having a plate of difficult-to-maintain spaghetti code.

devices in a chip set is determined by technology (analog versus digital process, clock domains, and so on), power consumption, and portability to multiple device applications rather than by any arbitrary layering prescribed by the OSI model.

An application can use the network at any layer, not just the Application layer. Just because the OSI model defines one layer specifically for application interfaces does not mean that real applications must use that layer as their entry point to the network. An application can access network services at any layer in the hierarchy as long as it is willing to accept the level of service provided at that layer. For example, an application that operates across only a single link can interface directly to the Data Link layer; there is no need to incur the overhead of Network and higher-layer processing if those functions are not needed by that particular application.

Similarly, there is no need for communications to pass through every layer between an entity and the underlying physical channel, even if they exist in the protocol suite in use. Layers can be skipped if the functionality adds no benefit. Figure 1-2 depicts an example of a multi-protocol end station architecture incorporating TCP/IP, Local Area Transport (LAT), AppleTalk, and IPX. Note that not all seven layers are present in any of these protocol suites. In addition, many higher-layer protocols and applications skip layers where appropriate, and some modules encompass the functionality of multiple layers.

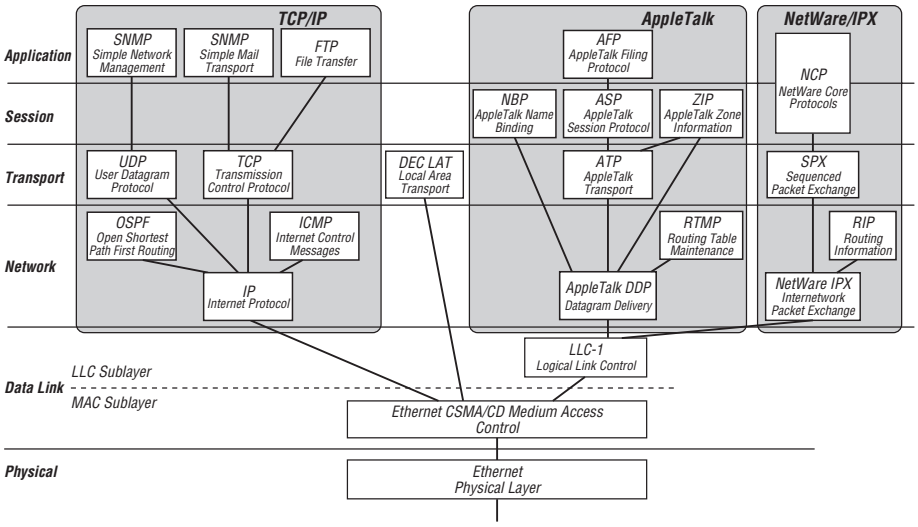


Figure 1-2 Multi-protocol layered architecture example

These important concepts are often lost between the study and the practice of networking. Layered architecture is how we describe the behavior of a

system; implementation is how we actually build it. Neither one should control the other. In fact, no popular network system in use today exactly maps, module-for-module, to the OSI model. Any attempt to build such a system (or to describe a system as if it did map this way) is futile; this is not the purpose of the model.

1.1.9 Inside the Data Link Layer

Because this is a book about LAN switches, we need to examine the innards of the Data Link more than any other layer in the architecture. In this section, we look at the different modes of Data Link layer operation, the architectural subdivision of the Data Link layer, and the operation of the Logical Link Control protocol (LLC).

1.1.9.1 Modes of Operation

Data Links can operate in either of two basic modes: connectionless or connection-oriented.

1.1.9.1.1 Connectionless Operation

A connectionless link provides best-effort service; frames are sent among devices and should be properly received with high probability, but no guarantees are made and no mechanisms are invoked to recover from errors if they do occur. Error detection will prevent corrupted frames from being delivered to a higher-layer client (to the level of robustness provided by the error check algorithm). However, in the event of an error, it is not the connectionless Data Link's responsibility to invoke retransmissions or other recovery mechanisms; connectionless links do not provide error control.

Similarly, if a target destination is unable to receive a frame due to lack of resources (for example, buffer memory), it is not a connectionless Data Link's responsibility to recover from this loss, or even to prevent transmission when such resources are not available; connectionless links do not normally provide flow control.

A connectionless link thus operates open loop; no feedback is provided from the receiver to the sender. No acknowledgments are generated, no information is provided about buffer availability, and no retransmission requests are produced in the event of frame loss. If connectionless operation is in use at the Data Link layer, and some higher-layer application requires a guarantee of successful data delivery, then reliable delivery mechanisms must be provided at a higher layer (typically Transport) or within the application itself.

1.1.9.1.2 Connection-Oriented Operation

A connection-oriented link usually provides for both error and flow control between the communicating partners. In general, this will require that the partners maintain a certain amount of state information about this ongoing stream of information being exchanged. In the event of an error, there must be some way to identify the particular frame(s) that were not received and to request their retransmission. Thus, sequence numbers are usually assigned to frames, and the communicating stations must keep track of which frames were received and which are either in process or require retransmission.

Prior to information exchange, partners in a connection-oriented link must generally invoke a call setup procedure, which establishes the link and initializes the sequence state information. Once set up, data can be exchanged, with error and/or flow control procedures operating to ensure orderly and error-free exchange during the course of the call. Once completed, the call can be torn down and the resources made available for other communications.

A connection-oriented link operates closed loop; once a connection is established, there is a continual exchange of data and feedback control information in both directions. Errors and frame loss can be corrected relatively quickly; the loop time constant need only accommodate the processing and propagation delays of the single link over which communication is occurring.

1.1.9.1.3 Connectionless Versus Connection-Oriented Operation

A connectionless link provides no guarantees regarding frame delivery to the target destination(s). Frames will be delivered with high probability, but there are sure to be some frames that are not delivered because of errors in the physical channel or buffer unavailability at a receiver. A connection-oriented link provides some assurance of proper data delivery to its client — unless the physical channel is inoperative (there is nothing that a Data Link protocol can do to deliver data across a non-functioning channel!). As always, there is no free lunch; a price must be exacted for this assurance, as explained in the following list:[8]

- **Protocol complexity:** The link protocol must necessarily be more complex for a connection-oriented link than for a connectionless link. A connectionless protocol can consider each frame completely independent of any other. A connection-oriented protocol must generally provide mechanisms for frame sequencing, error recovery, and flow control. Typically, this involves a Positive Acknowledgment and Retransmission (PAR) protocol for error recovery and either a

[8]Actually, there are two kinds of free lunches: those you have already paid for and those you have not yet paid for. If you are using a connection-oriented link, you have already bought lunch. If you are considering using a connection-oriented link, remember that those luscious pictures of dishes on the menu have prices next to them.

sliding window or buffer credit scheme for flow control. In addition, the connection-oriented protocol needs facilities for call setup and teardown, and possibly for restoration of a disrupted call.

- **Station complexity:** Stations participating in a connection-oriented link protocol must implement all of the functions demanded by that protocol (call setup/teardown, error control, flow control, and so on). For performance reasons, these functions are generally implemented in hardware within the link controller; the delay imposed by a software-based connection-oriented link protocol implementation is often unacceptable, particularly on high-speed links. This additional hardware adds to the cost of the link controller in every station using the protocol.
- **Connection-orientation:** The use of a connection-oriented link protocol presumes a connection orientation on the part of the higher-layer protocols and/or applications in the system. A connection-oriented link protocol may be appropriate if the communication indeed comprises a long-term stream of information exchanges. However, if the communicating applications exchange information only sporadically, the overhead of call setup and maintenance can be excessive. Examples of such sporadically communicating applications include most Network-layer routing protocols (RIP, OSPF, and so on) and infrequent polling of devices for network management statistics (SNMP).

Connectionless links are uncomplicated; minimal overhead is required in the frames exchanged, and the link hardware can be simpler and therefore lower in cost. Whether connectionless operation is acceptable depends primarily on the probability that frames will be delivered properly under normal operation. If the vast majority of frames are successfully delivered, connectionless operation is incredibly efficient. For the boundary case of a missing frame, higher-layer protocol mechanisms can still recover and maintain reliable delivery for the client application(s). Performance will suffer when errors occur, but if errors do not occur often, the effect is insignificant.

Connection-oriented links incur all of the overhead and complexity required for reliable delivery whether or not the underlying channel or the communicating devices ever need to invoke those mechanisms. If the communications channel is error prone, or if the communicating devices can be easily swamped by the speed of the channel (i.e., they have inadequate resources to prevent buffer overflow at the Data Link layer), then a connection-oriented link can provide efficient operation. Low-level hardware prevents such problems and limitations from propagating beyond the Data Link layer facilities; higher-layer protocols and applications are unaware that errors are being corrected and buffer overflow is being prevented.

The communications channel in a LAN environment is generally of exceedingly high quality. Unlike long-distance telephony circuits, microwave links, or satellite channels, LANs generally operate over carefully designed media in a controlled environment. The error rates encountered in a typical LAN are on the order of 1×10^{-12} or better. For a workgroup average frame length of 534 bytes [AMD96], this implies 1 lost frame due to bit errors for every 234 million frames sent. The complexity and overhead of a connection-oriented link are not easily justified for this level of errors. If the communications channel were an error prone wide area network (WAN) link with an error rate of 1×10^{-6} (one bit in a million in error), there would instead be 1 lost frame for every 234 frames sent. This is a much more significant level of frame loss and could easily justify the complexity of a connection-oriented Data Link protocol.

Thus, LANs generally use connectionless Data Link protocols. The notable exception is the IBM LAN architecture and its use of Token Ring; this is discussed in detail in Chapter 3, ''Bridging Between Technologies,'' and Chapter 6, ''Source Routing.''

1.1.9.2 Data Link Sublayering

LANs are somewhat special in that they often comprise a shared channel among many stations (as opposed to a point-to-point link, as provided by a telephony T-carrier). That is, in addition to providing a connectionless or connection-oriented service to its client, a LAN generally requires some means to arbitrate among the stations for use of the shared, common channel.

Thus, we separate the Data Link layer into two distinct sublayers, as depicted in Figure 1-3:

- **Logical Link Control (LLC):** This upper sublayer provides the Data Link service (connectionless or connection-oriented) to the higher-layer client, independent of the nature of the underlying LAN. In this manner, higher layer clients are relieved from having to deal with the details of the particular LAN technology being employed. They can use the same service interface to the Data Link, whether it is operating over an Ethernet, Token Ring, FDDI, or other technology.[9]
- **Medium Access Control (MAC):** This lower sublayer deals with the details of frame formats and channel arbitration associated with the particular LAN technology in use, independent of the class of service being provided to higher-layer clients by LLC.

[9]Note that a price is paid for this abstraction. With a uniform service interface, higher-layer clients can lose visibility into technology-specific features of the underlying LAN (for example, the priority access mechanism provided by Token Ring).

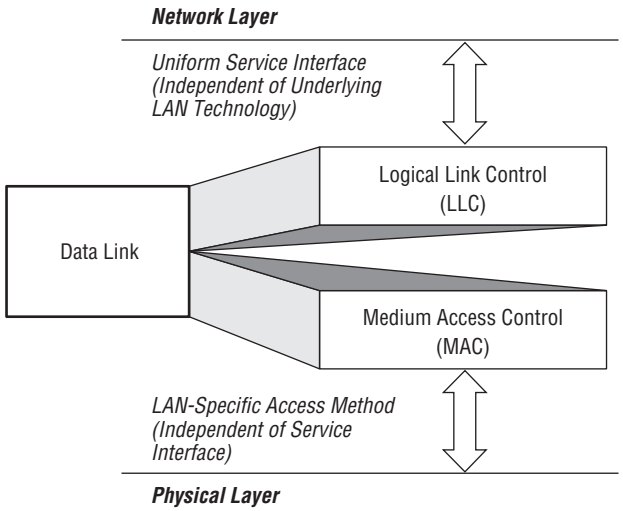


Figure 1-3 Data Link sublayering

1.1.9.3 Logical Link Control

The Logical Link Control protocol was developed and standardized within the IEEE 802.2 Working Group (see section 1.4.1) and provides for three types of service:

- **LLC Type 1: Connectionless Service.** This is a simple, best-effort delivery service. LLC-1 provides no call setup or maintenance procedures, no error recovery, and no flow control. The only protocol mechanism provided is for multiplexing of the Data Link to multiple higher-layer clients.
- **LLC Type 2: Connection-Oriented Service.** LLC-2 was derived directly from the High-Level Data Link Control protocol (HDLC) commonly used on wide area telecommunications links [IS093, ANSI79]. It operates from the same set of principles; the main differences are a reduction in the number of connection modes available and the inclusion of both source and destination client identifiers. LLC-2 includes procedures for call establishment and teardown, error recovery using Positive Acknowledgment and Retransmission, and flow control using a fixed-length sliding window of eight frames. Devices that implement LLC-2 must also implement LLC-1; connectionless operation is used to establish LLC-2 connections.
- **LLC Type 3: Acknowledged Connectionless Service.** LLC-3 is somewhat of a contrived, amalgamated service. It provides neither connections nor error or flow control, but does include support for

immediate acknowledgment of frame delivery. A client using LLC-3 can immediately detect whether an individual frame was properly delivered and take necessary action (for example, resubmitting the frame for transmission). In a true show of architectural impurity, LLC-3 was specifically designed to leverage a mechanism called Request-with-Response that is available in the IEEE 802.4 Token Bus LAN. Request-with-Response provides a low-level acknowledgment capability with very fast reaction time [IEEE90c]. As IEEE 802.4 Token Bus LANs never enjoyed widespread popularity, and applications that need an LLC-3 style of service never emerged, LLC-3 sees little (if any) commercial use.

Readers interested in the details of LLC protocol procedures should refer to [IEEE98b] for the complete set of specifications.

1.1.9.3.1 LLC Frame Format

Figure 1-4 depicts the format of an LLC frame for all three classes of service. The frame header comprises either 3 or 4 bytes; the longer version is used only for LLC-2 Information and Supervisory frames.

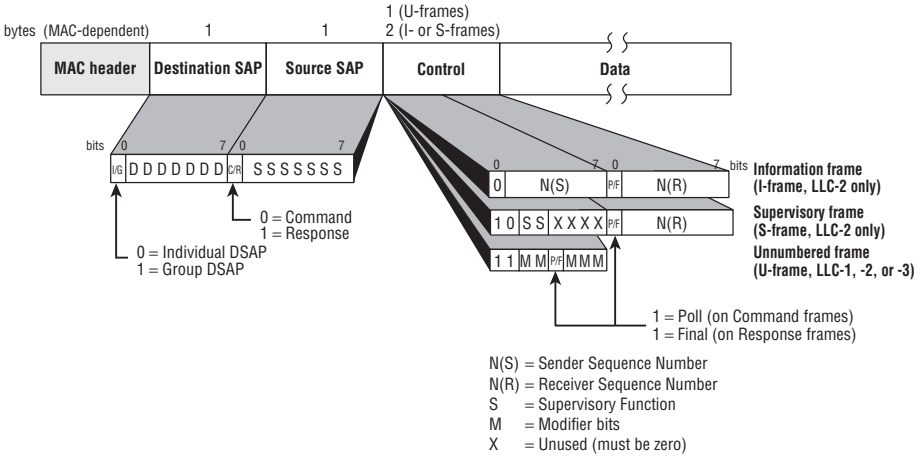


Figure 1-4 LLC frame format

LLC multiplexes among multiple higher-layer clients through the use of a Service Access Point (SAP) identifier. Both the client within the sender (Source SAP, or SSAP) and the target client within the receiver (Destination SAP, or DSAP) can be identified. SAP identifiers are 1 byte in length.

The first bit of the DSAP indicates whether the target is an individual client within the receiver or a set of multiple clients within the receiving station

that needs to see the received frame simultaneously.[10] This provision for SAP multicasting applies only to DSAPs; it is not even clear what would be meant by a multicast SSAP. The first bit of the SSAP is used to distinguish Command and Response frames.

1.1.9.3.2 SNAP Encapsulation

A problem arises with the use of LLC in its pure form. LLC SAPs (LSAPs[11]) are only 1 byte long; as a result, they can multiplex only among a maximum of 256 clients. However, as shown in Figure 1-4, the SAP space is further subdivided. Half of the space is reserved for group (i.e., multicast) SAPs, leaving only 128 multiplexing points for most purposes. Even within this restricted space, it is also common practice to use the second bit of the SAP to divide the space further, allowing for 64 publicly administered, globally unique SAPs and only 64 identifiers that can be locally administered for private use.

To overcome this limitation, an escape mechanism was built into the LLC SAP identifier. If the SAP is set equal to 0xAA, this indicates that the Sub-Network Access Protocol (SNAP) is in use.[12] As depicted in Figure 1-5, this scheme uses a standard LLC-1 header with fixed DSAP/SSAP values (0xAA) and provides an expansion of the SAP space through a pair of fields following the LLC-1 U-frame header. An Organizationally Unique Identifier (OUI) indicates the organization for which the Protocol Identifier (Pid) field is significant; the Pid is a higher-layer protocol identifier. (OUIs are explained in section 1.2.2.3.)

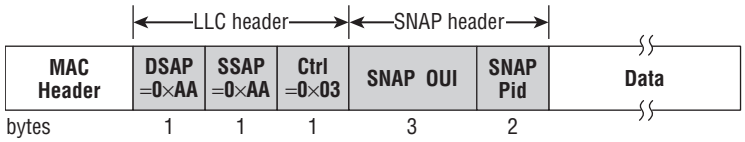


Figure 1-5 LLC-1/SNAP format

SNAP encapsulation allows any organization to have a set of 65,536 private higher-layer protocol identifiers, effectively eliminating the restriction of the 8-bit LSAP space.

[10]It is important to distinguish between the concept of a multicast DSAP, which identifies multiple clients within a single station that need to receive a given frame, and a multicast address, which identifies a set of stations on a LAN that need to receive the same frame (see section 1.2.2.1). While multicast addressing is an important LAN mechanism, no practical use for multicast DSAPs has ever emerged.

[11]LSAPs include both DSAPs and SSAPs.

[12]The use of the term Sub-Network here is misleading. It has nothing to do with the concept of subnetworks in the TCP/IP protocol suite. It is used primarily to make the acronym SNAP sound, well, snappy. It's even more fun when we get to talk about SNAP SAPs.

1.2 Addressing

By definition, a network comprises multiple stations.[13] The purpose of the network is to allow information exchange among these multiple stations. An address is the means used to uniquely identify each station either as a sender or receiver of information (or both).

Every layer that supports data exchange among multiple stations must provide a means of unique identification, that is, some form of addressing.[14] Many Data Link technologies (for example, LANs) allow multiple devices to share a single communications link; Data Link addresses allow unique identification of stations on that link. At the Network layer, you need to uniquely identify every station in a collection of multiple, interconnected links. Therefore, most network architectures provide for station addresses at both the Data Link and Network layers.

1.2.1 Local and Global Uniqueness

The only important characteristic of an address is its uniqueness; its purpose is to identify the particular sender and/or receiver of a given unit of information. Strictly speaking, an address need only be unique within the extent of the architectural layer at which it is operating. That is, a Data Link address need only be locally unique; it must unambiguously identify each station on a particular link (for example, a single LAN). It is not strictly necessary for a Data Link address to be able to distinguish stations on disjoint links because such stations cannot directly communicate at the Data Link layer.

At the Network layer, an address must uniquely identify each station in the entire internetwork. Network-layer addresses must therefore be globally unique. Traditionally, globally unique Network-layer addresses are constructed from locally unique Data Link addresses in a hierarchical manner, as depicted in Figure 1-6.

Note that each station's Data Link address (1, 2, 3, or 4) is locally unique on its particular LAN. While there are multiple stations with the same Data Link address (1, 2, 3, or 4), no two stations have the same address on the same LAN. Thus, there is no ambiguity when Station 1 and Station 4 on Network 1 communicate at the Data Link layer; there is only one Station 1 and only one Station 4 on the instant link comprising Network 1.

Communication among stations on separate LANs can be accomplished at the Network layer through the use of the internetwork routers. Each station's

[13] A network consisting of one station is about as useful as a single walkie-talkie.

[14] Strictly speaking, station addresses are necessary only when communications can occur among some proper subset of the totality of stations present. For example, if every transmission is always intended for receipt by every station, then there is no need to identify the target receiver(s). However, this is a rather unique case, somewhat artificially contrived. In practice, we need to provide station addresses at every layer where there are multiple communicating stations.

Network-layer address can be formed by a catenation of its locally unique Data Link address and a globally unique Network identifier (Network 1, 2, or 3). Thus, [Network 1 — Station 1] can communicate with [Network 3 — Station 1]; although they have the same Data Link address, there is no ambiguity at the Network layer.

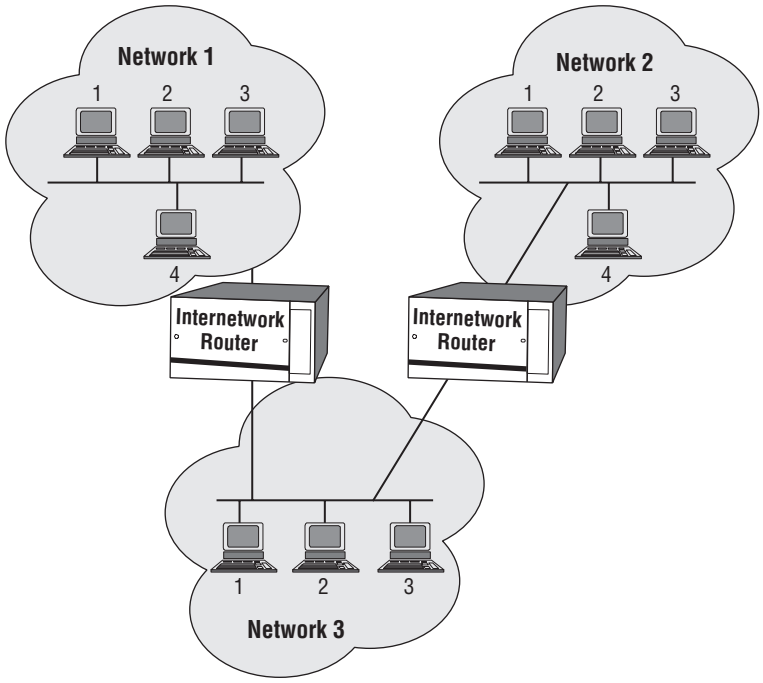


Figure 1-6 Hierarchical addresses

1.2.2 LAN Data Link Addresses

Until the 1980s, even large corporations rarely had more than a few hundred computers deployed throughout their entire organizations. Personal computers had not yet entered the workplace, and computing tended to be centralized under the control of a small cadre of knowledgeable technicians.

In this environment, using manual means to administer individual device addresses was both reasonable and manageable. Network and computer configurations did not change frequently, and the task of maintaining address uniqueness was not particularly onerous. As such, most Data Link technologies

at the time employed either 8- or 16-bit locally unique addresses. Address assignment was accomplished either through software configuration or by setting switches or jumpers on the network interface hardware. As a side benefit, the relatively small address space saved transmission and processing overhead.

In 1979, the team that designed the commercial 10 Mb/s Ethernet (including Rich Seifert, the author of the first edition of this book) recognized that this situation was about to change dramatically. The advent of personal computing was at hand; while a human administrator might be able to manage the address assignments of dozens of computers, this method offered little hope of success if there were tens of thousands of addressable devices in the enterprise, especially when devices were being constantly added and moved throughout the company.

In response to this anticipated situation, the Ethernet designers consciously took a different approach to Data Link layer addressing. Rather than trying to save transmission overhead by conserving bits, they instead opted to create a huge address space capable of providing a globally unique Data Link address to every device for all time. The Ethernet address space was designed to allow a unique address to be permanently assigned to every device that would ever attach to a LAN. Later, this same address scheme was endorsed and adopted by the IEEE 802 LAN Standards Committee in a slightly modified form.

Figure 1-7 depicts the format of the 48-bit addresses currently used in all industry-standard LANs. An address can identify either the sender (Source Address) or the target recipient (Destination Address) of a transmission. Because these addresses are used solely by the Medium Access Control sublayer within the Data Link, they are referred to as MAC addresses.

1.2.2.1 Unicast and Multicast Addresses

The 48-bit address space is divided into two halves:

- **A unicast address identifies a single device or network interface.**[15] When frames are sent to an individual station on a LAN, the unicast identifier of the target is typically used as the destination address

[15]There are actually two philosophies for interpreting unicast addresses. One philosophy follows the premise that a unicast address identifies a device (e.g., a workstation or server) as opposed to a network interface installed within the device. Under this philosophy, when a device has multiple interfaces, it uses the same address on all of them. This approach was used in the original Xerox Network System (XNS) and most Sun Microsystems products. The other philosophy, that an address uniquely identifies the interface rather than the device, sees more widespread application today, and is the model assumed in this book. Using the address-per-interface philosophy, a device with multiple interfaces will have multiple unicast addresses assigned to it. Both philosophies are valid in the sense that both can be made to work properly in a practical network. In neither case is there any ambiguity.

in all transmitted frames. The source address in transmitted frames (the identifier of the sender) is always unicast. Unicast addresses are sometimes called individual addresses, physical addresses, or hardware addresses; these terms are all synonymous.

- **A multicast address identifies a group of logically related devices.** Most LAN technologies provide many-to-many connectivity among multiple stations on a shared communications channel; multicast addressing provides the means to send a frame to multiple destinations with a single transmission. (See Chapter 10, ''Multicast Pruning,'' for a complete discussion of how multicast addresses are used.) Multicast addresses are sometimes called group addresses or logical addresses.

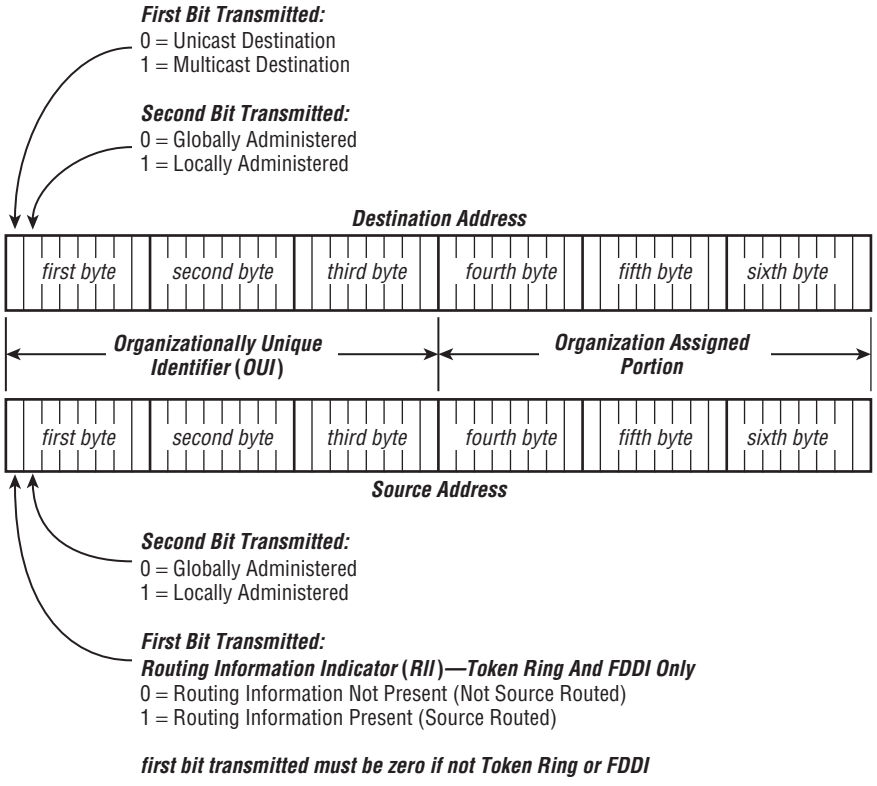


Figure 1-7 48-bit LAN address (MAC address) format

The first bit of a destination address (called the Individual/Group or I/G bit in the IEEE standards) indicates whether the target recipient is an individual destination (I/G = 0 for unicast) or a group of destinations (I/G = 1 for multicast). Thus, there are 2^{47} possible unicast addresses and 2^{47} possible multicast addresses. Source addresses are always unicast; a transmission always emanates from a single device.

The multicast mechanism provided by this address structure is considerably more powerful than the simple broadcast mechanism provided in many link technologies. Broadcasting allows a station to send a frame to all stations on a LAN simultaneously. Multicasting allows a station to send a frame to an arbitrary subset of all stations on the LAN; this prevents needlessly bothering (and using the resources of) those stations that are not interested in, or unable to understand, the transmission. In fact, the broadcast address (the Ethernet address where all bits are set to 1, in the IEEE standards) is just one of the 2^{47} multicast addresses available.

The first bit of a source address is always 0 on any LAN technology other than IEEE 802.5 Token Ring or FDDI. On those two technologies only, the first bit of the source address can be used as a Routing Information Indicator (RII), indicating the presence of source routing information. Source Routing and the use of the Routing Information Indicator are discussed in detail in Chapter 6.

1.2.2.2 Globally Unique and Locally Unique MAC Addresses

In the original Ethernet design, there was no concept of locally unique MAC addresses. All addresses were intended to be globally unique; no mechanism was provided for local control or address assignment. When the Ethernet address scheme was incorporated into the IEEE LAN standards, political considerations forced the adoption of a means to allow network administrators to manually assign addresses in a locally unique manner. The second bit of an address (called the Global/Local or G/L bit in the standards)[16] indicates whether the identifier is globally unique (G/L = 0) or unique only to the LAN on which the station resides (G/L = 1). As discussed in section 1.2.2.3, globally unique addresses are assigned by equipment manufacturers at the time a device is produced. Locally unique addresses are manually assigned by a network administrator; it becomes the responsibility of that administrator to ensure that addresses on a given LAN are unique.

[16] And sometimes the ''IBM bit'' by network architects.

IS 48 BITS THE RIGHT NUMBER?

The 48-bit LAN address structure in common use today has served us well since its inception in 1980, but will it last forever? Will we need some new, wider address space soon because of the proliferation of network devices?

Probably not. A 48-bit address provides 2^{48}, or about 281 million million, unique points in the address space. Even allowing for half of these to be used for multicast addresses, and further eliminating half of what is left for locally unique assignments, there is still enough space for almost 12,000 network-addressable devices for every man, woman, and child on the planet. (Even you don't have that many computers on your desk!)

In other words, if the industry produced 100 million LAN devices every day of the year (more than 500 times the current level of production), it would still take nearly 2,000 years to exhaust the address space.

Granted, the way that addresses are assigned tends to make sparse use of the available space, but there is no concern that we will run out of addresses anytime soon. More important, *we will not run out of addresses in time for any of us who instituted this scheme to be blamed for it!*

1.2.2.3 How LAN Addresses Are Assigned

As shown in Figure 1-7, the 48-bit address is divided into two parts. The first 24 bits of the address constitute an Organizationally Unique Identifier (OUI). The OUI indicates the organization (typically a manufacturer) responsible for unique assignment of the remaining 24 bits of the address.[17] If a company builds devices that need globally unique addresses assigned to them (for example, network interfaces), the company must first obtain an OUI from the IEEE. This is a relatively straightforward procedure involving the filling out of a simple form and an exchange of currency.[18]

The organization obtaining the OUI thus has 16,777,216 globally unique unicast addresses (from the 24 bits in the organization-assigned portion of the address) available to assign to the devices it produces. Normally, each

[17]In theory, any organization can obtain an OUI and assign globally unique addresses for its own use; however, the vast majority of OUIs are obtained by network equipment manufacturers.

[18]Information on obtaining OUIs can be found at `http://standards.ieee.org`. In the period when the Ethernet standard existed prior to the completion of the IEEE LAN standards, OUIs were administered and assigned by Xerox Corporation; the $1,000 fee was actually structured as a one-time, royalty-free license to build products that incorporated Xerox's patented

network-addressable device is configured with a Read-Only Memory (ROM) that contains the 48-bit address assigned by the manufacturer. (See Chapter 9, ''Link Aggregation,'' for a discussion on how device drivers can use and/or override this burned-in address.) It is the manufacturer's responsibility to ensure that it assigns the 24 bits under its control in a unique manner. It is the IEEE's responsibility to ensure that the same OUI is never assigned to two organizations. When a manufacturer uses up most of its 16 million-plus addresses, it can obtain another OUI.[19]

Note that an OUI assignment provides its owner with both 2^{24} unicast addresses and 2^{24} multicast addresses. By changing the first bit of the address from a 0 to a 1, the OUI holder gains a multicast space for its private use.

The second bit of an OUI is the Global/Local bit; if it is set to 1 (by a network administrator manually assigning addresses), this means that the entire scheme of OUIs and ROM-resident globally unique addresses is being ignored and overridden for the device(s) being configured. A network administrator using this mechanism assumes total responsibility for ensuring that any such manually configured station addresses are unique.

When assigning OUIs, the IEEE ensures that the G/L bit is always a 0; that is, the IEEE does not assign OUIs in the locally administered address space. As stated earlier, the original Ethernet standard did not provide for locally unique address assignment. Before the IEEE standards were produced, Xerox Corporation administered Ethernet OUI allocations. Xerox had no way of knowing that the address format would be modified in the future by the IEEE, and during that early period a number of OUI assignments were made that had their second bit set equal to 1.[20] These were globally unique assignments that later appeared to be locally unique because of the IEEE's modification of the 48-bit address semantics. Some of these devices still exist in the field; it is important that network administrators who manually assign locally unique addresses not use this portion of the address space.

Ethernet technology. Since that time, Xerox has waived its rights to those original patents; the IEEE now charges $1,650 for an OUI. This fee serves three purposes:

- It pays for the paperwork and administration of OUIs.
- It is low enough to be insignificant for any serious producer of network equipment.
- It is high enough to discourage every graduate student taking a networking class from obtaining his or her own OUI (an important factor preventing the rapid depletion of OUIs!).

[19] Any organization that has shipped over 16 million network devices can afford another OUI.
[20] In particular, 3Com received an assignment of 02-60-8 C-xx-yy-zz, and Digital Equipment Corp. received an assignment of AA-AA-03-xx-yy-zz.

The IEEE maintains a list of current public OUI assignments on its Web site at `http://standards.ieee.org`. When obtaining an OUI, an organization may choose to keep its assignment private;[21] private assignments are not published. There are no private assignments that encroach on the locally administered address space.

1.2.2.4 Written Address Conventions

Strictly speaking, an address is a sequence of 48 bits. It is not a number; that is, it serves as a unique identifier but has no numerical significance. However, it is rather inconvenient (not to mention user-unfriendly) to have to write down a string of 48 ones and zeroes every time we need to show an address; a 48-bit serial format is a rather unwieldy piece of baggage to lug around. Therefore, we use a shorthand convention when writing MAC addresses. It is important to remember that this is only a writing convention for the convenience of humans and network architects — regardless of how it looks on paper, the address is still the 48-bit string.

Addresses are normally written as a sequence of 12 hexadecimal digits separated by hyphens:[22]

```
aa-bb-cc-dd-ee-ff
```

Each pair of hexadecimal digits represents 1 byte of the 6-byte (48-bit) address. The bytes are written from left to right in the same order as transmitted; that is, the `aa` byte is transmitted first, the `bb` byte second, and the `ff` byte last. The `aa-bb-cc` portion is the OUI; the `dd-ee-ff` portion is normally assigned by the manufacturer.

The order of the bits transmitted within each byte varies with the LAN technology being employed (see sections 1.3.1.4, 1.3.2.4, and 1.3.3.3). However, unless stated otherwise, addresses in this book are assumed to be in canonical format; that is, the bits within each byte are transmitted from the least significant to the most significant. This is the standard order on Ethernet LANs. Thus, the least significant bit of the first byte is the Individual/Group bit.[23] Chapter 3 contains an extensive discussion of the bit-ordering differences among LANs and their impact on LAN switch technology.

[21] If an organization chooses to keep the OUI assignment private, there is an additional fee.

[22] Some manufacturers use colons instead of hyphens.

[23] This convention allows unicast addresses to be easily distinguished from multicast addresses. If the second hexadecimal digit in an address (i.e., aa-bb-cc-dd-ee-ff) is even (0, 2, 4, 6, 8, A, C, or E), the address is unicast. If the second digit is odd (1, 3, 5, 7, 9, B, D, or F), the address is multicast.

1.3 LAN Technology Review

In the context of this book, switches are used primarily to interconnect LANs.[24] Historically, dozens of different types and variations of LAN technology have been commercially produced; many have also been formalized and approved by official standards organizations or have become de facto standards through market forces. However, only a small number of these LAN technologies have truly achieved widespread use.

In this section, we look at the three most popular LAN technologies in the order of their product volume and importance: Ethernet, Token Ring, and the Fiber Distributed Data Interface (FDDI). The vast majority of LAN switches are designed specifically for one or more of these three systems. At the end of the section we briefly consider some other, less-used LAN technologies.

1.3.1 Ethernet

Ethernet was originally conceived and implemented at Xerox Corporation in Palo Alto, California, in 1973. The lab prototype, developed by Dr. Robert Metcalfe (generally regarded as the ''father of Ethernet''), operated at 2.94 million bits per second. This Experimental Ethernet was used in some early Xerox products, including the Xerox Alto, the world's first networked personal workstation with a graphical user interface.

During 1979, Digital Equipment Corporation (DEC) and Intel joined forces with Xerox to standardize, commercialize, and promote the use of network products using Ethernet technology. This DEC-Intel-Xerox (DIX) cartel developed and published the standard for a 10 Mb/s version of Ethernet in September 1980 (Ethernet Version 1.0) [DIX80]. In 1982, a second revision of this standard was published (Version 2) [DIX82] that made some minor changes to the signaling and incorporated some network management features.

In parallel to the DIX work, the IEEE formed its now-famous Project 802 to provide a broader industry framework for the standardization of LAN technology. When it became clear that the IEEE 802 committee could not agree on a single standard for all Local Area Networks (the original mission), the committee subdivided into various Working Groups (WGs), each focusing on different LAN technologies. (See section 1.4 for a complete discussion of the IEEE 802 history and organization.) In June of 1983, the IEEE 802.3 Working Group produced the first IEEE standard for LANs based on Ethernet technology. With a few minor differences, this was the same technology embodied in the DIX Ethernet specification. Indeed, much of the language

[24] Interconnections among geographically separated LANs using Wide Area Network (WAN) technologies are discussed in Chapter 3.

of the two documents is identical. As the marketplace for Ethernet grew, this base standard was augmented with a set of repeater specifications and a variety of physical medium options: thinner coaxial cable for low-cost desktop attachments, unshielded twisted pair, optical fibers for inter-building connections, and so on.

In 1991–1992, Grand Junction Networks developed a higher-speed version of Ethernet that had the same basic characteristics (frame format, software interface, access control dynamics) as Ethernet but operated at 100 Mb/s. This proved to be a huge success and once again fostered industry standards activity. The resulting Fast Ethernet standard [IEEE95b] spawned another wave of high-volume Ethernet products. In 1998, a version of Ethernet was standardized that operated at 1000 Mb/s (Gigabit Ethernet) [IEEE98e]. Work is continuing on even higher data rates and additional features for Ethernet-based systems.

Literally hundreds of millions of Ethernet interfaces are deployed throughout the world in personal computers, servers, internetworking devices, printers, test equipment, telephone switches, cable TV set-top boxes — the list is huge and increasing daily. In many products, there is no intent to connect the device to a traditional computer network; Ethernet is used simply as a low-cost, high-speed, general-purpose communications interface.

1.3.1.1 Ethernet Medium Access Control

The purpose of any Medium Access Control (MAC) algorithm is to allow stations to decide when it is permissible for them to transmit on a shared physical channel. Ethernet uses a distributed algorithm known as Carrier Sense Multiple Access with Collision Detect (CSMA/CD). The IEEE 802.3 standard contains a precise, formalized specification of the CSMA/CD algorithm in Pascal code. Readers interested in the nitty-gritty details of CSMA/CD operation should refer to the standard itself [IEEE98a]. The description provided here is qualitative in nature, and does not take into account some of the low-level minutiae that must be considered in an actual implementation. A simplified flowchart of the CSMA/CD algorithm is provided in Figure 1-8.

1.3.1.1.1 Frame Transmission

When a station has a frame queued for transmission, it checks the physical channel to determine if it is currently in use by another station. This process is referred to as sensing carrier. If the channel is busy, the station defers to the ongoing traffic to avoid corrupting a transmission in progress.

Following the end of the transmission in progress (i.e., when carrier is no longer sensed), the station waits for a period of time known as an interframe gap to allow the physical channel to stabilize and to additionally allow time

for receivers to perform necessary housekeeping functions such as adjusting buffer pointers, updating management counters, interrupting a host processor, and so on. After the expiration of the interframe gap time, the station begins its transmission.

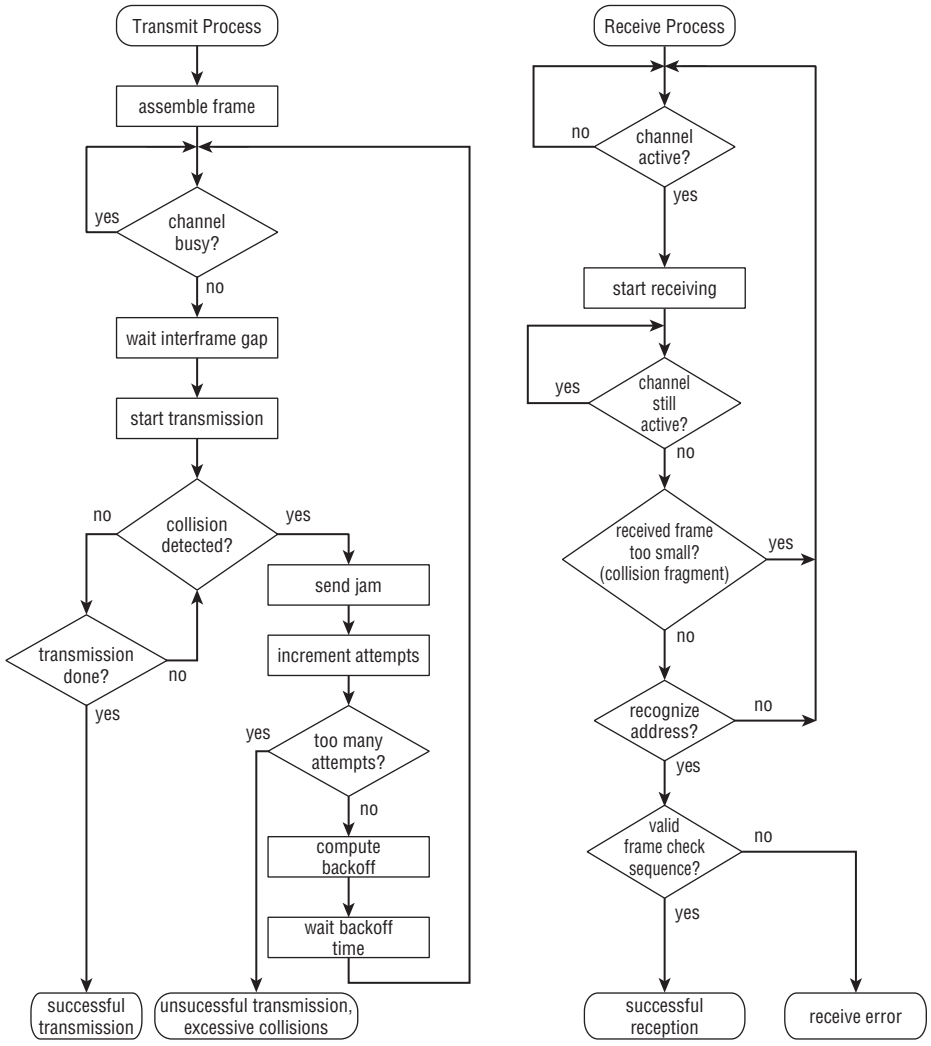


Figure 1-8 Ethernet MAC flow

If this is the only station on the network with a frame queued for transmission at this time, the station should be able to send its frame following the expiration of the interframe gap time with no interference from other stations. No other action is required, and the frame is considered delivered by the sending station

following the end of the frame transmission. The station can then go on to process the next frame in the transmit queue (if any), repeating the access control algorithm.

On the other hand, if there are multiple stations with frames queued at the same time, each will attempt to transmit after the interframe gap expires following the deassertion of carrier sense. The resulting interference is referred to as a collision. A collision will always occur if two or more stations have frames in their respective transmit queues and have been deferring to passing traffic. The collision resolution procedure is the means by which the stations on the Ethernet arbitrate among themselves to determine which one will be granted access to the shared channel.

In the event of a collision, all involved stations continue to transmit for a short period to ensure that the collision is obvious to all parties. This process is known as jamming. After jamming, the stations abort the remainder of their intended frames and wait for a random period of time.[25] This is referred to as backing off. After backing off, the station goes back to the beginning of the process and attempts to send the frame again. If the frame encounters 16 transmission attempts all resulting in collisions, the frame is discarded by the MAC, the backoff range is reset, the event is reported to management (or simply counted), and the station proceeds with the next frame in the transmit queue, if any.

The backoff time for any retransmission attempt is a random variable with an exponentially increasing range for repeated transmission attempts. The range of the random variable r selected on the nth transmission attempt of a given frame is:

$$0 \leq r < 2^k$$

where

$$k = \text{MIN}\ (n,\ 10).$$

Thus, the station starts with a range of 0 to 1 on the first collision encountered by a given frame, and increases the range to 0 to 3, 0 to 7, 0 to 15, and so on, up to the maximum range of 0 to 1,023 with repeated collisions encountered by the same frame. The backoff time is measured in units of the worst-case round-trip propagation delay of the channel, known as the slotTime.[26]

[25]It is the aborting of the remainder of the frame upon detecting a collision that gives Ethernet its high performance relative to ALOHA-type protocols [KLEIN75]. The detection and resolution of collisions occur very quickly, and channel time is not wasted sending the rest of a frame that has already been corrupted.

[26]The slotTime is 512 bit times for all Ethernet data rates except 1000 Mb/s. This translates to a quantum of 51.2 µs at 10 Mb/s and 5.12 µs at 100 Mb/s. On Gigabit Ethernet, the slotTime is defined as 4,096 bit times, or 4.096 µs [SEIF98].

1.3.1.1.2 Frame Reception

On the receive side, a station monitors the channel for an indication that a frame is being received. When the channel becomes non-idle, the station begins receiving bits from the channel, looking for the Preamble and Start-of-Frame delimiter that indicate the beginning of the MAC frame (see section 1.3.1.3). The station continues receiving until the end of the frame, as indicated by the underlying channel.

A receiving MAC discards any received frames that are less than one slotTime in length. This is because, by definition, these must be the result of a collision; valid frames will always be longer than the slotTime (i.e., the worst-case round-trip channel propagation delay). If the received frame meets the minimum length requirement, the Frame Check Sequence (FCS) is checked for validity and the frame discarded if the FCS does not match the proper value for the received frame. Assuming a valid FCS on a valid-length frame, the receiver will check the Destination Address to see if it matches either (1) the physical address of the receiving station (unicast) or (2) a multicast address that this station has been instructed to recognize. If either of these indicate that the frame is indeed destined for this station, the MAC passes the frame to its client (typically device driver software) and goes back to the beginning, looking for more frames to receive.

1.3.1.2 Ethernet Physical Layer Options and Nomenclature

When Ethernet first saw widespread commercial use, it supported only one data rate (10 Mb/s) and one physical medium (thick coaxial cable); the term Ethernet was therefore unambiguous in referring to this system. This clarity and simplicity was not to last, however. Ethernet was modified to operate over an increasing variety of physical media for reasons of cost, ease of installation and maintenance, use in electrically hostile environments, and so on. Later, the data rate changed, providing even more variations and physical media options. As such, there are a lot of very different communications systems available today, all called Ethernet.

In order to avoid having to say things like, ''10 Mb/s Ethernet using two pairs of Category 3 unshielded twisted pair,'' or ''Gigabit Ethernet on two optical fibers using long wavelength laser optics,'' the IEEE 802.3 committee developed a shorthand notation that allows us to refer to any particular standard implementation of Ethernet. A given flavor of Ethernet is referred to as:

n-signal-phy

where

- *n* is the data rate in md/s (10, 100, 1000, and so on).
- *signal* indicates either BASE, if the signaling used on the channel is baseband (i.e., the physical medium is dedicated to the Ethernet, with

no other communications system sharing the medium) or BROAD, if the signaling is broadband (i.e., the physical medium can simultaneously support Ethernet and other, possibly non-Ethernet services).[27]

- *phy* indicates the nature of the physical medium. In the first systems to which this notation was applied, phy indicated the maximum length of a cable segment in meters (rounded to the nearest 100 m). In later systems, this convention was dropped, and phy is simply a code for the particular media type.[28]

Table 1-1 (in the following section) provides a complete listing of the Ethernet reference designations that are currently defined.

Table 1-1 Ethernet Media Designations

1 MB/S SYSTEMS	
1BASE5	Unshielded twisted pair (UTP, 1 pair), 500 m maximum ("StarLAN")[1]
10 MB/S SYSTEMS	
10BASE5	Thick coaxial cable, 500 m maximum (Original Ethernet)
10BASE2	Thin coaxial cable, 185 m maximum ("Cheapernet")
10BROAD36	Broadband operation using three channels (each direction) of private CATV system, 3.6 km maximum diameter
10BASE-T	Two pairs of Category 3 (or better) UTP
10BASE-F systems	Generic designation for family of 10 Mb/s optical fiber
10 BASE-FL	Two multimode optical fibers with asynchronous active hub, 2 km maximum
10 BASE-FP	Two multimode optical fibers with passive hub, 1 km maximum
10 BASE-FB	Two multimode optical fibers for synchronous active hubs, 2 km maximum

(*continued*)

[27]The only Ethernet system using broadband signaling is 10BROAD36, which allows Ethernet to operate using three channels (in each direction) of a private CATV system. Other services (broadcast television, point-to-point modems, and so on) can use the other channels simultaneously. This system is not very popular, primarily due to its high cost.

[28]As part of this change in conventions, codes using the old style (length) convention do not use a hyphen between the signaling type and the physical medium designation (e.g., 10BASE5, 10BASE2); later designations always have a hyphen (e.g., 10BASE-T, 100BASE-FX) to show the change in meaning. In addition, the signaling designation is always capitalized. Now you can impress your coworkers and correct your boss when she writes 10BaseT instead of the strictly correct 10BASE-T. Please be sure to update your resume before trying this.

Table 1-1 (*Continued*)

100 MB/S SYSTEMS	
100BASE-T	Generic designation for all 100 Mb/s systems[2]
100BASE-X	Generic designation for 100BASE-T systems using 4B/5B encoding
100BASE-TX	Two pairs Category 5 UTP or STP, 100 m maximum
100BASE-FX	Two multimode optical fibers, 2 km maximum
100BASE -T4	Four pairs Category 3 (or better) UTP, 100 m maximum
100BASE -T2	Two pairs Category 3 (or better) UTP, 100 m maximum[3]
1000 MB/S SYSTEMS	
1000BASE-X	Generic designation for 1000 Mb/s systems using 8B/10B encoding
1000BASE-CX	Two pairs 150 Ω shielded twisted pair, 25 m maximum
1000BASE-SX	Two multimode or single-mode optical fibers using shortwave laser optics

[1]The 1BASE5 system was developed after the 10 Mb/s coaxial Ethernet but prior to 10BASE-T. It was never very successful commercially, and was rendered completely obsolete by 10BASE-T.
[2]Even though the 100 Mb/s family includes optical fiber, all are generically referred to as 100BASE-T.
[3]While there is an approved standard, no products using 100BASE-T2 signaling have ever been commercially produced.

1.3.1.3 Ethernet Frame Formats

Ethernet frames can take one of two forms, as depicted in Figure 1-9.

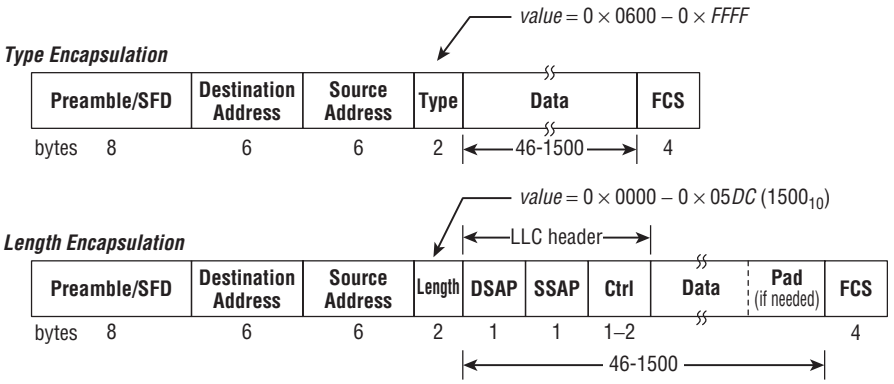


Figure 1-9 Ethernet frame formats

The Preamble/SFD, address, and Frame Check Sequence fields are common to both Type Encapsulated and Length Encapsulated Ethernet frames.

- **Preamble/Start-of-Frame Delimiter:** All Ethernet frames begin with an 8-byte field comprising a Preamble and a Start-of-Frame Delimiter (SFD). The Preamble allows receivers to synchronize on the incoming frame and comprises 7 bytes, each containing the value 0x55; the SFD contains the value 0xD5. The effect is to send a serial stream of alternating ones and zeroes (1 0 1 0 1 0 1 0 . . .) for 62 bits, followed by 2 ones, signifying the end of the delimiter sequence and the beginning of the Destination Address field.
- **Destination Address:** This field contains the 48-bit address of the target destination(s) of the frame. It may contain either a unicast or multicast address, as discussed in section 1.2.2.1.
- **Source Address:** This field contains the 48-bit unicast address of the station sending the frame.
- **Frame Check Sequence:** The FCS is a checksum computed on the contents of the frame from the Destination Address through the end of the data field, inclusive. The checksum algorithm is a 32-bit Cyclic Redundancy Check (CRC). The generator polynomial is:[29]

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

The FCS field is transmitted such that the first bit is the coefficient of the x^{31} term and the last bit is the coefficient of the x^0 term. Thus the bits of the CRC are transmitted: x31, x30, . . ., x1, x0.

1.3.1.3.1 Type Encapsulation

In the DIX Ethernet specifications (both Version 1 and Version 2), Type Encapsulation was the only frame format specified. For this reason, it is often called Ethernet Version 2 encapsulation, reflecting the more widely distributed version of that standard.

Type encapsulation provides fields so the specification of the upper-layer protocol type and data can be stated. An explanation of these follow:

- **Type field:** When Type Encapsulation is used, a Type field identifies the nature of the client protocol running above the Ethernet. Using Type fields, an Ethernet can upward multiplex among various higher-layer protocols (IP, IPX, AppleTalk, and so on).

[29] A complete discussion of CRCs is beyond the scope of this book. The reader is referred to [PETER72] for a general discussion, and to [HAMM75] for the detailed behavior of the particular algorithm used in Ethernet.

Ethernet controllers do not typically interpret this field, but use it to determine the destination process within the attached computer. Originally, Type field assignments were made by Xerox Corporation; however, in 1997 this responsibility was transferred to the IEEE.[30] Type fields are 16-bit values in the range of 0x0600 to 0xFFFF.

- **Data field:** The Data field encapsulates the higher-layer protocol information being transferred across the Ethernet. Ethernet frames must be of a certain minimum length due to the restrictions of the CSMA/CD algorithm.[31] When using Type Encapsulation, it is the responsibility of the higher-layer protocol client to ensure that there are always at least 46 bytes in the Data field. If fewer actual data bytes are required, the higher-layer protocol must implement some (unspecified) padding mechanism. The upper bound of the data field length is arbitrary and has been set at 1,500 bytes.[32]

1.3.1.3.2 Length Encapsulation

Originally, the IEEE 802.3 standard supported only Length Encapsulated frames and did not permit the use of Type Encapsulation. Despite this lack of official sanction, Type Encapsulation was (and is) widely used; it has always been the more popular frame format on Ethernet. In 1997, the standard was supplemented to include support for Type fields, and both encapsulations are now part of the IEEE standard.

When using Length Encapsulation, the 2 bytes following the Source Address are used as an indicator of the length of the valid data in the Data field rather than as an upward multiplexing mechanism.

Instead of specifying protocol types, Length Encapsulation will specify the data length as explained here:

- **Length and Pad fields:** The 16-bit Length field is used to indicate the number of valid bytes composing the Data field in the range of 0 to 1,500 bytes (0x0000 to 0x05DC). Note that this value may be less than the 46 byte minimum required for proper operation of the Ethernet MAC. When using Length Encapsulation, it is assumed that

[30]See `http://standards.ieee.org` for information about obtaining a Type field and the current list of assignments.

[31]A minimum frame length is necessary to ensure that collisions are always detected on a maximum-length Ethernet.

[32]There are many pros and cons of allowing longer Ethernet frames, including the effect on access latency and frame error rates. However, the real reason for the specified maximum was the cost of memory in 1979 (when the 10 Mb/s Ethernet was being designed) and the buffering requirements of low-cost LAN controllers. For a more detailed discussion, see [SEIF91]. Chapter 12, ''Virtual LANs: The IEEE Standard,'' discusses the need to increase the overall length of the Ethernet frame slightly to support Virtual LAN tags.

the Ethernet MAC will provide any needed pad (in the Pad field shown in Figure 1-9); the Length field contains the length of the unpadded data.

- **With the Type field eliminated:** There is now no way for the Ethernet MAC to indicate the higher-layer protocol client (upward multiplexing point) within the sending or receiving station. Frames using a Length field are therefore assumed to encapsulate Logical Link Control (LLC) data, as discussed in section 1.1.9.3. The first bytes of the Data field therefore contain the LLC header information shown in Figure 1-9. LLC (rather than a Type field) provides the mechanism for multiplexing among multiple clients of the Ethernet.

1.3.1.3.3 Length Versus Type Encapsulation

While these two approaches appear to be in conflict with each other (they use the same field in the frame for different purposes), in practice they can easily coexist. The 2-byte Type/Length field can carry a numerical value between 0 and $2^{16} - 1$ (65,535). The maximum allowable value for Length Encapsulated frames is 1,500, as this is the longest valid length of the Data field. Thus, the values between 1,501 and 65,535 can be used as Type field identifiers without interfering with the use of the same field as a Length indication. We have simply made sure that all of the Type field value assignments were made from this noninterfering space. In practice, all values of this field between 1,536 and 65,535 inclusive (0x0600 through 0xFFFF) are reserved for Type field assignment; all values from 0 to 1,500 are reserved for Length field assignment.[33]

In this manner, clients using Length Encapsulation and LLC can communicate among themselves, and clients using Type Encapsulation can communicate among themselves, on the same LAN. Of course, the two types of clients cannot intercommunicate, unless a device driver or higher-layer protocol understands both formats. Most higher-layer protocols, including TCP/IP, IPX (NetWare), DECnet Phase 4, and LAT (DEC's Local Area Transport), use Type Encapsulation. Length Encapsulation is most commonly used with AppleTalk, NetBIOS, and some IPX (NetWare) implementations. Chapter 3 contains a complete discussion of the issues related to the use of different encapsulation formats by different protocol clients.

Note that when Type Encapsulation is used, the Logical Link Control (LLC) protocol is not used, and need not even be present. If a device supports some clients that use Type Encapsulation and others that use Length Encapsulation, the MAC can upward multiplex to both sets of clients simultaneously, as depicted in Figure 1-2.

[33]The range of 1,501 to 1,535 was intentionally left undefined.

1.3.1.3.4 SNAP Encapsulation

Any frame encapsulating LLC data can use the LLC SNAP SAP[34] as discussed in section 1.1.9.3.2 to expand the upward-multiplexing capability of LLC.

Figure 1-10 depicts a Length Encapsulated Ethernet frame containing LLC/SNAP encapsulated data.

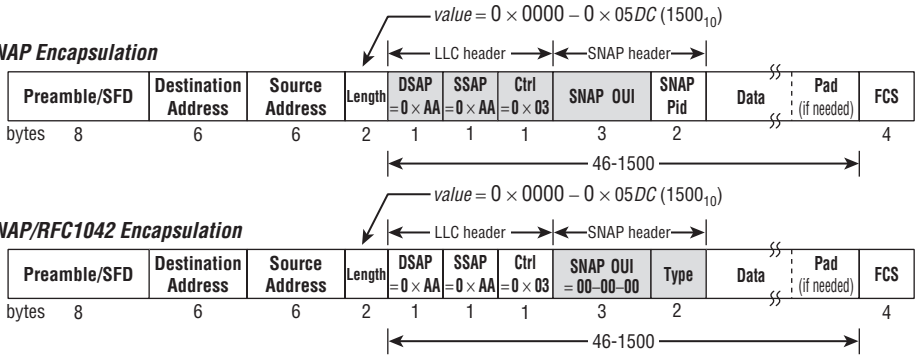


Figure 1-10 SNAP and RFC1042 encapsulation on Ethernet

In its generic form, the SNAP OUI indicates the globally unique identifier of the organization for which the SNAP Protocol Identifier (Pid) field is significant. For example, Apple Computer has an assigned OUI of 08-00-07, and most AppleTalk protocols are encapsulated using LLC/SNAP with this value of SNAP OUI.

A SNAP OUI of all zeroes (00-00-00) has a special meaning; it signifies that the SNAP Pid field has the same semantics as an Ethernet Type field. That is, the SNAP Pid field is interpreted as a globally unique Type field, with the same value assignments that are used for native Type Encapsulated frames, rather than being reserved for private use. This allows a client protocol to use Length Encapsulation, LLC, and SNAP along with the set of standard Type field assignments.

Using this approach, SNAP encapsulation can provide the equivalent of Type field encoding on LANs that do not provide native support for it (for example, Token Ring, discussed in section 1.3.2). If the SNAP OUI is not equal to 00-00-00, then there is no assurance that the SNAP Pid field has the same semantics as an Ethernet Type field; the organization controlling the non-zero OUI can define the Pid field in any manner it chooses.

The original purpose for this special case of SNAP encapsulation was to allow the TCP/IP protocol suite to use Type field encoding consistently across all LAN types regardless of their ability to natively support Type

[34]See, it really is fun to say this!

Encapsulation. The procedure is specified in [RFC1042] and is referred to as RFC1042 Encapsulation. Figure 1-10 depicts an Ethernet frame using RFC1042 Encapsulation.[35]

1.3.1.4 Bit-Ordering

When transmitting an Ethernet frame onto the LAN medium, bytes are transmitted in the order shown in Figure 1-9 from left to right. Within each byte, Ethernet (like most data communications systems) transmits bits from the least significant bit (the bit corresponding to the 2^0 numerical position) first, to the most significant bit (the bit corresponding to the 2^7 numerical position) last.

1.3.2 Token Ring

> *Token Ring (tō′ken ring) n. 1.* A LAN technology using distributed polling arbitration on a physical loop topology. 2. A piece of jewelry given when one wants to get married only for the weekend.

The Token Ring technology on which most commercial LAN implementations are based was developed by IBM at its laboratories in Zurich, Switzerland, in the late 1970s. Unlike Ethernet, Token Rings connect the attached stations in a loop configuration, as depicted in Figure 1-11. Each station can transmit information directly only to its downstream neighbor, and can receive directly only from its upstream neighbor. Many-to-many communication is affected by having each station repeat the signals from its upstream neighbor to the downstream one.

When a station transmits on its own behalf, it inserts its data into the circulating stream. All other (non-transmitting) stations repeat the data until it returns to the originating station, which is responsible for removing it from the ring.

Similar to what occurred with Ethernet, IBM both wrote its own specification formalizing Token Ring operation and also brought its technology to the IEEE 802 LAN Standards Committee during the early 1980s. IBM's Token Ring architecture became the basis of the IEEE 802.5 standard, first published in 1985.

The original Token Ring standard specified operation at either 4 or 16 Mb/s. Initially, most commercial products operated at the lower data rate. During the late 1980s and early 1990s, 16 Mb/s Token Ring products became more widely

[35]Often, the term SNAP Encapsulation is used when what is really meant is RFC1042 Encapsulation (i.e., SNAP Encapsulation with an OUI of all zeroes). The RFC1042 form is more common than SNAP used with a non-zero SNAP OUI.

available; by the late 1990s, virtually all Token Ring installations operated at the 16 Mb/s data rate. In 1998, the IEEE 802.5 Working Group approved a standard for Token Ring operation at 100 Mb/s; however, this has not seen significant commercial deployment.

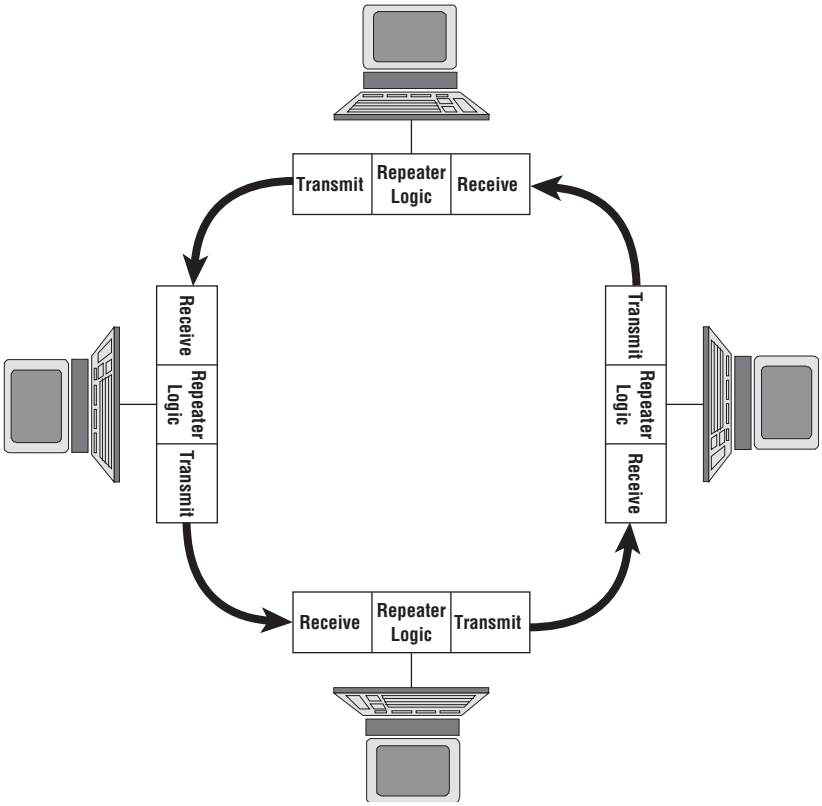


Figure 1-11 Token Ring configuration

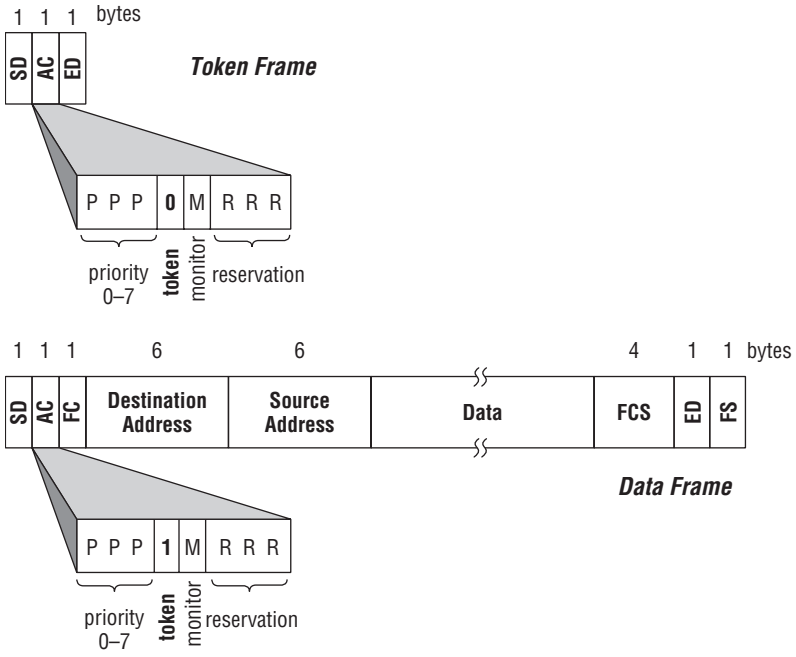
1.3.2.1 Token Ring Medium Access Control

Proper operation of a Token Ring requires that only one station be using it to transmit its own data at any given time. That is, only one station ever has permission to transmit; this ''permission slip'' is called a token.[36]

The token circulates from station to station around the ring. Each station repeats the token, effectively passing it to its downstream neighbor. When a station wishes to transmit data, it waits for the token to arrive and then

[36]In the first edition of this book, Rich made the following statement: ''I have discovered that many students find it easier to understand Token Ring operation if, when reading material on the subject, they substitute the phrase 'permission to transmit' for 'token' in their heads.''

changes a single bit (from a `0` to a `1`) in the token frame. This single bit change (the Token bit in the Access Control field, see Figure 1-12) converts the token to a data frame; the station then proceeds with the transmission of its data, which circulates throughout the ring.



SD = Start-of-Frame Delimiter
AC = Access Control (Token/Priority/Reservation/Monitor use)
FC = Frame Control (MAC/Data/Management frame type)
FCS = Frame Check Sequence
ED = End Delimiter
FS = Frame Status (Address Recognized/Frame Copied/Error indicator)

Figure 1-12 Token Ring frame format

All stations on the ring inspect the contents of the data frame as they repeat the data to their respective neighbors. If the destination address within the frame indicates that they are intended recipients, the station(s) copy the frame into a local buffer and pass it to the higher layer client (typically LLC) for further processing. In any event, the device continues to repeat the frame around the ring in case there are other potential receiving stations (for example, a multicast destination), and to allow the frame to return to the original sender.

When the original sender sees its own frame returning after its arduous journey, it removes the frame from the ring and transmits a new token to its

downstream neighbor (by flipping the Token bit back to a 0 value), effectively passing the baton to the next potential sender.[37]

1.3.2.2 Token Ring Physical Layer Options

The original Token Ring standard specified operation using a specialized shielded twisted pair cable developed as part of IBM's overall building wiring architecture [IBM82]. Virtually all early Token Ring customers used these cables, although they were extremely expensive, physically heavy, and difficult to terminate. The emergence of structured wiring systems using unshielded twisted pair (UTP) cable in the 1990s created a market demand for Token Ring products that could operate over this easier-to-use, lower-cost wire. While the official standard for Token Ring operation over UTP cable was not formally published until 1998 [IEEE98 h], many suppliers offered commercial products supporting Token Ring operation over UTP cable both at 4 and 16 Mb/s for a decade before the standard was approved.

In 1998, an amendment to IEEE 802.5 was approved that specified the operation of 16 Mb/s Token Ring over multimode optical fiber [IEEE98f]. Again, the date indicated only that the formal standard was catching up with common commercial practice many years after the fact.

The standard for Token Ring operation at 100 Mb/s uses the identical twisted pair physical layer specifications used for 100 Mb/s Ethernet (100BASE-TX).[38] That is, the same wiring system and physical layer signaling components can be used for either Ethernet or Token Ring operation at 100 Mb/s.

1.3.2.3 Token Ring Frame Formats

The following details the information contained within the fields of a Token Ring frame (refer to the Token Ring frame format example in Figure 1-12):

- **Start-of-Frame Delimiter (SD):** This field contains a unique pattern (as defined by the underlying physical signaling method) that allows a receiver to distinguish the start of a data frame from the idle signals used between frames.

[37] What is presented here is a highly simplified explanation of Token Ring operation. There are many special features (token reservation, early release, priority, etc.) and boundary conditions (ring initialization, lost token recovery, etc.) that can be understood only with a detailed examination of the Token Ring access protocol specifications. Interested readers should refer to [CARL98] or the standard itself [IEEE98 h]. Fortunately, a detailed understanding of the Token Ring MAC is not necessary for the purpose of understanding LAN switching in this book.

[38] Actually, Ethernet ''borrowed'' (a nicer word than ''ripped off'') these same Physical layer specifications from FDDI-over-copper [ANSI95] for use in 100 Mb/s Fast Ethernet. High Speed (100 Mb/s) Token Ring simply followed the long-standing tradition of using an existing, proven technology for a purpose other than that for which it was originally designed.

- **Access Control (AC):** This field comprises four subfields used to communicate information to the Token Ring access control process:
  - The priority bits (P) indicate the priority level of the token or data frame. (See Chapter 13 for a discussion of priority operation and the use of the priority bits.)
  - The token bit (T) is used to differentiate a token frame (T = 0) from a data frame (T = 1).
  - The monitor bit (M) is used by a special active monitor station to detect various error conditions (circulating high priority token, frame initiator no longer present to remove its frame from the ring, and so on).
  - The reservation bits (R) are used to request future use of a token at a specified priority level.
- **Frame Control (FC):** The Frame Control field is used to differentiate user data frames from network management data frames.
- **Destination Address:** This field contains the 48-bit address of the target destination(s) of the frame. It may contain either a unicast or multicast address, as discussed in section 1.2.2.1.
- **Source Address:** This field contains the 48-bit unicast address of the station sending the frame. (Chapter 6 discusses the use of the first bit of the Source Address as a Routing Information Indicator for Source Routing.)
- **Data Field:** The Data field encapsulates the higher-layer protocol information being transferred across the LAN. Unlike Ethernet frames, Token Ring frames have no minimum length; there can be 0 bytes present in the Data field. The maximum length of the Data field is 4,528 bytes when operating at 4 Mb/s, and 18,173 bytes when operating at 16 or 100 Mb/s.
- **Frame Check Sequence:** The FCS is a checksum computed on the contents of the frame from the FC field through the end of the data field inclusive. The checksum algorithm is identical to that used in Ethernet (see section 1.3.1.3).
- **End Delimiter (ED):** This field contains a unique pattern (as defined by the underlying physical signaling method) that allows a receiver to unambiguously determine the end of the frame.
- **Frame Status (FS):** This field contains status bits that allow a sending station to determine whether any intended recipients on the ring recognized the destination address, copied the frame into a local buffer, and/or detected an error during reception.

1.3.2.4 Bit-Ordering on Token Ring LANs

When a Token Ring frame is transmitted onto the LAN medium, bytes are transmitted in the order shown in Figure 1-12 from left to right. Within each byte, Token Ring transmits bits from the most significant bit (the bit corresponding to the 2^7 numerical position) first, to the least significant bit (the bit corresponding to the 2^0 numerical position) last. This is the opposite convention from that used in Ethernet.

Chapter 3 contains a complete discussion of the problems encountered as a result of this difference in bit-ordering as well as with the use of the Frame Status field when bridging between Ethernet and Token Ring LANs.

1.3.3 Fiber Distributed Data Interface

The Fiber Distributed Data Interface (FDDI) was the first standard local and metropolitan area network technology capable of operating at 100 Mb/s; until 1993, it was the only practical network alternative operating at a data rate in excess of 16 Mb/s. Developed under the auspices of the American National Standards Institute (ANSI) during the mid-1980s, it had support from dozens of network equipment manufacturers. FDDI is now an ISO International Standard [IS089a, IS089b, IS090]. While FDDI is not strictly part of the IEEE 802 family of standards, it is fully compatible with them. Architecturally, FDDI operates like an IEEE 802-style MAC, and it uses the same 48-bit address structure. Like its Token Ring cousin, FDDI requires LLC (with or without SNAP encapsulation) to upward multiplex among multiple client protocols and applications.

Originally, FDDI was intended for a wide variety of applications:

- **Front-end networks:** A high-speed replacement for desktop LANs (i.e., an Ethernet upgrade).
- **Back-end networks:** A processor-to-processor communications system for server interconnections, multiprocessing systems, and so on.
- **Backbone Networks:** A means of interconnecting other networks.

In practice, FDDI flourished only in the backbone application environment. This is primarily because of its initial use of fiber media exclusively (effectively eliminating widespread deployment in the high-volume desktop market) and its high cost.

1.3.3.1 FDDI Operation

FDDI was designed around the use of a shared fiber medium configured in a dual ring topology, with media arbitration accomplished through token passing (similar to the IEEE 802.5 Token Ring MAC). This is depicted in Figure 1-13.

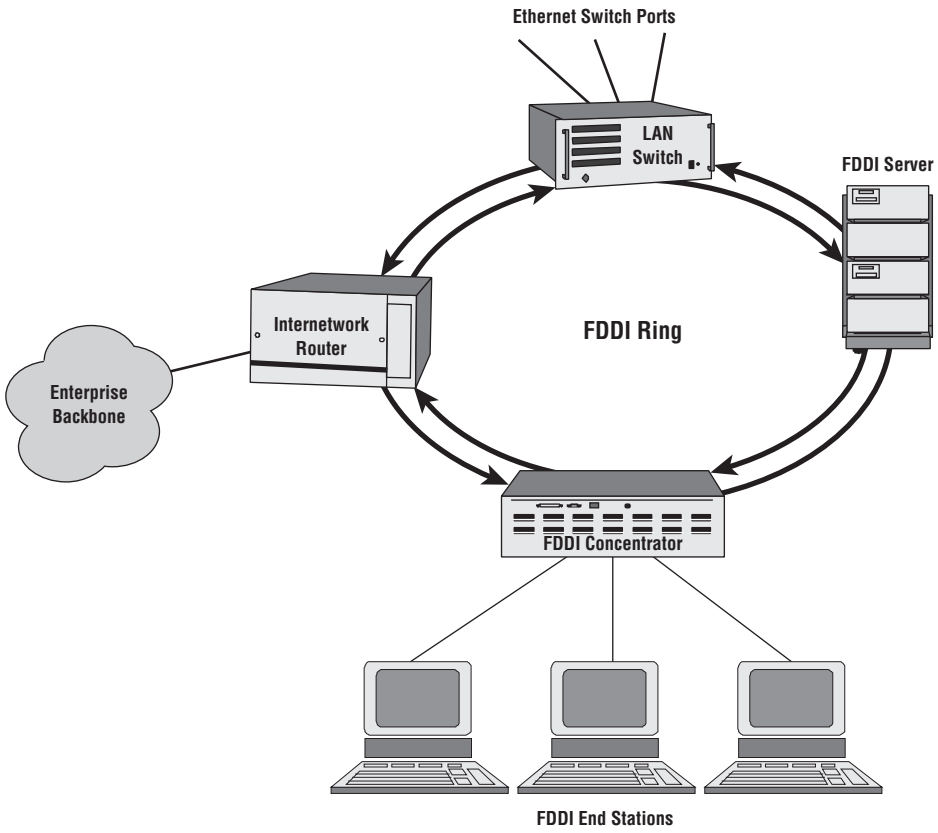


Figure 1-13 FDDI ring

FDDI supports a maximum of 500 stations in the ring. All stations operate at a single, common data rate of 100 Mb/s. A token circulates around the ring; when a station wishes to transmit data, it waits for the token to arrive and then substitutes its data frame(s) for the token. Following successful transmission of the data frame(s) in its transmit queue, the station releases a new token, allowing the next station in the ring to similarly send data.[39]

Tanenbaum's Doctrine

The nice thing about standards is that there are so many to choose from.

[39]Like the previous discussion of Token Ring operation, this is a highly simplified description of the FDDI access method. In addition to some of the same boundary conditions and features of the IEEE 802.5 Token Ring, FDDI also provides mechanisms for use of the redundant second ring, restricted token behavior, synchronous bandwidth reservation, and so on. These details are not presented here as they are not necessary for an understanding of the use of FDDI in a switched LAN environment. Readers interested in the details of the FDDI MAC should refer to [SHAH93] or the official standard [ISO89a].

1.3.3.2 FDDI Physical Signaling

Obviously from its name, FDDI was originally designed for fiber media (otherwise, we would just have called it DDI!). At the time it was being developed (around 1985), operation at 100 Mb/s over copper media was considered impractical. (At that time, even 10 Mb/s operation over twisted pair was inconceivable, much less 100 Mb/s operation!) In addition, backbone applications normally require relatively long-distance links (on the order of kilometers), which is impractical even today using low-cost copper media. Thus, the original FDDI standard specified the use of multimode optical fiber exclusively. The standard provided for distances of up to 2 km between stations on the ring.

In order to reduce the signaling rate on the fiber, a block data-encoding scheme was employed. FDDI systems encode four bits of data into five code bits (4B/5B encoding). This results in a channel signaling rate of 125 Mbaud (100 Mb/s $\times$ 5/4). The 4B/5B encoding system was later adopted for use in 100 Mb/s Ethernet systems.

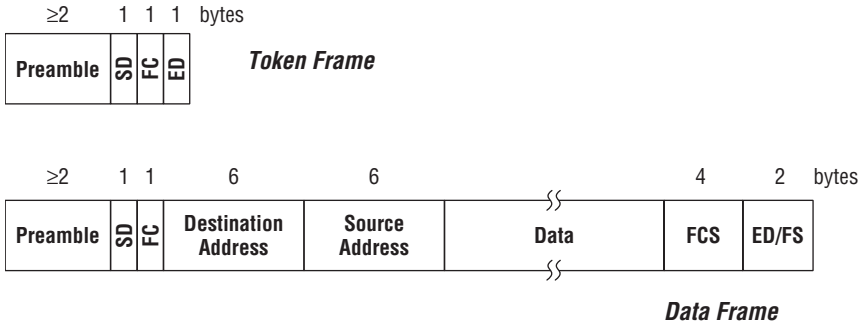
In an effort to expand the marketplace for FDDI, a system for operating FDDI over twisted pair copper media was developed during the early 1990s. The key technical challenge was to develop a means of signaling at 125 Mbaud in such a way that the signal was still decodable after passing through 100 m of UTP cable while not violating electromagnetic interference (EMI) regulations. The result was a line encoding system called Multi-Level Threshold (MLT-3), which used ternary (three level) rather than binary signaling to reduce the high-frequency spectrum of the transmissions.[40] The use of FDDI protocols over copper media is known as either Copper Distributed Data Interface (CDDI), a trade name of Cisco Systems, or Twisted Pair-Physical Medium-Dependent signaling (TP-PMD) in ISO terminology. The MLT-3 line encoding scheme is the same as that used in 100 Mb/s Ethernet and Token Ring.

FDDI is supported in standards-compliant devices over multimode optical fiber (2 km maximum), and both Category 5 UTP and STP (100 m maximum). Proprietary systems are also available that use singlemode fiber over longer distances.

1.3.3.3 FDDI Frame Format

FDDI, being a variant of Token Ring, uses a frame format similar to that of IEEE 802.5, as depicted in Figure 1-14.

[40] While officially MLT stands for Multi-Level Threshold, in reality it is the initials of the founders of the company that developed the technology (Crescendo Communications, later acquired by Cisco Systems): Mario Mozzola, Luca Cafiero, and Tazio De Nicolo.



SD = Start-of-Frame Delimiter
FC = Frame Control (Token/MAC/Data/Management frame type)
FCS = Frame Check Sequence
ED = End Delimiter
FS = Frame Status (Address Recognized/Frame Copied/Error indicator)

Figure 1-14 FDDI frame format

Explanations of the fields in the figure follow:

- **Preamble:** Similar to Ethernet, the FDDI Preamble allows receivers to synchronize on the incoming frame.
- **Frame control:** Rather than using a separate Access Control and Frame Control field as in the IEEE 802.5 Token Ring, FDDI uses a single field (FC) to differentiate among Tokens, user data, and management frames.

The remainder of the FDDI frame fields have the same use as their counterparts in the IEEE 802.5 Token Ring frame (see section 1.3.2.3). As with Token Ring, FDDI transmits bits from the most significant bit (the bit corresponding to the 2^7 numerical position) first, to the least significant bit (the bit corresponding to the 2^0 numerical position) last within each byte.

1.3.4 Other LAN Technologies

Ethernet, Token Ring, and FDDI make up the vast majority of installed LANs. However, there are numerous other technologies that have either existed at one time (and achieved some level of installed product base) or are just beginning to emerge in the LAN marketplace.

From a LAN switching perspective, most of these other technologies can be treated as variants of one of the more popular LANs. Differences in frame formats, shared medium access methods, or protocol semantics may affect the low-level implementation details of the devices employing a given technology, but do not change the fundamental behavior or operational algorithms in a switch from those used with the more popular technologies. In many cases,

generic Data Link protocols such as LLC shield higher-layer clients from the details of the underlying LAN.

Some of the LAN technologies that have ended up in a ditch along the shoulder of the Information Superhighway include:

- Token Bus, as specified in IEEE 802.4 and used in Manufacturing Automation Protocol (MAP) systems
- Distributed Queue/Dual Bus (DQDB) systems, as specified in IEEE 802.6 and used in Switched Multimegabit Data Service (SMDS) offerings
- isoEthemet, specified in IEEE 802.9a
- 100VG-AnyLAN, specified in IEEE 802.12

Virtually none of these systems ever achieved any significant degree of market penetration or installed base of product, nor are manufacturers actively developing new products using these technologies.

One alternative LAN technology that did achieve considerable success was the Attached Resource Computer network (ARCnet), developed by Datapoint Corporation during the mid-1970s. Designed to operate over coaxial cable using a branching tree topology and a Token Passing Bus access method, ARCnet was arguably the most widely installed LAN technology in the early 1980s. Until Ethernet components achieved commodity status, ARCnet's low cost made it attractive for small personal computer networks. Furthermore, its relatively low data rate of 2.5 Mb/s was not a limiting factor in that early, low-performance networking environment.[41]

The biggest problem with ARCnet (from a LAN switching perspective) is that it uses locally administered, manually configured, 8-bit addresses. When installing an ARCnet interface into a computer, the installer must select and configure the device's address, often by setting switches or jumpers on the interface card itself.

As discussed in Chapter 2, the fact that ARCnet does not use globally administered addresses makes it unsuitable for use with LAN bridges operating at the Data Link layer. While it is possible to build an internetwork of multiple LANs comprising a combination of ARCnet and other technologies, Network-layer routers (not LAN bridges or switches) must be used for the interconnection. ARCnet LANs cannot even be bridged to themselves, much less to Ethernets, Token Rings, or other standard LANs.

Fortunately for the deployment of LAN switches, ARCnet is rarely used today in commercial networks. It is occasionally encountered in embedded process control systems used for factory automation.

[41] A 20 Mb/s version (dubbed ARCnet Plus) was developed in the late 1980s, but never successfully commercialized.

1.4 IEEE LAN Standards

The Institute of Electrical and Electronics Engineers (IEEE) supports the development of industry standards in a wide range of technologies, from power generation to electronic instrumentation to programming languages. In February 1980, a meeting was held at the Jack Tar Hotel in San Francisco, California, to discuss the development of an industry standard for Local Area Networks.[42] That meeting (and many others subsequent to it) spawned the creation of IEEE Project 802, one of the largest and longest-running activities within the IEEE standards organization.

The original intent of Project 802 was to develop a single standard for local area computer networks. Recognizing that a number of different skill sets were involved in this endeavor, the group originally split into three subcommittees:

- A Higher-Layer Interface (HILI) group would be responsible for the development of service and programming interfaces for higher-layer protocol clients. In addition, this group would deal with all of the issues that are generally addressed with software technology, such as network management.
- A Medium Access Control (MAC) group would be responsible for the development of the MAC algorithm to be used on the LAN.
- A Physical Layer (PHY) group would be responsible for the actual wiring, encoding, and signaling used on the medium.

Over the next few years, it became clear that the MAC and PHY groups would never achieve consensus on a single technology or medium to be the standard for all users. There was one large faction of Ethernet supporters promoting CSMA/CD on baseband coaxial cable, a second faction arguing for the use of a Token Passing Bus on broadband coaxial cable, and a third group endorsing Token Ring on shielded twisted-pair cable.

Since no agreement would be forthcoming, the committee decided to pursue all three approaches. Each faction formed a separate Working Group and developed a set of specifications around its favorite technology. The HILI group split in two as well, one subcommittee dealing with architecture, interworking, and management issues, and the other with the definition of the generic Logical Link Control protocol.

[42]Rich was at that meeting. The Jack Tar Hotel no longer exists (the Cathedral Hill Hotel is now located at the same site), but the IEEE 802 LAN/MAN Standards Committee does. Interestingly, although much of the constituency of the IEEE 802 committee comes from Northern California (Silicon Valley), that first meeting was the only time that a full IEEE 802 (plenary) session was held anywhere in the San Francisco Bay Area. It's much more fun to travel to Maui, Montreal, New Orleans, and the like than to stay at home and work.

Thus, there were five Working Groups:

- **IEEE 802.1:** Overview, Architecture, Interworking, and Management
- **IEEE 802.2:** Logical Link Control
- **IEEE 802.3:** CSMA/CD MAC and PHY
- **IEEE 802.4:** Token Bus MAC and PHY
- **IEEE 802.5:** Token Ring MAC and PHY

Over the years, additional Working Groups were formed and these Working Groups produced a number of additional standards. IEEE Project 802 is internationally recognized as the official body responsible for the development of local and metropolitan area network standards. Many of the standards developed are adopted and endorsed by national standards bodies — for example, the American National Standards Institute (ANSI) in the United States, as well as the International Organization for Standardization (ISO). While IEEE 802 is ostensibly a U.S. organization, its membership is global, with significant representation from individuals and organizations in Europe, Asia, and Israel.

1.4.1 IEEE 802 Organization

Figure 1-15 depicts the current IEEE 802 organization, composed of 13 Working Groups (WGs) and 2 Technical Action Groups (TAGs). The groups in boldface are currently active and hold regular meetings to work on outstanding projects and issues. The others are in hibernation, meaning that there is no ongoing activity, no projects are outstanding, and no meetings are held; former members of those groups can be contacted if questions arise about the interpretation of a standard developed when the group was active.

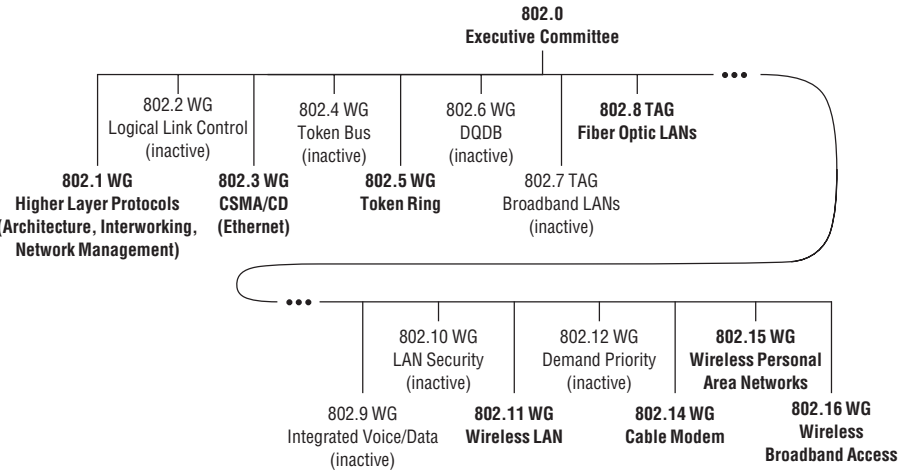


Figure 1-15 IEEE 802 organization

Information on current IEEE 802 Working Group activities and meetings, and on obtaining the standards themselves, can be found at `http://standards.ieee.org`.

As we wish to focus on those areas pertinent to LAN switching, sections 1.4.3, 1.4.4, and 1.4.5 look more closely at the activities of the IEEE 802.1 WG and the IEEE 802.3 and 802.5 WGs (Ethernet and Token Ring), respectively.

1.4.2 IEEE 802 Naming Conventions, or "Mind Your Ps and Qs"

You may notice (both in this book and in industry literature) that IEEE documents are sometimes called ''IEEE so-and-so,'' and at other times ''P802-dot-whatever.'' Also, some documents use lowercase letters and some uppercase. While some of the variation is due to sloppiness and inconsistency in publication practices, there is actually a method to the madness — a code system by which we assign designations to Task Forces and the documents that they produce. IEEE 802 uses the following conventions:

- A document that stands alone (i.e., that is self-contained rather than a supplement to another standard) either gets no letter designation (for example, IEEE 802.3) or gets a capital letter designation if the Working Group that produced the document is responsible for multiple standards (for example, IEEE 802.1D).
- A document that modifies or supplements a standalone document takes a lowercase letter designation. For example, IEEE 802.1k was a supplement to IEEE 802.1B, adding capability for the discovery and dynamic control of management event forwarding. When the foundation document (in this case, IEEE 802.1B) is revised and republished, all of the outstanding supplements get swept into the base document. The base document keeps its original designation (with a new revision date), and the supplements disappear completely from the scene.
- Letters are assigned in sequential order (a, B, c, D, E, and so on) as new projects are initiated within each Working Group, without respect to whether they are capitalized or not. There is no relationship between the letter designation assigned to a supplement and the letter of the foundation document that it modifies. The letter designation uniquely identifies both the document and the Working Group Task Force that develops it.
- Approved standards have an IEEE name (for example, IEEE 802.1Q). Unapproved drafts leading up to an approved standard are designated P802 (dot whatever), signifying that they are the work of an ongoing project (hence the P). The P802 name is usually followed by the draft number. Thus, P802.1p/D9 was the ninth (and final) draft

of a supplement to IEEE 802.1D. It turned out that the schedule for publication of IEEE 802.1p (the approved standard resulting from P802.1p) coincided with the time for the revision and republication of its foundation document, IEEE 802.ID. Thus, it was expedient to sweep the supplement into the base document immediately rather than waiting as much as a few years for the next revision cycle. So IEEE 802.Ip (an important standard on multicast pruning and priority bridge operation) was never published on its own.

Is it all clear now?

1.4.3 IEEE 802.1

IEEE 802.1 is the Working Group responsible for all of those aspects of LAN technology that cross MAC sublayer and Physical layer technology boundaries. As such, it has become a catch-all group for a variety of issues and standards development activities, including network management, addressing, and the IEEE 802 architecture itself.

Following is a list of IEEE 802.1 standards:

- **IEEE 802: Overview and Architecture:**[43] This is the foundation document for the entire series of IEEE 802 standards. It contains key concepts applicable across all LANs, including the structure of 48-bit addresses and Organizationally Unique Identifiers and the architectural reference model for the other IEEE 802 standards.
- **IEEE 802.1AB-2005:** This standard specifies the use of the Link Layer Discovery Protocol (LLDP) used to populate the physical topology and identify stations that are connected on the catenet.[44] This standard also identifies access points on network nodes for the purposes of device and system management.
- **IEEE 802.1AD-2005:** This standard outlines amendment 4 to the 802.1Q standard. It allows service providers the ability to use bridges to offer third parties what equates to a separate catenet.
- **IEEE Std 802.1AE-2006:** This standard defines the MAC security guidelines to assist in maintaining network data security and to recognize and react to data received and/or modified by devices that are not authorized in the catenet.

[43]Note that the name of this standard is IEEE 802, not 802-dot-anything. While not strictly an IEEE 802.1 document, it was developed under the auspices of the IEEE 802.1 WG, and by the same people who developed other IEEE 802.1 standards.

[44]Although the term ''catenet'' is considered somewhat obsolete, it is very applicable for the information that is covered in this book. The term is used throughout the book. Just remember that when you see this term, it is simply making reference to a system of networks interconnected to one another through a device or node.

- **IEEE 802.1B: Management:** This standard provides a protocol and mechanisms for remote management of IEEE 802 LANs. As the de facto standard for LAN Management is the Simple Network Management Protocol (SNMP) [RFC1157, ROSE96], IEEE 802.1B is largely ignored. There are few, if any, commercial implementations.
- **IEEE 802.1D: MAC Bridges:** This is the most widely known and implemented IEEE 802.1 standard; it forms the basis for all LAN bridges and switches. The operation of bridges built to this standard is discussed extensively in Chapters 2 to 6 of this book. The 1998 edition of the 802.1D standard includes facilities for multicast pruning and priority operation, as discussed in Chapter 10, ''Multicast Pruning,'' and Chapter 13, ''Priority Operation.'' The 2004 edition provided an update to the Spanning Tree Protocol, referred to as the Rapid Spanning Tree Protocol (see Chapter 5). Prior to these revisions, these functions were described in the supplement to IEEE 802.1D, designated P802.1p.
- **IEEE 802.1E: System Load Protocol:** This largely unknown but very useful and interesting standard defines a protocol for the reliable downloading of bulk data simultaneously to multiple devices using the multicast addressing capability of IEEE 802 LANs. The protocol avoids the need to individually download multiple devices with the same information (for example, operational code), yet provides for guaranteed delivery to all destinations, invoking retransmission only when needed by one of the multiple target devices.
- **IEEE 802.1F: Common Definitions for Management:** This standard provides a set of generic definitions for management information common across the range of other IEEE 802 standards. It is modeled toward use with the ISO Common Management Information Protocol (CMIP) [IS089c]. Because most systems today ignore CMIP and instead provide management based on SNMP, this standard sees little application.
- **IEEE 802.1G: Remote MAC Bridging:** This standard defines an architecture and set of requirements for bridges that interconnect geographically dispersed sites using Wide Area Network (WAN) technology. As discussed in Chapter 3, the standard provides little practical information and is generally ignored by both product developers and network administrators.
- **IEEE 802.1H: Bridging of Ethernet:** This important document is officially a Technical Recommendation, as opposed to a standard.[45] It describes the method used by bridges to convert between

[45]Technical Recommendations carry less weight than official standards. Indeed, the IEEE 802.1H recommendation comprises only 29 pages and weighs about 70 grams, as compared to the IEEE 802.1G standard, a weighty tome at over half a kilo!

Type Encapsulated frames (as used on Ethernet) and both SNAP and RFC1042 encapsulation (as used on LANs that do not support native Type Encapsulation). The procedures specified in this standard are discussed extensively in Chapter 3.

- **IEEE 802.1Q: Virtual Bridged LANs:** This standard defines the requirements for bridges operating in a Virtual LAN (VLAN) environment. VLAN operation is discussed in Chapter 11, ''Virtual LANs: Applications and Concepts,'' and Chapter 12, ''Virtual LANs: The IEEE Standard.''
- **IEEE 802.1X-2004:** This standard defines the manner in which a system can manage port access control in order to authenticate and authorize nodes that are approved as a catenet device and to block those that are not.

From a LAN switch perspective, the important IEEE 802.1 documents are IEEE 802, IEEE 802.1D, IEEE 802.1H, and IEEE 802.1Q.

1.4.4 IEEE 802.3

IEEE 802.3 is the standard for MAC and Physical Layer operation of CSMA/CD LANs. Unlike IEEE 802.1, only one self-contained document is controlled by the IEEE 802.3 Working Group — the IEEE 802.3 standard. Over the years, there have been numerous supplements and revisions; at the time of this writing, the letter designations for IEEE 802.3 projects have reached P802.3ba, having used all 26 letters twice and wrapped around to two-letter designations.

The 1998 version of IEEE 802.3 is over 1,200 pages long; it is affectionately referred to as the Doorstop Edition in deference to one of the alternative uses for such a massive work. It includes all of the supplements through IEEE 802.3z (Gigabit Ethernet).

There were two additional updates to the 802.3 standard. In 2002, the standard was updated to include the supplements 802.3ab, 802.3ac, and 802.3ad. In 2005, the standard included 802.3ae, 802.3af, 802.3 ah, and 802.3ak.

Some of the key supplements to IEEE 802.3 include:

- IEEE 802.3a (Thin wire coaxial cable, 10BASE2), 1988
- IEEE 802.3c (Repeater specifications), 1985
- IEEE 802.3d (Fiber optic inter-repeater link, FOIRL), 1987
- IEEE 802.3i (UTP cable, 10BASE-T), 1990
- IEEE 802.3j (Fiber optic LAN, 10BASE-F), 1993
- IEEE 802.3u (Fast Ethernet, 100BASE-T), 1995
- IEEE 802.3x (Full duplex operation and flow control), 1997

- IEEE 802.3z (Gigabit Ethernet over optical fiber), 1998
- IEEE 802.3ab (Gigabit Ethernet over UTP cable, 1000BASE-T), 1999
- IEEE 802.3ac (Frame Extensions for VLAN-tagging), 1998
- IEEE 802.3ad (Link Aggregation), 2000
- IEEE 802.3ae (10 Gbit/s Ethernet over fiber), 2003
- IEEE 802.3af (Power over Ethernet), 2003
- IEEE 802.3ah (Ethernet in the First Mile), 2004
- IEEE 802.3ak (Ethernet over Twinaxial), 2004

Throughout this book, the terms Ethernet and IEEE 802.3 are used interchangeably.

1.4.5 IEEE 802.5

IEEE 802.5 is the standard for MAC and Physical Layer operation of Token Ring LANs. As with IEEE 802.3, this Working Group controls only one base document — the IEEE 802.5 standard. That standard has been supplemented to include:

- IEEE 802.5c (Dual ring redundant configuration), 1991
- IEEE 802.5j (Optical fiber media), 1998
- IEEE 802.5r (Dedicated Token Ring/Full duplex operation), 1998
- IEEE 802.5t (100 Mb/s High Speed Token Ring), 2000
- IEEE 802.5v (Gigabit Token Ring), 2001

1.4.6 Other Standards Organizations

IEEE Project 802 is not the only organization in the world concerned with the development and dissemination of network standards. In particular, the formal charter of IEEE 802 restricts its activities to technologies operating at the Data Link and Physical layers exclusively. While these make up a large part of what most people consider a LAN, any practical networking system will also require services above these lower layers. Thus, some aspects of networking are outside the scope of IEEE 802 standardization.

The IEEE is a professional society existing for the benefit of its members and the engineering community. It derives whatever authority it has from the assent of the industry. It is not a government agency, nor does it have any power or ability to enforce compliance with its standards other than the willingness of the industry to accept those standards as meaningful.

Many national and international standards bodies look to IEEE 802 for guidance in setting their official standardization policies. As such, many of the important IEEE 802 standards have also been approved as standards by

the American National Standards Institute (ANSI). ANSI is the U.S. representative to the International Organization for Standardization; ISO has adopted many of the IEEE 802 standards and published them under the ISO name. Thus, IEEE 802.3 is also available as ISO 8802-3, IEEE 802.1D is available as ISO 15802-3, and so on. Typically, ISO adoption and publication of IEEE 802 standards lag approximately one year behind their approval by IEEE.

Some of the important standards organizations working on technologies related to LANs and LAN switching include:

- **American National Standards Institute:** ANSI is responsible for much of the work on higher-layer protocols in the ISO protocol suite, including the ISO Connectionless Network Protocol (CLNP), ISO Transport Protocol, and so on. ANSI was also responsible for the development of FDDI.
- **Internet Engineering Task Force:** The IETF is responsible for all of the higher-layer protocols in the TCP/IP family.
- **International Organization for Standardization:** While ISO does not generally develop standards itself, it does adopt and endorse standards developed by other organizations, including IEEE and ANSI. Many national governments and legal agencies adopt ISO standards either intact or as the foundation for their own national standards; in some cases, these agencies may forbid the sale of products in their countries that do not conform to the relevant ISO standard.
- **Electronic Industries Association/Telecommunications Industries Association:** The EIA/TIA vendor consortium has been responsible for the development of the standards for the structured building wiring on which most modern LANs operate. They also developed many other standard communications technologies, including RS-232, RS-422, and so on.
- **International Telecommunications Union:** The ITU-T (formerly known as the International Consultative Committee on Telephony and Telegraphy, or CCITT) is the agency of the United Nations responsible for many of the standards for long-distance telecommunications and interfaces to public data network equipment (for example, modems).

1.5 Terminology

Q: Does a station send frames or packets onto a network?

Q: Is a device that interconnects Ethernets to an FDDI backbone called a bridge or a router?

Q: Should you care?

A: Yes, yes, and yes.

Writers (especially in the lay or trade press) often get free and loose with terminology. *Packets* and *frames* are treated interchangeably; *routers* are called *bridges*, *bridges* are called *gateways*, or *multilayer switches* are called *switches* even though they do not fit the traditional definition of a switch. (Multilayer switches will be discussed in further detail in Chapter 4.) In many cases, it really does not matter whether the wording reflects strict technical correctness. Such details are usually not vital for a broad, high-level understanding of system behavior. We, however, have no such luxury here. This book examines the behavior of stations in internetworking devices at a highly detailed level. As such, we often need to be precise in our terminology, so that it is clear exactly what is being discussed or manipulated.

Network terminology tends to follow the layered architectural model. We use different terms to describe the devices, objects, and processes that implement the functions of each layer. This actually makes life somewhat easier; we can generally tell which architectural layer is being discussed just from the terms being used.

1.5.1 Applications, Clients, and Service Providers

As depicted in Figure 1-16, each layer in a suite of protocols can be seen to use the services of a lower-layer *service provider* and to provide services to a higher-layer *client*. Information is passed across the service interfaces between the layers, but a given layer simply does its defined job — processing transmission requests from higher-layer clients and passing them down to the next lower layer, or processing received information from a lower-layer service provider and delivering the contained payload up the stack. If we look at a protocol suite from a standpoint of architectural purity, each layer is unaware of the nature of the layers both above and below it. A network layer entity (for example, an IP implementation) is unaware whether its client is TCP, UDP, OSPF, or some other entity. Similarly, the IP entity performs the same functions whether it is operating over an Ethernet or WAN link.

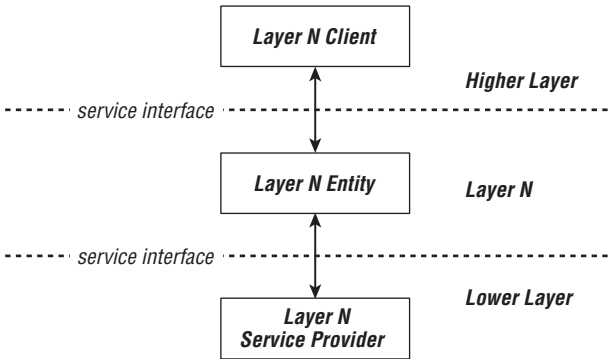


Figure 1-16 Clients and service providers

In particular, there is no way for a given layer to know the architectural layer at which its client resides. As discussed in section 1.1.6, an end user application may invoke network services at any layer of the architecture. The entity providing services at some middle layer has no idea whether its client is a protocol at the next higher layer or an end user application. From the perspective of any given layer, every entity using its services is a *client application*. In this book, we use the term ''application'' in this very broad sense, meaning any client using the services of the layer entity under discussion. Thus, from the perspective of IP, a Transport protocol (for example, SNMP/UDP), and a routing protocol (for example, OSPF) are all client applications. The terms ''higher layer protocols'' and ''client applications'' are synonymous.

1.5.2 Encapsulation

As data are passed from a user application down the protocol stack for ultimate transmission onto a network, each layer generally adds some control information to the transmission, so that its peer entity at the receiver can properly interpret it and take action upon its receipt. This process is known as *encapsulation*. Each layer entity takes the information provided by its higher-layer client (called a *Service Data Unit*, or *SDU*), adds layer-specific control information (typically in the form of a *protocol header* and/or *trailer*), and passes this combined *Protocol Data Unit (PDU)* to its lower-layer service provider. Similarly, as the information is processed in the receiving device, each layer parses, interprets, and acts upon its layer-specific control information, strips away the header and trailer, and passes the decapsulated information up to its higher-layer client. As depicted in Figure 1-17, a Transport layer PDU thus becomes the Network layer SDU; the Network layer adds a protocol header to its SDU to create a Network layer PDU, which becomes the SDU of the Data Link layer, and so on.

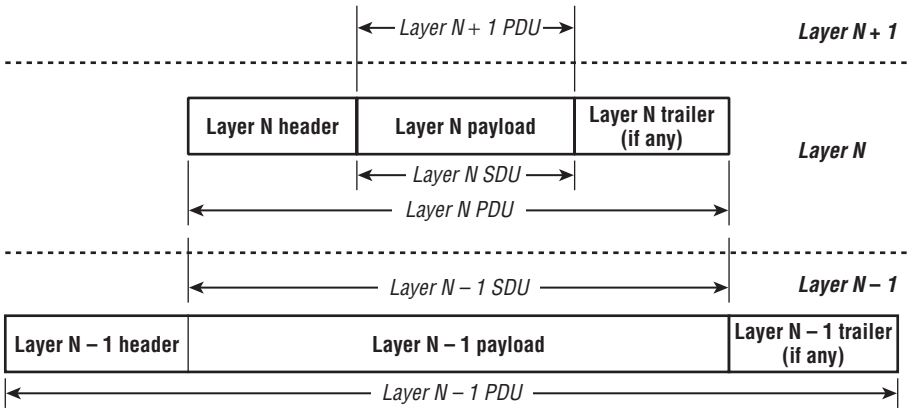


Figure 1-17 Service and Protocol Data Units

The unit of data at each protocol layer (the Protocol Data Unit) has a distinct name:

- A Transport PDU is called a *segment*, or in some protocol suites, a *message*.
- A Network PDU is called a *packet*. In the IP protocol suite, the term ''datagram'' is often used to denote a connectionless IP packet.
- A Data Link PDU is called a *frame*.
- A Physical layer PDU is called a *symbol stream*, or more commonly, a *stream*.

Thus, we can speak of TCP segments, IP packets, and Ethernet frames. Strictly speaking, there is no such thing as an Ethernet packet because an Ethernet entity (operating at the Data Link layer) deals only with its own encapsulation (a frame). An Ethernet frame encapsulates an IP packet, which encapsulates a TCP segment, as depicted in Figure 1-18. Throughout this book, these terms will take their strict interpretation as stated here.

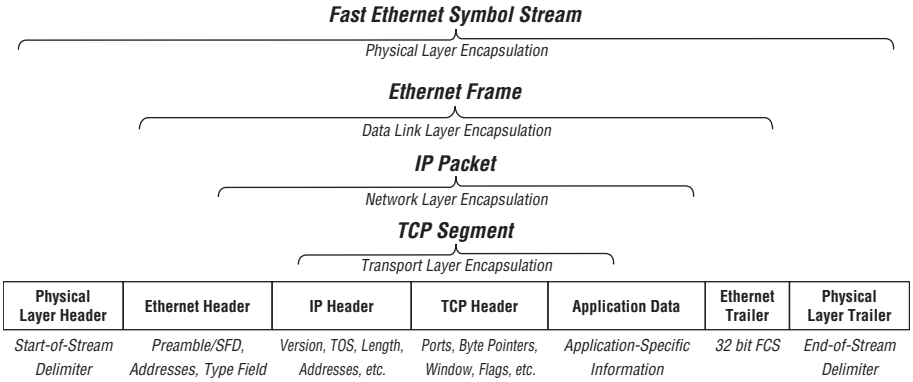


Figure 1-18 Example of layer encapsulation

A few points should be noted:

- If a frame is the encapsulation at the Data Link layer, then it should seem curious that the Ethernet Start-of-Frame Delimiter occurs *within* the Ethernet header; if it truly delimited the start of the frame, it should come at the end of the Physical layer encapsulation (see section 1.3.1.3). This is an artifact of the evolution of Ethernet from its DEC-Intel-Xerox roots to the IEEE 802.3 standard. In the original specification, the Ethernet Preamble was considered part of the Physical layer header; in the later IEEE 802.3 standard, the Preamble was defined to be part of the Data Link encapsulation. Thus, in the earlier standard, the Start-of-Frame Delimiter did actually mark the start of the Data Link frame.

- Depending on the nature of the Physical layer, there may be no need for any header or trailer fields at all. When using coaxial cable, 10 Mb/s Ethernet systems render the physical channel completely idle between frames; there is literally no voltage or current present if no station is transmitting. Thus, the presence of any signal at all can be used to indicate when the frame ends. In contrast, 100 Mb/s Ethernet systems provide a continuous signal on the physical channel even when frames are not being exchanged. Therefore, Fast Ethernet includes both a Start-of-Stream and an End-of-Stream Delimiter to explicitly indicate the boundaries of the Physical layer encapsulation.
- Strictly speaking, the Physical layer transmits *symbols*. A symbol is one or more encoded bits. Depending on the encoding scheme used, it may not even be possible to transmit a single bit on the physical channel. For example, Fast Ethernet used an encoding scheme whereby 4 bits of data are encoded into a 5-bit symbol (so-called 4B/5B encoding). Data is always transferred in 4-bit nibbles; it is not possible to send a single bit on a Fast Ethernet. On a 10 Mb/s Ethernet, the Manchester encoding scheme used does permit individual encoded bits to be transmitted. Regardless of the encoding scheme used, we often speak of the physical channel encapsulation as a *bit stream*, even though it is symbols (encoded bits) that are actually exchanged.

1.5.3 Stations and Interconnections

Any device that implements network services at the Data Link layer or above is called a *station*.[46] There are two types of stations:

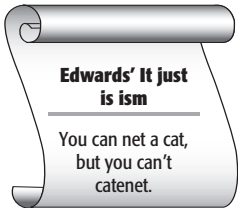
- **End stations** support one or more user network applications. End stations are the source and ultimate destination (sink) of all user data communicated across a network.
- **Intermediate stations** serve as relay devices, forwarding messages generated by end stations across a network so that they can reach their target end station destination(s).

A given device may have both end station and intermediate station capabilities. For example, a computer may be used both as an application server (end station) and as an internetwork router (intermediate station). The term ''workstation'' generally denotes a device that provides end station functionality exclusively.

[46] In general, devices operating completely within the Physical layer are not considered stations, as they are not addressable and do not implement any network protocols. Thus, cabling components and repeaters do not qualify as stations.

Intermediate stations are classified by the layer in the architecture at which they provide their relay function:

- **Data Link layer:** An intermediate station relaying frames among various Data Links is called a *bridge*. A collection of LANs bridged together is called a *catenet* or a *bridged* LAN.
- **Network layer:** An intermediate station relaying packets among networks is called an *internetwork router*, or simply a *router*. A collection of networks interconnected by routers is called an *internetwork*, or simply an *internet*. The Network-layer protocol used on an *internetwork* is called an *internetwork protocol*, or simply an *internet protocol*.



Note that ''internet'' and ''internet protocol'' are intentionally spelled here with lowercase letters. Any set of networks interconnected by routers is an internet, regardless of the internet protocol in use. Thus, one can build AppleTalk/DDP internets, NetWare/IPX internets, IP internets, and so on. One well-known IP internet is *the Internet* (capitalization intentional). The internet protocol used on the Internet is the Internet Protocol (IP).[47] This distinction between an internet and the Internet is maintained consistently throughout this book.

A *gateway* strictly refers to a device that provides interconnection between dissimilar protocol architectures. It is typically application-specific; for example, an e-mail gateway may be able to translate messages between the Simple Mail Transfer Protocol (SMTP) operating over a TCP/IP stack and the IBM proprietary PROFS system operating in an IBM SNA protocol environment, thus allowing mail interaction between users on a private SNA system and the Internet.

Historically, IP routers were often called gateways (IP gateways). While current literature and most modern products do use the (more correct) term ''router,'' many IP standards documents and protocols (and some old-time IP fogies) still carry this terminology baggage — for example, the Border Gateway Protocol (BGP).

In practice, ''gateway'' has become a marketing term for any device that connects anything to anything else. Because it evokes a much more pleasant visual image than either ''bridge'' or ''router,''[48] it has been used to sell everything from true Application-layer translators to RS-232 adaptor connectors. Fortunately, we do not need this term for anything in this book.

Table 1-2 shows the seven-layer OSI reference model along with the terms used for encapsulation and interconnection at each layer.

[47]This has to be one of the oddest sentences ever written in a networking text, but it does make complete sense.

[48]It has always reminded Rich of the Gateway Arch, and a wonderful time he had in St. Louis, Missouri, in 1983.

Table 1-2 Network Terminology Summary

	UNIT OF DATA	INTERCONNECTION DEVICE	MAXIMUM EXTENT (LAN CONTEXT)
APPLICATION	No standard	Gateway	No standard
PRESENTATION	No standard	No standard	No standard
SESSION	No standard	No standard	No standard
TRANSPORT	Segment or Message	No standard	internetwork or Internet
NETWORK	Packet	Router	
DATA LINK	Frame	Bridge	
			Catenet or Bridged LAN
PHYSICAL	Symbol or bit stream	Repeater	LAN

Where no standard term exists, the table reflects that. Fortunately, we will not need terms for these entries. (In fact, if there were a general need for these missing table entries, standard terms would have been developed long ago.)