**PHYSICAL LAYER**

**2.2 Guided Transmission Media**

The purpose of the physical layer is to transport a raw bit stream from one machine to another.

Various physical media can be used for the actual transmission. Each one has its own niche in terms of bandwidth, delay, cost, and ease of installation and maintenance. Media are roughly grouped into guided media, such as copper wire and fiber optics, and unguided media, such as radio and lasers through the air. We will look at all of these in the following sections.

**2.2.1 Magnetic Media**

One of the most common ways to transport data from one computer to another is to write them onto magnetic tape or removable media (e.g., recordable DVDs), physically transport the tape or disks to the destination machine, and read them back in again. Although this method is not as sophisticated as using a geosynchronous communication satellite, it is often more cost effective, especially for applications in which high bandwidth or cost per bit transported is the key factor.

A simple calculation will make this point clear. An industry standard Ultrium tape can hold 200

gigabytes. A box 60 x 60 x 60 cm can hold about 1000 of these tapes, for a total capacity of 200

terabytes, or 1600 terabits (1.6 petabits). A box of tapes can be delivered anywhere in the United

States in 24 hours by Federal Express and other companies. The effective bandwidth of this

transmission is 1600 terabits/86,400 sec, or 19 Gbps. If the destination is only an hour away by road, the bandwidth is increased to over 400 Gbps. No computer network can even approach this.

For a bank with many gigabytes of data to be backed up daily on a second machine (so the bank can continue to function even in the face of a major flood or earthquake), it is likely that no other

transmission technology can even begin to approach magnetic tape for performance. Of course,

networks are getting faster, but tape densities are increasing, too. If we now look at cost, we get a similar picture. The cost of an Ultrium tape is around $40 when bought in bulk. A tape can be reused at least ten times, so the tape cost is maybe $4000 per box per usage. Add to this another $1000 for shipping (probably much less), and we have a cost of roughly $5000 to ship 200 TB. This amounts to shipping a gigabyte for under 3 cents. No network can beat

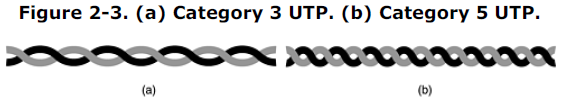
that. The moral of the story is:

**2.2.2 Twisted Pair**

Although the bandwidth characteristics of magnetic tape are excellent, the delay characteristics are poor. Transmission time is measured in minutes or hours, not milliseconds. For many applications an on-line connection is needed. One of the oldest and still most common transmission media is **twisted** **pair**. A twisted pair consists of two insulated copper wires, typically about 1 mm thick. The wires are twisted together in a helical form, just like a DNA molecule. Twisting is done because two parallel wires constitute a fine antenna. When the wires are twisted, the waves from different twists cancel out, so the wire radiates less effectively.

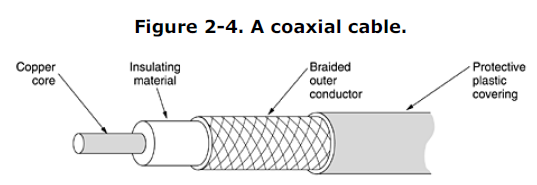
The most common application of the twisted pair is the telephone system. Nearly all telephones are connected to the telephone company (telco) office by a twisted pair. Twisted pairs can run several kilometers without amplification, but for longer distances, repeaters are needed. When many twisted pairs run in parallel for a substantial distance, such as all the wires coming from an apartment building to the telephone company office, they are bundled together and encased in a protective sheath. The pairs in these bundles would interfere with one another if it were not for the twisting. In parts of the world where telephone lines run on poles above ground, it is common to see bundles several centimeters in diameter. Twisted pairs can be used for transmitting either analog or digital signals. The bandwidth depends on the thickness of the wire and the distance traveled, but several megabits/sec can be achieved for a few kilometers in many cases. Due to their adequate performance and low cost, twisted pairs are widely used and are likely to remain so for years to come. Twisted pair cabling comes in several varieties, two of which are important for computer networks.

**Category 3** twisted pairs consist of two insulated wires gently twisted together. Four such pairs are typically grouped in a plastic sheath to protect the wires and keep them together. Prior to about 1988, most office buildings had one category 3 cable running from a central **wiring closet** on each floor into each office. This scheme allowed up to four regular telephones or two multiline telephones in each office to connect to the telephone company equipment in the wiring closet. Starting around 1988, the more advanced **category 5** twisted pairs were introduced. They are similar to category 3 pairs, but with more twists per centimeter, which results in less crosstalk and a better quality signal over longer distances, making them more suitable for high-speed computer communication. Up-and-coming categories are 6 and 7, which are capable of handling signals with bandwidths of 250 MHz and 600 MHz, respectively (versus a mere 16 MHz and 100 MHz for categories 3 and 5, respectively). All of these wiring types are often referred to as **UTP** (**Unshielded Twisted Pair**), to contrast them with the bulky, expensive, shielded twisted pair cables IBM introduced in the early 1980s, but which have not proven popular outside of IBM installations. Twisted pair cabling is illustrated in Fig. 2-3.

**2.2.3**

**Coaxial Cable**

Another common transmission medium is the **coaxial cable** (known to its many friends as just ''coax'' and pronounced ''co-ax''). It has better shielding than twisted pairs, so it can span longer distances at higher speeds. Two kinds of coaxial cable are widely used. One kind, 50-ohm cable, is commonly used when it is intended for digital transmission from the start. The other kind, 75-ohm cable, is commonly used for analog transmission and cable television but is becoming more important with the advent of Internet over cable. This distinction is based on historical, rather than technical, factors (e.g., early dipole antennas had an impedance of 300 ohms, and it was easy to use existing 4:1 impedance matching transformers). A coaxial cable consists of a stiff copper wire as the core, surrounded by an insulating material. The insulator is encased by a cylindrical conductor, often as a closely-woven braided mesh. The outer conductor is covered in a protective plastic sheath. A cutaway view of a coaxial cable is shown in Fig. 2-4.

****The

construction and shielding of the coaxial cable give it a good combination of high bandwidth and

excellent noise immunity. The bandwidth possible depends on the cable quality, length, and signal-to noise ratio of the data signal. Modern cables have a bandwidth of close to 1 GHz. Coaxial cables used to be widely used within the telephone system for long-distance lines but have now largely been replaced by fiber optics on long-haul routes. Coax is still widely used for cable television and metropolitan area networks, however.

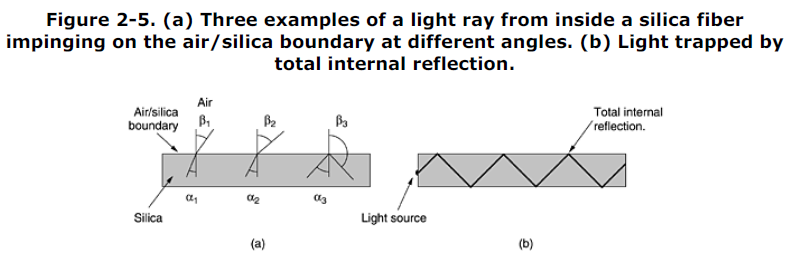
**2.2.4 Fiber Optics**

Many people in the computer industry take enormous pride in how fast computer technology is

improving. The original (1981) IBM PC ran at a clock speed of 4.77 MHz. Twenty years later, PCs could run at 2 GHz, a gain of a factor of 20 per decade. Not too bad. In the same period, wide area data communication went from 56 kbps (the ARPANET) to 1 Gbps (modern optical communication), a gain of more than a factor of 125 per decade, while at the same time the error rate went from 10-5 per bit to almost zero. Furthermore, single CPUs are beginning to approach physical limits, such as speed of light and heat dissipation problems. In contrast, with *current* fiber technology, the achievable bandwidth is certainly in excess of 50,000 Gbps (50 Tbps) and many people are looking very hard for better technologies and materials. The current practical signaling limit of about 10 Gbps is due to our inability to convert between electrical and optical signals any faster, although in the laboratory, 100 Gbps has been achieved on a single fiber.

In the race between computing and communication, communication won. The full implications of essentially infinite bandwidth (although not at zero cost) have not yet sunk in to a generation of computer scientists and engineers taught to think in terms of the low Nyquist and Shannon limits imposed by copper wire. The new conventional wisdom should be that all computers are hopelessly slow and that networks should try to avoid computation at all costs, no matter how much bandwidth that wastes. In this section we will study fiber optics to see how that transmission technology works. An optical transmission system has three key components: the light source, the transmission medium, and the detector. Conventionally, a pulse of light indicates a 1 bit and the absence of light indicates a 0 bit. The transmission medium is an ultra-thin fiber of glass. The detector generates an electrical pulse when light falls on it. By attaching a light source to one end of an optical fiber and a detector to the other, we have a unidirectional data transmission system that accepts an electrical signal, converts and transmits it by light pulses, and then reconverts the output to an electrical signal at the receiving end. This transmission system would leak light and be useless in practice except for an interesting principle

of physics. When a light ray passes from one medium to another, for example, from fused silica to air, the ray is refracted (bent) at the silica/air boundary, as shown in Fig. 2-5(a). Here we see a light ray incident on the boundary at an angle a1 emerging at an angle b1*.* The amount of refraction depends on the properties of the two media (in particular, their indices of refraction). For angles of incidence above a certain critical value, the light is refracted back into the silica; none of it escapes into the air. Thus, a light ray incident at or above the critical angle is trapped inside the fiber, as shown in Fig. 2-5(b), and can propagate for many kilometers with virtually no loss.

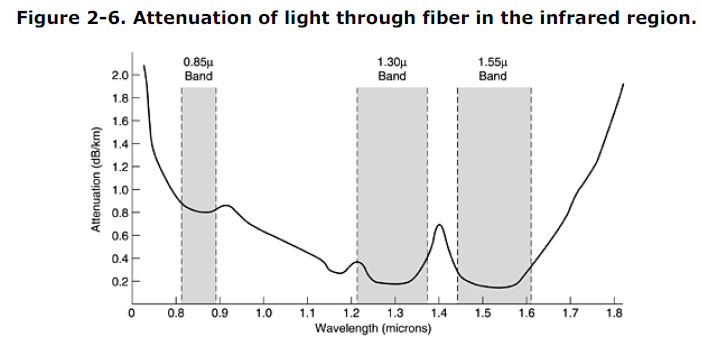
****

The sketch of Fig. 2-5(b) shows only one trapped ray, but since any light ray incident on the boundary above the critical angle will be reflected internally, many different rays will be bouncing around at different angles. Each ray is said to have a different **mode**, so a fiber having this property is called a **multimode fiber**. However, if the fiber's diameter is reduced to a few wavelengths of light, the fiber acts like a wave guide, and the light can propagate only in a straight line, without bouncing, yielding a **single-mode** **fiber**. Single-mode fibers are more expensive but are widely used for longer distances. Currently available single-mode fibers can transmit data at 50 Gbps for 100 km without amplification. Even higher data rates have been achieved in the laboratory for shorter distances. **Transmission of Light through Fiber**

Optical fibers are made of glass, which, in turn, is made from sand, an inexpensive raw material available in unlimited amounts. Glassmaking was known to the ancient Egyptians, but their glass had to be no more than 1 mm thick or the light could not shine through. Glass transparent enough to be useful for windows was developed during the Renaissance. The glass used for modern optical fibers is so transparent that if the oceans were full of it instead of water, the seabed would be as visible from the surface as the ground is from an airplane on a clear day.

The attenuation of light through glass depends on the wavelength of the light (as well as on some

physical properties of the glass). For the kind of glass used in fibers, the attenuation is shown in Fig. 2-6 in decibels per linear kilometer of fiber. The attenuation in decibels is given by the formula

****



For example, a factor of two loss gives an attenuation of 10 log10 2 = 3 dB. The figure shows the near infrared part of the spectrum, which is what is used in practice. Visible light has slightly shorter wavelengths, from 0.4 to 0.7 microns (1 micron is 10-6 meters). The true metric purist would refer to these wavelengths as 400 nm to 700 nm, but we will stick with traditional usage.

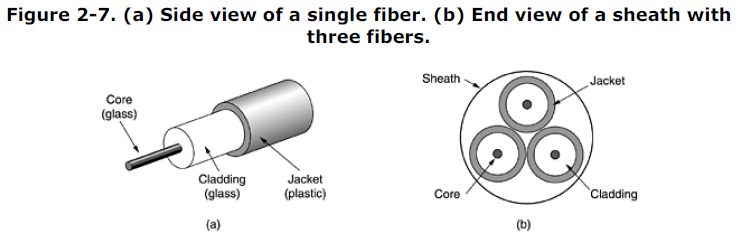
Three wavelength bands are used for optical communication. They are centered at 0.85, 1.30, and 1.55 microns, respectively. The last two have good attenuation properties (less than 5 percent loss per kilometer). The 0.85 micron band has higher attenuation, but at that wavelength the lasers and electronics can be made from the same material (gallium arsenide). All three bands are 25,000 to 30,000 GHz wide. Light pulses sent down a fiber spread out in length as they propagate. This spreading is called **chromatic dispersion**.

The amount of it is wavelength dependent. One way to keep these spread-out pulses from overlapping is to increase the distance between them, but this can be done only by reducing the signaling rate. Fortunately, it has been discovered that by making the pulses in a special shape related to the reciprocal of the hyperbolic cosine, nearly all the dispersion effects cancel out, and it is possible to send pulses for thousands of kilometers without appreciable shape distortion. These pulses are called **solitons**. A considerable amount of research is going on to take solitons out of the lab and into the field.

**Fiber Cables**

Fiber optic cables are similar to coax, except without the braid. Figure 2-7(a) shows a single fiber

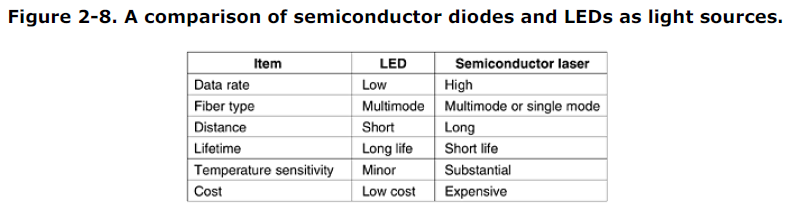
viewed from the side. At the center is the glass core through which the light propagates. In multimode fibers, the core is typically 50 microns in diameter, about the thickness of a human hair. In single mode fibers, the core is 8 to 10 microns.

****

The core is surrounded by a glass cladding with a lower index of refraction than the core, to keep all the light in the core. Next comes a thin plastic jacket to protect the cladding. Fibers are typically grouped in bundles, protected by an outer sheath. Figure 2-7(b) shows a sheath with three fibers. Terrestrial fiber sheaths are normally laid in the ground within a meter of the surface, where they are occasionally subject to attacks by backhoes or gophers. Near the shore, transoceanic fiber sheaths are buried in trenches by a kind of seaplow. In deep water, they just lie on the bottom, where they can be snagged by fishing trawlers or attacked by giant squid.

Fibers can be connected in three different ways. First, they can terminate in connectors and be plugged into fiber sockets. Connectors lose about 10 to 20 percent of the light, but they make it easy to reconfigure systems. Second, they can be spliced mechanically. Mechanical splices just lay the two carefully-cut ends next to each other in a special sleeve and clamp them in place. Alignment can be improved by passing light through the junction and then making small adjustments to maximize the signal. Mechanical splices take trained personnel about 5 minutes and result in a 10 percent light loss. Third, two pieces of fiber can be fused (melted) to form a solid connection. A fusion splice is almost as good as a single drawn fiber, but even here, a small amount of attenuation occurs. For all three kinds of splices, reflections can occur at the point of the splice, and the reflected energy can interfere with the signal. Two kinds of light sources are typically used to do the signaling, LEDs (Light Emitting Diodes) and semiconductor lasers. They have different properties, as shown in Fig. 2-8. They can be tuned in wavelength by inserting Fabry-Perot or Mach-Zehnder interferometers between the source and the fiber. Fabry-Perot interferometers are simple resonant cavities consisting of two parallel mirrors. The light is incident perpendicular to the mirrors. The length of the cavity selects out those wavelengths

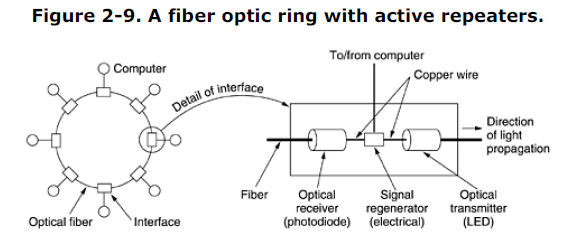
that fit inside an integral number of times. Mach-Zehnder interferometers separate the light into two beams. The two beams travel slightly different distances. They are recombined at the end and are in phase for only certain wavelengths.

****

The receiving end of an optical fiber consists of a photodiode, which gives off an electrical pulse when struck by light. The typical response time of a photodiode is 1 nsec, which limits data rates to about 1 Gbps. Thermal noise is also an issue, so a pulse of light must carry enough energy to be detected. By making the pulses powerful enough, the error rate can be made arbitrarily small.

**Fiber Optic Networks**

Fiber optics can be used for LANs as well as for long-haul transmission, although tapping into it is more complex than connecting to an Ethernet. One way around the problem is to realize that a ring network is really just a collection of point-to-point links, as shown in Fig. 2-9. The interface at each computer passes the light pulse stream through to the next link and also serves as a T junction to allow the computer to send and accept messages.

****

Two types of interfaces are used. A passive interface consists of two taps fused onto the main fiber. One tap has an LED or laser diode at the end of it (for transmitting), and the other has a photodiode (for receiving). The tap itself is completely passive and is thus extremely reliable because a broken LED or photodiode does not break the ring. It just takes one computer off-line.

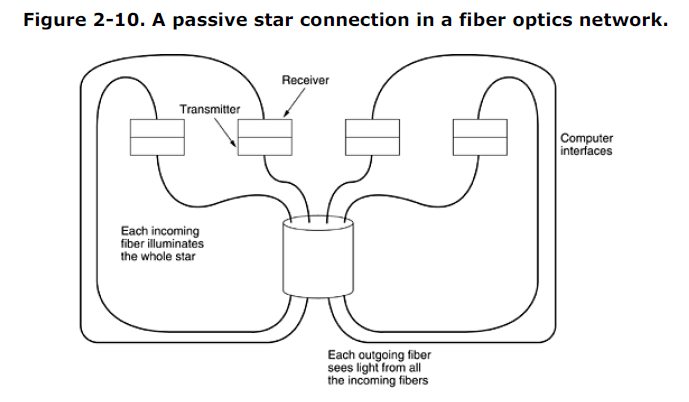
The other interface type, shown in Fig. 2-9, is the **active repeater**. The incoming light is converted to an electrical signal, regenerated to full strength if it has been weakened, and retransmitted as light. The interface with the computer is an ordinary copper wire that comes into the signal regenerator. Purely optical repeaters are now being used, too. These devices do not require the optical to electrical to optical conversions, which means they can operate at extremely high bandwidths.

If an active repeater fails, the ring is broken and the network goes down. On the other hand, since the signal is regenerated at each interface, the individual computer-to-computer links can be kilometers long, with virtually no limit on the total size of the ring. The passive interfaces lose light at each junction, so the number of computers and total ring length are greatly restricted.

A ring topology is not the only way to build a LAN using fiber optics. It is also possible to have

hardware broadcasting by using the **passive star** construction of Fig. 2-10. In this design, each

interface has a fiber running from its transmitter to a silica cylinder, with the incoming fibers fused to one end of the cylinder. Similarly, fibers fused to the other end of the cylinder are run to each of the receivers. Whenever an interface emits a light pulse, it is diffused inside the passive star to illuminate all the receivers, thus achieving broadcast. In effect, the passive star combines all the incoming signals and transmits the merged result on all lines. Since the incoming energy is divided among all the outgoing lines, the number of nodes in the network is limited by the sensitivity of the photodiodes.



**Comparison of Fiber Optics and Copper Wire**

It is instructive to compare fiber to copper. Fiber has many advantages. To start with, it can handle much higher bandwidths than copper. This alone would require its use in high-end networks. Due to the low attenuation, repeaters are needed only about every 50 km on long lines, versus about every 5 km for copper, a substantial cost saving. Fiber also has the advantage of not being affected by power surges, electromagnetic interference, or power failures. Nor is it affected by corrosive chemicals in the air, making it ideal for harsh factory environments.

Oddly enough, telephone companies like fiber for a different reason: it is thin and lightweight. Many existing cable ducts are completely full, so there is no room to add new capacity. Removing all the copper and replacing it by fiber empties the ducts, and the copper has excellent resale value to copper refiners who see it as very high grade ore. Also, fiber is much lighter than copper. One thousand twisted pairs 1 km long weigh 8000 kg. Two fibers have more capacity and weigh only 100 kg, which greatly reduces the need for expensive mechanical support systems that must be maintained. For new routes, fiber wins hands down due to its much lower installation cost. Finally, fibers do not leak light and are quite difficult to tap. These properties gives fiber excellent security against potential wiretappers. On the downside, fiber is a less familiar technology requiring skills not all engineers have, and fibers can be damaged easily by being bent too much. Since optical transmission is inherently unidirectional, two-way communication requires either two fibers or two frequency bands on one fiber. Finally, fiber

interfaces cost more than electrical interfaces. Nevertheless, the future of all fixed data communication for distances of more than a few meters is clearly with fiber. For a discussion of all aspects of fiber optics and their networks, see (Hecht, 2001).

**2.3 UNGUIDED TRANSMISSION MEDIA**

2.3.2 Radio Transmission

Radio waves are easy to generate, can travel long distances, and can penetrate buildings easily, so they are widely used for communication, both indoors and outdoors. Radio waves also are

omnidirectional, meaning that they travel in all directions from the source, so the transmitter and

receiver do not have to be carefully aligned physically.Sometimes omnidirectional radio is good, but sometimes it is bad. In the 1970s, General Motors decided to equip all its new Cadillacs with computer-controlled antilock brakes. When the driver stepped on the brake pedal, the computer pulsed the brakes on and off instead of locking them on hard. One fine day an Ohio Highway Patrolman began using his new mobile radio to call headquarters, and suddenly the Cadillac next to him began behaving like a bucking bronco. When the officer pulledthe car over, the driver claimed that he had done nothing and that the car had gone crazy. Eventually, a pattern began to emerge: Cadillacs would sometimes go berserk, but only on major highways in Ohio and then only when the Highway Patrol was watching. For a long, long time General Motors could not understand why Cadillacs worked fine in all the other states and also on minor roads in Ohio. Only after much searching did they discover that the Cadillac's wiring made a fine antenna for

the frequency used by the Ohio Highway Patrol's new radio system. The properties of radio waves are frequency dependent. At low frequencies, radio waves pass through obstacles well, but the power falls off sharply with distance from the source, roughly as 1/r2 in air. At high frequencies, radio waves tend to travel in straight lines and bounce off obstacles. They are also

absorbed by rain. At all frequencies, radio waves are subject to interference from motors and other electrical equipment. Due to radio's ability to travel long distances, interference between users is a problem. For this reason, all governments tightly license the use of radio transmitters, with one exception, discussed below.

In the VLF, LF, and MF bands, radio waves follow the ground, as illustrated in Fig. 2-12(a). These waves can be detected for perhaps 1000 km at the lower frequencies, less at the higher ones. AM radio broadcasting uses the MF band, which is why the ground waves from Boston AM radio stations cannot be heard easily in New York. Radio waves in these bands pass through buildings easily, which is why portable radios work indoors. The main problem with using these bands for data communication is their low bandwidth.

Figure 2-12. (a) In the VLF, LF, and MF bands, radio waves follow the

curvature of the earth. (b) In the HF band, they bounce off the ionosphere.



In the HF and VHF bands, the ground waves tend to be absorbed by the earth. However, the waves that reach the ionosphere, a layer of charged particles circling the earth at a height of 100 to 500 km, are refracted by it and sent back to earth, as shown in Fig. 2-12(b). Under certain atmospheric conditions, the signals can bounce several times. Amateur radio operators (hams) use these bands to talk long distance. The military also communicate in the HF and VHF bands.

2.3.3 Microwave Transmission

Above 100 MHz, the waves travel in nearly straight lines and can therefore be narrowly focused.

Concentrating all the energy into a small beam by means of a parabolic antenna (like the familiar

satellite TV dish) gives a much higher signal-to-noise ratio, but the transmitting and receiving

antennas must be accurately aligned with each other. In addition, this directionality allows multiple transmitters lined up in a row to communicate with multiple receivers in a row without interference, provided some minimum spacing rules are observed. Before fiber optics, for decades these microwaves formed the heart of the long-distance telephone transmission system. In fact, MCI, one of AT&T's first competitors after it was deregulated, built its entire system with microwave communications going from tower to tower tens of kilometers apart. Even the company's name reflected this (MCI stood for Microwave Communications, Inc.). MCI has since gone over to fiber and merged with WorldCom.Since the microwaves travel in a straight line, if the towers are too far apart, the earth will get in the way (think about a San Francisco to Amsterdam link). Consequently, repeaters are needed periodically. The higher the towers are, the farther apart they can be. The distance between repeaters goes up very roughly with the square root of the tower height. For 100-meter-high towers, repeaters can be spaced 80 km apart.

Unlike radio waves at lower frequencies, microwaves do not pass through buildings well. In addition, even though the beam may be well focused at the transmitter, there is still some divergence in space. Some waves may be refracted off low-lying atmospheric layers and may take slightly longer to arrive than the direct waves. The delayed waves may arrive out of phase with the direct wave and thus cancel the signal. This effect is called multipath fading and is often a serious problem. It is weather and frequency dependent. Some operators keep 10 percent of their channels idle as spares to switch on when multipath fading wipes out some frequency band temporarily.

The demand for more and more spectrum drives operators to yet higher frequencies. Bands up to 10 GHz are now in routine use, but at about 4 GHz a new problem sets in: absorption by water. These waves are only a few centimeters long and are absorbed by rain. This effect would be fine if one were planning to build a huge outdoor microwave oven for roasting passing birds, but for communication, it is a severe problem. As with multipath fading, the only solution is to shut off links that are being rained on and route around them.

In summary, microwave communication is so widely used for long-distance telephone communication, mobile phones, television distribution, and other uses that a severe shortage of spectrum has developed. It has several significant advantages over fiber. The main one is that no right of way is needed, and by buying a small plot of ground every 50 km and putting a microwave tower on it, one can bypass the telephone system and communicate directly. This is how MCI managed to get started as a new long-distance telephone company so quickly. (Sprint went a completely different route: it was formed by the Southern Pacific Railroad, which already owned a large amount of right of way and just buried fiber next to the tracks.)

Microwave is also relatively inexpensive. Putting up two simple towers (may be just big poles with four guy wires) and putting antennas on each one may be cheaper than burying 50 km of fiber through a congested urban area or up over a mountain, and it may also be cheaper than leasing the telephone company's fiber, especially if the telephone company has not yet even fully paid for the copper it ripped out when it put in the fiber.

2.3.4 Infrared and Millimeter Waves

Unguided infrared and millimeter waves are widely used for short-range communication. The remote controls used on televisions, VCRs, and stereos all use infrared communication. They are relatively directional, cheap, and easy to build but have a major drawback: they do not pass through solid objects (try standing between your remote control and your television and see if it still works). In general, as we go from long-wave radio toward visible light, the waves behave more and more like light and less and less like radio.

On the other hand, the fact that infrared waves do not pass through solid walls well is also a plus. It means that an infrared system in one room of a building will not interfere with a similar system in adjacent rooms or buildings: you cannot control your neighbor's television with your remote control. Furthermore, security of infrared systems against eavesdropping is better than that of radio systems precisely for this reason. Therefore, no government license is needed to operate an infrared system, in contrast to radio systems, which must be licensed outside the ISM bands. Infrared communication has a limited use on the desktop, for example, connecting notebook computers and printers, but it is not a major player in the communication game.

2.3.5 Lightwave Transmission

Unguided optical signaling has been in use for centuries. Paul Revere used binary optical signaling from the Old North Church just prior to his famous ride. A more modern application is to connect the LANs in two buildings via lasers mounted on their rooftops. Coherent optical signaling using lasers is inherently unidirectional, so each building needs its own laser and its own photodetector. This scheme offers very high bandwidth and very low cost. It is also relatively easy to install and, unlike microwave, does not require an FCC license.

The laser's strength, a very narrow beam, is also its weakness here. Aiming a laser beam 1-mm wide at a target the size of a pin head 500 meters away requires the marksmanship of a latter-day Annie Oakley. Usually, lenses are put into the system to defocus the beam slightly.

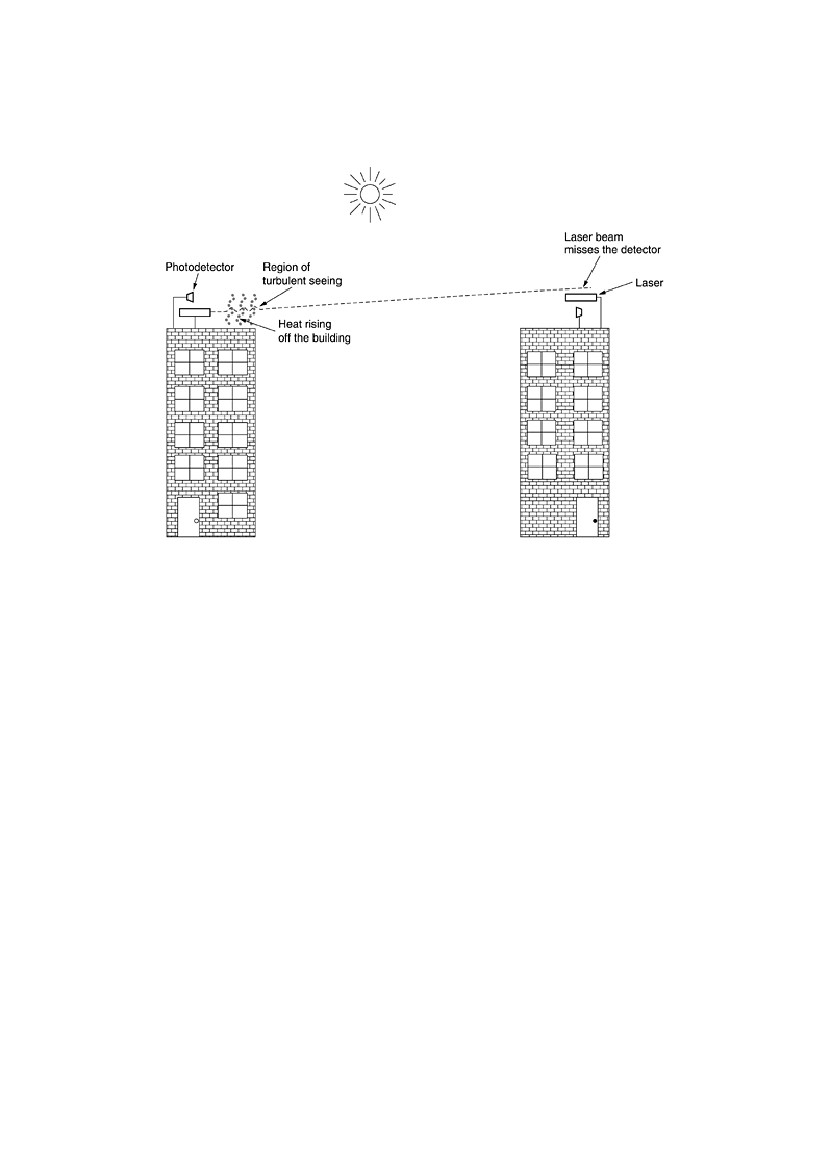
A disadvantage is that laser beams cannot penetrate rain or thick fog, but they normally work well on sunny days. However, the author once attended a conference at a modern hotel in Europe at which the conference organizers thoughtfully provided a room full of terminals for the attendees to read their e- mail during boring presentations. Since the local PTT was unwilling to install a large number of telephone lines for just 3 days, the organizers put a laser on the roof and aimed it at their university's computer science building a few kilometers away. They tested it the night before the conference and it worked perfectly. At 9 a.m. the next morning, on a bright sunny day, the link failed completely and stayed down all day. That evening, the organizers tested it again very carefully, and once again it worked absolutely perfectly. The pattern repeated itself for two more days consistently. After the conference, the organizers discovered the problem. Heat from the sun during the daytime caused convection currents to rise up from the roof of the building, as shown in Fig. 2-14. This turbulent air diverted the beam and made it dance around the detector. Atmospheric ''seeing'' like this makes the stars twinkle (which is why astronomers put their telescopes on the tops of mountains—to get above as much of the atmosphere as possible). It is also responsible for shimmering roads on a hot day and the wavy images seen when one looks out above a hot radiator

Figure 2-14. Convection currents can interfere with laser communication

systems. A bidirectional system with two lasers is pictured here.

**The Medium Access Control Sub layer**

As we discussed earlier 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 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, many of which use a multiaccess channel as the basis for 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 previous chapter. 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, 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 constant 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.

**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. **Station Model.** The model consists of *N* independent **stations** (e.g., computers, telephones, or personal communications ), each with a program or user that generates frames for transmission.

Stations are sometimes called **terminals**. The probability of a frame being generated in an interval of length *t* is *t*, where 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. **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 equivalent, although protocol software may assign priorities to them. **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 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. Two alternative assumptions about time are possible. 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. 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 use 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**

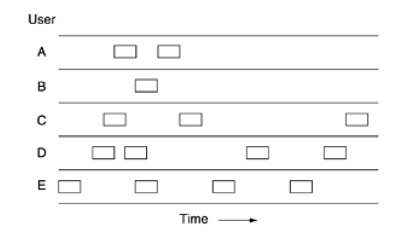
In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. Their work has been extended by many researchers since then (Abramson, 1985). Although Abramson's work, called the ALOHA system, used ground-based radio 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 time is divided 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 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 by allowing variable length frames.

**Figure 4-1. In pure ALOHA, frames are transmitted at completely arbitrary**

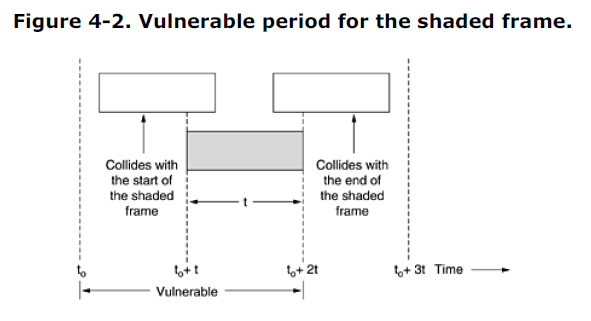
**times.**



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. An interesting question is: What is the efficiency of an ALOHA channel? In other words, 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).

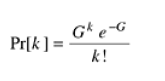
A user is always in one of two states: typing or waiting. Initially, all users are in the typing state. When a line is finished, the user stops typing, waiting for a response. The station then transmits a frame containing the line and checks the channel to see if it was successful. If so, the user sees 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 N.* At low load (i.e., *N* 0), there will be few collisions, hence few retransmissions, so *G 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 + 2*t* will bump into the end of the shaded frame.



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

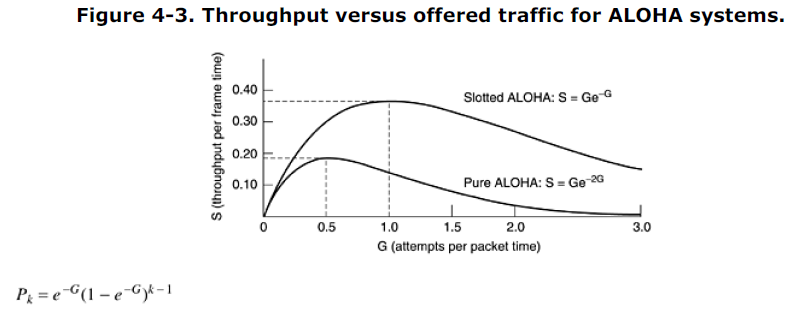
**Equation 4**



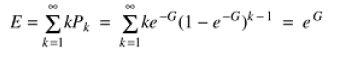
so the probability of zero frames is just *e*-*G.* In an interval two frame times long, the mean number of frames generated is 2*G.* The probability of no other traffic being initiated during the entire vulnerable period is thus given by *P*0 = *e* -2*G.* Using *S* = *GP*0, we get



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 just 1 - *e*-*G.* The probability of a transmission requiring exactly *k* attempts, (i.e., *k* – 1 collisions followed by one success) is,



The expected number of transmissions, *E*, per carriage return typed is then



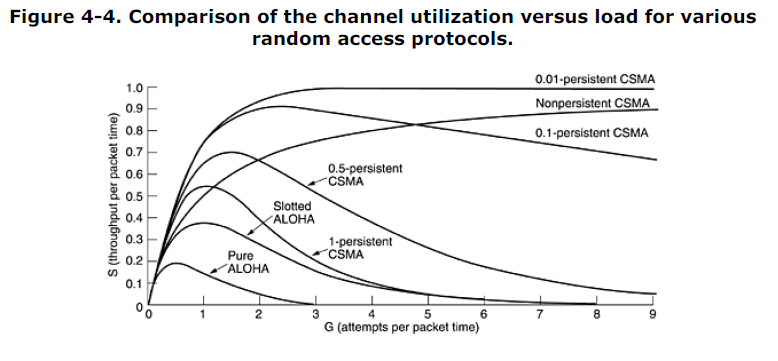
As a result of the exponential dependence of *E* upon *G*, small increases in the channel load can drastically reduce its performance. Slotted Aloha is important for a reason that may not be initially obvious. It was devised in the 1970s, used in a few early experimental systems, then almost forgotten. 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 also study many 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.

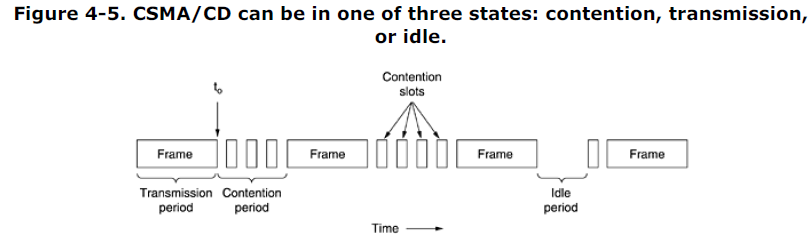
Protocols in which stations listen for a carrier (i.e., a transmission) and act accordingly are called **carrier sense protocols**. A number of them have been proposed. Kleinrock and Tobagi (1975) have analyzed several such protocols in detail. Below we will mention several versions of the carrier sense 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 when 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 approach will lead to a higher performance than pure ALOHA. Exactly the same holds for slotted ALOHA. A second carrier sense protocol is **non persistent 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. Consequently, this algorithm leads to better channel utilization but longer delays than 1-persistent CSMA. The last protocol is **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, the unlucky station 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.



**CSMA with Collision Detection**

Persistent and non persistent 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 **CSMA/CD** (**CSMA 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 *.* At *t*0, one station begins transmitting. At - , 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- *.* In other words, in the worst case a station cannot be sure that it has seized the channel until it has transmitted for 2without hearing a collision. For this reason we will model the contention interval as a slotted ALOHA system with slot width 2*.* On a 1-km long

coaxial cable, 5 Usec. 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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, listening 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

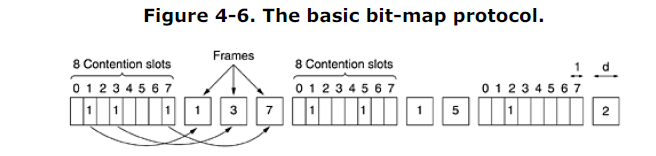
delivery. Even in the absence of collisions, the receiver may not have copied the frame correctly for various reasons (e.g., lack of buffer space or a missed interrupt).

**4.2.3 Collision-Free Protocols**

Although collisions do not occur with CSMA/CD once a station has unambiguously captured 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 ) and the frames are short. And CSMA/CD is not universally applicable. 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. Most of these are not currently used in major systems, but in a rapidly changing field, having some protocols with excellent properties available for future systems is often a good thing. In the protocols to be described, we assume 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 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.

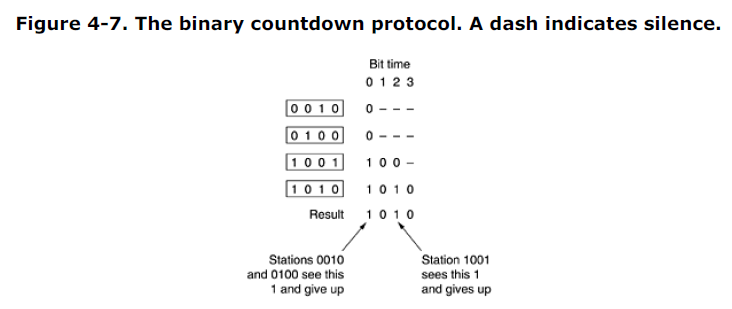
Consider the situation from the point of view of a low-numbered station, such as 0 or 1. Typically, when it becomes ready to send, the ''current'' slot will be somewhere in the middle of the bit map. On average, the station will have to wait *N/*2 slots for the current scan to finish and another full *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 average 1.5*N* slots and high-numbered stations must wait on average 0.5*N* 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.



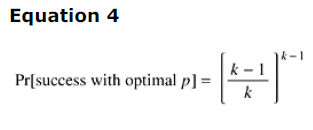
The channel efficiency of this method is *d/*(*d* + log2 *N*). If, however, the frame format has been cleverly chosen so that the sender's address is the first field in the frame, even these log2 *N* bits are not wasted, and the efficiency is 100 percent. Mok and Ward (1979) have described a variation 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.

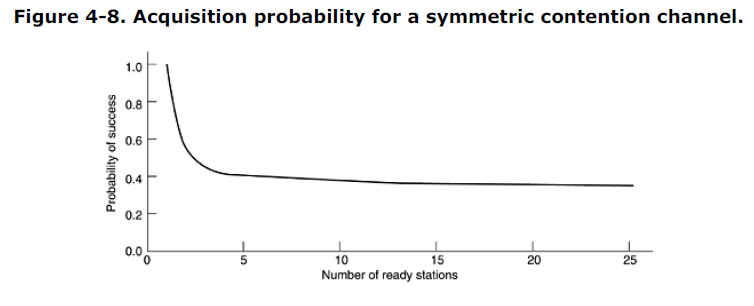
**4.2.4 Limited-Contention Protocols**

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 load 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 to 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



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.*

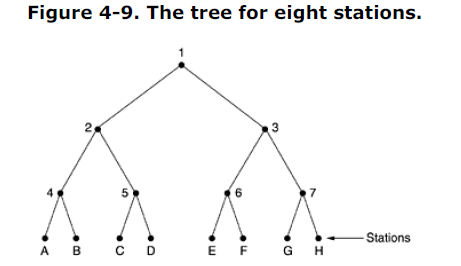


From Fig. 4-8, it is fairly obvious that the probability of some station acquiring the channel can be increased only by decreasing the amount of competition. The limited-contention protocols do precisely that. They first divide the stations into (not necessarily disjoint) groups. Only the members of 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 computerized 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.



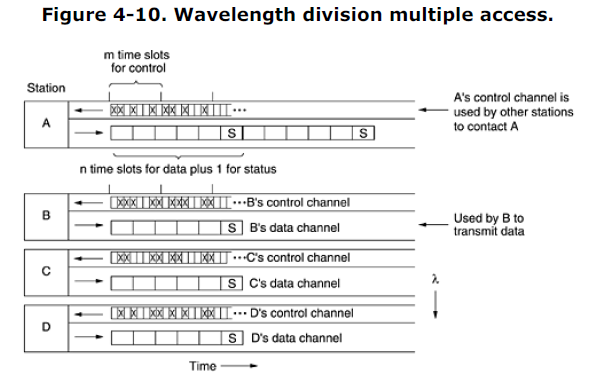
In essence, if a collision occurs during slot 0, the entire tree is searched, depth first, to locate all ready stations. Each bit slot is associated with some particular node in the tree. If a collision occurs, the search continues recursively with the node's left and right children. If a bit slot is idle or if only one station 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-*iq.* 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-*iq* = 1*.* Solving this equation, we find that *i* = log2 *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 subchannels using FDM, TDM, or both, and dynamically allocate them as needed. Schemes like this are commonly used on fiber optic LANs 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 into channels (wavelength bands), as shown in Fig. 2-31. 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. Each channel is divided into groups of time slots, as shown 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 traffic classes : (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 still 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 protocol are possible. For example, instead of each station having 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 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 farther apart because some control information is in the way. Numerous other WDMA protocols have been proposed and implemented, differing in various details. Some have only one control channel; others 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**

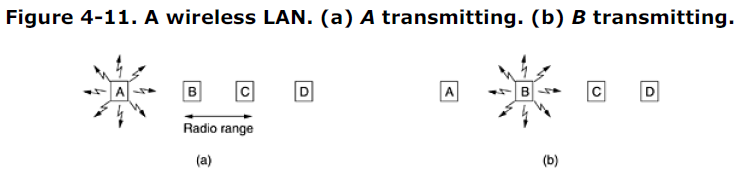
As the number of mobile computing and communication devices grows, so does the demand to connect them to the outside world. Even the very first mobile telephones had the ability to connect to other telephones. The first portable computers did not have this capability, but soon afterward, modems became commonplace 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 (also called access points) 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 and covering all the stations in its cell. Typically, its bandwidth is 11 to 54 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, in other words, we will not consider CDMA-type systems further in this discussion. 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

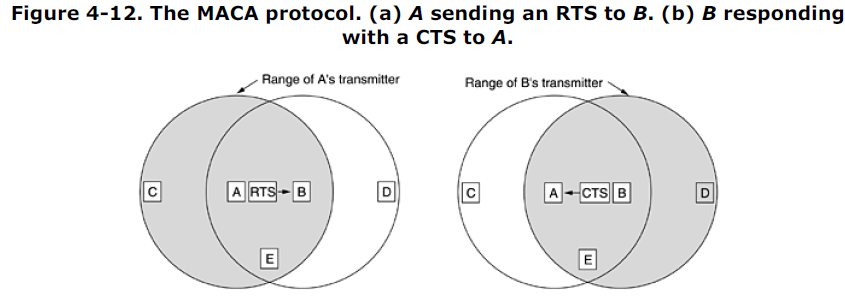
*B* and *D*, but not with *A*.



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 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 is called the **exposed station problem**. The problem is that before starting a transmission, a station really wants to know whether there is activity around the receiver.

CSMA merely tells it whether 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 for the duration of the upcoming (large) data frame. MACA is illustrated in Fig. 4-12.



Let us now 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 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 use, 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

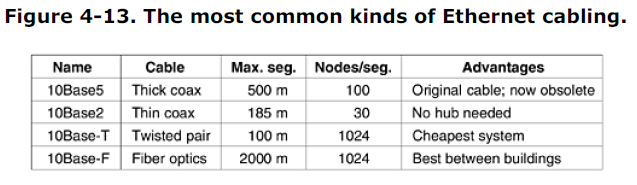
Association to weed out the unfit. The most important of the survivors are 802.3 (Ethernet) and 802.11 (wireless LAN). With 802.15 (Bluetooth) and 802.16 (wireless MAN), it is too early to tell. Please consult the 5th edition of this book to find out. Both 802.3 and 802.11 have different physical layers 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 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. The first number is the speed in Mbps. Then comes the word ''Base'' (or sometimes ''BASE'') to indicate baseband transmission. There used to be a broadband variant, 10Broad36, but it never caught on in the marketplace and has since vanished. Finally, if the medium is

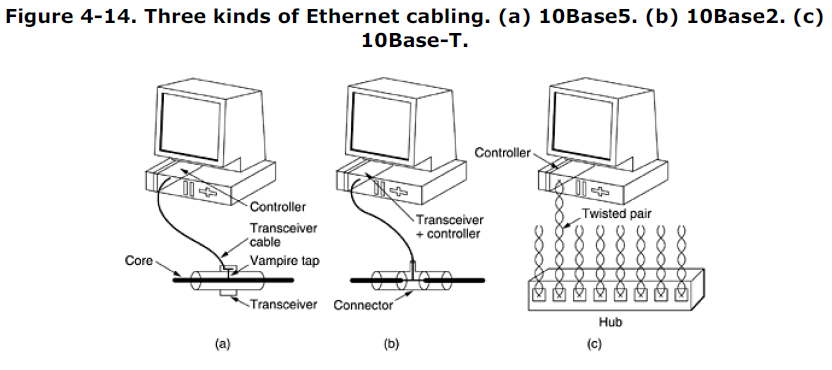
coax, its length is given rounded to units of 100 m after ''Base.''



Historically, the second cable type was **10Base2**, or **thin Ethernet,** which, in contrast to the gardenhose- like thick Ethernet, bends easily. Connections to it are made using industry-standard BNC connectors to form T junctions, rather than using vampire taps. BNC connectors are easier to use and more reliable. Thin Ethernet is much cheaper and easier to install, but it can run for only 185 meters per segment, each of which can handle only 30 machines. Detecting cable breaks, excessive length, 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**.

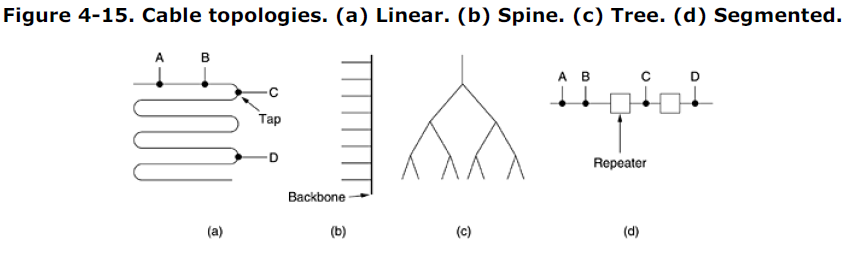
The problems associated with finding cable breaks drove systems toward a different kind of wiring pattern, in which all stations have a cable running to a central **hub** in which they are all connected electrically (as if they were soldered together). Usually, these wires are telephone company twisted pairs, since most office buildings are already wired this way, and normally plenty of spare pairs are available. This scheme is called **10Base-T**. Hubs do not buffer incoming traffic. We will discuss an improved version of this idea (switches), which do buffer incoming traffic 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** or **drop 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 shared cable at all, just the hub (a box full of electronics) to which each station is connected by a dedicated (i.e., not shared) cable. 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 wiretapping copper wire. Figure 4-15 shows different ways of wiring a building. In Fig. 4-15(a), a single cable is snaked from room to room, with each station tapping into 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 the spine 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.



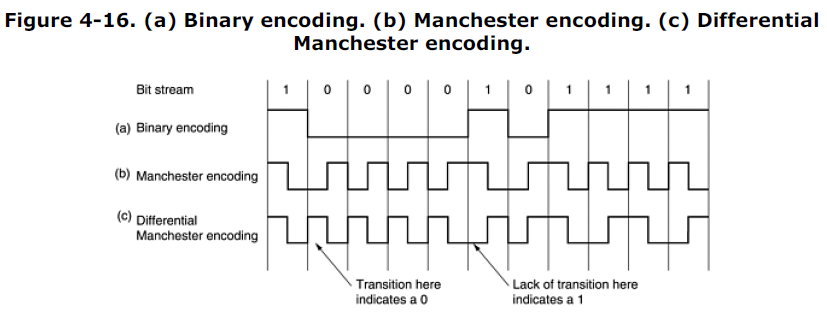
Each version of Ethernet has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by **repeaters**, as shown in Fig. 4-15(d). A repeater is a physical layer device. It receives, amplifies (regenerates), and retransmits signals in both directions. As far as the software is concerned, a series of cable segments connected by repeaters is no different from 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 uses 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). This problem can be solved by using +1 volts for a 1 and -1 volts for a 0, but there is still the problem of a receiver sampling the signal at a slightly different frequency than the sender used to generate it. Different clock speeds can cause the receiver and sender to get out of

synchronization about where the bit boundaries are, especially after a long run of consecutive 0s or a long run of consecutive 1s. 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

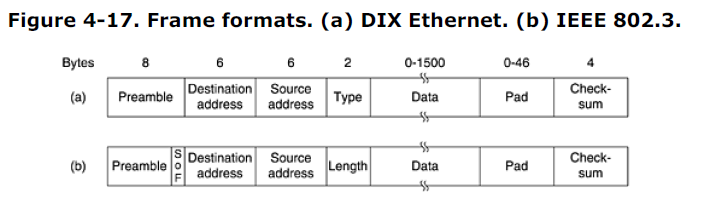
equal intervals. A binary 1 bit is sent by having the voltage set high during the first interval and low in the second one. A binary 0 is just the reverse: first low and then high. This scheme ensures that every bit period has a transition in the middle, making it easy for the receiver to synchronize 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 Usec 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.



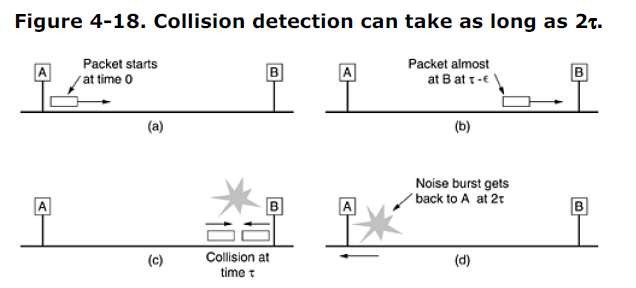
The frame contains two addresses, one for the destination and one for the source. The standard allows 2-byte and 6-byte addresses, but the parameters defined for the 10-Mbps baseband standard use only the 6-byte addresses. The high-order bit of the destination address is a 0 for ordinary addresses and 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 accepted by 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 centrally 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 x 1013 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. Multiple network-layer protocols may be 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 come the data, up to 1500 bytes. This limit was chosen somewhat arbitrarily at the time the DIX standard was cast in 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 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 . Just before the frame gets to the other end (i.e., at time -), 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, 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. 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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 Usec in the worst case, including the time to pass through the 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 than 64 bytes are padded out to 64 bytes with 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 the addition of 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 that 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 before 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 into discrete slots whose length is equal to the worst-case round-trip propagation time on the ether (2). To accommodate the longest path allowed by Ethernet, the slot time has been set to 512 bit times, or 51.2 Usec 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 23 - 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.

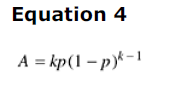
This algorithm, called **binary exponential backoff**, was chosen to dynamically adapt to the number of stations trying to send. If the randomization interval for all collisions was 1023, the chance of two stations colliding for a second time would be negligible, but the average wait after a collision 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 1 and the remaining station picked 0. 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. Truncating the backoff at 1023 keeps the bound 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.

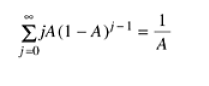
**4.3.5 Ethernet Performance**

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,

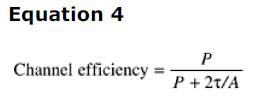


*A* is maximized when *p* = 1*/k*, with *A* 1*/e* as *k .* The probability that the contention interval

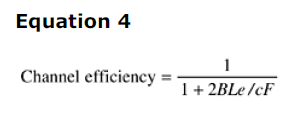
has exactly *j* slots in it is *A*(1 - *A*)*j* - 1, so the mean number of slots per contention is given by



Since each slot has a duration 2, the mean contention interval, *w*, is 2*/A.* Assuming optimal *p*, the mean number of contention slots is never more than *e*, so *w* is at most 2*e* 5.4. If the mean frame takes *P* sec to transmit, when many stations have frames to send,

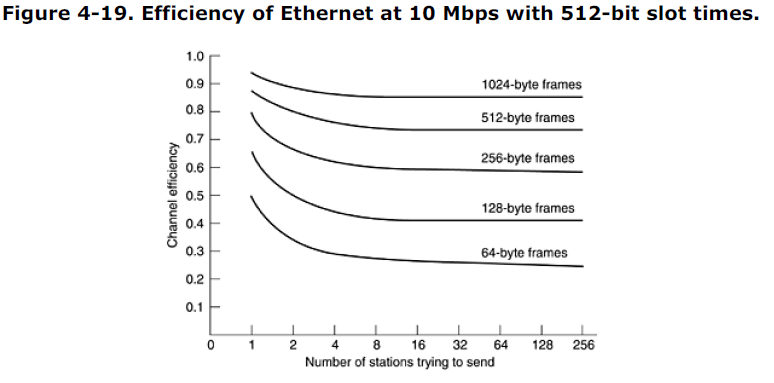


Here we see where the maximum cable distance between any two stations enters into the performance figures, giving rise to topologies other than that of Fig. 4-15(a). The longer the cable, the longer the contention interval. This observation is why the Ethernet standard specifies a maximum cable length. 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



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.

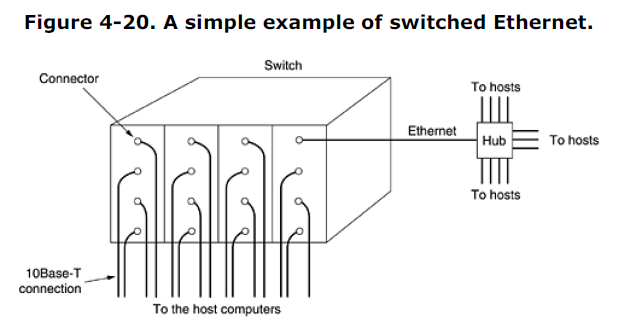
In Fig. 4-19, the channel efficiency is plotted versus number of ready stations for 2=51.2 Usec and a data rate of 10 Mbps, using Eq. (4-7). With a 64-byte slot time, it is not surprising that 64-byte frames are not efficient. On the other hand, with 1024-byte frames and an asymptotic value of *e* 64-byte 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 frames/sec, then when the system is in state *k*, the total input rate of all unblocked stations combined is *k*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 twisted pair connection to a single host computer.



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 only one station per collision domain, collisions are impossible and performance is improved. 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 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

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, they 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 usersof 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, various industry groups proposed two new ring-based optical LANs. One was called **FDDI** (**Fiber Distributed Data Interface**) and the

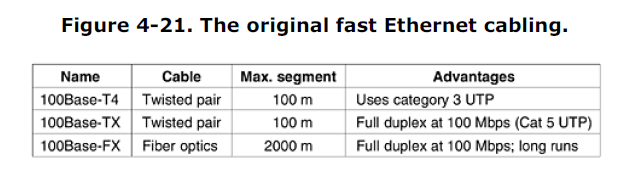
other was called **Fibre Channel** [ ]. To make a long story short, while both were used as backbone networks, neither one made the breakthrough to the desktop. In both cases, the station management was too complicated, which led to complex chips and high prices. The lesson that should have been learned here was KISS (Keep It Simple, Stupid). It is called ''fibre channel'' and not ''fiber channel'' because the document editor was British. 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 802.3 committee decided to go with a souped-up Ethernet for three primary reasons:

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 practically 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 either 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 being 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 farther. 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.



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). However, 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. Straight binary coding is not used; instead a scheme called used**4B/5B**is 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 algorithm, apply, so the system works just like old-fashioned Ethernet. In particular, only one station at a time can be transmitting. In other words, hubs require half-duplex communication. 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.

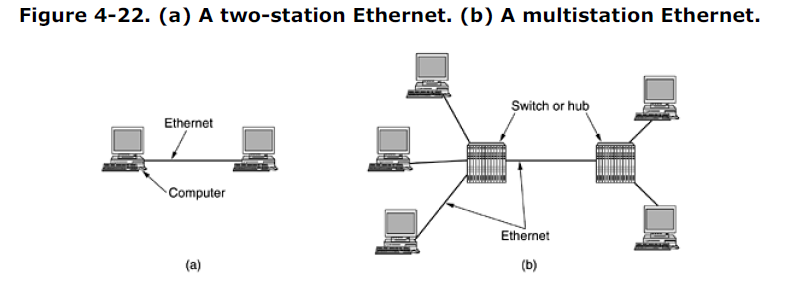
As a final note, virtually all switches can handle a mix of 10-Mbps and 100-Mbps stations, to make upgrading easier. As a site acquires more and more 100-Mbps workstations, all it has to do is buy the necessary number of new line cards and insert them into the switch. In fact, the standard itself provides 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 going to be the end of the line unless somebody quickly invents a new letter after z. 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 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 individual Ethernet cable has exactly two devices on it, no more and no fewer.



Gigabit Ethernet supports two different 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 no contention is possible, the CSMA/CD protocol is not used, so the maximum length of the cable is determined by signal strength issues rather than by 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 just as in fast Ethernet.

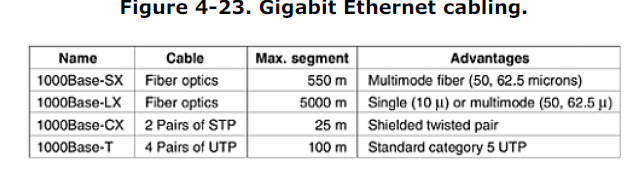
The other mode of operation, half-duplex, is used when the computers are connected to a hub rather than a switch. A hub does not buffer incoming frames. Instead, it electrically connects all the lines internally, simulating the multidrop cable used in classic Ethernet. In this mode, collisions are possible, 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 enough frames are 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 slash the performance of the new system is 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.



Three fiber diameters are permitted: 10, 50, and 62.5 microns. The first is for single mode and the last 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 its 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.

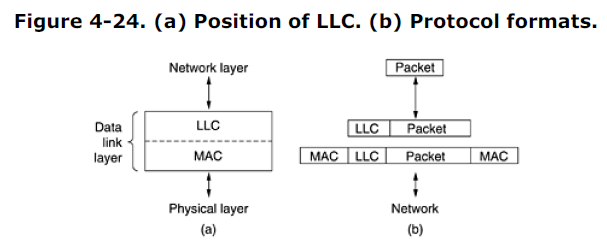
These choices were made to keep enough transitions in the stream to make sure the receiver stays in sync with the sender and also to keep the number of 0s and 1s on the fiber as close to equal as possible. In addition, many input bytes have two possible codewords assigned to them. When the encoder has 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 does not 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 approved by IEEE in 2002 as 802.3ae. 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 acknowledgements) and flow control (using a sliding window). In contrast, in this chapter, we have not said a word about reliable communication. All that Ethernet and the other 802 protocols offer is a best-efforts 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.



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, replacing the DIX *Type* field. The control field contains sequence and acknowledgement numbers, very much in the style of HDLC (see Fig. 3-24), 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-efforts 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 in sight, so it is likely to be around for many years to come. Few CPU architectures, operating systems, or programming languages have been king of the mountain for two decades going on three. Clearly, Ethernet did something right. What? Probably the main reason for its longevity 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

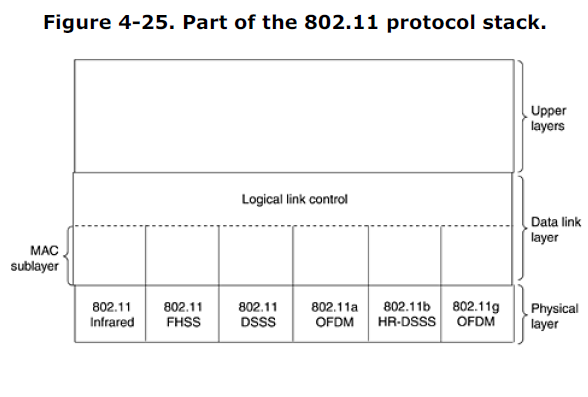
connectors, failures became extremely rare. People hesitate to replace something that works perfectly all the time, especially when they know that an awful lot of things in the computer industry work very poorly, so that many so-called ''upgrades'' are appreciably worse than what they replaced. Simple 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 there are 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 Ethernet, 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 introduced, but these changes have not required changing the software. When a network 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 except for ATM's use deep within the core of the telephone system.

**4.4 Wireless LANs**

Although Ethernet is widely used, it is about to get some competition. Wireless LANs are increasingly popular, and more and more office buildings, 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 them indistinguishable as far as the network layer is concerned. We studied the LLC when examining Ethernet earlier in this chapter and will not repeat that material here.

The1997 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. Cordless telephones and microwave ovens also use this band. All of these techniques operate at1 or 2 Mbps and at low enough power that they do not conflict too much. 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. In 2001, a second OFDM modulation was introduced, but in a different frequency band from the first one. 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 codeword containing fifteen 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**) uses 79 channels, each 1-MHz wide, starting at the low end of the 2.4-GHz ISM band. A pseudorandom number generator is used to produce the sequence of frequencies hopped to. 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.

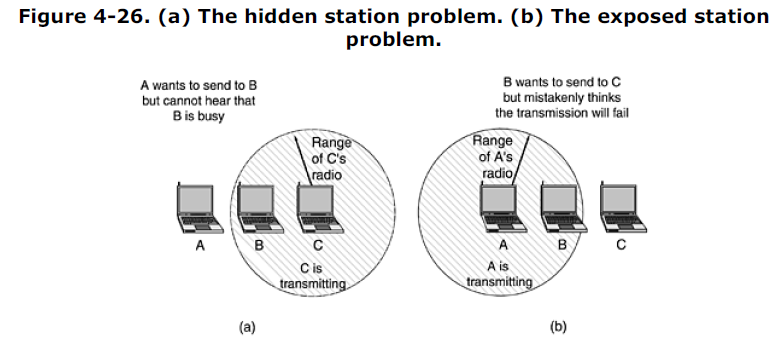
The amount of time spent at each frequency, the **dwell time**, is an adjustable parameter, but must be less than 400 msec. FHSS' randomization provides a fair way to allocate spectrum in the unregulated ISM band. It also provides a modicum of security since an intruder who does not know the hopping sequence or dwell time cannot eavesdrop on transmissions. Over longer distances, multipath fading can be an issue, and FHSS offers good resistance to it. It is also relatively insensitive to radio interference, which makes it popular for building-to-building links. Its main disadvantage is its low bandwidth. 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. For years, the FCC required all wireless communications equipment operating in the ISM bands in the U.S. to use spread spectrum, but in May 2002, that rule was dropped as new technologies emerged. The first of the high-speed wireless 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. Since transmissions are present on multiple frequencies at the same time, this technique is considered a form of spread spectrum, but different from both CDMA and FHSS. Splitting the signal into many narrow bands has some key advantages over using a single wide band, including better immunity to narrowband interference and the possibility of using noncontiguous bands. A complex encoding system is used, based on phase-shift modulation for speeds up to 18 Mbps and on 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 (Doufexi et al., 2002). The technique has a good spectrum efficiency in terms of bits/Hz and good immunity to multipath fading.

Next, 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. In fact, its standard was approved first and it got to market first. Data rates supported by 802.11b 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. In practice, the operating speed of 802.11b is nearly always 11 Mbps. 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 November 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 speed 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 (not to mention three low-speed wireless LANs). One can legitimately ask if this is a good thing for a standards committee to do. Maybe three was their lucky number.

**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 earlier and illustrated again in Fig. 4- 26(a). Since not all stations are within radio range of each other, transmissions going on in one part of a cell may not be 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*.

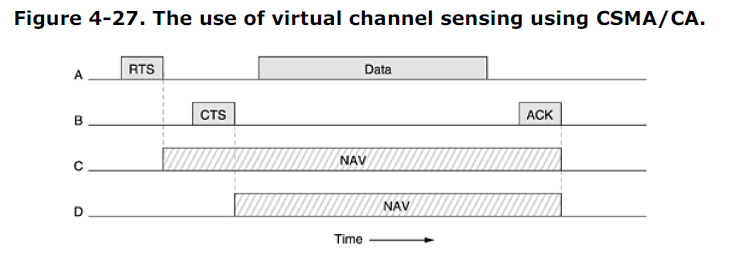


In addition, there is the inverse problem, the exposed station problem, illustrated in Fig. 4-26(b). Here *B* wants to send to *C* so it listens to the channel. When it hears a transmission, it falsely concludes that it may not send to *C*, even though *A* may be transmitting to *D* (not shown). In addition, 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, called **DCF** (**Distributed Coordination Function**), does not use any kind of central control (in that respect, similar toEthernet). The other, called **PCF** (**Point Coordination Function**), uses the base station to control allactivity in its cell. All implementations must support DCF but PCF is optional. We will now discuss thesetwo 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 ofoperation are supported by CSMA/CA. In the first method, when a station wants to transmit, it sensesthe channel. If it is idle, it just starts transmitting. It does not sense the channel while transmitting butemits its entire frame, which may well be destroyed at the receiver due to interference there. If thechannel is busy, the sender defers until it goes idle and then starts transmitting. If a collision occurs,

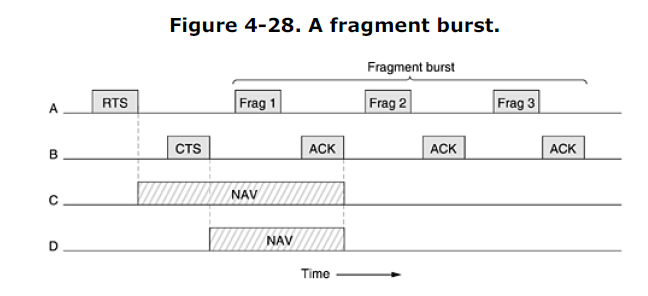
the colliding stations wait a random time, using the Ethernet binary exponential backoff algorithm, andthen try again later. The other mode of CSMA/CA operation is based on MACAW and uses virtual channel sensing, as illustrated in Fig. 4-27. In this example, *A* wants to send to *B*. C is a station within 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*.



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 grantpermission, in which case it sends a CTS frame back. Upon receipt of the CTS, *A* now sends its frameand 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 protocolis 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

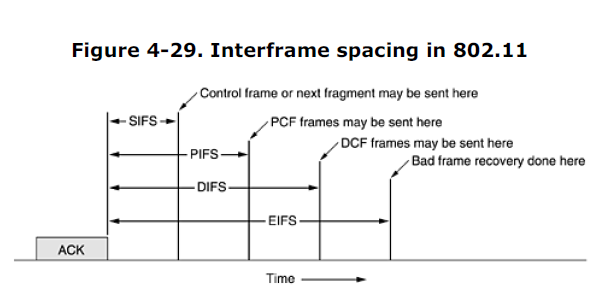
a certain period of time. In contrast to wired networks, wireless networks are noisy and unreliable, in no small part due to microwave ovens, which also use the unlicensed ISM bands. As a consequence, the probability of a frame making it through successfully decreases with frame length. If the probability 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. In summary, if a frame is too long, it has very little chance of getting through undamaged and will probably have to be retransmitted. 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, multiple fragments can be sent in a row, as shown in Fig. 4-28. sequence of fragments is called 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 acknowledgement, 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 sent, 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.



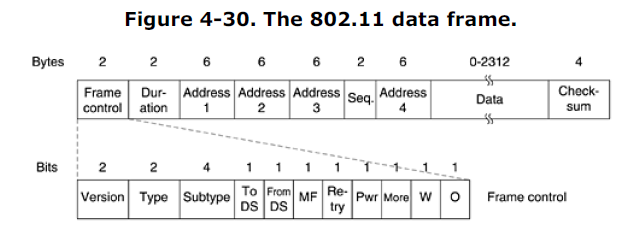
The shortest interval is **SIFS** (**Short InterFrame Spacing**). It is used to allow the parties in a single dialog the chance to go first. This includes letting the receiver send a CTS to respond to an RTS, letting the receiver send an ACK for a fragment or full data frame, and letting the sender of 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 different classes of frames on the wire: 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 techniques used, so we will not discuss them 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 into 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 format similar to that of data frames, except without one of the base station

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 it is used correctly, no data will be lost as a consequence of the handover. (But 802.11, like Ethernet, is just a best-efforts service.)

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 (i.e., relate to actions within a single cell). 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.** For information sent over a wireless LAN to be kept 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 naturally 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 some parameters that can be inspected and, in some cases, adjusted. They relate to encryption, timeout intervals, data rates, beacon frequency, and so on. Wireless LANs based on 802.11 are starting to be deployed in office buildings, airports, hotels, restaurants, and campuses around the world. Rapid growth is expected. For some experience about the widespread deployment of 802.11 at CMU, see (Hills, 2001).

**4.5 Broadband Wireless**

We have been indoors too long. Let us now go outside and see if any interesting networking is going on there. It turns out that quite a bit is going on there, and some of it has to do with the so-called last mile. With the deregulation of the telephone system in many countries, competitors to the entrenched telephone company are now often allowed to offer local voice and high-speed Internet service. There is certainly plenty of demand. The problem is that running fiber, coax, or even category 5 twisted pair to millions of homes and businesses is prohibitively expensive. What is a competitor to do? The answer is broadband wireless. Erecting a big antenna on a hill just outside of town and installing

antennas directed at it on customers' roofs is much easier and cheaper than digging trenches and stringing cables. Thus, competing telecommunication companies have a great interest in providing a multimegabit wireless communication service for voice, Internet, movies on demand, etc. As we saw in Fig. 2-30, LMDS was invented for this purpose. However, until recently, every carrier devised its own system. This lack of standards meant that hardware and software could not be mass produced, which kept prices high and acceptance low. Many people in the industry realized that having a broadband wireless standard was the key elementmissing, so IEEE was asked to form a committee composed of people from key companies and academia to draw up the standard. The next number available in the 802 numbering space was **802.16**, so the standard got this number. Work was started in July 1999, and the final standard was

approved in April 2002. Officially the standard is called ''Air Interface for Fixed Broadband Wireless Access Systems.'' However, some people prefer to call it a **wireless MAN (Metropolitan Area** **Network)** or a **wireless local loop**. We regard all these terms as interchangeable. Like some of the other 802 standards, 802.16 was heavily influenced by the OSI model, including the (sub)layers, terminology, service primitives, and more. Unfortunately, also like OSI, it is fairly complicated. In the following sections we will give a brief description of some of the highlights of 802.16, but this treatment is far from complete and leaves out many details. For additional information about broadband wireless in general, see (Bolcskei et al., 2001; and Webb, 2001). For information about 802.16 in particular, see (Eklund et al., 2002).

**4.5.1 Comparison of 802.11 with 802.16**

At this point you may be thinking: Why de vise a new standard? Why not just use 802.11? There are some very good reasons for not using 802.11, primarily because 802.11 and 802.16 solve different problems. Before getting into the technology of 802.16, it is probably worthwhile saying a few words about why a new standard is needed at all. The environments in which 802.11 and 802.16 operate are similar in some ways, primarily in that they were designed to provide high-bandwidth wireless communications. But they also differ in some major ways. To start with, 802.16 provides service to buildings, and buildings are not mobile. They do not migrate from cell to cell often. Much of 802.11 deals with mobility, and none of that is relevant here. Next, buildings can have more than one computer in them, a complication that does not occur when the end station is a single notebook computer. Because building owners are generally willing to spend much more money for communication gear than are notebook owners, better radios are available. This difference means that 802.16 can use full-duplex communication, something 802.11 avoids to keep the cost of the radios low. Because 802.16 runs over part of a city, the distances involved can be several kilometers, which

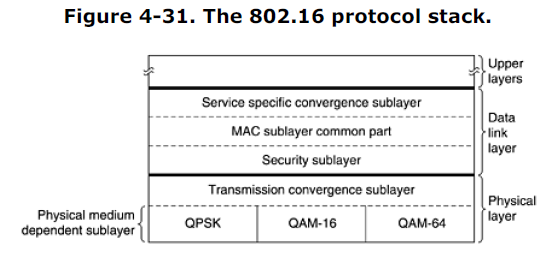
means that the perceived power at the base station can vary widely from station to station. This variation affects the signal-to-noise ratio, which, in, turn, dictates multiple modulation schemes. Also, open communication over a city means that security and privacy are essential and mandatory. Furthermore, each cell is likely to have many more users than will a typical 802.11 cell, and these users are expected to use more bandwidth than will a typical 802.11 user. After all it is rare for a company to invite 50 employees to show up in a room with their laptops to see if they can saturate the 802.11 wireless network by watching 50 separate movies at once. For this reason, more spectrum is

needed than the ISM bands can provide, forcing 802.16 to operate in the much higher 10-to-66 GHz frequency range, the only place unused spectrum is still available.But these millimeter waves have different physical properties than the longer waves in the ISM bands, which in turn requires a completely different physical layer. One property that millimeter waves have is that they are strongly absorbed by water (especially rain, but to some extent also by snow, hail, and with a bit of bad luck, heavy fog). Consequently, error handling is more important than in an indoor environment. Millimeter waves can be focused into directional beams (802.11 is omnidirectional), so choices made in 802.11 relating to multipath propagation are moot here. Another issue is quality of service. While 802.11 provides some support for real-time traffic (using PCF

mode), it was not really designed for telephony and heavy-duty multimedia usage. In contrast, 802.16 is expected to support these applications completely because it is intended for residential as well as business use. In short, 802.11 was designed to be mobile Ethernet, whereas 802.16 was designed to be wireless, but stationary, cable television. These differences are so big that the resulting standards are very different as they try to optimize different things. A very brief comparison with the cellular phone system is also worthwhile. With mobile phones, we are talking about narrow-band, voice-oriented, low-powered, mobile stations that communicate using medium-length microwaves. Nobody watches high-resolution, two-hour movies on GSM mobile phones (yet). Even UMTS has little hope of changing this situation. In short, the wireless MAN world is far more demanding than is the mobile phone world, so a completely different system is needed. Whether 802.16 could be used for mobile devices in the future is an interesting question. It was not optimized for them, but the possibility is there. For the moment it is focused on fixed wireless.

**4.5.2 The 802.16 Protocol Stack**

The 802.16 protocol stack is illustrated in Fig. 4-31. The general structure is similar to that of the other 802 networks, but with more sublayers. The bottom sublayer deals with transmission. Traditional narrow-band radio is used with conventional modulation schemes. Above the physical transmission layer comes a convergence sublayer to hide the different technologies from the data link layer. Actually, 802.11 has something like this too, only the committee chose not to formalize with an OSItype name.



Although we have not shown them in the figure, work is already underway to add two new physical layer protocols. The 802.16a standard will support OFDM in the 2-to-11 GHz frequency range. The 802.16b standard will operate in the 5-GHz ISM band. Both of these are attempts to move closer to 802.11. The data link layer consists of three sublayers. The bottom one deals with privacy and security, which is far more crucial for public outdoor networks than for private indoor networks. It manages encryption, decryption, and key management. Next comes the MAC sublayer common part. This is where the main protocols, such as channel management, are located. The model is that the base station controls the system. It can schedule the downstream (i.e., base to subscriber) channels very efficiently and plays a major role in managing the upstream

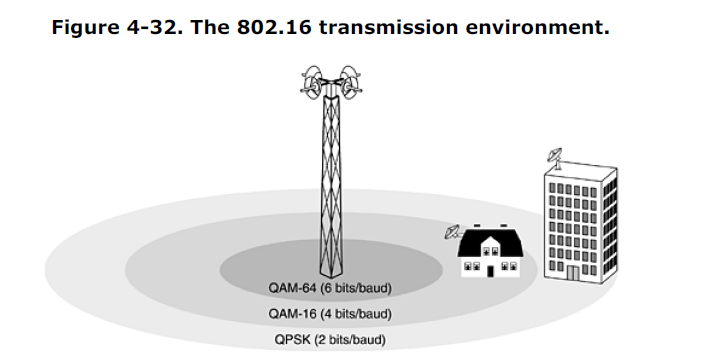
(i.e., subscriber to base) channels as well. An unusual feature of the MAC sublayer is that, unlike those of the other 802 networks, it is completely connection oriented, in order to provide quality-of-service guarantees for telephony and multimedia communication. The service-specific convergence sublayer takes the place of the logical link sublayer in the other 802 protocols. Its function is to interface to the network layer. A complication here is that 802.16 was designed to integrate seamlessly with both datagram protocols (e.g., PPP, IP, and Ethernet) and ATM. The problem is that packet protocols are connectionless and ATM is connection oriented. This means that every ATM connection has to map onto an 802.16 connection, in principle a straightforward matter. But onto which 802.16 connection should an incoming IP packet be mapped? That problem is

dealt with in this sublayer.

**4.5.3 The 802.16 Physical Layer**

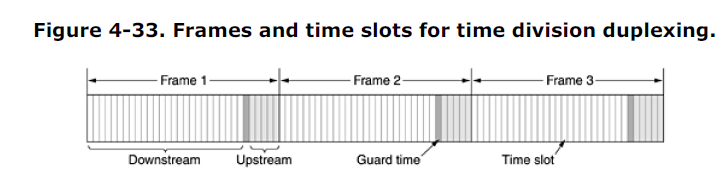
As mentioned above, broadband wireless needs a lot of spectrum, and the only place to find it is in the 10-to-66 GHz range. These millimeter waves have an interesting property that longer microwaves do not: they travel in straight lines, unlike sound but similar to light. As a consequence, the base station can have multiple antennas, each pointing at a different sector of the surrounding terrain, as shown in Fig. 4-32. Each sector has its own users and is fairly independent of the adjoining ones, something not true of cellular radio, which is omnidirectional.

Because signal strength in the millimeter band falls off sharply with distance from the base station, the signal-to-noise ratio also drops with distance from the base station. For this reason, 802.16 employs three different modulation schemes, depending on how far the subscriber station is from the base station. For close-in subscribers, QAM-64 is used, with 6 bits/baud. For medium-distance subscribers,m QAM-16 is used, with 4 bits/baud. For distant subscribers, QPSK is used, with 2 bits/baud. For example, for a typical value of 25 MHz worth of spectrum, QAM-64 gives 150 Mbps, QAM-16 gives 100 Mbps, and QPSK gives 50 Mbps. In other words, the farther the subscriber is from the base station, the lower the data rate (similar to what we saw with ADSL in Fig. 2-27).



The constellation diagrams for these three modulation techniques were shown in Fig. 2-25. Given the goal of producing a broadband system, and subject to the above physical constraints, the 802.16 designers worked hard to use the available spectrum efficiently. One thing they did not like was the way GSM and DAMPS work. Both of those use different but equal frequency bands for upstream and downstream traffic. For voice, traffic is probably symmetric for the most part, but for Internet access, there is often more downstream traffic than upstream traffic. Consequently, 802.16 provides a more flexible way to allocate the bandwidth. Two schemes are used, **FDD** (**Frequency Division**

**Duplexing**) and **TDD** (**Time Division Duplexing**). The latter is illustrated in Fig. 4-33. Here the base station periodically sends out frames. Each frame contains time slots. The first ones are for downstream traffic. Then comes a guard time used by the stations to switch direction. Finally, we have slots for upstream traffic. The number of time slots devoted to each direction can be changed dynamically to match the bandwidth in each direction to the traffic.



Downstream traffic is mapped onto time slots by the base station. The base station is completely in control for this direction. Upstream traffic is more complex and depends on the quality of service required. We will come to slot allocation when we discuss the MAC sublayer below. Another interesting feature of the physical layer is its ability to pack multiple MAC frames back-to back in a single physical transmission. The feature enhances spectral efficiency by reducing the number of preambles and physical layer headers needed. Also noteworthy is the use of Hamming codes to do forward error correction in the physical layer. Nearly all other networks simply rely on checksums to detect errors and request retransmission when frames are received in error. But in the wide area broadband environment, so many transmission errors are expected that error correction is employed in the physical layer, in addition to checksums in the higher layers. The net effect of the error correction is to make the channel look better than it really is (in the same way that CD-ROMs appear to be very reliable, but only because more than half the total bits are devoted to error correction in the physical layer).

**4.5.4 The 802.16 MAC Sublayer Protocol**

The data link layer is divided into three sublayers, as we saw in Fig. 4-31. Since we will not study cryptography until Chap. 8, it is difficult to explain now how the security sublayer works. Suffice it to say that encryption is used to keep secret all data transmitted. Only the frame payloads are encrypted; the headers are not. This property means that a snooper can see who is talking to whom but cannot tell what they are saying to each other. If you already know something about cryptography, here comes a one-paragraph explanation of the security sublayer. If you know nothing about cryptography, you are not likely to find the next paragraph terribly enlightening (but you might consider rereading it after finishing Chap. 8). At the time a subscriber connects to a base station, they perform mutual authentication with RSA public-key cryptography using X.509 certificates. The payloads themselves are encrypted using a symmetric-key system, either DES with cipher block chaining or triple DES with two keys. AES (Rijndael) is likely to be added soon. Integrity checking uses SHA-1. Now that was not so bad, was it?

Let us now look at the MAC sublayer common part. MAC frames occupy an integral number of physical layer time slots. Each frame is composed of sub-frames, the first two of which are the downstream and upstream maps. These maps tell what is in which time slot and which time slots are free. The downstream map also contains various system parameters to inform new stations as they come online. The downstream channel is fairly straightforward. The base station simply decides what to put in which subframe. The upstream channel is more complicated since there are competing uncoordinated subscribers that need access to it. Its allocation is tied closely to the quality-of-service issue. Four classes of service are defined as follows:

1. Constant bit rate service.

2. Real-time variable bit rate service.

3. Non-real-time variable bit rate service.

4. Best-efforts service.

All service in 802.16 is connection-oriented, and each connection gets one of the above classes of service, determined when the connection is set up. This design is very different from that of 802.11 or Ethernet, which have no connections in the MAC sublayer.

Constant bit rate service is intended for transmitting uncompressed voice such as on a T1 channel. This service needs to send a predetermined amount of data at predetermined time intervals. It is accommodated by dedicating certain time slots to each connection of this type. Once the bandwidth has been allocated, the time slots are available automatically, without the need to ask for each one. Real-time variable bit rate service is for compressed multimedia and other soft real-time applications in which the amount of bandwidth needed each instant may vary. It is accommodated by the base station

polling the subscriber at a fixed interval to ask how much bandwidth is needed this time.

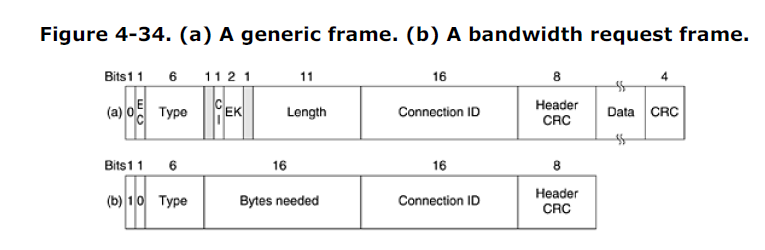
Non-real-time variable bit rate service is for heavy transmissions that are not real time, such as large file transfers. For this service the base station polls the subscriber often, but not at rigidly-prescribed time intervals. A constant bit rate customer can set a bit in one of its frames requesting a poll in order to send additional (variable bit rate) traffic.

If a station does not respond to a poll *k* times in a row, the base station puts it into a multicast group and takes away its personal poll. Instead, when the multicast group is polled, any of the stations in it can respond, contending for service. In this way, stations with little traffic do not waste valuable polls. Finally, best-efforts service is for everything else. No polling is done and the subscriber must contend for bandwidth with other best-efforts subscribers. Requests for bandwidth are done in time slots marked in the upstream map as available for contention. If a request is successful, its success will be noted in the next downstream map. If it is not successful, unsuccessful subscribers have to try again later. To minimize collisions, the Ethernet binary exponential backoff algorithm is used. The standard defines two forms of bandwidth allocation: per station and per connection. In the former case, the subscriber station aggregates the needs of all the users in the building and makes collective requests for them. When it is granted bandwidth, it doles out that bandwidth to its users as it sees fit.

In the latter case, the base station manages each connection directly.

**4.5.5 The 802.16 Frame Structure**

All MAC frames begin with a generic header. The header is followed by an optional payload and an optional checksum (CRC), as illustrated in Fig. 4-34. The payload is not needed in control frames, for example, those requesting channel slots. The checksum is (surprisingly) also optional dueto the error correction in the physical layer and the fact that no attempt is ever made to retransmit real-time frames. If no retransmissions will be attempted, why even bother with a checksum?



A quick rundown of the header fields of Fig. 4-34(a) is as follows. The *EC* bit tells whether the payload is encrypted. The *Type* field identifies the frame type, mostly telling whether packing and fragmentation are present. The *CI* field indicates the presence or absence of the final checksum. The *EK* field tells which of the encryption keys is being used (if any). The *Length* field gives the complete length of the frame, including the header. The *Connection identifier* tells which connection this frame belongs to. Finally, the *HeaderCRC* field is a checksum over the header only, using the polynomial *x*8 + *x*2 + *x* + 1. A second header type, for frames that request bandwidth, is shown in Fig. 4-34(b). It starts with a 1 bit instead of a 0 bit and is similar to the generic header except that the second and third bytes form a 16-bit number telling how much bandwidth is needed to carry the specified number of bytes. Bandwidth request frames do not carry a payload or full-frame CRC. A great deal more could be said about 802.16, but this is not the place to say it. For more information, please consult the standard itself.

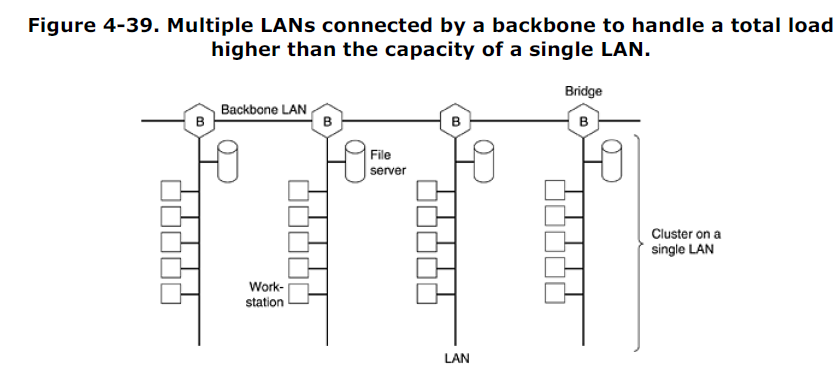
**4.7 Data Link Layer Switching**

Many organizations have multiple LANs and wish to connect them. LANs can be connected by devices called **bridges**, which operate in the data link layer. Bridges examine the data layer link addresses to do routing. Since they are not supposed to examine the payload field of the frames they route, they can transport IPv4 (used in the Internet now), IPv6 (will be used in the Internet in the future), AppleTalk, ATM, OSI, or any other kinds of packets. In contrast, *routers* examine the addresses in packets and route based on them. Although this seems like a clear division between bridges and routers, some modern developments, such as the advent of switched Ethernet, have muddied the waters, as we will see later. In the following sections we will look at bridges and switches, especially for connecting different 802 LANs. For a comprehensive treatment of bridges, switches, and related topics, see (Perlman, 2000). Before getting into the technology of bridges, it is worthwhile taking a look at some common situations

in which bridges are used. We will mention six reasons why a single organization may end up with multiple LANs.

First, many university and corporate departments have their own LANs, primarily to connect their own personal computers, workstations, and servers. Since the goals of the various departments differ, different departments choose different LANs, without regard to what other departments are doing. Sooner or later, there is a need for interaction, so bridges are needed. In this example, multiple LANs came into existence due to the autonomy of their owners. Second, the organization may be geographically spread over several buildings separated by considerable distances. It may be cheaper to have separate LANs in each building and connect them with bridges and laser links than to run a single cable over the entire site. Third, it may be necessary to split what is logically a single LAN into separate LANs to accommodate the load. At many universities, for example, thousands of workstations are available for student and faculty computing. Files are normally kept on file server machines and are downloaded to users' machines upon request. The enormous scale of this system precludes putting all the workstations on a single LAN—the total bandwidth needed is far too high. Instead, ultiple LANs connected by bridges are used, as shown in Fig. 4-39.

Each LAN contains a cluster of workstations with its own file server so that most traffic is restricted to a single LAN and does not add load to the backbone.



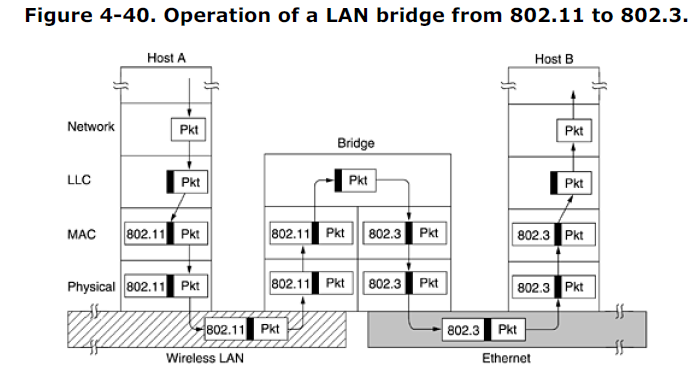
It is worth noting that although we usually draw LANs as multidrop cables as in Fig. 4-39 (the classic look), they are more often implemented with hubs or especially switches nowadays. However, a long multidrop cable with multiple machines plugged into it and a hub with the machines connected inside the hub are functionally identical. In both cases, all the machines belong to the same collision domain, and all use the CSMA/CD protocol to send frames. Switched LANs are different, however, as we saw before and will see again shortly. Fourth, in some situations, a single LAN would be adequate in terms of the load, but the physical distance between the most distant machines is too great (e.g., more than 2.5 km for Ethernet).

Even if laying the cable is easy to do, the network would not work due to the excessively long round-trip delay. The only solution is to partition the LAN and install bridges between the segments. Using bridges, the total physical distance covered can be increased. Fifth, there is the matter of reliability. On a single LAN, a defective node that keeps outputting a continuous stream of garbage can cripple the LAN. Bridges can be inserted at critical places, like fire doors in a building, to prevent a single node that has gone berserk from bringing down the entire system. Unlike a repeater, which just copies whatever it sees, a bridge can be programmed to exercise some discretion about what it forwards and what it does not forward. Sixth, and last, bridges can contribute to the organization's security. Most LAN interfaces have a **promiscuous mode**, in which *all* frames are given to the computer, not just those addressed to it. Spies and busybodies love this feature.

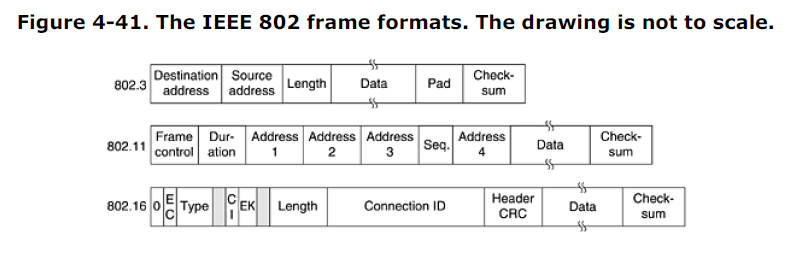
By inserting bridges at various places and being careful not to forward sensitive traffic, a system administrator can isolate parts of the network so that its traffic cannot escape and fall into the wrong hands. Ideally, bridges should be fully transparent, meaning it should be possible to move a machine from one cable segment to another without changing any hardware, software, or configuration tables. Also, it should be possible for machines on any segment to communicate with machines on any other segment without regard to the types of LANs being used on the two segments or on segments in between them. This goal is sometimes achieved, but not always.

**4.7.1 Bridges from 802.x to 802.y**

Having seen why bridges are needed, let us now turn to the question of how they work. Figure 4-40 illustrates the operation of a simple two-port bridge. Host *A* on a wireless (802.11) LAN has a packet to send to a fixed host, *B*, on an (802.3) Ethernet to which the wireless LAN is connected. The packet descends into the LLC sublayer and acquires an LLC header (shown in black in the figure). Then it passes into the MAC sublayer and an 802.11 header is prepended to it (also a trailer, not shown in the figure). This unit goes out over the air and is picked up by the base station, which sees that it needs to go to the fixed Ethernet. When it hits the bridge connecting the 802.11 network to the 802.3 network, 239 it starts in the physical layer and works its way upward. In the MAC sublayer in the bridge, the 802.11 header is stripped off. The bare packet (with LLC header) is then handed off to the LLC sublayer in the bridge. In this example, the packet is destined for an 802.3 LAN, so it works its way down the 802.3 side of the bridge and off it goes on the Ethernet. Note that a bridge connecting *k* different LANs will have *k* different MAC sublayers and *k* different physical layers, one for each type.



So far it looks like moving a frame from one LAN to another is easy. Such is not the case. In this section we will point out some of the difficulties that one encounters when trying to build a bridge between the various 802 LANs (and MANs). We will focus on 802.3, 802.11, and 802.16, but there are others as well, each with its unique problems. To start with, each of the LANs uses a different frame format (see Fig. 4-41). Unlike the differences between Ethernet, token bus, and token ring, which were due to history and big corporate egos, here the differences are to some extent legitimate. For example, the *Duration* field in 802.11 is there due to the MACAW protocol and makes no sense in Ethernet. As a result, any copying between different LANs requires reformatting, which takes CPU time, requires a new checksum calculation, and introduces the possibility of undetected errors due to bad bits in the bridge's memory.



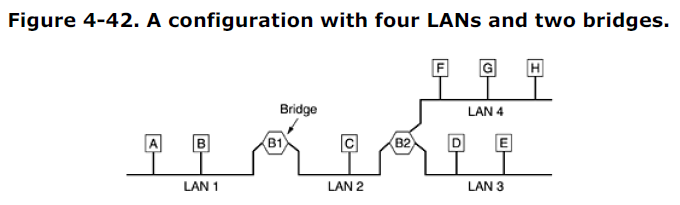
A second problem is that interconnected LANs do not necessarily run at the same data rate. When forwarding a long run of back-to-back frames from a fast LAN to a slower one, the bridge will not be able to get rid of the frames as fast as they come in. For example, if a gigabit Ethernet is pouring bits into an 11-Mbps 802.11b LAN at top speed, the bridge will have to buffer them, hoping not to run out of memory. Bridges that connect three or more LANs have a similar problem when several LANs are trying to feed the same output LAN at the same time even if all the LANs run at the same speed. A third problem, and potentially the most serious of all, is that different 802 LANs have different maximum frame lengths. An obvious problem arises when a long frame must be forwarded onto a LAN that cannot accept it. Splitting the frame into pieces is out of the question in this layer. All the protocols assume that frames either arrive or they do not. There is no provision for reassembling frames out of smaller units. This is not to say that such protocols could not be devised.

They could be and have been. It is just that no data link protocols provide this feature, so bridges must keep their hands off the frame payload. Basically, there is no solution. Frames that are too large to be forwarded must be discarded. So much for transparency. Another point is security. Both 802.11 and 802.16 support encryption in the data link layer. Ethernet does not. This means that the various encryption services available to the wireless networks are lost when traffic passes over an Ethernet. Worse yet, if a wireless station uses data link layer encryption, there will be no way to decrypt it when it arrives over an Ethernet. If the wireless station does not use encryption, its traffic will be exposed over the air link. Either way there is a problem. One solution to the security problem is to do encryption in a higher layer, but then the 802.11 station has to know whether it is talking to another station on an 802.11 network (meaning use data link layer encryption) or not (meaning do not use it). Forcing the station to make a choice destroys transparency. A final point is quality of service. Both 802.11 and 802.16 provide it in various forms, the former using PCF mode and the latter using constant bit rate connections. Ethernet has no concept of quality of service, so traffic from either of the others will lose its quality of service when passing over an Ethernet.

**4.7.2 Local Internetworking**

The previous section dealt with the problems encountered in connecting two different IEEE 802 LANs via a single bridge. However, in large organizations with many LANs, just interconnecting them all raises a variety of issues, even if they are all just Ethernet. Ideally, it should be possible to go out and buy bridges designed to the IEEE standard, plug the connectors into the bridges, and everything should work perfectly, instantly. There should be no hardware changes required, no software changes required, no setting of address switches, no downloading of routing tables or parameters, nothing. Just plug in the cables and walk away. Furthermore, the operation of the existing LANs should not be affected by the bridges at all.

In other words, the bridges should be completely transparent (invisible to all the hardware and software). Surprisingly enough, this is actually possible. Let us now take a look at how this magic is accomplished. In its simplest form, a transparent bridge operates in promiscuous mode, accepting every frame transmitted on all the LANs to which it is attached. As an example, consider the configuration of Fig. 4-42. Bridge B1 is connected to LANs 1 and 2, and bridge B2 is connected to LANs 2, 3, and 4. A frame arriving at bridge B1 on LAN 1 destined for *A* can be discarded immediately, because it is already on the correct LAN, but a frame arriving on LAN 1 for *C* or *F* must be forwarded.



When a frame arrives, a bridge must decide whether to discard or forward it, and if the latter, on which LAN to put the frame. This decision is made by looking up the destination address in a big (hash) table inside the bridge. The table can list each possible destination and tell which output line (LAN) it belongs on. For example, B2's table would list *A* as belonging to LAN 2, since all B2 has to know is which LAN to put frames for *A* on. That, in fact, more forwarding happens later is not of interest to it. When the bridges are first plugged in, all the hash tables are empty. None of the bridges know where any of the destinations are, so they use a flooding algorithm: every incoming frame for an unknown destination is output on all the LANs to which the bridge is connected except the one it arrived on. As time goes on, the bridges learn where destinations are, as described below. Once a destination is known, frames destined for it are put on only the proper LAN and are not flooded.

The algorithm used by the transparent bridges is **backward learning**.As mentioned above, the bridges operate in promiscuous mode, so they see every frame sent on any of their LANs. By looking at the source address, they can tell which machine is accessible on which LAN. For example, if bridge B1 in Fig. 4-42 sees a frame on LAN 2 coming from *C*, it knows that *C* must be reachable via LAN 2, so it makes an entry in its hash table noting that frames going to *C* should use LAN 2. Any subsequent frame addressed to *C* coming in on LAN 1 will be forwarded, but a frame for *C* coming in on LAN 2 will be discarded. The topology can change as machines and bridges are powered up and down and moved around. To handle dynamic topologies, whenever a hash table entry is made, the arrival time of the frame is noted in the entry. Whenever a frame whose source is already in the table arrives, its entry is updated with the current time. Thus, the time associated with every entry tells the last time a frame from that machine was seen. Periodically, a process in the bridge scans the hash table and purges all entries more than a few minutes old.

In this way, if a computer is unplugged from its LAN, moved around the building, and plugged in again somewhere else, within a few minutes it will be back in normal operation, without any manual intervention. This algorithm also means that if a machine is quiet for a few minutes, any traffic sent to it will have to be flooded until it next sends a frame itself. The routing procedure for an incoming frame depends on the LAN it arrives on (the source LAN) and

the LAN its destination is on (the destination LAN), as follows:

1. If destination and source LANs are the same, discard the frame.

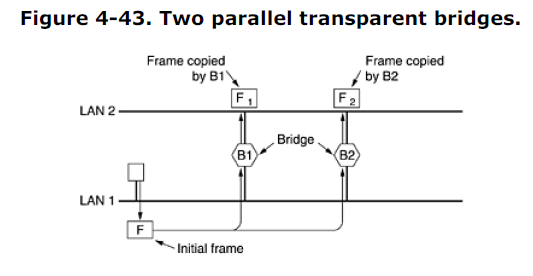
2. If the destination and source LANs are different, forward the frame.

3. If the destination LAN is unknown, use flooding.

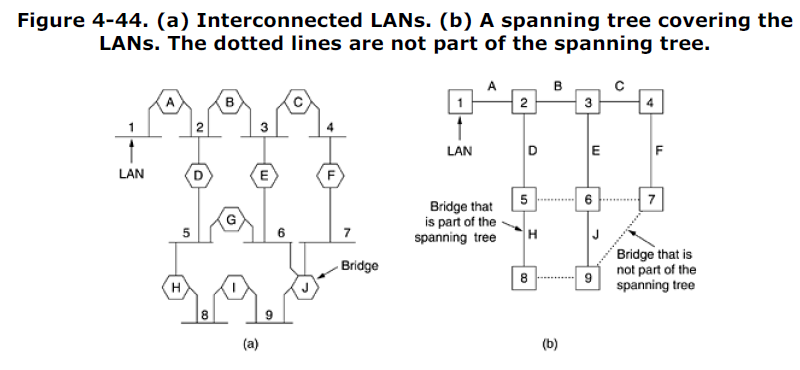
As each frame arrives, this algorithm must be applied. Special-purpose VLSI chips do the lookup and update the table entry, all in a few microseconds.

**4.7.3 Spanning Tree Bridges**

To increase reliability, some sites use two or more bridges in parallel between pairs of LANs, as shown in Fig. 4-43. This arrangement, however, also introduces some additional problems because it creates loops in the topology.

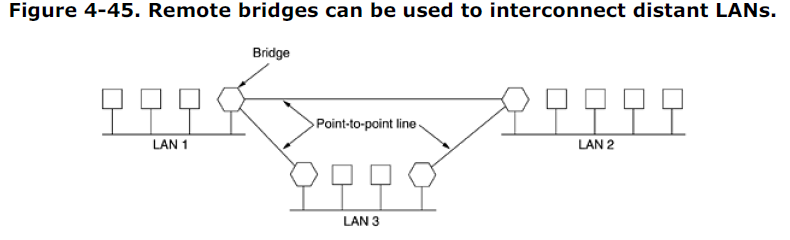


A simple example of these problems can be seen by observing how a frame, *F*, with unknown destination is handled in Fig. 4-43. Each bridge, following the normal rules for handling unknown destinations, uses flooding, which in this example just means copying it to LAN 2. Shortly thereafter, bridge 1 sees *F*2, a frame with an unknown destination, which it copies to LAN 1, generating *F*3 (not shown). Similarly, bridge 2 copies *F*1 to LAN 1 generating *F*4 (also not shown). Bridge 1 now forwards *F*4 and bridge 2 copies *F*3. This cycle goes on forever. The solution to this difficulty is for the bridges to ommunicate with each other and overlay the actual topology with a spanning tree that reaches every LAN. In effect, some potential connections between LANs are ignored in the interest of constructing a fictitious loop-free topology. For example, in Fig. 4-44(a) we see nine LANs interconnected by ten bridges. This configuration can be abstracted into a graph with the LANs as the nodes. An arc connects any two LANs that are connected by a bridge. The graph can be reduced to a spanning tree by dropping the arcs shown as dotted lines in Fig. 4-44(b). Using this spanning tree, there is exactly one path from every LAN to every other LAN. Once the bridges have agreed on the spanning tree, all forwarding between LANs follows the spanning tree. Since there is a unique path from each source to each destination, loops are impossible.

To build the spanning tree, first the bridges have to choose one bridge to be the root of the tree. They make this choice by having each one broadcast its serial number, installed by the manufacturer and guaranteed to be unique worldwide. The bridge with the lowest serial number becomes the root. Next, a tree of shortest paths from the root to every bridge and LAN is constructed. This tree is the spanning tree. If a bridge or LAN fails, a new one is computed. The result of this algorithm is that a unique path is established from every LAN to the root and thus to every other LAN. Although the tree spans all the LANs, not all the bridges are necessarily present in the tree (to prevent loops). Even after the spanning tree has been established, the algorithm continues to run during normal operation in order to automatically detect topology changes and update the tree. The distributed algorithm used for constructing the spanning tree was invented by Radia Perlman and is described in detail in (Perlman, 2000). It is standardized in IEEE 802.1D.

**4.7.4 Remote Bridges**

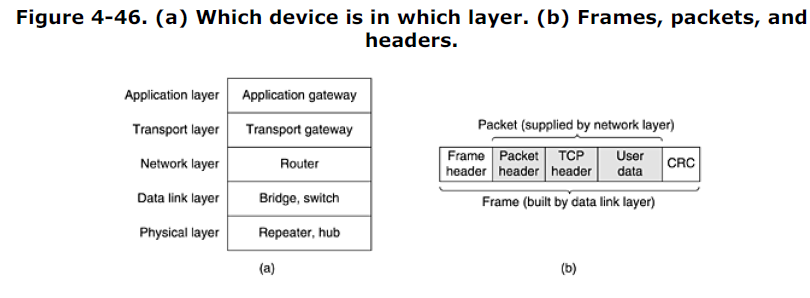
A common use of bridges is to connect two (or more) distant LANs. For example, a company might have plants in several cities, each with its own LAN. Ideally, all the LANs should be interconnected, so the complete system acts like one large LAN.This goal can be achieved by putting a bridge on each LAN and connecting the bridges pairwise with point-to-point lines (e.g., lines leased from a telephone company). A simple system, with three LANs, is illustrated in Fig. 4-45. The usual routing algorithms apply here. The simplest way to see this is to regard the three point-to-point lines as hostless LANs. Then we have a normal system of six LANS interconnected by four bridges. Nothing in what we have studied so far says that a LAN must have hosts on it.

Various protocols can be used on the point-to-point lines. One possibility is to choose some standard point-to-point data link protocol such as PPP, putting complete MAC frames in the payload field. This strategy works best if all the LANs are identical, and the only problem is getting frames to the correct LAN. Another option is to strip off the MAC header and trailer at the source bridge and put what is left in the payload field of the point-to-point protocol. A new MAC header and trailer can then be generated at the destination bridge. A disadvantage of this approach is that the checksum that arrives at the destination host is not the one computed by the source host, so errors caused by bad bits in a bridge's memory may not be detected.

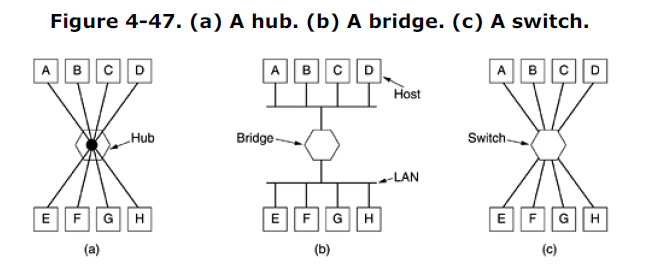
**4.7.5 Repeaters, Hubs, Bridges, Switches, Routers, and Gateways**

So far in this book we have looked at a variety of ways to get frames and packets from one cable segment to another. We have mentioned repeaters, bridges, switches, hubs, routers, and gateways. All of these devices are in common use, but they all differ in subtle and not-so-subtle ways. Since there are so many of them, it is probably worth taking a look at them together to see what the similarities and differences are. To start with, these devices operate in different layers, as illustrated in Fig. 4-46(a). The layer matters because different devices use different pieces of information to decide how to switch. In a typical scenario, the user generates some data to be sent to a remote machine. Those data are passed to the transport layer, which then adds a header, for example, a TCP header, and passes the resulting unit down to the network layer. The network layer adds its own header to form a network layer packet, for example, an IP packet. In Fig. 4-46(b) we see the IP packet shaded in gray. Then the packet goes to the data link layer, which adds its own header and checksum (CRC) and gives the resulting frame to

the physical layer for transmission, for example, over a LAN.



Now let us look at the switching devices and see how they relate to the packets and frames. At the bottom, in the physical layer, we find the repeaters. These are analog devices that are connected to two cable segments. A signal appearing on one of them is amplified and put out on the other. Repeaters do not understand frames, packets, or headers. They understand volts. Classic Ethernet, for example, was designed to allow four repeaters, in order to extend the maximum cable length from 500 meters to 2500 meters. Next we come to the hubs. A hub has a number of input lines that it joins electrically. Frames arriving on any of the lines are sent out on all the others. If two frames arrive at the same time, they will collide, just as on a coaxial cable. In other words, the entire hub forms a single collision domain. All the lines coming into a hub must operate at the same speed. Hubs differ from repeaters in that they do not (usually) amplify the incoming signals and are designed to hold multiple line cards each with multiple inputs, but the differences are slight. Like repeaters, hubs do not examine the 802 addresses or use them in any way. A hub is shown in Fig. 4-47(a).

Now let us move up to the data link layer where we find bridges and switches. We just studied bridges at some length. A bridge connects two or more LANs, as shown in Fig. 4-47(b). When a frame arrives, software in the bridge extracts the destination address from the frame header and looks it up in a table to see where to send the frame. For Ethernet, this address is the 48-bit destination address shown in Fig. 4-17. Like a hub, a modern bridge has line cards, usually for four or eight input lines of a certain type. A line card for Ethernet cannot handle, say, token ring frames, because it does not know where to find the destination address in the frame header.

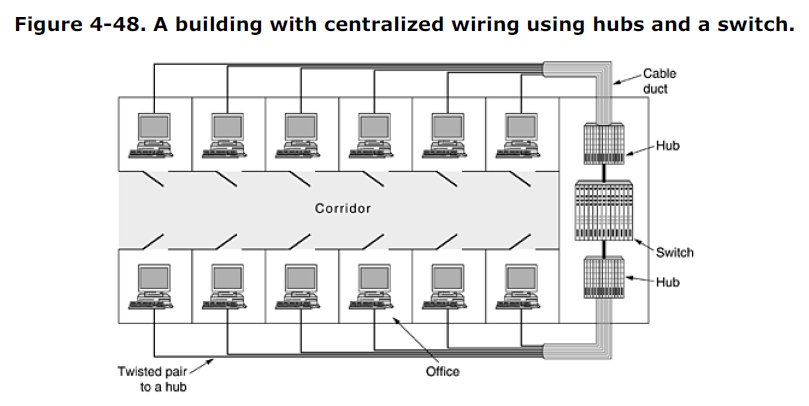
However, a bridge may have line cards for different network types and different speeds. With a bridge, each line is its own collision domain, in contrast to a hub. Switches are similar to bridges in that both route on frame addresses. In fact, many people uses the terms interchangeably. The main difference is that a switch is most often used to connect individual computers, as shown in Fig. 4-47(c). As a consequence, when host *A* in Fig. 4-47(b) wants to send a frame to host *B*, the bridge gets the frame but just discards it. In contrast, in Fig. 4-47(c), the switch must actively forward the frame from *A* to *B* because there is no other way for the frame to get there. Since each switch port usually goes to a single computer, switches must have space for many more line cards than do bridges intended to connect only LANs. Each line card provides buffer space for frames arriving on its ports. Since each port is its own collision domain, switches never lose frames to collisions. However, if frames come in faster than they can be retransmitted, the switch may run out of buffer space and have to start discarding frames. To alleviate this problem slightly, modern switches start forwarding frames as soon as the destination header field has come in, but before the rest of the frame has arrived (provided the output line is available, of course).

These switches do not use store-and-forward switching. Sometimes they are referred to as **cut-through switches**. Usually, cut-through is handled entirely in hardware, whereas bridges traditionally contained an actual CPU that did store-and-forward switching in software. But since all modern bridges and switches contain special integrated circuits for switching, the difference between a switch and bridge is more a marketing issue than a technical one. So far we have seen repeaters and hubs, which are quite similar, as well as bridges and switches, which are also very similar to each other. Now we move up to routers, which are different from all of the above.

When a packet comes into a router, the frame header and trailer are stripped off and the packet located in the frame's payload field (shaded in Fig. 4-46) is passed to the routing software. This software uses the packet header to choose an output line. For an IP packet, the packet header will contain a 32-bit (IPv4) or 128-bit (IPv6) address, but not a 48-bit 802 address. The routing software does not see the frame addresses and does not even know whether the packet came in on a LAN or a point-to-point line. We will study routers and routing in next chapters. Up another layer we find transport gateways. These connect two computers that use different connection-oriented transport protocols. For example, suppose a computer using the connectionoriented TCP/IP protocol needs to talk to a computer using the connection-oriented ATM transport protocol. The transport gateway can copy the packets from one connection to the other, reformatting them as need be. Finally, application gateways understand the format and contents of the data and translate messages from one format to another. An e-mail gateway could translate Internet messages into SMS messages for mobile phones, for example.

**4.7.6 Virtual LANs**

In the early days of local area networking, thick yellow cables snaked through the cable ducts of many office buildings. Every computer they passed was plugged in. Often there were many cables, which were connected to a central backbone (as in Fig. 4-39) or to a central hub. No thought was given to which computer belonged on which LAN. All the people in adjacent offices were put on the same LAN whether they belonged together or not. Geography trumped logic. With the advent of 10Base-T and hubs in the 1990s, all that changed. Buildings were rewired (at considerable expense) to rip out all the yellow garden hoses and install twisted pairs from every office to central wiring closets at the end of each corridor or in a central machine room, as illustrated in Fig. 4-48. If the Vice President in Charge of Wiring was a visionary, category 5 twisted pairs were installed;if he was a bean counter, the existing (category 3) telephone wiring was used (only to be replaced a few years later when fast Ethernet emerged).



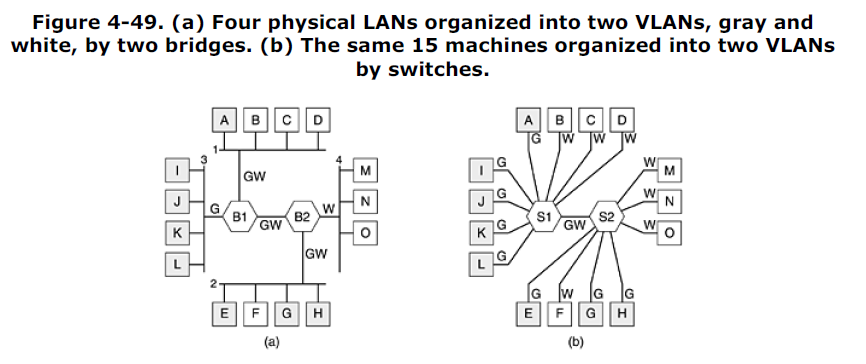
With hubbed (and later, switched) Ethernet, it was often possible to configure LANs logically rather than physically. If a company wants *k* LANs, it buys *k* hubs. By carefully choosing which connectors to plug into which hubs, the occupants of a LAN can be chosen in a way that makes organizational sense, without too much regard to geography. Of course, if two people in the same department work in different buildings, they are probably going to be on different hubs and thus different LANs. Nevertheless, the ituation is a lot better than having LAN membership entirely based on geography.Does it matter who is on which LAN? After all, in virtually all organizations, all the LANs are interconnected. In short, yes, it often matters. Network administrators like to group users on LANs to reflect the organizational structure rather than the physical layout of the building for a variety of reasons. One issue is security.

Any network interface can be put in promiscuous mode, copying all the traffic that comes down the pipe. Many departments, such as research, patents, and accounting, have information that they do not want passed outside their department. In such a situation, putting all the people in a department on a single LAN and not letting any of that traffic off the LAN makes sense.Management does not like hearing that such an arrangement is impossible unless all the people in each department are located in adjacent offices with no interlopers. A second issue is load. Some LANs are more heavily used than others and it may be desirable to separate them at times. For example, if the folks in research are running all kinds of nifty experiments that sometimes get out of hand and saturate their LAN, the folks in accounting may not be enthusiastic about donating some of their capacity to help out.

A third issue is broadcasting. Most LANs support broadcasting, and many upper-layer protocols use this feature extensively. For example, when a user wants to send a packet to an IP address *x*, how does it know which MAC address to put in the frame? We will study this question in Chap. 5, but briefly summarized, the answer is that it broadcasts a frame containing the question: Who owns IP address *x*? Then it waits for an answer. And there are many more examples of where broadcasting is used. As more and more LANs get interconnected, the number of broadcasts passing each machine tends to increase linearly with the number of machines. Related to broadcasts is the problem that once in a while a network interface will break down and begin generating an endless stream of broadcast frames.

The result of this **broadcast storm** is that (1) the entire LAN capacity is occupied by these frames, and (2) all the machines on all the interconnected LANs are crippled just processing and discarding all the frames being broadcast. At first it might appear that broadcast storms could be limited in scope by separating the LANs with bridges or switches, but if the goal is to achieve transparency (i.e., a machine can be moved to a different LAN across the bridge without anyone noticing it), then bridges have to forward broadcast frames. Having seen why companies might want multiple LANs with restricted scope, let us get back to the problem of decoupling the logical topology from the physical topology. Suppose that a user gets shifted within the company from one department to another without changing offices or changes offices without changing departments. With hubbed wiring, moving the user to the correct LAN means having the networkadministrator walk down to the wiring closet and pull the connector for the user's machine from one hub and put it into a new hub. In many companies, organizational changes occur all the time, meaning that system administrators spend a lot of time pulling out plugs and pushing them back in somewhere else. Also, in some cases,the change cannot be made at all because the twisted pair from the user's machine is too far from the correct hub (e.g., in the wrong building). In response to user requests for more flexibility, network vendors began working on a way to rewire buildings entirely in software. The resulting concept is called a **VLAN** (**Virtual LAN**) and has even been standardized by the 802 committee. It is now being deployed in many organizations. Let us now take a look at it.

For additional information about VLANs, see (Breyer and Riley, 1999; and Seifert, 2000). VLANs are based on specially-designed VLAN-aware switches, although they may also have some hubs on the periphery, as in Fig. 4-48. To set up a VLAN-based network, the network administrator decides how many VLANs there will be, which computers will be on which VLAN, and what the VLANs will be called. Often the VLANs are (informally) named by colors, since it is then possible to print color diagrams showing the physical layout of the machines, with the members of the red LAN in red, members of the green LAN in green, and so on. In this way, both the physical and logical layouts are visible in a single view. As an example, consider the four LANs of Fig. 4-49(a), in which eight of the machines belong to the G (gray) VLAN and seven of them belong to the W (white) VLAN. The four physical LANs are connected by two bridges, *B1* and *B2*. If centralized twisted pair wiring is used, there might also be four hubs (not shown), but logically a multidrop cable and a hub are the same thing. Drawing it this way just makes the figure a little less cluttered. Also, the term ''bridge'' tends to be used nowadays mostly when thereare multiple machines on each port, as in this figure, but otherwise, ''bridge'' and ''switch'' areessentially interchangeable. Fig. 4-49shows the same machines and same VLANs using switches with a single computer on each port.

To make the VLANs function correctly, configuration tables have to be set up in the bridges or switches. These tables tell which VLANs are accessible via which ports (lines). When a frame comes in from, say, the gray VLAN, it must be forwarded on all the ports marked G. This holds for ordinary (i.e., unicast) traffic as well as for multicast and broadcast traffic. Note that a port may be labeled with multiple VLAN colors. We see this most clearly in Fig. 4-49(a). Suppose that machine *A* broadcasts a frame. Bridge *B1* receives the frame and sees that it came from a machine on the gray VLAN, so it forwards it on all ports labeled G (except the incoming port). Since *B1* has only two other ports and both of them are labeled G, the frame is sent to both of them. At *B2* the story is different. Here the bridge knows that there are no gray machines on LAN 4, so the frame is not forwarded there. It goes only to LAN 2. If one of the users on LAN 4 should hange

departments and be moved to the gray VLAN, then the tables inside *B2* have to be updated to relabel that port as GW instead of W. If machine *F* goes gray, then the port to LAN 2 has to be changed to G instead of GW. Now let us imagine that all the machines on both LAN 2 and LAN 4 become gray. Then not only do *B2*'s ports to LAN 2 and LAN 4 get marked G, but *B1*'s port to *B2* also has to change from GW to G since white frames arriving at *B1* from LANs 1 and 3 no longer have to be forwarded to *B2*. In Fig. 4-49(b)

the same situation holds, only here all the ports that go to a single machine are labeled with a single color because only one VLAN is out there. So far we have assumed that bridges and switches somehow know what color an incoming frame is. How do they know this? Three methods are in use, as follows:

1. Every port is assigned a VLAN color.

2. Every MAC address is assigned a VLAN color.

3. Every layer 3 protocol or IP address is assigned a VLAN color.

In the first method, each port is labeled with VLAN color. However, this method only works if all machines on a port belong to the same VLAN. In Fig. 4-49(a), this property holds for *B1* for the port to LAN 3 but not for the port to LAN 1.In the second method, the bridge or switch has a table listing the 48-bit MAC address of each machine connected to it along with the VLAN that machine is on. Under these conditions, it is possible to mix VLANs on a physical LAN, as in LAN 1 in Fig. 4-49(a). When a frame arrives, all the bridge or switch has to do is to extract the MAC address and look it up in a table to see which VLAN the frame came from.

The third method is for the bridge or switch to examine the payload field of the frame, for example, to classify all IP machines as belonging to one VLAN and all AppleTalk machines as belonging to another. For the former, the IP address can also be used to identify the machine. This strategy is most useful when many machines are notebook computers that can be docked in any one of several places. Since each docking station has its own MAC address, just knowing which docking station was used does not say anything about which VLAN the notebook is on. The only problem with this approach is that it violates the most fundamental rule of networking: independence of the layers. It is none of the data link layer's business what is in the payload field. It should not be examining the payload and certainly not be making decisions based on the contents. A consequence of using this approach is that a change to the layer 3 protocol (for example, an upgrade from IPv4 to IPv6) suddenly causes the switches to fail. Unfortunately, switches that work this way are on the market. Of course, there is nothing wrong with routing based on IP addresses—nearly all of Chap. 5 is devoted to IP routing—but mixing the layers is looking for trouble. A switch vendor might pooh-pooh this argument saying that its switches understand both IPv4 and IPv6, so everything is fine. But what happens when IPv7 happens? The vendor would probably say: Buy new switches, is that so bad?

**The IEEE 802.1Q Standard**

Some more thought on this subject reveals that what actually matters is the VLAN of the frame itself, not the VLAN of the sending machine. If there were some way to identify the VLAN in the frame header, then the need to inspect the payload would vanish. For a new LAN, such as 802.11 or 802.16, it would have been easy enough to just add a VLAN field in the header. In fact, the *Connection Identifier* field in 802.16 is somewhat similar in spirit to a VLAN identifier. But what to do about Ethernet, which is the dominant LAN, and does not have any spare fields lying around for the VLAN identifier?

The IEEE 802 committee had this problem thrown into its lap in 1995. After much discussion, it did the unthinkable and changed the Ethernet header. The new format was published in IEEE standard **802.1Q**, issued in 1998. The new format contains a VLAN tag; we will examine it shortly. Not surprisingly, changing something as well established as the Ethernet header is not entirely trivial. A few questions that come to mind are:

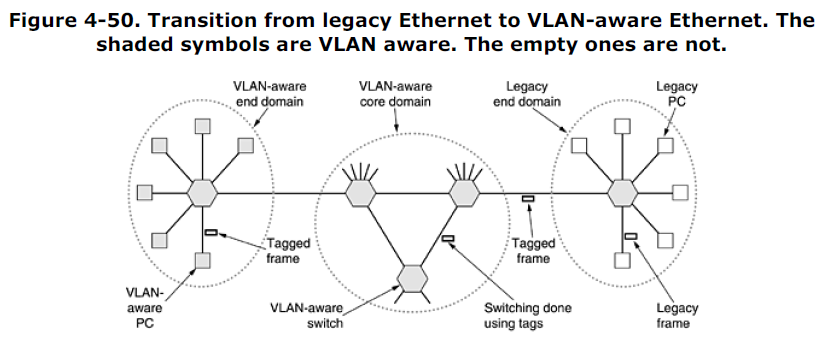
1. Need we throw out several hundred million existing Ethernet cards?

2. If not, who generates the new fields?

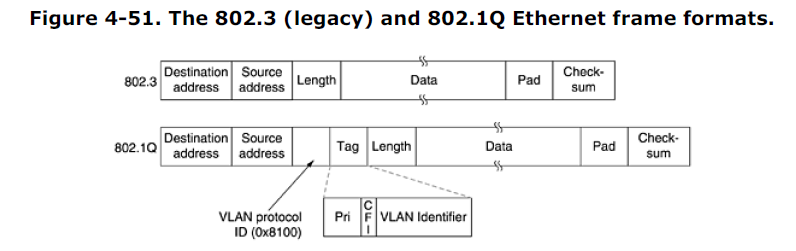
3. What happens to frames that are already the maximum size?

Of course, the 802 committee was (only too painfully) aware of these problems and had to come up with solutions, which it did. The key to the solution is to realize that the VLAN fields are only actually used by the bridges and switches and not by the user machines. Thus in Fig. 4-49, it is not really essential that they are present on the lines going out to the end stations as long as they are on the line between the bridges or switches. Thus, to use VLANs, the bridges or switches have to be VLAN aware, but that was already a requirement. Now we are only introducing the additional requirement that they are 802.1Q aware, which new ones already are. As to throwing out all existing Ethernet cards, the answer is no. Remember that the 802.3 committee could not even get people to change the *Type* field into a *Length* field. You can imagine the reaction to an announcement that all existing Ethernet cards had to be thrown out. However, as new Ethernet cards come on the market, the hope is that they will be 802.1Q compliant and correctly fill in the VLAN fields.

So if the originator does not generate the VLAN fields, who does? The answer is that the first VLANaware bridge or switch to touch a frame adds them and the last one down the road removes them. But how does it know which frame belongs to which VLAN? Well, the first bridge or switch could assign a VLAN number to a port, look at the MAC address, or (heaven forbid) examine the payload. Until Ethernet cards are all 802.1Q compliant, we are kind of back where we started. The real hope here is that all gigabit Ethernet cards will be 802.1Q compliant from the start and that as people upgrade to gigabit Ethernet, 802.1Q will be introduced automatically. As to the problem of frames longer than 1518 bytes, 802.1Q just raised the limit to 1522 bytes. During the transition process, many installations will have some legacy machines (typically classic or fast Ethernet) that are not VLAN aware and others (typically gigabit Ethernet) that are. This situation is illustrated in Fig. 4-50, where the shaded symbols are VLAN aware and the empty ones are not. For simplicity, we assume that all the switches are VLAN aware. If this is not the case, the first VLANaware switch can add the tags based on MAC or IP addresses.



In this figure, VLAN-aware Ethernet cards generate tagged (i.e., 802.1Q) frames directly, and further switching uses these tags. To do this switching, the switches have to know which VLANs are reachable on each port, just as before. Knowing that a frame belongs to the gray VLAN does not help much until the switch knows which ports connect to machines on the gray VLAN. Thus, the switch needs a table indexed by VLAN telling which ports to use and whether they are VLAN aware or legacy. When a legacy PC sends a frame to a VLAN-aware switch, the switch builds a new tagged frame based on its knowledge of the sender's VLAN (using the port, MAC address, or IP address). From that point on, it no longer matters that the sender was a legacy machine. Similarly, a switch that needs to deliver a tagged frame to a legacy machine has to reformat the frame in the legacy format before delivering it. Now let us take a look at the 802.1Q frame format. It is shown in Fig. 4-51. The only change is the addition of a pair of 2-byte fields. The first one is the *VLAN protocol ID*. It always has the value 0x8100. Since this number is greater than 1500, all Ethernet cards interpret it as a type rather than a length. What a legacy card does with such a frame is moot since such frames are not supposed to be sent to legacy cards.



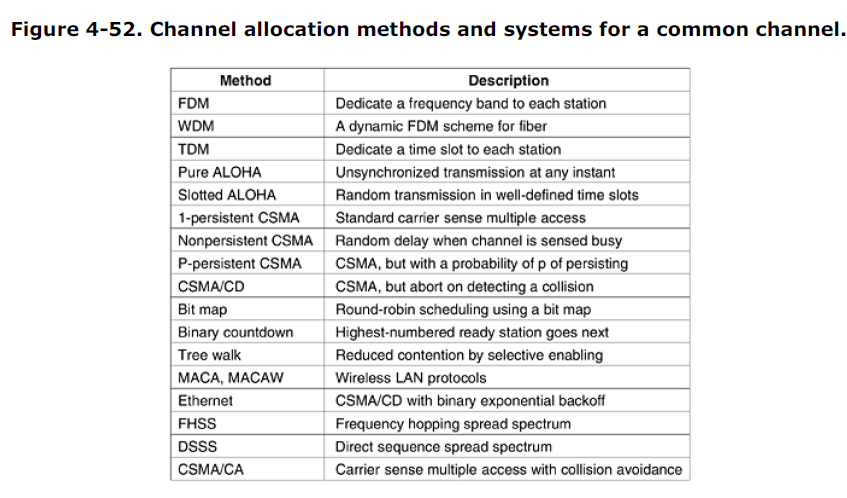
The second 2-byte field contains three subfields. The main one is the *VLAN identifier*, occupying thelow-order 12 bits. This is what the whole thing is about—which VLAN does the frame belong to? The 3-bit *Priority* field has nothing to do with VLANs at all, but since changing the Ethernet header is a oncein-a-decade event taking three years and featuring a hundred people, why not put in some other good things while you are at it? This field makes it possible to distinguish hard real-time traffic from soft real-time traffic from time-insensitive traffic in order to provide better quality of service over Ethernet.

It is needed for voice over Ethernet (although in all fairness, IP has had a similar field for a quarter of a century and nobody ever used it). The last bit, *CFI* (*Canonical Format Indicator*) should have been called the *CEI* (*Corporate Ego Indicator*).

It was originally intended to indicate little-endian MAC addresses versus big-endian MAC addresses, but that use got lost in other controversies. Its presence now indicates that the payload contains a freeze-dried 802.5 frame that is hoping to find another 802.5 LAN at the destination while being carried by Ethernet in between. This whole arrangement, of course, has nothing whatsoever to do with VLANs. But standards' committee politics is not unlike regular politics: if you vote for my bit, I will vote for your bit. As we mentioned above, when a tagged frame arrives at a VLAN-aware switch, the switch uses the VLAN ID as an index into a table to find out which ports to send it on. But where does the table come from? If it is manually constructed, we are back to square zero: manual configuration of bridges. The beauty of the transparent bridge is that it is plug-and-play and does not require any manual configuration. It would be a terrible shame to lose that property. Fortunately, VLAN-aware bridges can also autoconfigure themselves based on observing the tags that come by. If a frame tagged as VLAN 4 comes in on port 3, then apparently some machine on port 3 is on VLAN 4. The 802.1Q standard explains how to build the tables dynamically, mostly by referencing appropriate portions of Perlman's algorithm standardized in 802.1D. Before leaving the subject of VLAN routing, it is worth making one last observation. Many people in the Internet and Ethernet worlds are fanatically in favor of connectionless networking and violently opposed to anything smacking of connections in the data link or network layers. Yet VLANs introduce something that is surprisingly similar to a connection. To use VLANs properly, each frame carries a new special identifier that is used as an index into a table inside the switch to look up where the frame is supposed to be sent. That is precisely what happens in connection-oriented networks. In connectionless networks, it is the destination address that is used for routing, not some kind of connection identifier. We will see more of this creeping connectionism in Chap. 5.

**4.8 Summary**

Some networks have a single channel that is used for all communication. In these networks, the key design issue is the allocation of this channel among the competing stations wishing to use it. Numerous channel allocation algorithms have been devised. A summary of some of the more important channel allocation methods is given in Fig. 4-52. The simplest allocation schemes are FDM and TDM. These are efficient when the number of stations is small and fixed and the traffic is continuous. Both are widely used under these circumstances, forexample, for dividing up the bandwidth on telephone trunks.



When the number of stations is large and variable or the traffic is fairly bursty, FDM and TDM are poor choices. The ALOHA protocol, with and without slotting, has been proposed as an alternative. ALOHA and its many variants and derivatives have been widely discussed, analyzed, and used in real systems. When the state of the channel can be sensed, stations can avoid starting a transmission while another station is transmitting. This technique, carrier sensing, has led to a variety of protocols that can be used on LANs and MANs. A class of protocols that eliminates contention altogether, or at least reduce it considerably, is well known. Binary countdown completely eliminates contention. The tree walk protocol reduces it by dynamically dividing the stations into two disjoint groups, one of which is permitted to transmit and one of which is not. It tries to make the division in such a way that only one station that is ready to send is permitted to do so.

Wireless LANs have their own problems and solutions. The biggest problem is caused by hidden stations, so CSMA does not work. One class of solutions, typified by MACA and MACAW, attempts to stimulate transmissions around the destination, to make CSMA work better. Frequency hopping spread spectrum and direct sequence spread spectrum are also used. IEEE 802.11 combines CSMA and MACAW to produce CSMA/CA.

Ethernet is the dominant form of local area networking. It uses CSMA/CD for channel allocation. Older versions used a cable that snaked from machine to machine, but now twisted pairs to hubs and switches are most common. Speeds have risen from 10 Mbps to 1 Gbps and are still rising. Wireless LANs are becoming common, with 802.11 dominating the field. Its physical layer allows five different transmission modes, including infrared, various spread spectrum schemes, and a multichannel FDM system. It can operate with a base station in each cell, but it can also operate without one. The

protocol is a variant of MACAW, with virtual carrier sensing. Wireless MANs are starting to appear. These are broadband systems that use radio to replace the last

mile on telephone connections. Traditional narrowband modulation techniques are used. Quality of service is important, with the 802.16 standard defining four classes (constant bit rate, two variable bit rate, and one best efforts). The Bluetooth system is also wireless but aimed more at the desktop, for connecting headsets and other peripherals to computers without wires. It is also intended to connect peripherals, such as fax machines, to mobile telephones. Like 801.11, it uses frequency hopping spread spectrum in the ISM band. Due to the expected noise level of many environments and need for real-time interaction, elaborate forward error correction is built into its various protocols. With so many different LANs, a way is needed to interconnect them all. Bridges and switches are used for this purpose. The spanning tree algorithm is used to build plug-and-play bridges. A new development in the LAN interconnection world is the VLAN, which separates the logical topology of the LANs from their physical topology. A new format for Ethernet frames (802.1Q) has been introduced to ease the introduction of VLANs into organizations.