As we pointed out in Chap. 1, networks can be divided into two categories: those using point-to-point connections and those using broadcast channels. This chapter deals with broadcast networks and their protocols.

In any broadcast network, the key issue is how to determine who gets to use the channel when there is competition for it. To make this point clearer, consider a conference call in which six people, on six different telephones, are all connected together so that each one can hear and talk to all the others. It is very likely that when one of them stops speaking, two or more will start talking at once, leading to chaos. In a face-to-face meeting, chaos is avoided by external means, for example, at a meeting, people raise their hands to request permission to speak. When only a single channel is available, determining who should go next is much harder. Many protocols for solving the problem are known and form the contents of this chapter. In the literature, broadcast channels are sometimes referred to as multiaccess channels or random access channels.

The protocols used to determine who goes next on a multiaccess channel belong to a sublayer of the data link layer called the MAC (Medium Access Control) sublayer. The MAC sublayer is especially important in LANs, nearly all of which use a multiaccess channel as the basis of their communication. WANs, in contrast, use point-to-point links, except for satellite networks. Because multiaccess channels and LANs are so closely related, in this chapter we will discuss LANs in general, including a few issues that are not strictly part of the MAC sublayer.
Technically, the MAC sublayer is the bottom part of the data link layer, so logically we should have studied it before examining all the point-to-point protocols in Chap. 3. Nevertheless, for most people, understanding protocols involving multiple parties is easier after two-party protocols are well understood. For that reason we have deviated slightly from a strict bottom-up order of presentation.

4.1 THE CHANNEL ALLOCATION PROBLEM

The central theme of this chapter is how to allocate a single broadcast channel among competing users. We will first look at static and dynamic schemes in general. Then we will examine a number of specific algorithms.

4.1.1 Static Channel Allocation in LANs and MANs

The traditional way of allocating a single channel, such as a telephone trunk, among multiple competing users is Frequency Division Multiplexing (FDM). If there are $N$ users, the bandwidth is divided into $N$ equal sized portions (see Fig. 2-30), each user being assigned one portion. Since each user has a private frequency band, there is no interference between users. When there is only a small and fixed number of users, each of which has a heavy (buffered) load of traffic (e.g., carriers’ switching offices), FDM is a simple and efficient allocation mechanism.

However, when the number of senders is large and continuously varying, or the traffic is bursty, FDM presents some problems. If the spectrum is cut up into $N$ regions, and fewer than $N$ users are currently interested in communicating, a large piece of valuable spectrum will be wasted. If more than $N$ users want to communicate, some of them will be denied permission, for lack of bandwidth, even if some of the users who have been assigned a frequency band hardly ever transmit or receive anything.

However, even assuming that the number of users could somehow be held constant at $N$, dividing the single available channel into static subchannels is inherently inefficient. The basic problem is that when some users are quiescent, their bandwidth is simply lost. They are not using it, and no one else is allowed to use it either. Furthermore, in most computer systems, data traffic is extremely bursty (peak traffic to mean traffic ratios of 1000:1 are common). Consequently, most of the channels will be idle most of the time.

The poor performance of static FDM can easily be seen from a simple queueing theory calculation. Let us start with the mean time delay, $T$, for a channel of capacity $C$ bps, with an arrival rate of $\lambda$ frames/sec, each frame having a length drawn from an exponential probability density function with mean $1/\mu$ bits/frame. With these parameters the arrival rate is $\lambda$ frames/sec and the service rate is $\mu C$ frames/sec so
For example, if $C$ is 100 Mbps, the mean frame length, $1/\mu$, is 10,000 bits, and the frame arrival rate, $\lambda$, is 5000 frames/sec, then $T = 200 \mu$sec. Note that if we ignored the queueing delay and just asked how long does it take to send a 10,000 bit frame on a 100-Mbps network we would get the (incorrect) answer of 100 $\mu$sec. That result only holds when there is no contention for the channel.

Now let us divide the single channel up into $N$ independent subchannels, each with capacity $C/N$ bps. The mean input rate on each of the subchannels will now be $\lambda/N$. Recomputing $T$ we get

$$T_{\text{FDM}} = \frac{1}{\mu(C/N) - (\lambda/N)} = \frac{N}{\mu C - \lambda} = NT$$

The mean delay using FDM is $N$ times worse than if all the frames were somehow magically arranged orderly in a big central queue.

Precisely the same arguments that apply to FDM also apply to time division multiplexing (TDM). Each user is statically allocated every $N$th time slot. If a user does not use the allocated slot, it just lies fallow. The same holds if we split the networks up physically. Using our previous example again, if we were to replace the 100-Mbps network with 10 networks of 10 Mbps each and statically allocate each user to one of them, the mean delay would jump from 200 $\mu$sec to 2 msec.

Since none of the traditional static channel allocation methods work well with bursty traffic, we will now explore dynamic methods.

### 4.1.2 Dynamic Channel Allocation in LANs and MANs

Before we get into the first of the many channel allocation methods to be discussed in this chapter, it is worthwhile carefully formulating the allocation problem. Underlying all the work done in this area are five key assumptions, described below.

1. **Station Model.** The model consists of $N$ independent stations (computers, telephones, personal communicators, etc.), each with a program or user that generates frames for transmission. Sometimes stations are called terminals. The probability of a frame being generated in an interval of length $\Delta t$ is $\lambda \Delta t$, where $\lambda$ is a constant (the arrival rate of new frames). Once a frame has been generated, the station is blocked and does nothing until the frame has been successfully transmitted.

2. **Single Channel Assumption.** A single channel is available for all communication. All stations can transmit on it and all can receive from it. As far as the hardware is concerned, all stations are equiv-
alent, although protocol software may assign priorities to them.

3. **Collision Assumption.** If two frames are transmitted simultaneously, they overlap in time and the resulting signal is garbled. This event is called a *collision*. All stations can detect collisions. A collided frame must be transmitted again later. There are no errors other than those generated by collisions.

4a. **Continuous Time.** Frame transmission can begin at any instant. There is no master clock dividing time into discrete intervals.

4b. **Slotted Time.** Time is divided into discrete intervals (slots). Frame transmissions always begin at the start of a slot. A slot may contain 0, 1, or more frames, corresponding to an idle slot, a successful transmission, or a collision, respectively.

5a. **Carrier Sense.** Stations can tell if the channel is in use before trying to use it. If the channel is sensed as busy, no station will attempt to use it until it goes idle.

5b. **No Carrier Sense.** Stations cannot sense the channel before trying to use it. They just go ahead and transmit. Only later can they determine whether or not the transmission was successful.

Some discussion of these assumptions is in order. The first one says that stations are independent, and that work is generated at a constant rate. It also implicitly assumes that each station only has one program or user, so while the station is blocked, no new work is generated. More sophisticated models allow multiprogrammed stations that can generate work while a station is blocked, but the analysis of these stations is much more complex.

The single channel assumption is the heart of the model. There are no external ways to communicate. Stations cannot raise their hands to request that the teacher call on them.

The collision assumption is also basic, although in some systems (notably spread spectrum), this assumption is relaxed, with surprising results. Also, some LANs, such as token rings, pass a special token from station to station, possession of which allows the current holder to transmit a frame. But in the coming sections we will stick to the single channel with contention and collisions model.

There are two alternative assumptions about time. Either it is continuous (4a) or it is slotted (4b). Some systems use one and some systems use the other, so we will discuss and analyze both. Obviously, for a given system, only one of them holds.

Similarly, a network can either have carrier sensing (5a) or not have it (5b). LANs generally have carrier sense. However, wireless networks cannot do it effectively because not every station may be within radio range of every other station. Stations on wired carrier sense networks can terminate their transmission
prematurely if they discover that it is colliding with another transmission. Collision detection is rarely done on wireless networks, for engineering reasons. Note that the word “carrier” in this sense refers to an electrical signal on the cable and has nothing to do with the common carriers (e.g., telephone companies) that date back to the Pony Express days.

4.2 MULTIPLE ACCESS PROTOCOLS

Many algorithms for allocating a multiple access channel are known. In the following sections we will study a small sample of the more interesting ones and give some examples of their use.

4.2.1 ALOHA

In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. Their work has been extended by many researchers since then (Abramson, 1985). Although Abramson’s work, called the ALOHA system, used ground-based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single shared channel.

We will discuss two versions of ALOHA here: pure and slotted. They differ with respect to whether or not time is divided up into discrete slots into which all frames must fit. Pure ALOHA does not require global time synchronization; slotted ALOHA does.

Pure ALOHA

The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent. There will be collisions, of course, and the colliding frames will be damaged. However, due to the feedback property of broadcasting, a sender can always find out whether or not its frame was destroyed by listening to the channel, the same way other users do. With a LAN, the feedback is immediate; with a satellite, there is a delay of 270 msec before the sender knows if the transmission was successful. If listening while transmitting is not possible for some reason, acknowledgements are needed. If the frame was destroyed, the sender just waits a random amount of time and sends it again. The waiting time must be random or the same frames will collide over and over, in lockstep. Systems in which multiple users share a common channel in a way that can lead to conflicts are widely known as contention systems.

A sketch of frame generation in an ALOHA system is given in Fig. 4-1. We have made the frames all the same length because the throughput of ALOHA systems is maximized by having a uniform frame size rather than allowing variable
Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled. If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed, and both will have to be retransmitted later. The checksum cannot (and should not) distinguish between a total loss and a near miss. Bad is bad.

A most interesting question is: What is the efficiency of an ALOHA channel? That is, what fraction of all transmitted frames escape collisions under these chaotic circumstances? Let us first consider an infinite collection of interactive users sitting at their computers (stations). A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. The station then transmits a frame containing the line and checks the channel to see if it was successful. If so, the user sees the reply and goes back to typing. If not, the user continues to wait and the frame is retransmitted over and over until it has been successfully sent.

Let the “frame time” denote the amount of time needed to transmit the standard, fixed-length frame (i.e., the frame length divided by the bit rate). At this point we assume that the infinite population of users generates new frames according to a Poisson distribution with mean $N$ frames per frame time. (The infinite-population assumption is needed to ensure that $N$ does not decrease as users become blocked.) If $N > 1$, the user community is generating frames at a higher rate than the channel can handle, and nearly every frame will suffer a collision. For reasonable throughput we would expect $0 < N < 1$.

In addition to the new frames, the stations also generate retransmissions of frames that previously suffered collisions. Let us further assume that the probability of $k$ transmission attempts per frame time, old and new combined, is also Poisson, with mean $G$ per frame time. Clearly, $G \geq N$. At low load (i.e., $N \approx 0$), there will be few collisions, hence few retransmissions, so $G \approx N$. At high load
there will be many collisions, so $G > N$. Under all loads, the throughput, $S$, is just the offered load, $G$, times the probability, $P_0$ of a transmission succeeding—that is, $S = GP_0$, where $P_0$ is the probability that a frame does not suffer a collision.

A frame will not suffer a collision if no other frames are sent within one frame time of its start, as shown in Fig. 4-2. Under what conditions will the shaded frame arrive undamaged? Let $t$ be the time required to send a frame. If any other user has generated a frame between time $t_0$ and $t_0 + t$, the end of that frame will collide with the beginning of the shaded one. In fact, the shaded frame’s fate was already sealed even before the first bit was sent, but since in pure ALOHA a station does not listen to the channel before transmitting, it has no way of knowing that another frame was already underway. Similarly, any other frame started between $t_0 + t$ and $t_0 + 2t$ will bump into the end of the shaded frame.

![Figure 4-2. Vulnerable period for the shaded frame.](image)

The probability that $k$ frames are generated during a given frame time is given by the Poisson distribution:

$$\Pr[k] = \frac{G^k e^{-G}}{k!} \quad (4-2)$$

so the probability of zero frames is just $e^{-G}$. In an interval two frame times long, the mean number of frames generated is $2G$. The probability of no other traffic being initiated during the entire vulnerable period is thus given by $P_0 = e^{-2G}$. Using $S = GP_0$, we get

$$S = Ge^{-2G}$$

The relation between the offered traffic and the throughput is shown in Fig. 4-3. The maximum throughput occurs at $G = 0.5$, with $S = 1/2e$, which is about 0.184. In other words, the best we can hope for is a channel utilization of 18 percent. This result is not very encouraging, but with everyone transmitting at will, we could hardly have expected a 100 percent success rate.
In 1972, Roberts published a method for doubling the capacity of an ALOHA system (Roberts, 1972). His proposal was to divide time up into discrete intervals, each interval corresponding to one frame. This approach requires the users to agree on slot boundaries. One way to achieve synchronization would be to have one special station emit a pip at the start of each interval, like a clock.

In Roberts’ method, which has come to be known as slotted ALOHA, in contrast to Abramson’s pure ALOHA, a computer is not permitted to send whenever a carriage return is typed. Instead, it is required to wait for the beginning of the next slot. Thus the continuous pure ALOHA is turned into a discrete one. Since the vulnerable period is now halved, the probability of no other traffic during the same slot as our test frame is $e^{-G}$ which leads to

$$S = Ge^{-G} \quad (4-3)$$

As you can see from Fig. 4-3, slotted ALOHA peaks at $G = 1$, with a throughput of $S = 1/e$ or about 0.368, twice that of pure ALOHA. If the system is operating at $G = 1$, the probability of an empty slot is 0.368 (from Eq. 4-2). The best we can hope for using slotted ALOHA is 37 percent of the slots empty, 37 percent successes, and 26 percent collisions. Operating at higher values of $G$ reduces the number of empties but increases the number of collisions exponentially. To see how this rapid growth of collisions with $G$ comes about, consider the transmission of a test frame. The probability that it will avoid a collision is $e^{-G}$, the probability that all the other users are silent in that slot. The probability of a collision is then $1 - e^{-G}$. The probability of a transmission requiring exactly $k$ attempts, (i.e., $k - 1$ collisions followed by one success) is

$$P_k = e^{-G}(1 - e^{-G})^{k-1}$$

The expected number of transmissions, $E$, per carriage return typed is then
\[ E = \sum_{k=1}^{\infty} k P_k = \sum_{k=1}^{\infty} k e^{-G} (1 - e^{-G})^{k-1} = e^G \]

As a result of the exponential dependence of \( E \) upon \( G \), small increases in the channel load can drastically reduce its performance.

Slotted Aloha is important for a reason that may not be initially obvious. It was devised in the 1970s, used in a few early experimental systems, then almost forgotten. When Internet access over the cable was invented, all of a sudden there was a problem of how to allocate a shared channel among multiple competing users, and slotted Aloha was pulled out of the garbage can to save the day. It has often happened that protocols that are perfectly valid fall into disuse for political reasons (e.g., some big company wants everyone to do things its way), but years later some clever person realizes that a long-discarded protocol solves his current problem. For this reason, in this chapter we will study a number of elegant protocols that are not currently in widespread use, but might easily be used in future applications, provided that enough network designers are aware of them. Of course, we will study various protocols that are in current use as well.

4.2.2 Carrier Sense Multiple Access Protocols

With slotted ALOHA the best channel utilization that can be achieved is \( 1/e \). This is hardly surprising, since with stations transmitting at will, without paying attention to what the other stations are doing, there are bound to be many collisions. In local area networks, however, it is possible for stations to detect what other stations are doing, and adapt their behavior accordingly. These networks can achieve a much better utilization than \( 1/e \). In this section we will discuss some protocols for improving performance.

Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called carrier sense protocols. A number of them have been proposed. Kleinrock and Tobagi (1975) have analyzed several such protocols in detail. Below we will mention several versions of the carrier sense protocols.

Persistent and Nonpersistent CSMA

The first carrier sense protocol that we will study here is called 1-persistent CSMA (Carrier Sense Multiple Access). When a station has data to send, it first listens to the channel to see if anyone else is transmitting at that moment. If the channel is busy, the station waits until it becomes idle. When the station detects an idle channel, it transmits a frame. If a collision occurs, the station waits a random amount of time and starts all over again. The protocol is called 1-persistent because the station transmits with a probability of 1 whenever it finds the channel idle.

The propagation delay has an important effect on the performance of the protocol. There is a small chance that just after a station begins sending, another
station will become ready to send and sense the channel. If the first station’s signal has not yet reached the second one, the latter will sense an idle channel and will also begin sending, resulting in a collision. The longer the propagation delay, the more important this effect becomes, and the worse the performance of the protocol.

Even if the propagation delay is zero, there will still be collisions. If two stations become ready in the middle of a third station’s transmission, both will wait politely until the transmission ends and then both will begin transmitting exactly simultaneously, resulting in a collision. If they were not so impatient, there would be fewer collisions. Even so, this protocol is far better than pure ALOHA, because both stations have the decency to desist from interfering with the third station’s frame. Intuitively, this will lead to a higher performance than pure ALOHA. Exactly the same holds for slotted ALOHA.

A second carrier sense protocol is nonpersistent CSMA. In this protocol, a conscious attempt is made to be less greedy than in the previous one. Before sending, a station senses the channel. If no one else is sending, the station begins doing so itself. However, if the channel is already in use, the station does not continually sense it for the purpose of seizing it immediately upon detecting the end of the previous transmission. Instead, it waits a random period of time and then repeats the algorithm. Intuitively this algorithm should lead to better channel utilization and longer delays than 1-persistent CSMA.

The last protocol is \textbf{p-persistent CSMA}. It applies to slotted channels and works as follows. When a station becomes ready to send, it senses the channel. If it is idle, it transmits with a probability \( p \). With a probability \( q = 1 - p \) it defers until the next slot. If that slot is also idle, it either transmits or defers again, with probabilities \( p \) and \( q \). This process is repeated until either the frame has been transmitted or another station has begun transmitting. In the latter case, it acts as if there had been a collision (i.e., it waits a random time and starts again). If the station initially senses the channel busy, it waits until the next slot and applies the above algorithm. Figure 4-4 shows the computed throughput versus offered traffic for all three protocols, as well as for pure and slotted ALOHA.

\textbf{CSMA with Collision Detection}

Persistent and nonpersistent CSMA protocols are clearly an improvement over ALOHA because they ensure that no station begins to transmit when it senses the channel busy. Another improvement is for stations to abort their transmissions as soon as they detect a collision. In other words, if two stations sense the channel to be idle and begin transmitting simultaneously, they will both detect the collision almost immediately. Rather than finish transmitting their frames, which are irretrievably garbled anyway, they should abruptly stop transmitting as soon as the collision is detected. Quickly terminating damaged frames saves time and bandwidth. This protocol, known as \textbf{CSMA/CD} (CSMA
SEC. 4.2 MULTIPLE ACCESS PROTOCOLS

Figure 4-4. Comparison of the channel utilization versus load for various random access protocols.

with Collision Detection) is widely used on LANs in the MAC sublayer. In particular, it is the basis of the popular Ethernet LAN, so it is worth devoting some
time to looking at it in detail.

CSMA/CD, as well as many other LAN protocols, uses the conceptual model of Fig. 4-5. At the point marked \( t_0 \), a station has finished transmitting its frame. Any other station having a frame to send may now attempt to do so. If two or more stations decide to transmit simultaneously, there will be a collision. Collisions can be detected by looking at the power or pulse width of the received signal and comparing it to the transmitted signal.

After a station detects a collision, it aborts its transmission, waits a random period of time, and then tries again, assuming that no other station has started transmitting in the meantime. Therefore, our model for CSMA/CD will consist of alternating contention and transmission periods, with idle periods occurring when all stations are quiet (e.g., for lack of work).

Now let us look closely at the details of the contention algorithm. Suppose that two stations both begin transmitting at exactly time \( t_0 \). How long will it take them to realize that there has been a collision? The answer to this question is vital to determining the length of the contention period, and hence what the delay and throughput will be. The minimum time to detect the collision is then just the time it takes the signal to propagate from one station to the other.

Based on this reasoning, you might think that a station not hearing a collision for a time equal to the full cable propagation time after starting its transmission could be sure it had seized the cable. By “seized,” we mean that all other stations knew it was transmitting and would not interfere. This conclusion is wrong. Consider the following worst-case scenario. Let the time for a signal to propagate
between the two farthest stations be $\tau$. At $t_0$, one station begins transmitting. At $\tau - \varepsilon$, an instant before the signal arrives at the most distant station, that station also begins transmitting. Of course, it detects the collision almost instantly and stops, but the little noise burst caused by the collision does not get back to the original station until time $2\tau - \varepsilon$. In other words, in the worst case a station cannot be sure that it has seized the channel until it has transmitted for $2\tau$ without hearing a collision. For this reason we will model the contention interval as a slotted ALOHA system with slot width $2\tau$. On a 1-km long coaxial cable, $\tau \approx 5 \mu$s.

For simplicity we will assume that each slot contains just 1 bit. Once the channel has been seized, a station can transmit at any rate it wants to, of course, not just at $1$ bit per $2\tau$ sec.

It is important to realize that collision detection is an analog process. The station’s hardware must listen to the cable while it is transmitting. If what it reads back is different from what it is putting out, it knows that a collision is occurring. The implication is that the signal encoding must allow collisions to be detected (e.g., a collision of two 0-volt signals may well be impossible to detect). For this reason, special encoding is commonly used.

It is also worth noting that a sending station must continually monitor the channel for noise bursts that might indicate a collision. For this reason, CSMA/CD with a single channel is inherently a half-duplex system. It is impossible for a station to transmit and receive frames at the same time because the receiving logic is in use looking for collisions during every transmission.

To avoid any misunderstanding, it is worth noting that no MAC-sublayer protocol guarantees reliable delivery. Even in the absence of collisions, the receiver may not have copied the frame correctly due to various reasons (e.g., lack of buffer space or a missed interrupt).
4.2.3 Collision-Free Protocols

Although collisions do not occur with CSMA/CD once a station has unambiguously seized the channel, they can still occur during the contention period. These collisions adversely affect the system performance, especially when the cable is long (i.e., large $\tau$) and the frames are short. As very long, high-bandwidth fiber optic networks come into use, the combination of large $\tau$ and short frames will become an increasingly serious problem. In this section, we will examine some protocols that resolve the contention for the channel without any collisions at all, not even during the contention period.

In the protocols to be described, we make the assumption that there are exactly $N$ stations, each with a unique address from 0 to $N - 1$ “wired” into it. It does not matter that some stations may be inactive part of the time. We also assume that propagation delay is negligible. The basic question remains: Which station gets the channel after a successful transmission? We continue using the model of Fig. 4-5 with its discrete contention slots.

A Bit-Map Protocol

In our first collision-free protocol, the basic bit-map method, each contention period consists of exactly $N$ slots. If station 0 has a frame to send, it transmits a 1 bit during the zeroth slot. No other station is allowed to transmit during this slot. Regardless of what station 0 does, station 1 gets the opportunity to transmit a 1 during slot 1, but only if it has a frame queued. In general, station $j$ may announce the fact that it has a frame to send by inserting a 1 bit into slot $j$. After all $N$ slots have passed by, each station has complete knowledge of which stations wish to transmit. At that point, they begin transmitting in numerical order (see Fig. 4-6).

Since everyone agrees on who goes next, there will never be any collisions. After the last ready station has transmitted its frame, an event all stations can easily monitor, another $N$ bit contention period is begun. If a station becomes ready just after its bit slot has passed by, it is out of luck and must remain silent until every station has had a chance and the bit map has come around again. Protocols like this in which the desire to transmit is broadcast before the actual
transmission are called **reservation protocols**.

Let us briefly analyze the performance of this protocol. For convenience, we will measure time in units of the contention bit slot, with data frames consisting of $d$ time units. Under conditions of low load, the bit map will simply be repeated over and over, for lack of data frames.

Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, when it becomes ready to send, the “current” slot will be somewhere in the middle of the bit map. On the average, the station will have to wait $N/2$ slots for the current scan to finish and another full $N$ slots for the following scan to run to completion before it may begin transmitting.

The prospects for high-numbered stations are brighter. Generally, these will only have to wait half a scan ($N/2$ bit slots) before starting to transmit. High-numbered stations rarely have to wait for the next scan. Since low-numbered stations must wait on the average $1.5N$ slots and high-numbered stations must wait on the average $0.5N$ slots, the mean for all stations is $N$ slots. The channel efficiency at low load is easy to compute. The overhead per frame is $N$ bits, and the amount of data is $d$ bits, for an efficiency of $d/(N + d)$.

At high load, when all the stations have something to send all the time, the $N$ bit contention period is prorated over $N$ frames, yielding an overhead of only 1 bit per frame, or an efficiency of $d/(d + 1)$. The mean delay for a frame is equal to the sum of the time it queues inside its station, plus an additional $N(d + 1)/2$ once it gets to the head of its internal queue.

**Binary Countdown**

A problem with the basic bit-map protocol is that the overhead is 1 bit per station so it does not scale well to networks with thousands of stations. We can do better than that by using binary station addresses. A station wanting to use the channel now broadcasts its address as a binary bit string, starting with the high-order bit. All addresses are assumed to be the same length. The bits in each address position from different stations are BOOLEAN ORed together. We will call this protocol **binary countdown**. It was used in Datakit (Fraser, 1987). It implicitly assumes that the transmission delays are negligible so that all stations see asserted bits essentially instantaneously.

To avoid conflicts, an arbitration rule must be applied: as soon as a station sees that a high-order bit position that is 0 in its address has been overwritten with a 1, it gives up. For example, if stations 0010, 0100, 1001, and 1010 are all trying to get the channel, in the first bit time the stations transmit 0, 0, 1, and 1, respectively. These are ORed together to form a 1. Stations 0010 and 0100 see the 1 and know that a higher-numbered station is competing for the channel, so they give up for the current round. Stations 1001 and 1010 continue.

The next bit is 0, and both stations continue. The next bit is 1, so station 1001 gives up. The winner is station 1010, because it has the highest address. After
winning the bidding, it may now transmit a frame, after which another bidding cycle starts. The protocol is illustrated in Fig. 4-7. It has the property that higher-numbered stations have a higher priority than lower-numbered stations, which may be either good or bad, depending on the context.

![Figure 4-7. The binary countdown protocol. A dash indicates silence.]

The channel efficiency of this method is \( \frac{d}{d + \log_2 N} \). If, however, the frame format has been cleverly chosen so that the sender’s address is the first field in the frame, even these \( \log_2 N \) bits are not wasted, and the efficiency is 100 percent.

Mok and Ward (1979) have described a variation of binary countdown using a parallel rather than a serial interface. They also suggest using virtual station numbers, with the virtual station numbers from 0 up to and including the successful station being circularly permuted after each transmission, in order to give higher priority to stations that have been silent unusually long. For example, if stations \( C, H, D, A, G, B, E, F \) have priorities 7, 6, 5, 4, 3, 2, 1, and 0, respectively, then a successful transmission by \( D \) puts it at the end of the list, giving a priority order of \( C, H, A, G, B, E, F, D \). Thus \( C \) remains virtual station 7, but \( A \) moves up from 4 to 5 and \( D \) drops from 5 to 0. Station \( D \) will now only be able to acquire the channel if no other station wants it.

Binary countdown is an example of a simple, elegant, and efficient protocol that is waiting to be rediscovered. Hopefully, it will find a new home some day.

### 4.2.4 Limited-Contention Protocols

We have now considered two basic strategies for channel acquisition in a cable network: contention, as in CSMA, and collision-free methods. Each strategy can be rated as to how well it does with respect to the two important
performance measures, delay at low load and channel efficiency at high load. Under conditions of light load, contention (i.e., pure or slotted ALOHA) is preferable due to its low delay. As the load increases, contention becomes increasingly less attractive, because the overhead associated with channel arbitration becomes greater. Just the reverse is true for the collision-free protocols. At low load, they have high delay, but as the load increases, the channel efficiency improves rather than gets worse as it does for contention protocols.

Obviously, it would be nice if we could combine the best properties of the contention and collision-free protocols, arriving at a new protocol that used contention at low loads to provide low delay, but used a collision-free technique at high load to provide good channel efficiency. Such protocols, which we will call limited-contention protocols, do, in fact, exist, and will conclude our study of carrier sense networks.

Up until now the only contention protocols we have studied have been symmetric, that is, each station attempts to acquire the channel with some probability, $p$, with all stations using the same $p$. Interestingly enough, the overall system performance can sometimes be improved by using a protocol that assigns different probabilities to different stations.

Before looking at the asymmetric protocols, let us quickly review the performance of the symmetric case. Suppose that $k$ stations are contending for channel access. Each has a probability $p$ of transmitting during each slot. The probability that some station successfully acquires the channel during a given slot is then $kp(1-p)^{k-1}$. To find the optimal value of $p$, we differentiate with respect to $p$, set the result to zero, and solve for $p$. Doing so, we find that the best value of $p$ is $1/k$. Substituting $p = 1/k$ we get

$$\Pr[\text{success with optimal } p] = \left[ \frac{k-1}{k} \right]^{k-1}$$

This probability is plotted in Fig. 4-8. For small numbers of stations, the chances of success are good, but as soon as the number of stations reaches even five, the probability has dropped close to its asymptotic value of $1/e$.

From Fig. 4-8, it is fairly obvious that the probability of some station acquiring the channel can be increased only by decreasing the amount of competition. The limited-contention protocols do precisely that. They first divide the stations up into (not necessarily disjoint) groups. Only the members of group 0 are permitted to compete for slot 0. If one of them succeeds, it acquires the channel and transmits its frame. If the slot lies fallow or if there is a collision, the members of group 1 contend for slot 1, etc. By making an appropriate division of stations into groups, the amount of contention for each slot can be reduced, thus operating each slot near the left end of Fig. 4-8.

The trick is how to assign stations to slots. Before looking at the general case, let us consider some special cases. At one extreme, each group has but one
member. Such an assignment guarantees that there will never be collisions, because at most one station is contending for any given slot. We have seen such protocols before (e.g., binary countdown). The next special case is to assign two stations per group. The probability that both will try to transmit during a slot is $p^2$, which for small $p$ is negligible. As more and more stations are assigned to the same slot, the probability of a collision grows, but the length of the bit-map scan needed to give everyone a chance shrinks. The limiting case is a single group containing all stations (i.e., slotted ALOHA). What we need is a way to assign stations to slots dynamically, with many stations per slot when the load is low and few (or even just one) station per slot when the load is high.

**The Adaptive Tree Walk Protocol**

One particularly simple way of performing the necessary assignment is to use the algorithm devised by the U.S. Army for testing soldiers for syphilis during World War II (Dorfman, 1943). In short, the Army took a blood sample from $N$ soldiers. A portion of each sample was poured into a single test tube. This mixed sample was then tested for antibodies. If none were found, all the soldiers in the group were declared healthy. If antibodies were present, two new mixed samples were prepared, one from soldiers 1 through $N/2$ and one from the rest. The process was repeated recursively until the infected soldiers were determined.

For the computer version of this algorithm (Capetanakis, 1979) it is convenient to think of the stations as the leaves of a binary tree, as illustrated in Fig. 4-9. In the first contention slot following a successful frame transmission, slot 0, all stations are permitted to try to acquire the channel. If one of them does so, fine. If there is a collision, then during slot 1 only those stations falling under node 2 in the tree may compete. If one of them acquires the channel, the slot
following the frame is reserved for those stations under node 3. If, on the other hand, two or more stations under node 2 want to transmit, there will be a collision during slot 1, in which case it is node 4’s turn during slot 2.

![Figure 4-9. The tree for eight stations.](image)

In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node’s left and right children. If a bit slot is idle or if there is only one station that transmits in it, the searching of its node can stop, because all ready stations have been located. (Were there more than one, there would have been a collision.)

When the load on the system is heavy, it is hardly worth the effort to dedicate slot 0 to node 1, because that makes sense only in the unlikely event that precisely one station has a frame to send. Similarly, one could argue that nodes 2 and 3 should be skipped as well for the same reason. Put in more general terms, at what level in the tree should the search begin? Clearly, the heavier the load, the farther down the tree the search should begin. We will assume that each station has a good estimate of the number of ready stations, \( q \), for example, from monitoring recent traffic.

To proceed, let us number the levels of the tree from the top, with node 1 in Fig. 4-9 at level 0, nodes 2 and 3 at level 1, etc. Notice that each node at level \( i \) has a fraction \( 2^{-i} \) of the stations below it. If the \( q \) ready stations are uniformly distributed, the expected number of them below a specific node at level \( i \) is just \( 2^{-i}q \). Intuitively, we would expect the optimal level to begin searching the tree as the one at which the mean number of contending stations per slot is 1, that is, the level at which \( 2^{-i}q = 1 \). Solving this equation we find that \( i = \log_2 q \).

Numerous improvements to the basic algorithm have been discovered and are discussed in some detail by Bertsekas and Gallager (1992). For example, consider the case of stations \( G \) and \( H \) being the only ones wanting to transmit. At node 1 a collision will occur, so 2 will be tried and discovered idle. It is pointless
to probe node 3 since it is guaranteed to have a collision (we know that two or more stations under 1 are ready and none of them are under 2 so they must all be under 3). The probe of 3 can be skipped and 6 tried next. When this probe also turns up nothing, 7 can be skipped and node $G$ tried next.

### 4.2.5 Wavelength Division Multiple Access Protocols

A different approach to channel allocation is to divide the channel into sub-channels using FDM, TDM, or both, and dynamically allocate them as needed. Schemes like this are commonly used on fiber optic LANs in order to permit different conversations to use different wavelengths (i.e., frequencies) at the same time. In this section we will examine one such protocol (Humblet et al., 1992).

A simple way to build an all-optical LAN is to use a passive star coupler (see Fig. 2-10). In effect, two fibers from each station are fused to a glass cylinder. One fiber is for output to the cylinder and one is for input from the cylinder. Light output by any station illuminates the cylinder and can be detected by all the other stations. Passive stars can handle hundreds of stations.

To allow multiple transmissions at the same time, the spectrum is divided up into channels (wavelength bands), as shown in Fig. 2-30. In this protocol, **WDMA (Wavelength Division Multiple Access)**, each station is assigned two channels. A narrow channel is provided as a control channel to signal the station, and a wide channel is provided so the station can output data frames.

![Diagram of Wavelength Division Multiple Access](image)

**Figure 4-10.** Wavelength division multiple access.

Each channel is divided up into groups of time slots, as depicted in Fig. 4-10.
Let us call the number of slots in the control channel \( m \) and the number of slots in the data channel \( n + 1 \), where \( n \) of these are for data and the last one is used by the station to report on its status (mainly, which slots on both channels are free). On both channels, the sequence of slots repeats endlessly, with slot 0 being marked in a special way so latecomers can detect it. All channels are synchronized by a single global clock.

The protocol supports three classes of traffic: (1) constant data rate connection-oriented traffic, such as uncompressed video, (2) variable data rate connection-oriented traffic, such as file transfer, and (3) datagram traffic, such as UDP packets. For the two connection-oriented protocols, the idea is that for \( A \) to communicate with \( B \), it must first insert a CONNECTION REQUEST frame in a free slot on \( B \)’s control channel. If \( B \) accepts, communication can take place on \( A \)’s data channel.

Each station has two transmitters and two receivers, as follows:

1. A fixed-wavelength receiver for listening to its own control channel.
2. A tunable transmitter for sending on other stations’ control channels.
3. A fixed-wavelength transmitter for outputting data frames.
4. A tunable receiver for selecting a data transmitter to listen to.

In other words, every station listens to its own control channel for incoming requests but has to tune to the transmitter’s wavelength to get the data. Wavelength tuning is done by a Fabry-Perot or Mach-Zehnder interferometer that filters out all wavelengths except the desired wavelength band.

Let us now consider how station \( A \) sets up a class 2 communication channel with station \( B \) for, say, file transfer. First, \( A \) tunes its data receiver to \( B \)’s data channel and waits for the status slot. This slot tells which control slots are currently assigned and which are free. In Fig. 4-10, for example, we see that of \( B \)’s eight control slots, 0, 4, and 5 are free. The rest are occupied (indicated by crosses).

\( A \) picks one of the free control slots, say 4, and inserts its CONNECTION REQUEST message there. Since \( B \) constantly monitors its control channel, it sees the request and grants it by assigning slot 4 to \( A \). This assignment is announced in the status slot of \( B \)’s data channel. When \( A \) sees the announcement, it knows it has a unidirectional connection. If \( A \) asked for a two-way connection, \( B \) now repeats the same algorithm with \( A \).

It is possible that at the same time \( A \) tried to grab \( B \)’s control slot 4, \( C \) did the same thing. Neither will get it, and both will notice the failure by monitoring the status slot in \( B \)’s control channel. They now each wait a random amount of time and try again later.

At this point, each party has a conflict-free way to send short control messages to the other one. To perform the file transfer, \( A \) now sends \( B \) a control
message saying, for example, “Please watch my next data output slot 3. There is a data frame for you in it.” When B gets the control message, it tunes its receiver to A’s output channel to read the data frame. Depending on the higher-layer protocol, B can use the same mechanism to send back an acknowledgement if it wishes.

Note that a problem arises if both A and C have connections to B and each of them suddenly tells B to look at slot 3. B will pick one of these requests at random, and the other transmission will be lost.

For constant rate traffic, a variation of this protocol is used. When A asks for a connection, it simultaneously says something like: Is it all right if I send you a frame in every occurrence of slot 3? If B is able to accept (i.e., has no previous commitment for slot 3), a guaranteed bandwidth connection is established. If not, A can try again with a different proposal, depending on which output slots it has free.

Class 3 (datagram) traffic uses yet another variation. Instead of writing a CONNECTION REQUEST message into the control slot it just found (4), it writes a DATA FOR YOU IN SLOT 3 message. If B is free during the next data slot 3, the transmission will succeed. Otherwise, the data frame is lost. In this manner, no connections are ever needed.

Several variants of the entire protocol are possible. For example, instead of giving each station its own control channel, a single control channel can be shared by all stations. Each station is assigned a block of slots in each group, effectively multiplexing multiple virtual channels onto one physical one.

It is also possible to make do with a single tunable transmitter and a single tunable receiver per station by having each station’s channel be divided up into $m$ control slots followed by $n + 1$ data slots. The disadvantage here is that senders have to wait longer to capture a control slot and consecutive data frames are further apart because some control information is in the way.

Numerous other WDMA protocols have been proposed, differing in the details. Some have one control channel, some have multiple control channels. Some take propagation delay into account, others do not; some make tuning time an explicit part of the model, others ignore it. The protocols also differ in terms of processing complexity, throughput and scalability. When a large number of frequencies are being used, the system is sometimes called DWDM (Dense Wavelength Division Multiplexing). For more information see (Bogineni et al., 1993; Chen, 1994; Goralski, 2001; Kartalopoulos, 1999; and Levine and Akyildiz, 1995).

### 4.2.6 Wireless LAN Protocols

As the number of mobile computing and communication devices grows, so does the demand to connect them to the outside world. Even the very first mobile telephones had the ability to connect to other telephones. The first portable
computers did not have this capability, but soon afterward, modems became common- 
place on notebook computers. To go on-line, these computers had to be plugged into a telephone wall socket. Requiring a wired connection to the fixed network meant that the computers were portable, but not mobile.

To achieve true mobility, notebook computers need to use radio (or infrared) signals for communication. In this manner, dedicated users can read and send e-mail while hiking or boating. A system of notebook computers that communicate by radio can be regarded as a wireless LAN, as we discussed in Sec. 1.5.4. These LANs have somewhat different properties than conventional LANs and require special MAC sublayer protocols. In this section we will examine some of these protocols. More information about wireless LANs can be found in (Geier, 2002; and O’Hara and Petrick, 1999).

A common configuration for a wireless LAN is an office building with base stations strategically placed around the building. All the base stations are wired together using copper or fiber. If the transmission power of the base stations and notebooks is adjusted to have a range of 3 or 4 meters, then each room becomes a single cell, and the entire building becomes a large cellular system, as in the traditional cellular telephony systems we studied in Chap. 2. Unlike cellular telephone systems, each cell has only one channel, covering the entire available bandwidth. Typically its bandwidth is 11 to 50 Mbps.

In our discussions below, we will make the simplifying assumption that all radio transmitters have some fixed range. When a receiver is within range of two active transmitters, the resulting signal will generally be garbled and useless (but with certain exceptions to be discussed later). It is important to realize that in some wireless LANs, not all stations are within range of one another, which leads to a variety of complications. Furthermore, for indoor wireless LANs, the presence of walls between stations can have a major impact on the effective range of each station.

A naive approach to using a wireless LAN might be to try CSMA: just listen for other transmissions and only transmit if no one else is doing so. The trouble is, this protocol is not really appropriate because what matters is interference at the receiver, not at the sender. To see the nature of the problem, consider Fig. 4-11, where four wireless stations are illustrated. For our purposes, it does not matter which are base stations and which are notebooks. The radio range is such that A and B are within each other’s range and can potentially interfere with one another. C can also potentially interfere with both B and D, but not with A.

First consider what happens when A is transmitting to B, as depicted in Fig. 4-11(a). If C senses the medium, it will not hear A because A is out of range, and thus falsely conclude that it can transmit to B. If C does start transmitting, it will interfere at B, wiping out the frame from A. The problem of a station not being able to detect a potential competitor for the medium because the competitor is too far away is called the hidden station problem.

Now let us consider the reverse situation: B transmitting to A, as shown in
SEC. 4.2 MULTIPLE ACCESS PROTOCOLS

A (a) 
B C D

Radio range

Fig. 4-11. A wireless LAN. (a) A transmitting. (b) B transmitting.

Fig. 4-11(b). If C senses the medium, it will hear an ongoing transmission and falsely conclude that it may not send to D, when in fact such a transmission would cause bad reception only in the zone between B and C, where neither of the intended receivers is located. This situation is called the exposed station problem.

The problem is that before starting a transmission, a station really wants to know whether or not there is activity around the receiver. CSMA merely tells it whether or not there is activity around the station sensing the carrier. With a wire, all signals propagate to all stations so only one transmission can take place at once anywhere in the system. In a system based on short-range radio waves, multiple transmissions can occur simultaneously if they all have different destinations and these destinations are out of range of one another.

Another way to think about this problem is to imagine an office building in which every employee has a wireless notebook computer. Suppose that Linda wants to send a message to Milton. Linda’s computer senses the local environment and, detecting no activity, starts sending. However, there may still be a collision in Milton’s office because a third party may currently be sending to him from a location so far from Linda that her computer could not detect it.

MACA and MACAW

An early protocol designed for wireless LANs is MACA (Multiple Access with Collision Avoidance) (Karn, 1990). The basic idea behind it is for the sender to stimulate the receiver into outputting a short frame, so stations nearby can detect this transmission and avoid transmitting themselves for the duration of the upcoming (large) data frame. MACA is illustrated in Fig. 4-12.

Let us consider how A sends a frame to B. A starts by sending an RTS (Request To Send) frame to B, as shown in Fig. 4-12(a). This short frame (30 bytes) contains the length of the data frame that will eventually follow. Then B replies with a CTS (Clear To Send) frame, as shown in Fig. 4-12(b). The CTS frame contains the data length (copied from the RTS frame). Upon receipt of the CTS frame, A begins transmission.

Now let us see how stations overhearing either of these frames react. Any station hearing the RTS is clearly close to A and must remain silent long enough
for the CTS to be transmitted back to A without conflict. Any station hearing the CTS is clearly close to B and must remain silent during the upcoming data transmission, whose length it can tell by examining the CTS frame.

In Fig. 4-12, C is within range of A but not within range of B. Therefore it hears the RTS from A but not the CTS from B. As long as it does not interfere with the CTS, it is free to transmit while the data frame is being sent. In contrast, D is within range of B but not A. It does not hear the RTS but does hear the CTS. Hearing the CTS tips it off that it is close to a station that is about to receive a frame, so it defers from sending anything until that frame is expected to be finished. Station E hears both control messages, and like D, must be silent until the data frame is complete.

Despite these precautions, collisions can still occur. For example, B and C could both send RTS frames to A at the same time. These will collide and be lost. In the event of a collision, an unsuccessful transmitter (i.e., one that does not hear a CTS within the expected time interval) waits a random amount of time and tries again later. The algorithm used is binary exponential backoff, which we will study when we come to Ethernet.

Based on simulation studies of MACA, Bharghavan et al. (1994) fine tuned MACA to improve its performance and renamed their new protocol MACAW (MACA for Wireless). To start with, they noticed that without data link layer acknowledgements, lost frames were not retransmitted until the transport layer noticed their absence, much later. They solved this problem by introducing an ACK frame after each successful data frame. They also observed that CSMA has some utility, namely to keep a station from transmitting an RTS at the same time another nearby station is also doing so to the same destination, so carrier sensing was added. In addition, they decided to run the backoff algorithm separately for
each data stream (source-destination pair), rather than for each station. This change improves the fairness of the protocol. Finally, they added a mechanism for stations to exchange information about congestion, and a way to make the backoff algorithm react less violently to temporary problems, to improve system performance.

4.3 ETHERNET

We have now finished our general discussion of channel allocation protocols in the abstract, so it is time to see how these principles apply to real systems, in particular, LANs. As discussed in Sec. 1.5.3, the IEEE has standardized a number of local area networks and metropolitan area networks under the name of IEEE 802. A few have survived but many have not, as we saw in Fig. 1-38 (???). Some people who believe in reincarnation think that Charles Darwin came back as a member of the IEEE Standards Association to weed out the unfit. The most important survivors are 802.3 (Ethernet) and 802.11 (wireless LAN). With 802.15 (Bluetooth) and 802.16 (wireless MAN) it is too early to tell. Please consult the 5th edition of this book to find out. Both 802.3 and 802.11 have different physical layers and different MAC sublayers, but converge on the same logical link control sublayer (defined in 802.2), so they have the same interface to the network layer.

We introduced Ethernet in Sec. 1.5.3 and will not repeat that material here. Instead we will focus on the technical details of Ethernet, the protocols, and recent developments in high-speed (gigabit) Ethernet. Since Ethernet and IEEE 802.3 are identical except for two minor differences that we will discuss shortly, many people use the terms “Ethernet” and “IEEE 802.3” interchangeably, and we will do so, too. For more information about Ethernet, see (Breyer and Riley, 1999; Seifert, 1998; and Spurgeon, 2000)

4.3.1 Ethernet Cabling

Since the name “Ethernet” refers to the cable (the ether), let us start our discussion there. Four types of cabling are commonly used, as shown in Fig. 4-13. Historically, 10Base5 cabling, popularly called thick Ethernet, came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go. (The 802.3 standard does not actually require the cable to be yellow, but it does suggest it.) Connections to it are generally made using vampire taps, in which a pin is very carefully forced halfway into the coaxial cable’s core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500 meters. In other words, the first number is the speed in Mbps, then comes the word “Base” (initially broadband was considered but rejected), and finally an indication of the transmission medium. If the medium is coax, its length is given in units of 100 m.
Historically, the second cable type was 10Base2 or thin Ethernet, which, in contrast to the garden-hose-like thick Ethernet, bends easily. Connections to it are made using industry standard BNC connectors to form T junctions, rather than using vampire taps. These are easier to use and more reliable. Thin Ethernet is much cheaper and easier to install, but it can run for only 200 meters per segment, each of which can handle only 30 machines.

Detecting cable breaks, bad taps, or loose connectors can be a major problem with both media. For this reason, techniques have been developed to track them down. Basically, a pulse of known shape is injected into the cable. If the pulse hits an obstacle or the end of the cable, an echo will be generated and sent back. By carefully timing the interval between sending the pulse and receiving the echo, it is possible to localize the origin of the echo. This technique is called time domain reflectometry.

The problems associated with finding cable drove systems toward a different kind of wiring pattern, in which all stations have a cable running to a central hub in which they are all connected electrically. Usually, these wires are telephone company twisted pairs, since most office buildings are already wired this way, and there are normally plenty of spare pairs available. This scheme is called 10Base-T. We will discuss an improved version of this idea—switches—later in this chapter.

These three wiring schemes are illustrated in Fig. 4-14. For 10Base5, a transceiver is clamped securely around the cable so that its tap makes contact with the inner core. The transceiver contains the electronics that handle carrier detection and collision detection. When a collision is detected, the transceiver also puts a special invalid signal on the cable to ensure that all other transceivers also realize that a collision has occurred.

With 10Base5, a transceiver cable connects the transceiver to an interface board in the computer. The transceiver cable may be up to 50 meters long and contains five individually shielded twisted pairs. Two of the pairs are for data in and data out, respectively. Two more are for control signals in and out. The fifth pair, which is not always used, allows the computer to power the transceiver electronics. Some transceivers allow up to eight nearby computers to be attached to them, to reduce the number of transceivers needed.

The transceiver cable terminates on an interface board inside the computer.
The interface board contains a controller chip that transmits frames to, and receives frames from, the transceiver. The controller is responsible for assembling the data into the proper frame format, as well as computing checksums on outgoing frames and verifying them on incoming frames. Some controller chips also manage a pool of buffers for incoming frames, a queue of buffers to be transmitted, direct memory transfers with the host computers, and other aspects of network management.

With 10Base2, the connection to the cable is just a passive BNC T-junction connector. The transceiver electronics are on the controller board, and each station always has its own transceiver.

With 10Base-T, there is no cable at all, just the hub (a box full of electronics). Adding or removing a station is simpler in this configuration, and cable breaks can be detected easily. The disadvantage of 10Base-T is that the maximum cable run from the hub is only 100 meters, maybe 200 meters if very high-quality category 5 twisted pairs are used. Nevertheless, 10Base-T quickly became dominant due to its use of existing wiring and the ease of maintenance that it offers. A faster version of 10Base-T (100Base-T) will be discussed later in this chapter.

A fourth cabling option for Ethernet is **10Base-F**, which uses fiber optics. This alternative is expensive due to the cost of the connectors and terminators, but it has excellent noise immunity and is the method of choice when running between buildings or widely separated hubs. Runs of up to km are allowed. It also offers good security since wiretapping fiber is much more difficult than wire-tapping copper wire.

Figure 4-15 shows different ways of wiring up a building. In Fig. 4-15(a), a single cable is snaked from room to room, with each station tapping onto it at the nearest point. In Fig. 4-15(b), a vertical spine runs from the basement to the roof,
with horizontal cables on each floor connected to it by special amplifiers (repeaters). In some buildings the horizontal cables are thin, and the backbone is thick. The most general topology is the tree, as in Fig. 4-15(c), because a network with two paths between some pairs of stations would suffer from interference between the two signals.

![Diagram of cable topologies](image)

**Figure 4-15.** Cable topologies. (a) Linear. (b) Spine. (c) Tree. (d) Segmented.

Each version of Ethernet has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by **repeaters**, as shown in Fig. 4-15(d). A repeater is a physical layer device. It receives, amplifies, and retransmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different than a single cable (except for some delay introduced by the repeaters). A system may contain multiple cable segments and multiple repeaters, but no two transceivers may be more than 2.5 km apart and no path between any two transceivers may traverse more than four repeaters.

### 4.3.2 Manchester Encoding

None of the versions of Ethernet use straight binary encoding with 0 volts for a 0 bit and 5 volts for a 1 bit because it leads to ambiguities. If one station sends the bit string 0001000, others might falsely interpret it as 10000000 or 01000000 because they cannot tell the difference between an idle sender (0 volts) and a 0 bit (0 volts).

What is needed is a way for receivers to unambiguously determine the start, end, or middle of each bit without reference to an external clock. Two such approaches are called **Manchester encoding** and **differential Manchester encoding**. With Manchester encoding, each bit period is divided into two equal intervals. A binary 1 bit is sent by having the voltage set high during the first interval and low in the second one. A binary 0 is just the reverse: first low and then high. This scheme ensures that every bit period has a transition in the middle, making it easy for the receiver to synchronize with the sender. A
disadvantage of Manchester encoding is that it requires twice as much bandwidth as straight binary encoding, because the pulses are half the width. For example, to send data at 10 Mbps, the signal has to change 20 million times/sec. Manchester encoding is shown in Fig. 4-16(b).

Differential Manchester encoding, shown in Fig. 4-16(c), is a variation of basic Manchester encoding. In it, a 1 bit is indicated by the absence of a transition at the start of the interval. A 0 bit is indicated by the presence of a transition at the start of the interval. In both cases, there is a transition in the middle as well. The differential scheme requires more complex equipment but offers better noise immunity. All Ethernet systems use Manchester encoding due to its simplicity. The high signal is +0.85 volts and the low signal is −0.85 volts, giving a DC value of 0 volts. Ethernet does not use differential Manchester encoding, but other LANs (e.g., the 802.5 token ring) do use it.

4.3.3 The Ethernet MAC Sublayer Protocol

The original DIX (DEC, Intel, Xerox) frame structure is shown in Fig. 4-17(a). Each frame starts with a Preamble of 8 bytes, each containing the bit pattern 10101010. The Manchester encoding of this pattern produces a 10-MHz square wave for 6.4 µsec to allow the receiver’s clock to synchronize with the sender’s. They are required to stay synchronized for the rest of the frame using the Manchester encoding to keep track of the bit boundaries.

The frame contains two addresses, one for the destination and one for the source. The standard allows 2-byte and 6-byte addresses, but the parameters defined for the 10-Mbps baseband standard use only the 6-byte addresses. The high-order bit of the destination address is a 0 for ordinary addresses and 1 for group addresses. Group addresses allow multiple stations to listen to a single address. When a frame is sent to a group address, all the stations in the group
receive it. Sending to a group of stations is called **multicast**. The address consisting of all 1 bits is reserved for **broadcast**. A frame containing all 1s in the destination field is delivered to all stations on the network. The difference between multicast and broadcast is important enough to warrant repeating. A multicast frame is sent to a selected group of stations on the Ethernet; a broadcast frame is sent to all stations on the Ethernet. Multicast is more selective, but involves group management. Broadcasting is coarser but does not require any group management.

Another interesting feature of the addressing is the use of bit 46 (adjacent to the high-order bit) to distinguish local from global addresses. Local addresses are assigned by each network administrator and have no significance outside the local network. Global addresses, in contrast, are assigned by IEEE to ensure that no two stations anywhere in the world have the same global address. With $48 - 2 = 46$ bits available, there are about $7 \times 10^{13}$ global addresses. The idea is that any station can uniquely address any other station by just giving the right 48-bit number. It is up to the network layer to figure out how to locate the destination.

Next comes the **Type** field, which tells the receiver what to do with the frame. There may be multiple network-layer protocols in use at the same time on the same machine, so when an Ethernet frame arrives, the kernel has to know which one to hand the frame to. The **Type** field specifies which process to give the frame to.

Next comes the data itself, up to 1500 bytes. This limit was chosen somewhat arbitrarily at the time the DIX standard was cast into stone, mostly based on the fact that a transceiver needs enough RAM to hold an entire frame, and RAM was expensive in 1978. A larger upper limit would have meant more RAM, hence a more expensive transceiver.

In addition to there being a maximum frame length, there is also a minimum frame length. While a data field of 0 bytes is sometimes useful, it causes a problem. When a transceiver detects a collision, it truncates the current frame, which means that stray bits and pieces of frames appear on the cable all the time. To make it easier to distinguish valid frames from garbage, Ethernet requires that

---

**Figure 4-17.** Frame formats. (a) DIX Ethernet. (b) IEEE 802.3.
valid frames must be at least 64 bytes long, from destination address to checksum, including both. If the data portion of a frame is less than 46 bytes, the Pad field is used to fill out the frame to the minimum size.

Another (and more important) reason for having a minimum length frame is to prevent a station from completing the transmission of a short frame before the first bit has even reached the far end of the cable, where it may collide with another frame. This problem is illustrated in Fig. 4-18. At time 0, station A, at one end of the network, sends off a frame. Let us call the propagation time for this frame to reach the other end $\tau$. Just before the frame gets to the other end (i.e., at time $\tau - \varepsilon$) the most distant station, B, starts transmitting. When B detects that it is receiving more power than it is putting out, it knows that a collision has occurred, so it aborts its transmission and generates a 48-bit noise burst to warn all other stations. In other words, it jams the ether to make sure the sender does not miss the collision. At about time $2\tau$, the sender sees the noise burst and aborts its transmission, too. It then waits a random time before trying again.

If a station tries to transmit a very short frame, it is conceivable that a collision occurs, but the transmission completes before the noise burst gets back at $2\tau$. The sender will then incorrectly conclude that the frame was successfully sent. To prevent this situation from occurring, all frames must take more than $2\tau$ to send so that the transmission is still taking place when the noise burst gets back to the sender. For a 10-Mbps LAN with a maximum length of 2500 meters and four repeaters (from the 802.3 specification), the round trip time (including time to propagate through the four repeaters) has been determined to be nearly 50 $\mu$sec in the worst case, including the time to pass through four repeaters, which is most certainly not zero. Therefore the minimum frame must take at least this long to transmit. At 10 Mbps, a bit takes 100 nsec, so 500 bits is the smallest frame that is guaranteed to work. To add some margin of safety this number was rounded up to 512 bits or 64 bytes. Frames with fewer bytes are padded out to 64 bytes using
the Pad field.

As the network speed goes up, the minimum frame length must go up or the maximum cable length must come down, proportionally. For a 2500-meter LAN operating at 1 Gbps, the minimum frame size would have to be 6400 bytes. Alternatively, the minimum frame size could be 640 bytes and the maximum distance between any two stations 250 meters. These restrictions are becoming increasingly painful as we move toward multigigabit networks.

The final Ethernet field is the Checksum. It is effectively a 32-bit hash code of the data. If some data bits are erroneously received (due to noise on the cable), the checksum will almost certainly be wrong, and the error will be detected. The checksum algorithm is a cyclic redundancy check (CRC) of the kind discussed in Chap. 3. It just does error detection, not forward error correction.

When IEEE standardized Ethernet, the committee made two changes to the DIX format, as shown in Fig. 4-17(b). The first one was to reduce the preamble to 7 bytes and use the last byte for a Start of Frame delimiter, for compatibility with 802.4 and 802.5. The second one was to change the Type field into a Length field. Of course, now there was no way for the receiver to figure out what to do with an incoming frame, but that problem was handled by adding a small header to the data portion itself to provide this information. We will discuss the format of the data portion when we come to logical link control later in this chapter.

Unfortunately, by the time 802.3 was published, so much hardware and software for DIX Ethernet was already in use, few manufacturers and users were enthusiastic about converting the Type field into a Length field. In 1997 IEEE threw in the towel and said that both ways were fine with it. Fortunately, all the Type fields in use prior to 1997 were greater than 1500. Consequently, any number there less than or equal to 1500 can be interpreted as Length, and any number greater than 1500 can be interpreted as Type. Now IEEE can maintain that everyone is using its standard and everybody else can keep on doing what they were already doing without feeling guilty about it.

4.3.4 The Binary Exponential Backoff Algorithm

Let us now see how randomization is done when a collision occurs. The model is that of Fig. 4-5. After a collision, time is divided up into discrete slots whose length is equal to the worst case round-trip propagation time on the ether \(2\tau\). To accommodate the longest path allowed by Ethernet (2.5 km and four repeaters), the slot time has been set to 512 bit times, or 51.2 µsec as mentioned above.

After the first collision, each station waits either 0 or 1 slot times before trying again. If two stations collide and each one picks the same random number, they will collide again. After the second collision, each one picks either 0, 1, 2, or 3 at random and waits that number of slot times. If a third collision occurs (the probability of this happening is 0.25), then the next time the number of slots to
wait is chosen at random from the interval 0 to \(2^3 - 1\).

In general, after \(i\) collisions, a random number between 0 and \(2^i - 1\) is chosen, and that number of slots is skipped. However, after ten collisions have been reached, the randomization interval is frozen at a maximum of 1023 slots. After 16 collisions, the controller throws in the towel and reports failure back to the computer. Further recovery is up to higher layers.

This algorithm, called **binary exponential backoff**, was chosen to dynamically adapt to the number of stations trying to send. If the randomization interval for all collisions was 1023, the chance of two stations colliding for a second time would be negligible, but the average wait after a collision would be hundreds of slot times, introducing significant delay. On the other hand, if each station always delayed for either zero or one slots, then if 100 stations ever tried to send at once, they would collide over and over until 99 of them picked 0 and the remaining station picked 1, or vice versa. This might take years. By having the randomization interval grow exponentially as more and more consecutive collisions occur, the algorithm ensures a low delay when only a few stations collide but also ensures that the collision is resolved in a reasonable interval when many stations collide. By truncating the backoff at 1023, the bound is kept from growing too large.

As described so far, CSMA/CD provides no acknowledgements. Since the mere absence of collisions does not guarantee that bits were not garbled by noise spikes on the cable, for reliable communication the destination must verify the checksum, and if correct, send back an acknowledgement frame to the source. Normally, this acknowledgement would be just another frame as far as the protocol is concerned and would have to fight for channel time just like a data frame. However, a simple modification to the contention algorithm would allow speedy confirmation of frame receipt (Tokoro and Tamaru, 1977). All that would be needed is to reserve the first contention slot following successful transmission for the destination station. Unfortunately, the standard does not provide for this possibility.

### 4.3.5 Ethernet Performance

Now let us briefly examine the performance of Ethernet under conditions of heavy and constant load, that is, \(k\) stations always ready to transmit. A rigorous analysis of the binary exponential backoff algorithm is complicated. Instead we will follow Metcalfe and Boggs (1976) and assume a constant retransmission probability in each slot. If each station transmits during a contention slot with probability \(p\), the probability \(A\) that some station acquires the channel in that slot is

\[ A = kp(1 - p)^{k-1} \] (4-5)

\(A\) is maximized when \(p = 1/k\), with \(A \to 1/e\) as \(k \to \infty\). The probability that the contention interval has exactly \(j\) slots in it is \(A(1 - A)^{j-1}\), so the mean number of slots per contention is given by
\[ \sum_{j=0}^{\infty} jA (1 - A)^{j-1} = \frac{1}{A} \]

Since each slot has a duration \(2\tau\), the mean contention interval, \(w\), is \(2\tau/A\). Assuming optimal \(p\), the mean number of contention slots is never more than \(e\), so \(w\) is at most \(2\tau e = 5.4\tau\).

If the mean frame takes \(P\) sec to transmit, when many stations have frames to send,

\[ \text{Channel efficiency} = \frac{P}{P + 2\tau A} \quad (4-6) \]

Here we see where the maximum cable distance between any two stations enters into the performance figures, giving rise to topologies other than that of Fig. 4-15(a). The longer the cable, the longer the contention interval. By allowing no more than 2.5 km of cable and four repeaters between any two transceivers, the round-trip time can be bounded to 51.2 µsec, which at 10 Mbps corresponds to 512 bits or 64 bytes, the minimum frame size.

It is instructive to formulate Eq. (4-6) in terms of the frame length, \(F\), the network bandwidth, \(B\), the cable length, \(L\), and the speed of signal propagation, \(c\), for the optimal case of \(e\) contention slots per frame. With \(P = F/B\), Eq. (4-6) becomes

\[ \text{Channel efficiency} = \frac{1}{1 + 2BLe/cF} \quad (4-7) \]

When the second term in the denominator is large, network efficiency will be low. More specifically, increasing network bandwidth or distance (the \(BL\) product) reduces efficiency for a given frame size. Unfortunately, much research on network hardware is aimed precisely at increasing this product. People want high bandwidth over long distances (fiber optic MANs, for example), which suggests that Ethernet implemented in this manner may not be the best system for these applications. We will see other ways of implementing Ethernet when we come to switched Ethernet later in this chapter.

In Fig. 4-19, the channel efficiency is plotted versus number of ready stations for \(2\tau = 51.2\) µsec and a data rate of 10 Mbps using Eq. (4-7). With a 64-byte slot time, it is not surprising that 64-byte frames are not efficient. On the other hand, with 1024-byte frames and an asymptotic value of \(e\) 64-byte slots per contention interval, the contention period is 174 bytes long and the efficiency is 0.85.

To determine the mean number of stations ready to transmit under conditions of high load, we can use the following (crude) observation. Each frame ties up the channel for one contention period and one frame transmission time, for a total of \(P + w\) sec. The number of frames per second is therefore \(1/(P + w)\). If each station generates frames at a mean rate of \(\lambda\) frames/sec, when the system is in state \(k\) the total input rate of all unblocked stations combined is \(k\lambda\) frames/sec.
Since in equilibrium the input and output rates must be identical, we can equate these two expressions and solve for $k$. (Notice that $w$ is a function of $k$.) A more sophisticated analysis is given in (Bertsekas and Gallager, 1992).

It is probably worth mentioning that there has been a large amount of theoretical performance analysis of Ethernet (and other networks). Virtually all of this work has assumed that traffic is Poisson. As researchers have begun looking at real data, it now appears that network traffic is rarely Poisson, but self-similar (Paxson and Floyd, 1994; and Willinger et al., 1995). What this means is that averaging over long periods of time does not smooth out the traffic. The average number of frames in each minute of an hour has as much variance as the average number of frames in each second of a minute. The consequence of this discovery is that most models of network traffic do not apply to the real world and should be taken with a grain (or better yet, a metric ton) of salt.

### 4.3.6 Switched Ethernet

As more and more stations are added to an Ethernet, the traffic will go up. Eventually, the LAN will saturate. One way out is to go to a higher speed, say from 10 Mbps to 100 Mbps. But with the growth of multimedia, even a 100 Mbps or 1 Gbps Ethernet can become saturated.

Fortunately, there is an additional way to deal with increased load: switched Ethernet, as shown in Fig. 4-20. The heart of this system is a switch containing a high-speed backplane and room for typically 4 to 32 plug-in line cards, each containing one to eight connectors. Most often, each connector has a 10Base-T
When a station wants to transmit an Ethernet frame, it outputs a standard frame to the switch. The plug-in card getting the frame may check to see if it is destined for one of the other stations connected to the same card. If so, the frame is copied there. If not, the frame is sent over the high-speed backplane to the destination station’s card. The backplane typically runs at many Gbps using a proprietary protocol.

What happens if two machines attached to the same plug-in card transmit frames at the same time? It depends on how the card has been constructed. One possibility is for all the ports on the card to be wired together to form a local on-card LAN. Collisions on this on-card LAN will be detected and handled the same as any other collisions on a CSMA/CD network—with retransmissions using the binary exponential backoff algorithm. With this kind of plug-in card, only one transmission per card is possible at any instant, but all the cards can be transmitting in parallel. With this design, each card forms its own collision domain, independent of the others.

With the other kind of plug-in card, each input port is buffered, so incoming frames are stored in the card’s on-board RAM as they arrive. This design allows all input ports to receive (and transmit) frames at the same time, for parallel, full-duplex operation, something not possible with CSMA/CD on a single channel. Once a frame has been completely received, the card can then check to see if the frame is destined for another port on the same card, or for a distant port. In the former case it can be transmitted directly to the destination. In the latter case, it must be transmitted over the backplane to the proper card. With this design, each port is a separate collision domain, so collisions do not occur. The total system throughput can often be increased by an order of magnitude over 10Base5, which has a single collision domain for the entire system.
Since the switch just expects standard Ethernet frames on each input port, it is possible to use some of the ports as concentrators. In Fig. 4-20, the port in the upper right-hand corner is connected not to a single station, but to a 12-port hub. As frames arrive at the hub, they contend for the ether in the usual way, including collisions and binary backoff. Successful frames make it to the switch, and are treated there like any other incoming frames: they are switched to the correct output line over the high-speed backplane. Hubs are cheaper than switches, but due to falling switch prices, are rapidly becoming obsolete. Nevertheless, legacy hubs still exist.

4.3.7 Fast Ethernet

At first, 10 Mbps seemed like heaven, just as 1200-bps modems seemed like heaven to the early users of 300-bps acoustic modems. But the novelty wore off quickly. As a kind of corollary to Parkinson's Law ("Work expands to fill the time available for its completion"), it seemed that data expanded to fill the bandwidth available for their transmission. To pump up the speed, two new ring-based optical LANs were proposed by various industry groups. One was called FDDI (Fiber Distributed Data Interface) and the other was called Fibre Channel†. To make a long story short, while both were used as backbone networks, neither one made the breakthrough to the desktop. In both cases, the station management was too complicated, which led to complex chips and high prices. The lesson that should have been learned here was KISS (Keep It Simple, Stupid).

In any event, the failure of the optical LANs to catch fire left a gap for garden-variety Ethernet at speeds above 10 Mbps. Many installations needed more bandwidth and thus had numerous 10-Mbps LANs connected by a maze of repeaters, bridges, routers, and gateways, although to the network managers it sometimes felt that they were being held together by bubble gum and chicken wire.

It was in this environment that IEEE reconvened the 802.3 committee in 1992 with instructions to come up with a faster LAN. One proposal was to keep 802.3 exactly as it was, but just make it go faster. Another proposal was to redo it totally, to give it lots of new features, such as real-time traffic and digitized voice, but just keep the old name (for marketing reasons). After some wrangling, the committee decided to keep 802.3 the way it was, but just make it go faster. The people behind the losing proposal did what any computer-industry people would have done under these circumstances—they stomped off and formed their own committee and standardized their LAN anyway (eventually as 802.12). It flopped miserably.

The three primary reasons that the 802.3 committee decided to go with a souped-up Ethernet were:

† It is called “fibre channel” and not “fiber channel” because the document editor was British.
1. The need to be backward compatible with existing Ethernet LANs.
2. The fear that a new protocol might have unforeseen problems.
3. The desire to get the job done before the technology changed.

The work was done quickly (by standards committees’ norms), and the result, 802.3u, was officially approved by IEEE in June 1995. Technically, 802.3u is not a new standard, but an addendum to the existing 802.3 standard (to emphasize its backward compatibility). Since everyone calls it fast Ethernet, rather than 802.3u, we will do that, too.

The basic idea behind fast Ethernet was simple: keep all the old frame formats, interfaces, and procedural rules, but just reduce the bit time from 100 nsec to 10 nsec. Technically, it would have been possible to copy 10Base-5 or 10Base-2 and still detect collisions on time by just reducing the maximum cable length by a factor of ten. However, the advantages of 10Base-T wiring were so overwhelming that fast Ethernet is based entirely on this design. Thus all fast Ethernet systems use hubs and switches; multidrop cables with vampire taps or BNC connectors are not permitted.

Nevertheless, some choices still had to be made, the most important of which was which wire types to support. One contender was category 3 twisted pair. The argument for it was that practically every office in the Western world has at least four category 3 (or better) twisted pairs running from it to a telephone wiring closet within 100 meters. Sometimes two such cables exist. Thus using category 3 twisted pair would make it possible to wire up desktop computers using fast Ethernet without having to rewire the building, an enormous advantage for many organizations.

The main disadvantage of category 3 twisted pair is its inability to carry 200 megabaud signals (100 Mbps with Manchester encoding) 100 meters, the maximum computer-to-hub distance specified for 10Base-T (see Fig. 4-13). In contrast, category 5 twisted pair wiring can handle 100 meters easily, and fiber can go much further. The compromise chosen was to allow all three possibilities, as shown in Fig. 4-21, but to pep up the category 3 solution to give it the additional carrying capacity needed.

<table>
<thead>
<tr>
<th>Name</th>
<th>Cable</th>
<th>Max. segment</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>100Base-T4</td>
<td>Twisted pair</td>
<td>100 m</td>
<td>Uses category 3 UTP</td>
</tr>
<tr>
<td>100Base-TX</td>
<td>Twisted pair</td>
<td>100 m</td>
<td>Full duplex at 100 Mbps</td>
</tr>
<tr>
<td>100Base-FX</td>
<td>Fiber optics</td>
<td>2000 m</td>
<td>Full duplex at 100 Mbps; long runs</td>
</tr>
</tbody>
</table>

Figure 4-21. The original fast Ethernet cabling.

The category 3 UTP scheme, called 100Base-T4, uses a signaling speed of 25 MHz, only 25 percent faster than standard Ethernet’s 20 MHz (remember that
Manchester encoding, as shown in Fig. 4-16, requires two clock periods for each of the 10 million bits each second. To achieve the necessary bandwidth, 100Base-T4 requires four twisted pairs. Since standard telephone wiring for decades has had four twisted pairs per cable, most offices are able to handle this. Of course, it means giving up your office telephone, but that is surely a small price to pay for faster e-mail.

Of the four twisted pairs, one is always to the hub, one is always from the hub, and the other two are switchable to the current transmission direction. To get the necessary bandwidth, Manchester encoding is not used, but with modern clocks and such short distances, it is no longer needed. In addition, ternary signals are sent, so that during a single clock period the wire can contain a 0, a 1, or a 2. With three twisted pairs going in the forward direction and ternary signaling, any one of 27 possible symbols can be transmitted, making it possible to send 4 bits with some redundancy. Transmitting 4 bits in each of the 25 million clock cycles per second gives the necessary 100 Mbps. In addition, there is always a 33.3 Mbps reverse channel using the remaining twisted pair. This scheme, known as 8B/6T, (8 bits map to 6 trits) is not likely to win any prizes for elegance, but it works with the existing wiring plant.

For category 5 wiring, the design, 100Base-TX, is simpler because the wires can handle clock rates of 125 MHz. Only two twisted pairs per station are used, one to the hub and one from it. Rather than just use straight binary coding, a scheme called 4B/5B is used at 125 MHz. It is taken from FDDI and compatible with it. Every group of five clock periods, each containing one of two signal values, yields 32 combinations. Sixteen of these combinations are used to transmit the four bit groups 0000, 0001, 0010, ..., 1111. Some of the remaining 16 are used for control purposes such as marking frames boundaries. The combinations used have been carefully chosen to provide enough transitions to maintain clock synchronization. The 100Base-TX system is full-duplex; stations can transmit at 100 Mbps and receive at 100 Mbps at the same time. Often 100Base-TX and 100Base-T4 are collectively referred to as 100Base-T.

The last option, 100Base-FX, uses two strands of multimode fiber, one for each direction, so it, too, is full duplex with 100 Mbps in each direction. In addition, the distance between a station and the hub can be up to 2 km.

In response to popular demand, in 1997 the 802 committee added a new cabling type: 100Base-T2 allowing fast Ethernet to run over two pairs of existing category 3 wiring. However, a sophisticated digital signal processor is needed to handle the encoding scheme required, making this option fairly expensive. So far, it is rarely used due to its complexity, cost, and the fact that many office buildings have already been rewired with category 5 UTP.

Two kinds of interconnection devices are possible with 100Base-T: hubs and switches, as shown in Fig 4-20. In a hub, all the incoming lines (or at least all the lines arriving at one plug-in card) are logically connected, forming a single collision domain. All the standard rules, including the binary exponential backoff
In a switch, each incoming frame is buffered on a plug-in line card and passed over a high-speed backplane from the source card to the destination card, if need be. The backplane has not been standardized, nor does it need to be, since it is entirely hidden deep inside the switch. If past experience is any guide, switch vendors will compete vigorously to produce ever faster backplanes in order to improve system throughput. Because 100Base-FX cables are too long for the normal Ethernet collision algorithm, they must be connected to switches so each one is a collision domain unto itself. Hubs are not permitted with 100Base-FX.

As a final note, virtually all switches can handle a mix of 10-Mbps and 100-Mbps stations, to make upgrading easier. As a site acquires more and more 100-Mbps workstations, all it has to do is buy the necessary number of new line cards and insert them into the switch. In fact, the standard itself provides a way for two stations to automatically negotiate the optimum speed (10 or 100 Mbps) and duplexity (half or full). Most fast Ethernet products use this feature to autoconfigure themselves.

### 4.3.8 Gigabit Ethernet

The ink was barely dry on the fast Ethernet standard when the 802 committee began working on a yet faster Ethernet (1995). It was quickly dubbed **gigabit Ethernet** and was ratified by IEEE in 1998 under the name 802.3z. This identifier suggests that gigabit Ethernet is the end of the line unless somebody invents a new letter after z quickly. Below we will discuss some of the key features of gigabit Ethernet. More information can be found in (Seifert, 1998).

The 802.3z committee’s goals were essentially the same as the 802.3u committee’s goals: make Ethernet go 10 times faster yet remain backward compatible with all existing Ethernet standards. In particular, gigabit Ethernet had to offer unacknowledged datagram service with both unicast and multicast, use the same 48-bit addressing scheme already in use, and maintain the same frame format, including the minimum and maximum frame sizes. The final standard met all these goals.

All configurations of gigabit Ethernet are technically point-to-point rather than multidrop as in the original 10 Mbps standard, now honored as **classic Ethernet**. In the simplest gigabit Ethernet configuration, illustrated in Fig. 4-22(a), two computers are directly connected to each other. The more common case, however, is having a switch or a hub connected to multiple computers and possibly additional switches or hubs, as shown in Fig. 4-22(b). In both configurations each Ethernet cable has exactly two devices on it.

Gigabit Ethernet supports two modes of operation: full-duplex mode and half-duplex mode. The “normal” mode is full-duplex mode, which allows traffic in both directions at the same time. This mode is used when there is a central
switch connected to computers (or other switches) on the periphery. In this configuration, all lines are buffered so each computer and switch is free to send frames whenever it wants to. The sender does not have to sense the channel to see if anybody else is using it because contention is impossible. On the line between a computer and a switch, the computer is the only possible sender on that line to the switch and the transmission succeeds even if the switch is currently sending a frame to the computer (because the line is full duplex). Since there is no contention possible, the CSMA/CD protocol is not used, so the maximum length of the cable is determined by signal strength issues rather than how long it takes for a noise burst to propagate back to the sender in the worst case. Switches are free to mix and match speeds. Autoconfiguration is supported as in fast Ethernet.

The other mode of operation, half-duplex, is used when the computers are connected to a hub rather than a switch. A hub does not buffer incoming frames. Instead, it connects all the lines electrically internally, simulating the multidrop cable used in classic Ethernet. In this mode, collisions are possible, so the standard CSMA/CD protocol is required. Because a minimum (i.e., 64-byte) frame can now be transmitted 100 times faster than in classic Ethernet, the maximum distance is 100 times less or 25 meters to maintain the essential property that the sender is still transmitting when the noise burst gets back to it, even in the worst case. With a 2500-meter long cable, the sender of a 64-byte frame at 1 Gbps would be long done before the frame got even a tenth of the way to the other end, let alone to the end and back.

The 802.3z committee considered a radius of 25 meters to be unacceptable and added two features to the standard to increase the radius. The first feature, called carrier extension, essentially tells the hardware to add its own padding after the normal frame to extend the frame to 512 bytes. Since this padding is added by the sending hardware and removed by the receiving hardware, the software is unaware of it, meaning that no changes are needed to existing software. Of course, using 512 bytes worth of bandwidth to transmit 46 bytes of
user data (the payload of a 64-byte frame) has a line efficiency of 9%.

The second feature, called **frame bursting**, allows a sender to transmit a concatenated sequence of multiple frames in a single transmission. If the total burst is less than 512 bytes, the hardware pads it again. If there are enough frames waiting for transmission, this scheme is highly efficient and preferred over carrier extension. These new features extend the radius of the network to 200 meters, which is probably enough for most offices.

In all fairness, it is hard to imagine an organization going to the trouble of buying and installing gigabit Ethernet cards to get high performance and then connecting the computers with a hub to simulate classic Ethernet with all its collisions. While hubs are somewhat cheaper than switches, gigabit Ethernet interface cards are still relatively expensive. To then economize by buying a cheap hub and reducing the performance of the new system considerably seems foolish. Still, backward compatibility is sacred in the computer industry, so the 802.3z committee was required to put it in.

Gigabit Ethernet supports both copper and fiber cabling, as listed in Fig. 4-23. Signaling at or near 1 Gbps over fiber means that the light source has to be turned on and off in under 1 nsec. LEDs simply cannot operate this fast, so lasers are required. Two wavelengths are permitted: 0.85 microns (Short) and 1.3 microns (Long). Lasers at 0.85 microns are cheaper but do not work on single-mode fiber.

<table>
<thead>
<tr>
<th>Name</th>
<th>Cable</th>
<th>Max. segment</th>
<th>Advantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000Base-SX</td>
<td>Fiber optics</td>
<td>550 m</td>
<td>Multimode fiber (50, 62.5 microns)</td>
</tr>
<tr>
<td>1000Base-LX</td>
<td>Fiber optics</td>
<td>5000 m</td>
<td>Single (10 µ) or multimode (50, 62.5 µ)</td>
</tr>
<tr>
<td>1000Base-CX</td>
<td>2 Pairs of STP</td>
<td>25 m</td>
<td>Shielded twisted pair</td>
</tr>
<tr>
<td>1000Base-T</td>
<td>4 Pairs of UTP</td>
<td>100 m</td>
<td>Standard category 5 UTP</td>
</tr>
</tbody>
</table>

**Figure 4-23.** Gigabit Ethernet cabling.

Three fiber diameters are permitted: 10, 50, and 62.5 microns. The former is for single mode and the latter two are for multimode. Not all six combinations are allowed, however, and the maximum distance depends on the combination used. The numbers given in Fig. 4-23 are for the best case. In particular, 5000 meters is only achievable with 1.3 micron lasers operating over 10 micron fiber in single mode, but this is the best choice for campus backbones and is expected to be popular, despite it being the most expensive choice.

The 1000Base-CX option uses short shielded copper cables. Its problem is that it is competing with high performance fiber from above and cheap UTP from below. It is unlikely to be used much, if at all.

The last option is bundles of four category 5 UTP wires working together. Because so much of this wiring is already installed, it is likely to be the poor man’s gigabit Ethernet.

Gigabit Ethernet uses new encoding rules on the fibers. Manchester encoding...
at 1 Gbps would require a 2 Gbaud signal, which was considered too difficult and also too wasteful of bandwidth. Instead a new scheme, called 8B/10B, was chosen, based on fibre channel. Each 8-bit byte is encoded on the fiber as 10 bits, hence the name 8B/10B. Since there are 1024 possible output codewords for each input byte, some leeway was available in choosing which codewords to allow. The following two rules were used in making the choices:

1. No codeword may have more than four identical bits in a row.
2. No codeword may have more than six 0s or six 1s.

These choices were made to keep enough transitions in the stream to make sure the receiver stays in sync with the sender and also to keep the number of 0s and 1s on the fiber as close to equal as possible. In addition, many input bytes have two possible codewords assigned to them. When the encoder has a choice of codewords, it always chooses the codeword that moves in the direction of equalizing the number of 0s and 1s transmitted so far. This emphasis of balancing 0s and 1s is needed to keep the DC component of the signal as low as possible to allow it to pass through transformers unmodified. While computer scientists are not fond of having the properties of transformers dictate their coding schemes, life is like that sometimes.

Gigabit Ethernets using 1000Base-T use a different encoding scheme since clocking data onto copper wire in 1 nsec is too difficult. This solution uses four category 5 twisted pairs to allow four symbols to be transmitted in parallel. Each symbol is encoded using one of five voltage levels. This scheme allows a single symbol to encode 00, 01, 10, 11, or a special value for control purposes. Thus there are 2 data bits per twisted pair or 8 data bits per clock cycle. The clock runs at 125 MHz, allowing 1 Gbps operation. The reason for allowing five voltage levels instead of four is to have combinations left over for framing and control purposes.

A speed of 1 Gbps is quite fast. For example, if a receiver is busy with some other task for even 1 msec and doesn’t empty the input buffer on some line, up to 1953 frames may have accumulated there in that 1 ms gap. Also, when a computer on a gigabit Ethernet is shipping data down the line to a computer on a classic Ethernet, buffer overruns are very likely. As a consequence of these two observations, gigabit Ethernet supports flow control (as does fast Ethernet, although the two are different).

The flow control consists of one end sending a special control frame to the other end telling it to pause for some period of time. Control frames are normal Ethernet frames containing a type of 0x8808. The first two bytes of the data field give the command; succeeding bytes provide the parameters, if any. For flow control, PAUSE frames are used, with the parameter telling how long to pause, in units of the minimum frame time. For gigabit Ethernet, the time unit is 512 nsec, allowing for pauses as long as 33.6 msec.
As soon as gigabit Ethernet was standardized, the 802 committee got bored and wanted to get back to work. IEEE told them to start on 10-gigabit Ethernet. After searching hard for a letter to follow z, they abandoned that approach and went over to two-letter suffixes. They got to work and that standard was completed in 2001 and approved by IEEE in 2002 (???) as 802.ae. Can 100-gigabit Ethernet be far behind?

4.3.9 IEEE 802.2: Logical Link Control

It is now perhaps time to step back and compare what we have learned in this chapter with what we studied in the previous one. In Chap. 3, we saw how two machines could communicate reliably over an unreliable line by using various data link protocols. These protocols provided error control (using acknowledgments) and flow control (using a sliding window).

In contrast, in this chapter, we have not said a word about reliable communication. All Ethernet and the other 802 protocols offer is a best-effort datagram service. Sometimes, this service is adequate. For example, for transporting IP packets, no guarantees are required or even expected. An IP packet can just be inserted into an 802 payload field and sent on its way. If it gets lost, so be it.

Nevertheless, there are also systems in which an error-controlled, flow-controlled data link protocol is desired. IEEE has defined one that can run on top of Ethernet and the other 802 protocols. In addition, this protocol, called LLC (Logical Link Control), hides the differences between the various kinds of 802 networks by providing a single format and interface to the network layer. This format, interface, and protocol are all closely based on the HDLC protocol we studied in Chap. 3. LLC forms the upper half of the data link layer, with the MAC sublayer below it, as shown in Fig. 4-24.

![Figure 4-24. (a) Position of LLC. (b) Protocol formats.](image)

Typical usage of LLC is as follows. The network layer on the sending machine passes a packet to LLC using the LLC access primitives. The LLC sublayer then adds an LLC header, containing sequence and acknowledgement
numbers. The resulting structure is then inserted into the payload field of an 802
frame and transmitted. At the receiver, the reverse process takes place.

LLC provides three service options: unreliable datagram service, acknowledged datagram service, and reliable connection-oriented service. The LLC
header contains three fields, a destination access point, a source access point, and
a control field. The access points tell which process the frame came from and
where it is to be delivered to, replacing the DIX Type field. The control field con-
tains sequence and acknowledgement numbers, very much in the style of HDLC
(see Fig. 3-19 ??), but not identical to it. These fields are primarily used when a
reliable connection is needed at the data link level, in which case protocols similar
to the ones discussed in Chap. 3 would be used. For the Internet, best-effort
attempts to deliver IP packets is sufficient, so no acknowledgements at the LLC
level are required.

4.3.10 Retrospective on Ethernet

Ethernet has been around for over 20 years and has no serious competitors at
this time, so it is likely to be around for many years to come. Few CPU architec-
tures, operating systems, or programming languages have been king of the moun-
tain for two decades going on three. Clearly, Ethernet did something right. What?

Probably the main reason is that Ethernet is simple and flexible. In practice,
simple translates into reliable, cheap, and easy to maintain. Once the vampire
taps were replaced by BNC connectors, failures became extremely rare. People
tend to be hesitant to replace something that works perfectly all the time, espe-
cially when they know that an awful lot of things in the computer industry work
very poorly, so that many so-called “upgrades” are appreciably worse than what
they replaced.

Simple also translates into cheap. Thin Ethernet and twisted pair wiring is
relatively inexpensive. The interface cards are also low cost. Only when hubs
and switches were introduced were substantial investments required, but by the
time they were in the picture, Ethernet was already well established.

Ethernet is easy to maintain. There is no software to install (other than the
drivers) and no configuration tables to manage (and get wrong). Also adding new
hosts is as simple as just plugging them in.

Another point is that Ethernet interworks easily with TCP/IP, which has
become dominant. IP is a connectionless protocol, so it fits perfectly with Ether-
net, which is also connectionless. IP fits much less well with ATM, which is
connection-oriented. This mismatch definitely hurt ATM’s chances.

Lastly, Ethernet has been able to evolve in certain crucial ways. Speeds have
gone up by several orders of magnitude and hubs and switches have been intro-
duced, but these changes have not required changing the software. When a net-
work salesman shows up at a large installation and says: “I have this fantastic
new network for you. All you have to do is throw out all your hardware and rewrite all your software’’ he has a problem. FDDI, Fibre Channel, and ATM were all faster than Ethernet when introduced, but they were incompatible with Ethernet, far more complex, and harder to manage. Eventually, Ethernet caught up with them in terms of speed so they had no advantages left and quietly died off.
While Ethernet is widely used, it is about to get some competition. Wireless LANs are increasingly popular and more and more office building, airports, and other public places are being outfitted with them. Wireless LANs can operate in one of two configurations as we saw in Fig. 1-35: with a base station and without a base station. Consequently, the 802.11 LAN standard takes this into account and makes provision for both arrangements, as we will see shortly.

We gave some background information on 802.11 in Sec. 1.5.4. Now is the time to take a closer look at the technology. In the following sections we will look at the protocol stack, physical layer radio transmission techniques, MAC sublayer protocol, frame structure, and services. For more information about 802.11, see (Crow et al., 1997; Geier, 2002; Heegard et al., 2001; Kapp, 2002; O’Hara and Petrick, 1999; and Severance, 1999). To hear the truth from the mouth of the horse, consult the published 802.11 standard itself.

### 4.4.1 The 802.11 Protocol Stack

The protocols used by all the 802 variants, including Ethernet, have a certain commonality of structure. A partial view of the 802.11 protocol stack is given in Fig. 4-25. The physical layer corresponds to the OSI physical layer fairly well, but the data link layer in all the 802 protocols is split into two or more sublayers. In 802.11, the MAC (Medium Access Control) sublayer determines how the channel is allocated, that is, who gets to transmit next. Above it is the LLC (Logical Link Control) sublayer, whose job it is to hide the differences between the different 802 variants and make different 802 LANs look identical to the network layer. We studied the LLC when examining Ethernet earlier in this chapter and will not repeat that material here.

The 1997 802.11 standard specifies three transmission techniques allowed in the physical layer. The infrared method uses much the same technology as television remote controls do. The other two use short-range radio using techniques called FHSS and DSSS. Both of these use a part of the spectrum that does not require licensing (the 2.4 GHz ISM band). Radio-controlled garage door openers also use this piece of the spectrum, so your notebook computer may find itself in competition with your garage door. All of these techniques operate at 1 or 2 Mbps. In 1999, two new techniques were introduced to achieve higher bandwidth. These are called OFDM and HR-DSSS. They operate at up to 54 Mbps and 11 Mbps, respectively. Now we will examine each of them briefly. Technically, these belong to the physical layer and should have been examined in Chapter 2, but since they are so closely tied to LANs in general and the 802.11 MAC sublayer, we treat them here instead.
### 4.4.2 The 802.11 Physical Layer

Each of the five permitted transmission techniques makes it possible to send a MAC frame from one station to another. They differ, however, in the technology used and speeds achievable. A detailed discussion of these technologies is far beyond the scope of this book, but a few words on each one, along with some of the key words may provide interested readers with terms to search for on the Internet or elsewhere for more information.

The infrared option uses diffused (i.e., not line of sight) transmission at 0.85 or 0.95 microns. Two speeds are permitted: 1 Mbps and 2 Mbps. At 1 Mbps, an encoding scheme is used in which a group of 4 bits is encoded as a 16 bit code-word containing 15 0s and a single 1 using what is called **Gray code**. This code has the property that a small error in time synchronization leads to only a single bit error in the output. At 2 Mbps, the encoding takes 2 bits and produces a 4-bit codeword, also with only a single 1, that is one of 0001, 0010, 0100, or 1000. Infrared signals cannot penetrate walls, so cells in different rooms are well isolated from each other. Nevertheless, due to the low bandwidth (and the fact that sunlight swamps infrared signals) this is not a popular option.

**FHSS (Frequency Hopping Spread Spectrum)** is not popular either and for the same reason (low bandwidth). It uses 79 channels, each 1 MHz wide, starting at the low end of the 2.4 GHz ISM band. The order in which the frequencies are hopped to is generated by a pseudorandom number sequence. As long as all stations use the same seed to the pseudorandom number generator and stay synchronized in time, they will hop to the same frequencies simultaneously. The amount of time spent at each frequency, the **dwell time** is an adjustable
parameter, but must be less than 400 msec. FHSS’ randomization provides a fair way to allocate spectrum in the unregulated ISM band. It also provides good security since an intruder who does not know the hopping sequence or dwell time cannot eavesdrop on transmissions. Over longer distances, multipath fading can be an issue, and FHSS offers some resistance to it.

The third modulation method, DSSS (Direct Sequence Spread Spectrum), is also restricted to 1 or 2 Mbps. The scheme used has some similarities to the CDMA system we examined in Sec. 2.6.2 (???), but differs in other ways. Each bit is transmitted as 11 chips using what is called a Barker sequence. It uses phase shift modulation at 1 Mbaud, transmitting 1 bit per baud when operating at 1 Mbps and 2 bits per baud when operating at 2 Mbps.

The first of the high-speed LANs, 802.11a, uses OFDM (Orthogonal Frequency Division Multiplexing) to deliver up to 54 Mbps in the wider 5 GHz ISM band. As the term FDM suggests, different frequencies are used—52 of them, 48 for data and 4 for synchronization—not unlike ADSL. A complex encoding system is used, based on phase-shift modulation for speeds up to 18 Mbps and QAM above that. At 54 Mbps, 216 data bits are encoded into 288-bit symbols. Part of the motivation for OFDM is compatibility with the European HiperLAN/2 system. Also, the technique has a good spectrum efficiency in terms of bits/Hz and good immunity to multipath fading.

Finally, we come to HR-DSSS (High Rate Direct Sequence Spread Spectrum), another spread spectrum technique, which uses 11 million chips/sec to achieve 11 Mbps in the 2.4 GHz band. It is called 802.11b but is not a follow-up to 802.11a. Data rates that are supported are 1, 2, 5.5, and 11 Mbps. The two slow rates run at 1 Mbaud, with 1 and 2 bits per baud, respectively, using phase shift modulation (for compatibility with DSSS). The two faster rates run at 1.375 Mbaud, with 4 and 8 bits per baud, respectively, using Walsh/Hadamard codes. The data rate may be dynamically adapted during operation to achieve the optimum speed possible under current conditions of load and noise. Although 802.11b is slower than 802.11a, its range is about 7 times greater, which is more important in many situations.

An enhanced version of 802.11b, 802.11g, was approved by IEEE in Nov. 2001 after much politicking about whose patented technology it would use. It uses the OFDM modulation method of 802.11a but operates in the narrow 2.4 GHz ISM band along with 802.11b. In theory it can operate at up to 54 MBps. It is not yet clear whether this goal will be realized in practice. What it does mean is that the 802.11 committee has produced three different high-speed wireless LANs, 802.11a, 802.11b, and 802.11g (plus a couple of duds). One can legitimately ask if this is a good thing for a standards committee to do.
4.4.3 The 802.11 MAC Sublayer Protocol

Let us now return from the land of electrical engineering to the land of computer science. The 802.11 MAC sublayer protocol is quite different from that of Ethernet due to the inherent complexity of the wireless environment compared to that of a wired system. With Ethernet, a station just waits until the ether goes silent and starts transmitting. If it does not receive a noise burst back within the first 64 bytes, the frame has almost assuredly been delivered correctly. With wireless, this situation does not hold.

To start with, there is the hidden station problem mentioned above and illustrated again in Fig. 4-26(a). Since not all stations are within radio range of each other, transmissions may be going on in one part of a cell that are not received elsewhere in the same cell. In this example, station $C$ is transmitting to station $B$. If $A$ senses the channel, it will not hear anything and falsely conclude that it may now start transmitting to $B$.

![Figure 4-26. (a) The hidden station problem. (b) The exposed station problem.](image)

In addition, there is the inverse problem, the exposed station problem, illustrated in Fig. 4-26(b). Here $B$ wants to send to $C$ so it listens to the channel. When it hears a transmission, it falsely concludes that it may not send to $C$, even though $A$ may be transmitting to $D$ (not shown). In addition, most radios are half duplex, meaning that they cannot transmit and listen for noise bursts at the same time on a single frequency. As a result of these problems, 802.11 does not use CSMA/CD, as Ethernet does.

To deal with this problem, 802.11 supports two modes of operation. The first is called DCF (Distributed Coordination Function), which does not use any kind of central control (in that respect, similar to Ethernet). The other is called PCF (Point Coordination Function) and uses the base station to control all activity in its cell. We will now discuss these two modes in turn.
When DCF is employed, 802.11 uses a protocol called **CSMA/CA (CSMA with Collision Avoidance)**. In this protocol, both physical channel sensing and virtual channel sensing are used. Two methods of operation are supported by CSMA/CA. In the first method, when a station wants to transmit, it senses the channel. If it is idle, it just starts transmitting. It does not sense the channel while transmitting, but emits its entire frame, which may well be destroyed at the receiver due to interference there. If the channel is busy, the sender defers until it goes idle and then starts transmitting. If a collision occurs, the colliding stations wait a random time using the Ethernet binary exponential backoff algorithm and then try again later.

The other mode of CSMA/CA operation is based on MACAW and uses virtual channel sensing, as illustrated in Fig. 4-27. In this example, A wants to send to B. C is a station within range of A (and possibly within range of B, but that does not matter). D is a station within range of B but not within range of A.

![Figure 4-27. The use of virtual channel sensing using CSMA/CA.](image)

The protocol starts when A decides it wants to send data to B. It begins by sending an RTS frame to B to request permission to send it a frame. When B receives this request, it may decide to grant permission, in which case it sends a CTS frame back. Upon receipt of the CTS, A now sends its frame and starts an ACK timer. Upon correct receipt of the data frame, B responds with an ACK frame, terminating the exchange. If A’s ACK timer expires before the ACK gets back to it, the whole protocol is run again.

Now let us consider this exchange from the viewpoints of C and D. C is within range of A, so it may receive the RTS frame. If it does, it realizes that someone is going to send data soon, so for the good of all it desists from transmitting anything until the exchange is completed. From the information provided in the RTS request, it can estimate how long the sequence will take, including the final ACK, so it asserts a kind of virtual channel busy for itself, indicated by NAV (Network Allocation Vector) in Fig. 4-27. D does not hear the RTS, but it does hear the CTS, so it also asserts the NAV signal for itself. Note that the NAV signals are not transmitted; they are just internal reminders to keep quiet for a certain
period of time.

In contrast to wired networks, wireless networks are noisy and unreliable, in no small part due to microwave ovens, which also use the unlicensed ISM bands. As a consequence, the probability of a frame making it through successfully decreases with frame length. If the probability of any bit being in error is $p$, then the probability of an $n$-bit frame being received entirely correctly is $(1 - p)^n$. For example, for $p = 10^{-4}$, the probability of receiving a full Ethernet frame (12,144 bits) correctly is less than 30%. If $p = 10^{-5}$, about one frame in 9 will be damaged. Even if $p = 10^{-6}$, over 1% of the frames will be damaged, which amounts to almost a dozen per second, and more if frames shorter than the maximum are used.

To deal with the problem of noisy channels, 802.11 allows frames to be fragmented into smaller pieces, each with its own checksum. The fragments are individually numbered and acknowledged using a stop-and-wait protocol (i.e., the sender may not transmit fragment $k + 1$ until it has received the acknowledgment for fragment $k$). Once the channel has been acquired using RTS and CTS frames, multiple fragments can be sent in a row, as shown in Fig. 4-28. A sequence of fragments is called a fragment burst.

![Fragment burst](image)

Figure 4-28. A fragment burst.

Fragmentation increases the throughput by restricting retransmissions to the bad fragments rather than the entire frame. The fragment size is not fixed by the standard but is a parameter of each cell and can be adjusted by the base station. The NAV mechanism keeps other stations quiet only until the next acknowledgment, but another mechanism (described below) is used to allow a whole fragment burst to be sent without interference.

All of the above discussion applies to the 802.11 DCF mode. In this mode, there is no central control, and stations compete for air time, just as they do with Ethernet. The other allowed mode is PCF, in which the base station polls the other stations, asking them if they have any frames to send. Since transmission order is completely controlled by the base station in PCF mode, no collisions ever
occur. The standard prescribes the mechanism for polling, but not the polling frequency, polling order, or even whether all stations need to get equal service.

The basic mechanism is for the base station to broadcast a beacon frame periodically (10 to 100 times per second). The beacon frame contains system parameters, such as hopping sequences and dwell times (for FHSS), clock synchronization, etc. It also invites new stations to sign up for polling service. Once a station has signed up for polling service at a certain rate, it is effectively guaranteed a certain fraction of the bandwidth, thus making it possible to give quality of service guarantees.

Battery life is always an issue with mobile wireless devices, so 802.11 pays attention to the issue of power management. In particular, the base station can direct a mobile station to go into sleep state until explicitly awakened by the base station or the user. Having told a station to go to sleep, however, means that the base station has the responsibility for buffering any frames directed at it while the mobile station is asleep. These can be collected later.

PCF and DCF can coexist within one cell. At first it might seem impossible to have central control and distributed control operating at the same time, but 802.11 provides a way to achieve this goal. It works by carefully defining the interframe time interval. After a frame has been completed, a certain amount of dead time is required before any station may send a frame. Four different intervals are defined, each for a specific purpose. The four intervals are depicted in Fig. 4-29.

The shortest interval is **SIFS** (Short InterFrame Spacing). It is used to allow the parties in a single dialog the chance to go first. This includes letting the receiver send a CTS to respond to an RTS, letting the receiver send an ACK for a fragment or full data frame, and letting the sender of a fragment burst transmit the next fragment without having to send an RTS again.

There is always exactly one station that is entitled to respond after a SIFS interval. If it fails to make use of its chance and a time **PIFS** (PCF InterFrame Spacing) elapses, the base station may send a beacon frame or poll frame. This
mechanism allows a station sending a data frame or fragment sequence to finish its frame without anyone else getting in the way, but gives the base station a chance to grab the channel when the previous sender is done without having to compete with eager users.

If the base station has nothing to say and a time DIFS (DCF InterFrame Spacing) elapses, any station may attempt to acquire the channel to send a new frame. The usual contention rules apply, and binary exponential backoff may be needed if a collision occurs.

The last time interval, EIFS (Extended InterFrame Spacing) is used only by a station that has just received a bad or unknown frame to report the bad frame. The idea of giving this event the lowest priority is that since the receiver may have no idea of what is going on, it should wait a substantial time to avoid interfering with an ongoing dialog between two stations.

4.4.4 The 802.11 Frame Structure

The 802.11 standard defines three classes of frames: data, control, and management. Each of these has a header with a variety of fields used within the MAC sublayer. In addition, there are some headers used by the physical layer but these mostly deal with the modulation so we will not discuss them further here.

The format of the data frame is shown in Fig. 4-30. First comes the Frame Control field. It itself has 11 subfields. The first of these is the Protocol version, which allows two versions of the protocol to operate at the same time in the same cell. Then come the Type (data, control, or management) and Subtype fields (e.g., RTS or CTS). The To DS and From DS bits indicate the frame is going to or coming from the intercell distribution system (e.g., Ethernet). The MF bit means that more fragments will follow. The Retry bit marks a retransmission of a frame sent earlier. The Power management bit is used by the base station to put the receiver in sleep state or take it out of sleep state. The More bit indicates that the sender has additional frames for the receiver. The W bit specifies that the frame body has been encrypted using the WEP (Wired Equivalent Privacy) algorithm. Finally, the O bit tells the receiver that a sequence of frames with this bit on must be processed strictly in order.

The second field of the data frame, the Duration field, tells how long the frame and its acknowledgement will occupy the channel. This field is also present in the control frames, and is how other stations manage the NAV mechanism. The frame header contains four addresses, all in standard IEEE 802 format. The source and destination are obviously needed, but what are the other two for? Remember that frames may enter or leave a cell via a base station. The other two addresses are used for the source and destination base stations for intercell traffic.

The Sequence field allows fragments to be numbered. Of the 16 bits available, 12 identify the frame and 4 identify the fragment. The Data field contains the payload, up to 2312 bytes, followed by the usual Checksum.
Management frames have a similar format to data frames, except without one of the base station addresses, because management frames are restricted to a single cell. Control frames are shorter still, having only one or two addresses, no Data field, and no Sequence field. The key information here is in the Subtype field, usually RTS, CTS, or ACK.

### 4.4.5 Services

The 802.11 standard states that each conformant wireless LAN must provide nine services. These services are divided into two categories: five distribution services and four station services. The distribution services relate to managing cell membership and interacting with stations outside the cell. In contrast, the station services relate to activity within a single cell.

The five distribution services are provided by the base stations and deal with station mobility as they enter and leave cells, attaching themselves to and detaching themselves from base stations. They are as follows.

1. **Association.** This service is used by mobile stations to connect themselves to base stations. Typically it is used just after a station moves within the radio range of the base station. Upon arrival, it announces its identity and capabilities. The capabilities include the data rates supported, need for PCF services (i.e., polling), and power management requirements. The base station may accept or reject the mobile station. If the mobile station is accepted, it must then authenticate itself.

2. **Disassociation.** Either the station or the base station may disassociate, thus breaking the relationship. A station should use this service before shutting down or leaving, but the base station may also use it before going down for maintenance.

3. **Reassociation.** A station may change its preferred base station using this service. This facility is useful for mobile stations moving from...
one cell to another. If used correctly, no data will be lost during the handover.

4. **Distribution.** This service determines how to route frames sent to the base station. If the destination is local to the base station, the frames can be sent out directly over the air. Otherwise, they will have to be forwarded over the wired network.

5. **Integration.** If a frame needs to be sent through a non-802.11 network with a different addressing scheme or frame format, this service handles the translation from the 802.11 format to the format required by the destination network.

The remaining four services are intracell. They are used after association has taken place and are as follows.

1. **Authentication.** Because wireless communication can easily be sent or received by unauthorized stations, a station must authenticate itself before it is permitted to send data. After a mobile station has been associated by the base station (i.e., accepted into its cell), the base station sends a special challenge frame to it to see if the mobile station knows the secret key (password) that has been assigned to it. It proves its knowledge of the secret key by encrypting the challenge frame and sending it back to the base station. If the result is correct, the mobile is fully enrolled in the cell. In the initial standard, the base station does not have to prove its identity to the mobile station, but work to repair this defect in the standard is underway.

2. **Deauthentication.** When a previously authenticated station wants to leave the network, it is deauthenticated. After deauthentication, it may no longer use the network.

3. **Privacy.** In order to keep information sent over a wireless LAN confidential, it must be encrypted. This service manages the encryption and decryption. The encryption algorithm specified is RC4, invented by Ronald Rivest of M.I.T.

4. **Data delivery.** Finally, data transmission is what it is all about, so 802.11 provides a way to transmit and receive data. Since 802.11 is modeled on Ethernet, and transmission over Ethernet is not guaranteed to be 100% reliable, transmission over 802.11 is not guaranteed to be reliable either. Higher layers must deal with detecting and correcting errors.

An 802.11 cell has many parameters that can be inspected and, in some cases, adjusted. They relate to encryption, timeout intervals, data rates, beacon...
frequency, and many others.

Wireless LANs based on 802.11 are starting to be deployed in office buildings, airports, and campuses around the world. Rapid growth is expected. For some experience about the widespread deployment of 802.11 at CMU, see (Hills, 2001).