2.6 Ethernet (802.3)

The Ethernet is easily the most successful local area networking technology of the last 20 years. Developed in the mid-1970s by researchers at the Xerox Palo Alto Research Center (PARC), the Ethernet is a working example of the more general carrier sense, multiple access with collision detect (CSMA/CD) local area network technology.

As indicated by the CSMA name, the Ethernet is a multiple-access network, meaning that a set of nodes send and receive frames over a shared link. You can, therefore, think of an Ethernet as being like a bus that has multiple stations plugged into it. The "carrier sense" in CSMA/CD means that all the nodes can distinguish between an idle and a busy link, and "collision detect" means that a node listens as it transmits and can therefore detect when a frame it is transmitting has interfered (collided) with a frame transmitted by another node.

The Ethernet has its roots in an early packet radio network, called Aloha, developed at the University of Hawaii to support computer communication across the Hawaiian Islands. Like the Aloha network, the fundamental problem faced by the Ethernet is how to mediate access to a shared medium fairly and efficiently (in Aloha the medium was the atmosphere, while in Ethernet the medium is a coax cable). That is, the core idea in both Aloha and the Ethernet is an algorithm that controls when each node can transmit.

Digital Equipment Corporation and Intel Corporation joined Xerox to define a 10-Mbps Ethernet standard in 1978. This standard then formed the basis for IEEE standard 802.3. With one exception that we will see in Section 2.6.2, it is fair to view the 1978 Ethernet standard as a proper subset of the 802.3 standard; 802.3 additionally defines a much wider collection of physical media over which Ethernet can operate, and more recently, it has been extended to include a 100-Mbps version called Fast Ethernet, and a 1,000-Mbps version called Gigabit Ethernet. The rest of this section focuses on the 10-Mbps Ethernet since it is typically used in multiple-access mode, and we are interested in how multiple hosts share a single link. Both 100- and 1,000-Mbps Ethernets are designed to be used in full-duplex, point-to-point configurations, which means that they are typically used in switched networks, as described in the next chapter.

2.6.1 Physical Properties

An Ethernet segment is implemented on a coaxial cable of up to 500 m. This cable is similar to the type used for cable TV, except that it typically has an impedance of 50 ohms instead of cable TV's 75 ohms. Hosts connect to an Ethernet segment by tapping into it; taps must be at least 2.5 m apart. A *transceiver*—a small device directly attached to the tap—detects when the line is idle and drives the signal when the host is transmitting. It also receives incoming signals. The transceiver is, in turn, connected to an Ethernet adaptor, which is plugged into the host. All the logic that makes up the Ethernet pro-

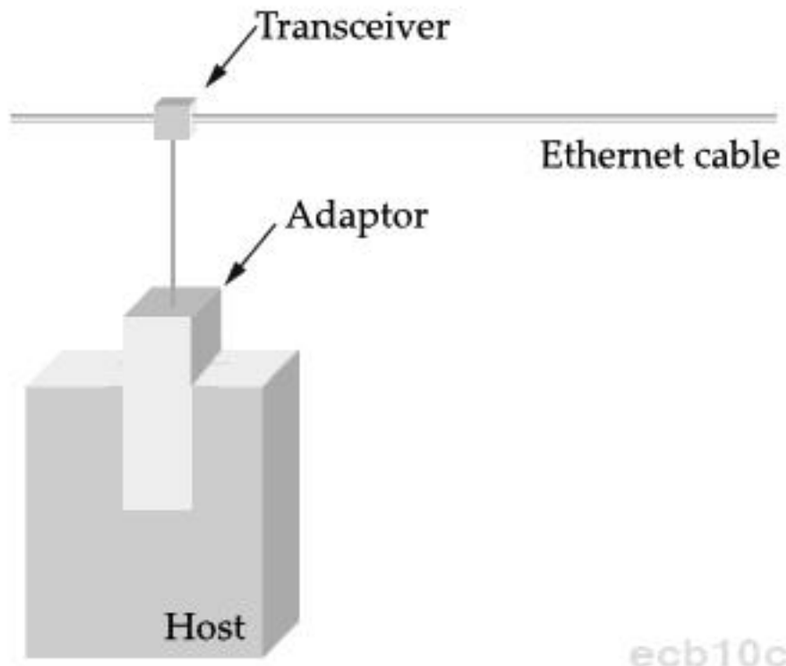

Figure 2.27 Ethernet transceiver and adaptor.

tocol, as described in this section, is implemented in the adaptor (not the transceiver). This configuration is shown in Figure 2.27.

Multiple Ethernet segments can be joined together by *repeaters*. A repeater is a device that forwards digital signals, much like an amplifier forwards analog signals. However, no more than four repeaters may be positioned between any pair of hosts, meaning that an Ethernet has a total reach of only 2,500 m. For example, using just two repeaters between any pair of hosts supports a configuration similar to the one illustrated in Figure 2.28, that is, a segment running down the spine of a building with a segment on each floor. All told, an Ethernet is limited to supporting a maximum of 1,024 hosts.

Any signal placed on the Ethernet by a host is broadcast over the entire network, that is, the signal is propagated in both directions, and repeaters forward the signal on all outgoing segments. Terminators attached to the end of each segment absorb the signal and keep it from bouncing back and interfering with trailing signals. The Ethernet uses the Manchester encoding scheme described in Section 2.2.

In addition to the system of segments and repeaters just described, alternative technologies have been introduced over the years. For example, rather than using a 50-ohm coax cable, an Ethernet can be constructed from a thinner cable known as 10Base2; the original cable is called 10Base5 (the two cables are commonly called *thin-net* and *thick-net*, respectively). The "10" in 10Base2 means that the network operates at 10 Mbps, "Base" refers to the fact that the cable is used in a *baseband* system, and the "2" means that a given segment can be no longer than 200 m (a segment of the original 10Base5 cable can be up to 500 m long). Today, a third cable technology is predominantly used, called 10BaseT, where the "T" stands for twisted pair. Typically, Category 5 twisted pair wiring is used. A 10BaseT segment is usually limited to under 100 m in length. (Both

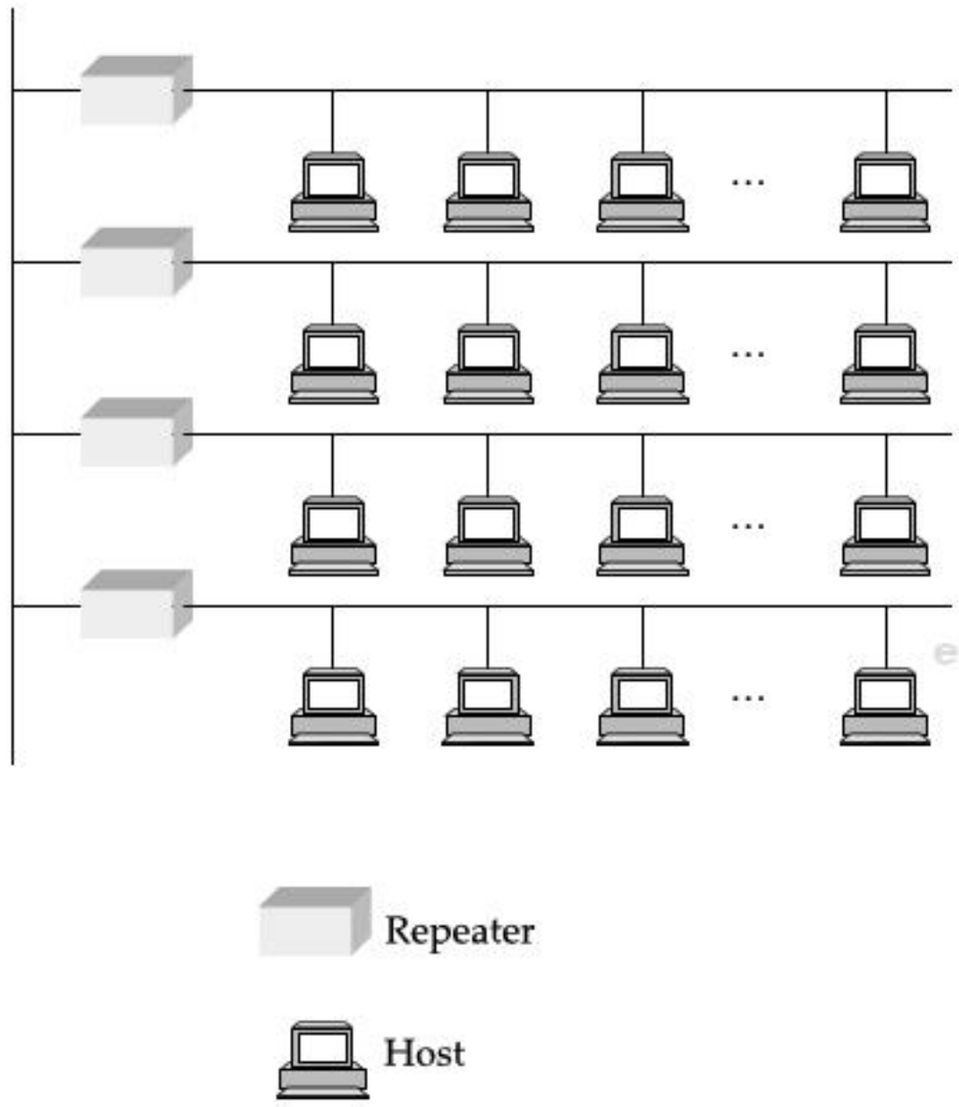

Figure 2.28 Ethernet repeater.

100- and 1,000-Mbps Ethernets also run up over Category 5 twisted pair, up to distances of 100 m.)

Because the cable is so thin, you do not tap into a 10Base2 or 10BaseT cable in the same way as you would with 10Base5 cable. With 10Base2, a T-joint is spliced into the cable. In effect, 10Base2 is used to daisy-chain a set of hosts together. With 10BaseT, the common configuration is to have several point-to-point segments coming out of a multiway repeater, sometimes called a *hub*, as illustrated in Figure 2.29. Multiple

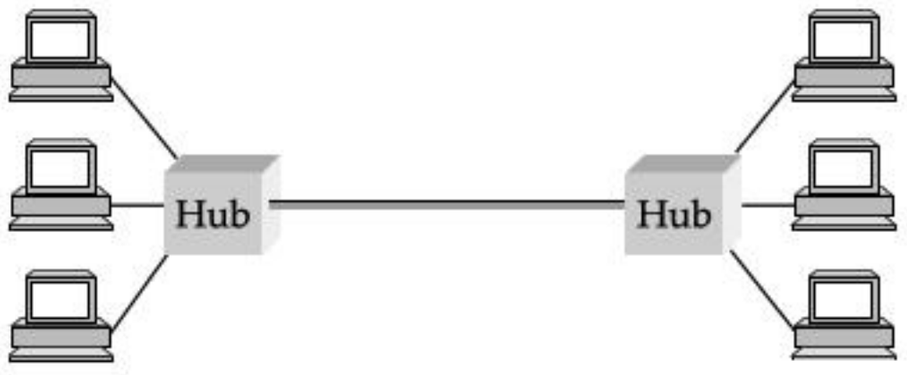

Figure 2.29 Ethernet hub.

100-Mbps Ethernet segments can also be connected by a hub, but the same is not true of 1,000-Mbps segments.

It is important to understand that whether a given Ethernet spans a single segment, a linear sequence of segments connected by repeaters, or multiple segments connected in a star configuration by a hub, data transmitted by any one host on that Ethernet reaches all the other hosts. This is the good news. The bad news is that all these hosts are competing for access to the same link, and as a consequence, they are said to be in the same *collision domain*.

2.6.2 Access Protocol

We now turn our attention to the algorithm that controls access to the shared Ethernet link. This algorithm is commonly called the Ethernet's *media access control (MAC)*. It is typically implemented in hardware on the network adaptor. We will not describe the hardware per se, but instead focus on the algorithm it implements. First, however, we describe the Ethernet's frame format and addresses.

Frame Format

Each Ethernet frame is defined by the format given in Figure 2.30. The 64-bit preamble allows the receiver to synchronize with the signal; it is a sequence of alternating 0s and 1s. Both the source and destination hosts are identified with a 48-bit address. The packet type field serves as the demultiplexing key, that is, it identifies to which of possibly many higher-level protocols this frame should be delivered. Each frame contains up to 1,500 bytes of data. Minimally, a frame must contain at least 46 bytes of data, even if this means the host has to pad the frame before transmitting it. The reason for this minimum frame size is that the frame must be long enough to detect a collision; we discuss this more below. Finally, each frame includes a 32-bit CRC. Like the HDLC protocol described in Section 2.3.2, the Ethernet is a bit-oriented framing protocol. Note that from the host's perspective, an Ethernet frame has a 14-byte header: two 6-byte addresses and a 2-byte type field. The sending adaptor attaches the preamble, CRC, and postamble before transmitting, and the receiving adaptor removes them.

The frame format just described is taken from the Digital-Intel-Xerox Ethernet standard. The 802.3 frame format is exactly the same, except it substitutes a 16-bit length

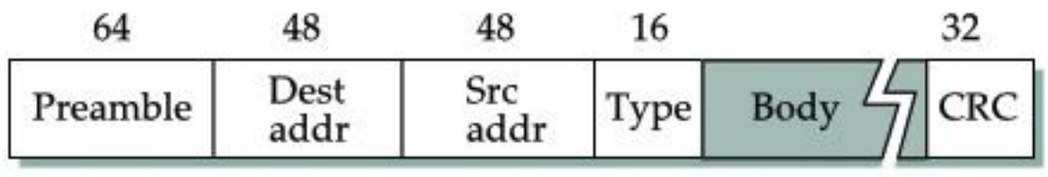

Figure 2.30 Ethernet frame format.

field for the 16-bit type field. 802.3 is usually paired with an encapsulation standard that defines a type field used to demultiplex incoming frames. This type field is the first thing in the data portion of the 802.3 frames, that is, it immediately follows the 802.3 header. Fortunately, since the Ethernet standard has avoided using any type values less than 1,500 (the maximum length found in an 802.3 header), and the type and length fields are in the same location in the header, it is possible for a single device to accept both formats, and for the device driver running on the host to interpret the last 16 bits of the header as either a type or a length. In practice, most hosts follow the Digital-Intel-Xerox format and interpret this field as the frame's type.

Addresses

Each host on an Ethernet—in fact, every Ethernet host in the world—has a unique Ethernet address. Technically, the address belongs to the adaptor, not the host; it is usually burned into ROM. Ethernet addresses are typically printed in a form humans can read as a sequence of six numbers separated by colons. Each number corresponds to 1 byte of the 6-byte address and is given by a pair of hexadecimal digits, one for each of the 4-bit nibbles in the byte; leading 0s are dropped. For example, 8:0:2b:e4:b1:2 is the human-readable representation of Ethernet address

00001000 00000000 00101011 11100100 10110001 00000010

To ensure that every adaptor gets a unique address, each manufacturer of Ethernet devices is allocated a different prefix that must be prepended to the address on every adaptor they build. For example, Advanced Micro Devices has been assigned the 24-bit prefix x080020 (or 8:0:20). A given manufacturer then makes sure the address suffixes it produces are unique.

Each frame transmitted on an Ethernet is received by every adaptor connected to that Ethernet. Each adaptor recognizes those frames addressed to its address and passes only those frames on to the host. (An adaptor can also be programmed to run in *promiscuous* mode, in which case it delivers all received frames to the host, but this is not the normal mode.) In addition to these *unicast* addresses, an Ethernet address consisting of all 1s is treated as a *broadcast* address; all adaptors pass frames addressed to the broadcast address up to the host. Similarly, an address that has the first bit set to 1 but is not the broadcast address is called a *multicast* address. A given host can program its adaptor to accept some set of multicast addresses. Multicast addresses are used to send messages to some subset of the hosts on an Ethernet (e.g., all file servers). To summarize, an Ethernet adaptor receives all frames and accepts

- Frames addressed to its own address;
- Frames addressed to the broadcast address;

- Frames addressed to a multicast address, if it has been instructed to listen to that address;
- All frames, if it has been placed in promiscuous mode.

It passes to the host only the frames that it accepts.

Transmitter Algorithm

As we have just seen, the receiver side of the Ethernet protocol is simple; the real smarts are implemented at the sender's side. The transmitter algorithm is defined as follows.

When the adaptor has a frame to send and the line is idle, it transmits the frame immediately; there is no negotiation with the other adaptors. The upper bound of 1,500 bytes in the message means that the adaptor can occupy the line for only a fixed length of time.

When an adaptor has a frame to send and the line is busy, it waits for the line to go idle and then transmits immediately.[4] The Ethernet is said to be a *1-persistent* protocol because an adaptor with a frame to send transmits with probability 1 whenever a busy line goes idle. In general, a *p-persistent* algorithm transmits with probability $0 \leq p \leq 1$ after a line becomes idle, and defers with probability $q = 1 - p$. The reasoning behind choosing a $p < 1$ is that there might be multiple adaptors waiting for the busy line to become idle, and we don't want all of them to begin transmitting at the same time. If each adaptor transmits immediately with a probability of, say, 33%, then up to three adaptors can be waiting to transmit and the odds are that only one will begin transmitting when the line becomes idle. Despite this reasoning, an Ethernet adaptor always transmits immediately after noticing that the network has become idle and has been very effective in doing so.

To complete the story about p-persistent protocols for the case when $p < 1$, you might wonder how long a sender that loses the coin flip (i.e., decides to defer) has to wait before it can transmit. The answer for the Aloha network, which originally developed this style of protocol, was to divide time into discrete slots, with each slot corresponding to the length of time it takes to transmit a full frame. Whenever a node has a frame to send and it senses an empty (idle) slot, it transmits with probability p and defers until the next slot with probability $q = 1 - p$. If that next slot is also empty, the node again decides to transmit or defer, with probabilities p and q, respectively. If that next slot is not empty—that is, some other station has decided to transmit—then the node simply waits for the next idle slot and the algorithm repeats.

Returning to our discussion of the Ethernet, because there is no centralized control it is possible for two (or more) adaptors to begin transmitting at the same time, either

[4] To be more precise, all adaptors wait 9.6 μs after the end of one frame before beginning to transmit the next frame. This is true for both the sender of the first frame, as well as those nodes listening for the line to become idle.

because both found the line to be idle or because both had been waiting for a busy line to become idle. When this happens, the two (or more) frames are said to *collide* on the network. Each sender, because the Ethernet supports collision detection, is able to determine that a collision is in progress. At the moment an adaptor detects that its frame is colliding with another, it first makes sure to transmit a 32-bit jamming sequence and then stops the transmission. Thus, a transmitter will minimally send 96 bits in the case of a collision: 64-bit preamble plus 32-bit jamming sequence.

One way that an adaptor will send only 96 bits—which is sometimes called a *runt frame*—is if the two hosts are close to each other. Had the two hosts been farther apart, they would have had to transmit longer, and thus send more bits, before detecting the collision. In fact, the worst-case scenario happens when the two hosts are at opposite ends of the Ethernet. To know for sure that the frame it just sent did not collide with another frame, the transmitter may need to send as many as 512 bits. Not coincidentally, every Ethernet frame must be at least 512 bits (64 bytes) long: 14 bytes of header plus 46 bytes of data plus 4 bytes of CRC.

Why 512 bits? The answer is related to another question you might ask about an Ethernet: Why is its length limited to only 2,500 m? Why not 10 or 1,000 km? The answer to both questions has to do with the fact that the farther apart two nodes are, the longer it takes for a frame sent by one to reach the other, and the network is vulnerable to a collision during this time.

Figure 2.31 illustrates the worst-case scenario, where hosts A and B are at opposite ends of the network. Suppose host A begins transmitting a frame at time t, as shown in (a). It takes it one link latency (let's denote the latency as d) for the frame to reach host B. Thus, the first bit of A's frame arrives at B at time $t + d$, as shown in (b). Suppose an instant before host A's frame arrives (i.e., B still sees an idle line), host B begins to transmit its own frame. B's frame will immediately collide with A's frame, and this collision will be detected by host B (c). Host B will send the 32-bit jamming sequence, as described above. (B's frame will be a runt.) Unfortunately, host A will not know that the collision occurred until B's frame reaches it, which will happen one link latency later, at time $t + 2 \times d$, as shown in (d). Host A must continue to transmit until this time in order to detect the collision. In other words, host A must transmit for $2 \times d$ to be sure that it detects all possible collisions. Considering that a maximally configured Ethernet is 2,500 m long, and that there may be up to four repeaters between any two hosts, the round-trip delay has been determined to be 51.2 μs, which on a 10-Mbps Ethernet corresponds to 512 bits. The other way to look at this situation is that we need to limit the Ethernet's maximum latency to a fairly small value (e.g., 51.2 μs) for the access algorithm to work; hence, an Ethernet's maximum length must be something on the order of 2,500 m.

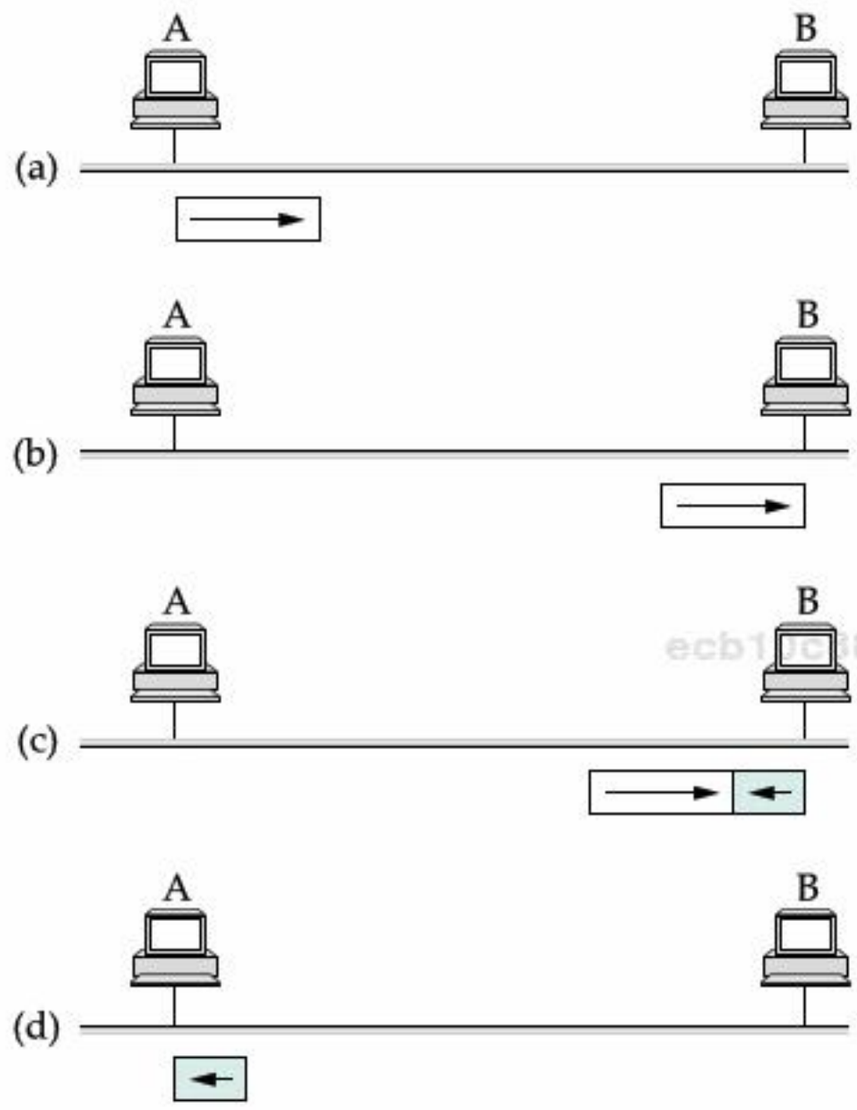

Figure 2.31 Worst-case scenario: (a) A sends a frame at time *t*; (b) A's frame arrives at B at time *t + d*; (c) B begins transmitting at time *t + d* and collides with A's frame; (d) B's runt (32-bit) frame arrives at A at time *t + 2d*.

Once an adaptor has detected a collision and stopped its transmission, it waits a certain amount of time and tries again. Each time it tries to transmit but fails, the adaptor doubles the amount of time it waits before trying again. This strategy of doubling the delay interval between each retransmission attempt is a general technique known as *exponential backoff*. More precisely, the adaptor first delays either 0 or 51.2 μs, selected at random. If this effort fails, it then waits 0, 51.2, 102.4, or 153.6 μs (selected randomly) before trying again; this is $k \times 51.2$ for $k = 0..3$. After the third collision, it waits $k \times 51.2$ for $k = 0..2^3 - 1$, again selected at random. In general, the algorithm randomly selects a k between 0 and $2^n - 1$ and waits $k \times 51.2$ μs, where n is the number of collisions experienced so far. The adaptor gives up after a given number of tries and reports a transmit error to the host. Adaptors typically retry up to 16 times, although the backoff algorithm caps n in the above formula at 10.

2.6.3 Experience with Ethernet

Because Ethernets have been around for so many years and are so popular, we have a great deal of experience in using them. One of the most important observations people

have made about Ethernets is that they work best under lightly loaded conditions. This is because under heavy loads—typically, a utilization of over 30% is considered heavy on an Ethernet—too much of the network's capacity is wasted by collisions.

Fortunately, most Ethernets are used in a far more conservative way than the standard allows. For example, most Ethernets have fewer than 200 hosts connected to them, which is far fewer than the maximum of 1,024. (See if you can discover a reason for this upper limit of around 200 hosts in Chapter 4.) Similarly, most Ethernets are far shorter than 2,500 m, with a round-trip delay of closer to 5 μs than 51.2 μs. Another factor that makes Ethernets practical is that, even though Ethernet adaptors do not implement link-level flow control, the hosts typically provide an end-to-end flow-control mechanism. As a result, it is rare to find situations in which any one host is continuously pumping frames onto the network.

Finally, it is worth saying a few words about why Ethernets have been so successful, so that we can understand the properties we should emulate with any LAN technology that tries to replace it. First, an Ethernet is extremely easy to administer and maintain: There are no switches that can fail, no routing or configuration tables that have to be kept up-to-date, and it is easy to add a new host to the network. It is hard to imagine a simpler network to administer. Second, it is inexpensive: Cable is cheap, and the only other cost is the network adaptor on each host. Any switch-based approach will involve an investment in some relatively expensive infrastructure (the switches), in addition to the incremental cost of each adaptor. As we will see in the next chapter, the most successful LAN switching technology in use today is itself based on Ethernet.

2.7 Rings (802.5, FDDI, RPR)

Ring networks, like Ethernets, are shared-media networks. This section will focus on the type that was for years the most prevalent, known as the IBM Token Ring. Like the Xerox Ethernet, IBM's Token Ring has a nearly identical IEEE standard, known as 802.5. 802.5 and the later Fiber Distributed Data Interface (FDDI) token ring are no longer in widespread use. Resilient Packet Ring (RPR) is a relatively recent technology, and its corresponding IEEE standard is known as 802.17; it remains to be seen how popular RPR will be.

As the name suggests, a ring network consists of a set of nodes connected in a ring (see Figure 2.32). Data always flows in a particular direction around the ring, with each node receiving frames from its upstream neighbor and then forwarding them to its downstream neighbor. This ring-based topology is in contrast to the Ethernet's bus topology. Like the Ethernet, however, the ring is viewed as a single shared medium; it does not behave as a collection of independent point-to-point links that just happen to be configured in a loop. Thus, a ring network shares two key features with an Ethernet:

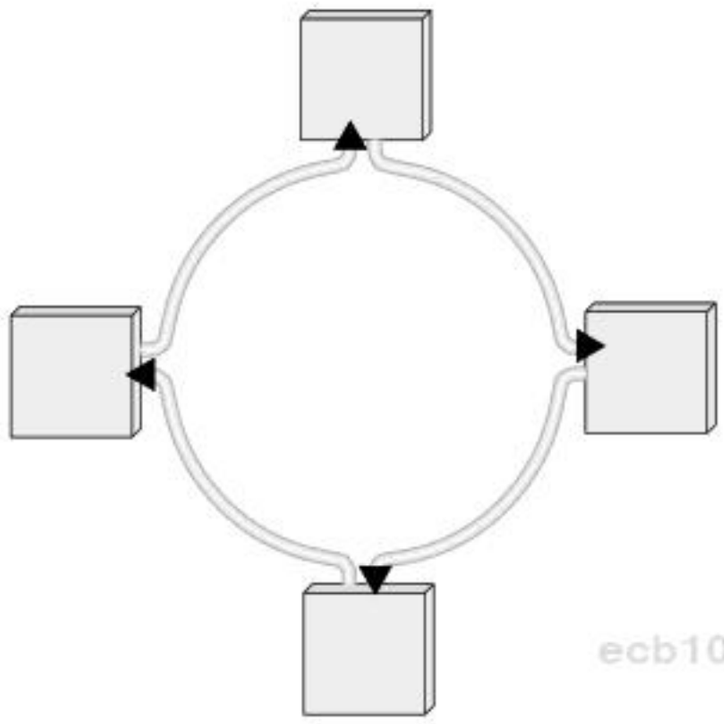

Figure 2.32 A ring network.

First, it involves a distributed algorithm that controls when each node is allowed to transmit, and second, all nodes typically[5] see all frames, with the node identified in the frame header as the destination saving a copy of the frame as it flows past.

The most common early forms of ring network were all *token* rings. The word "token" comes from the way access to the shared ring is managed. The idea is that a token, which is really just a special sequence of bits, circulates around the ring; each node receives and then forwards the token. When a node that has a frame to transmit sees the token, it takes the token off the ring (i.e., it does not forward the special bit pattern) and instead inserts its frame into the ring. Each node along the way simply forwards the frame, with the destination node saving a copy and forwarding the message onto the next node on the ring. When the frame makes its way back around to the sender, this node strips its frame off the ring (rather than continuing to forward it) and reinserts the token. In this way, some node downstream will have the opportunity to transmit a frame. The media access algorithm is fair in the sense that as the token circulates around the ring, each node gets a chance to transmit. Nodes are serviced in a round-robin fashion.

One of the first things you might worry about with a ring topology is that any link or node failure would render the whole network useless. The problem of node failure may be addressed by connecting each station into the ring using an electromechanical relay. As long as the station is healthy, the relay is open and the station is included in the ring. If the station stops providing power, the relay closes and the ring automatically bypasses the station. This is illustrated in Figure 2.33. Note that this approach is only effective when the transmission medium is electrical cable, not optical fiber.

[5] We will see an exception to this in Section 2.7.4.

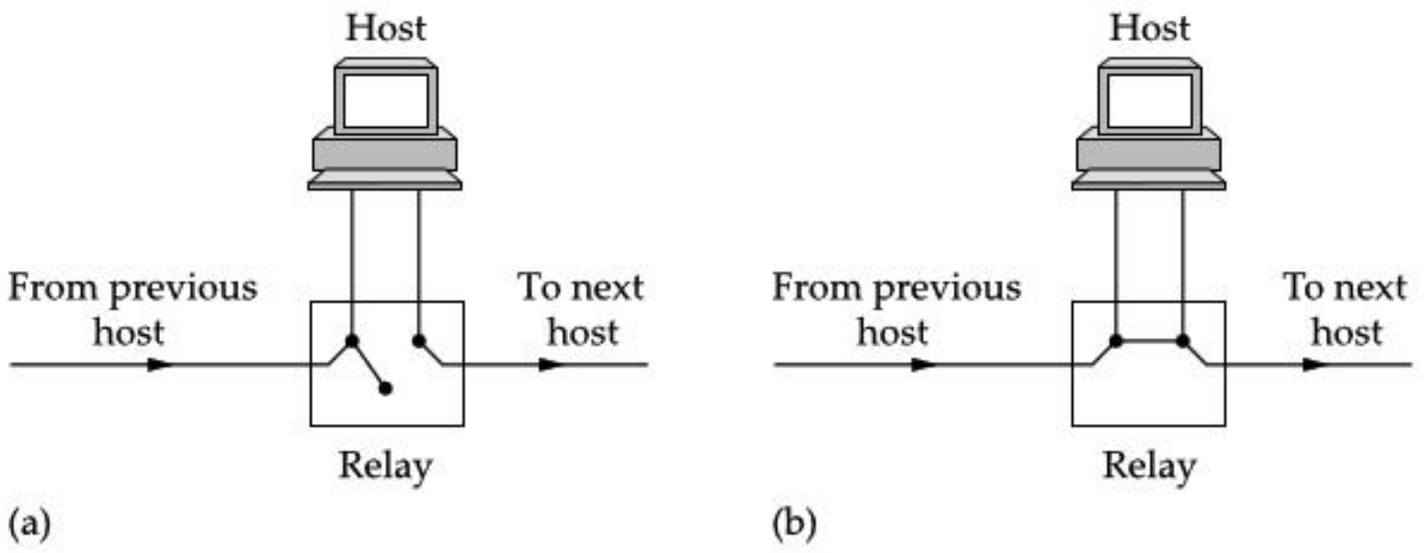

Figure 2.33 Relay used on a token ring: (a) relay open—host active; (b) relay closed—host bypassed.

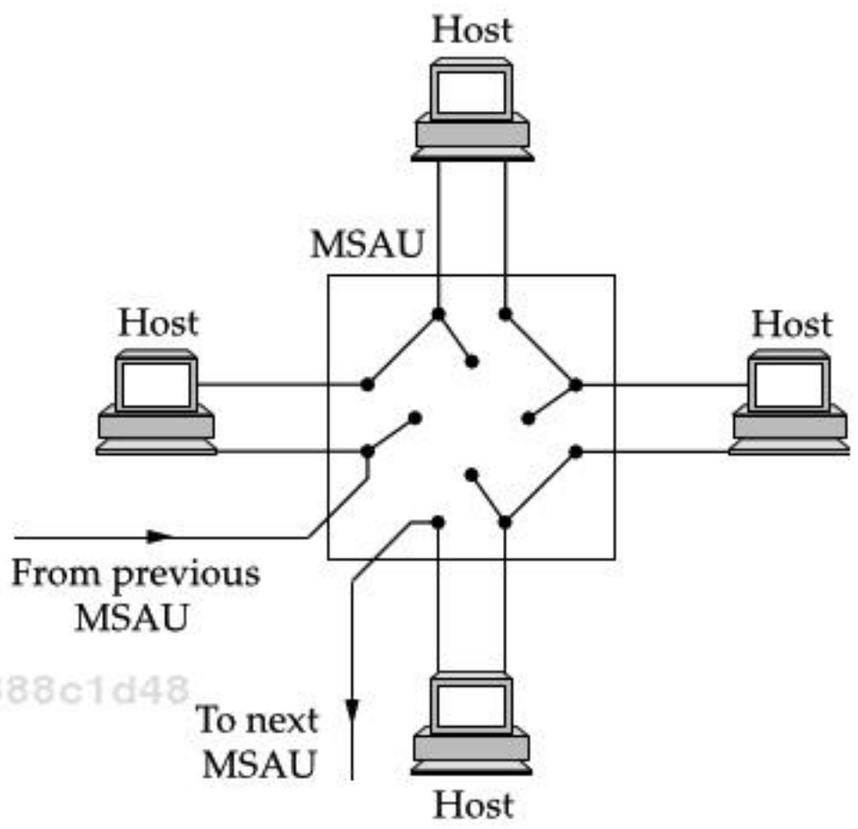

Figure 2.34 Multistation access unit.

Several of these relays are usually packed into a single box, known as a multistation access unit (MSAU). This has the interesting effect of making a token ring actually look more like a star topology, as shown in Figure 2.34. Any failure of a link outside the MSAU is then equivalent to a host failure, hence solved by the same relay mechanism. It also makes it very easy to add stations to and remove stations from the network, since they can just be plugged into or unplugged from the nearest MSAU, while the overall wiring of the network can be left unchanged.

2.7.1 Token Ring Media Access Control

It is now time to look a little more closely at how the MAC protocol operates on a token ring. The network adaptor for a token ring contains a receiver and a transmitter. Most of the time, when a node is neither the source nor the destination of the data on the ring, its adaptor is simply retransmitting the data that its receiver receives. When none of the stations connected to the ring has anything to send, the token circulates around the ring. As it does so, any station that has data to send may "seize" the token, that is, not retransmit it and begin sending data. Once a station has the token, it is allowed to send one or more packets—exactly how many more depends on some factors described below.

Each transmitted packet contains the destination address of the intended receiver; it may also contain a multicast (or broadcast) address if it is intended to reach more than one (or all) receivers. As the packet flows past each node on the ring, each node looks inside the packet to see if it is the intended recipient. If so, it copies the packet into a buffer as it flows through the network adaptor, but it does not remove the packet from the ring. The sending station has the responsibility of removing the packet from the ring.

One issue we must address is how much data a given node is allowed to transmit each time it possesses the token or, equivalently, how long a given node is allowed to hold the token: the *token holding time (THT)*. If we assume that most nodes on the network do not have data to send at any given time—a reasonable assumption, and certainly one that the Ethernet takes advantage of—then we could make a case for letting a node that possesses the token transmit as much data as it has before passing the token on to the next node, in effect setting the THT to infinity. The danger is that a single station could monopolize the ring for an arbitrarily long time, but we could certainly set the THT to significantly more than the time to send one packet.

It is easy to see that the more bytes a node can send each time it has the token, the better the utilization of the ring you can achieve in the situation in which only a single node has data to send. The downside, of course, is that this strategy does not work well when multiple nodes have data to send—it favors nodes that have a lot of data to send over nodes that have only a small message to send, even when it is important to get this small message delivered as soon as possible.

That issue is addressed by the 802.5 protocol's support for different levels of priority. The token contains a 3-bit priority field, so we can think of the token having a certain priority n at any time. Each device that wants to send a packet assigns a priority to that packet, and the device can only seize the token to transmit a packet if the packet's priority is at least as great as the token's. The priority of the token changes over time due to the use of three *reservation* bits in the frame header. For example, a station X waiting to send a priority n packet may set these bits to n if it sees a data frame going past and

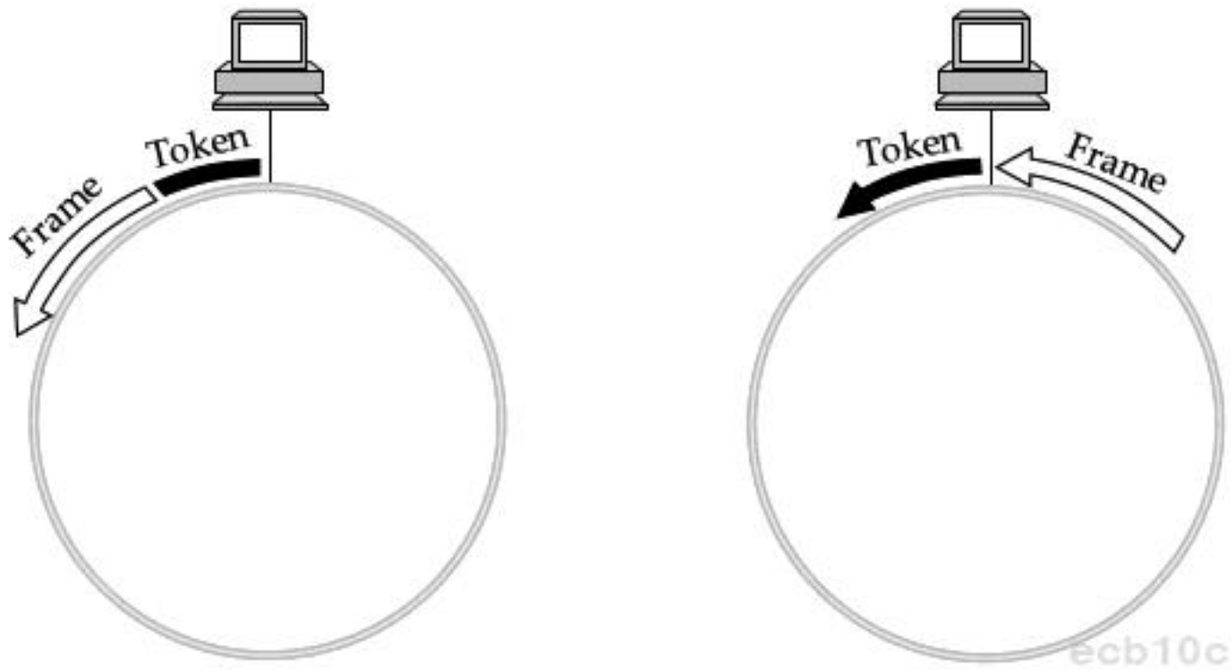

Figure 2.35 Token release: (a) early versus (b) delayed.

the bits have not already been set to a higher value. This causes the station that currently holds the token to elevate its priority to *n* when it releases it. Station X is responsible for lowering the token priority to its old value when it is done.

Note that this is a *strict* priority scheme, in the sense that no lower-priority packets get sent when higher-priority packets are waiting. This may cause lower-priority packets to be locked out of the ring for extended periods if there is a sufficient supply of high-priority packets.

The 802.5 protocol provides a form of reliable delivery using 2 bits in the packet trailer, the A and C bits. These are both 0 initially. When a station sees a frame for which it is the intended recipient, it sets the A bit in the frame. When it copies the frame into its adaptor, it sets the C bit. If the sending station sees the frame come back over the ring with the A bit still 0, it knows that the intended recipient is not functioning or absent. If the A bit is set but not the C bit, this implies that for some reason (e.g., lack of buffer space), the destination could not accept the frame. Thus, the frame might reasonably be retransmitted later in the hope that buffer space had become available.

One final issue will complete our discussion of the MAC protocol, which is the matter of exactly when the sending node releases the token. As illustrated in Figure 2.35, the sender can insert the token back onto the ring immediately following its frame (this is called *early release*) or after the frame it transmits has gone all the way around the ring and been removed (this is called *delayed release*). Clearly early release allows better bandwidth utilization, especially on large rings. 802.5 originally used delayed token release, but support for early release was subsequently added.

2.7.2 Token Ring Maintenance

Each 802.5 token ring has one station designated as a *monitor*. The monitor's job is to ensure the health of the ring by, for example, making sure that the token is not lost. Any station on the ring can become the monitor, and there are defined procedures by which the monitor is elected when the ring is first connected or on the failure of the current monitor. A healthy monitor periodically announces its presence with a special control message; if a station fails to see such a message for some period of time, it will assume that the monitor has failed and will try to become the monitor. The procedures for electing a monitor are the same whether the ring has just come up or the active monitor has just failed.

When a station decides that a new monitor is needed, it transmits a "claim token" frame, announcing its intent to become the new monitor. If that token circulates back to the sender, it can assume that it is okay for it to become the monitor. If some other station is also trying to become the monitor at the same instant, the sender might see a claim token message from that other station first. In this case, it will be necessary to break the tie using some well-defined rule like "highest address wins."

One responsibility of the monitor is to make sure that there is always a token somewhere in the ring, either circulating or currently held by a station. It should be clear that a token may vanish for several reasons, such as a bit error, or a crash on the part of a station that was holding it. To detect a missing token, the monitor watches for a passing token and maintains a timer equal to the maximum possible token rotation time. This interval equals

$$\mathsf{NumStations} \times \mathsf{THT} + \mathsf{RingLatency}$$

where NumStations is the number of stations on the ring, and RingLatency is the total propagation delay of the ring. If the timer expires without the monitor seeing a token, it creates a new one.

The monitor also checks for corrupted or orphaned frames. The former have checksum errors or invalid formats, and without monitor intervention, they could circulate forever on the ring. The monitor drains them off the ring before reinserting the token. An orphaned frame is one that was transmitted correctly onto the ring but whose "parent" died, that is, the sending station went down before it could remove the frame from the ring. These are detected using another header bit, the "monitor" bit. This is 0 on transmission and set to 1 the first time the packet passes the monitor. If the monitor sees a packet with this bit set, it knows the packet is going by for the second time and it drains the packet off the ring.

One additional ring maintenance function is the detection of dead stations. The relays in the MSAU can automatically bypass a station that has been disconnected or powered down, but may not detect more subtle failures. If any station suspects a failure

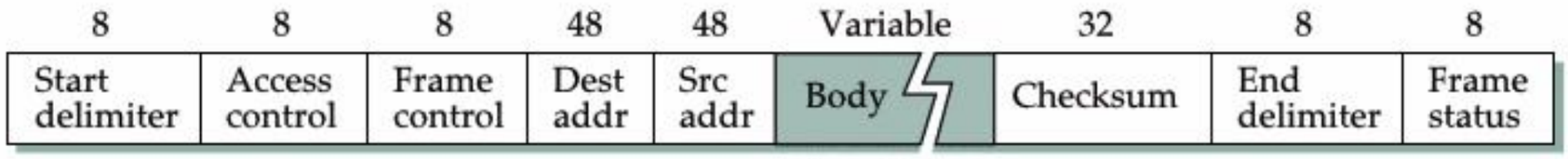

Figure 2.36 802.5/token ring frame format.

on the ring, it can send a *beacon* frame to the suspect destination. Based on how far this frame gets, the status of the ring can be established, and malfunctioning stations can be bypassed by the relays in the MSAU.

Frame Format

We are now ready to define the 802.5 frame format, which is depicted in Figure 2.36. 802.5 uses differential Manchester encoding. This fact is used by the frame format, which uses "illegal" Manchester codes in the start and end delimiters. After the start delimiter comes the access control byte, which includes the frame priority and the reservation priority mentioned above. The frame control byte is a demux key that identifies the higher-layer protocol.

Similar to the Ethernet, 802.5 addresses are 48 bits long. The frame also includes a 32-bit CRC. This is followed by the frame status byte, which includes the A and C bits for reliable delivery.

2.7.3 FDDI

Although FDDI is similar to 802.5 in many respects, there are significant differences. For one, FDDI runs on fiber, not copper (although a later standard, CDDI, was defined to allow copper links to be used). A more interesting difference is that an FDDI network consists of a dual ring—two independent rings that transmit data in opposite directions, as illustrated in Figure 2.37(a). The second ring is not used during normal operation but instead comes into play only if the primary ring fails, as depicted in Figure 2.37(b). That is, the ring loops back on the secondary fiber to form a complete ring, and as a consequence, an FDDI network is able to tolerate a single break in the cable or the failure of one station.

Another interesting difference is that instead of designating one node as a monitor, all the nodes participate equally in maintaining the FDDI ring. Each node maintains an estimate of the token rotation time (TRT)—the expected maximum time for the token to make one complete trip around the ring. A node then measures the time between successive arrivals of the token. If too much time elapses, suggesting that the token has been lost, the node transmits a "claim" frame, in which it includes its current TRT estimate. This claim serves two functions. First, the claim frame is a vote for a particular value of the TRT. Any node that wants to vote for a shorter TRT will replace that claim

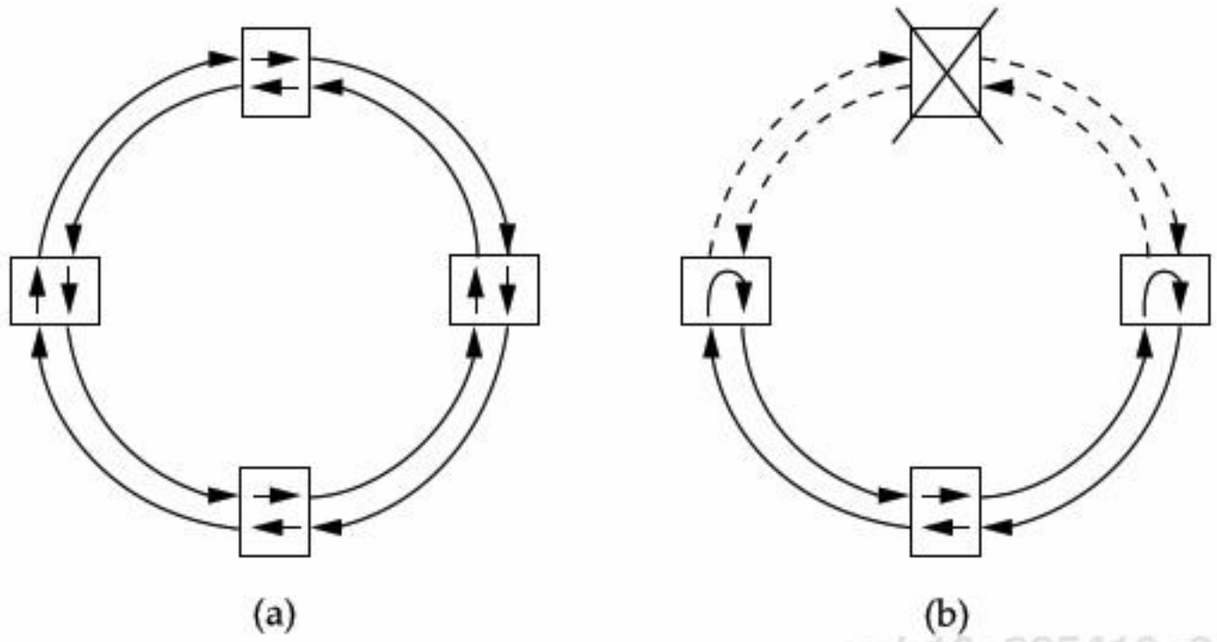

Figure 2.37 Dual-fiber ring: (a) normal operation; (b) failure of the primary ring.

frame with its own claim frame, otherwise it will accept the new time and forward the claim frame. Second, the claim frame is a request for authorization to regenerate the token. If a claim frame makes it all the way back around to the original sender, that node knows not only that the TRT it voted for was the shortest and has been accepted by all the other nodes, but also that it has been authorized to regenerate the token.

When the token arrives at a node with time to spare (i.e., in less than one TRT since the last time this node saw the token), the node can transmit data so long as it does not make the token fall behind schedule; otherwise, the node cannot transmit data. A shortcoming of this basic scheme is that it cannot guarantee any particular node the opportunity to transmit regularly, even if that node has data that is sensitive to jitter, because an upstream node could consume all the available time. To account for this possibility, FDDI defines two classes of traffic: *synchronous* and *asynchronous*. When a node receives a token, it is always allowed to send synchronous data, without regard for whether the token is early or late. In contrast, a node can send asynchronous traffic only when the token is early.

2.7.4 Resilient Packet Ring (802.17)

Resilient Packet Ring (RPR) is a relatively recent technology that has been standardized by the IEEE as 802.17. While it bears some similarity to the ring technologies described above, it was designed with slightly different goals in mind, which led to some key differences in the protocol. Resiliency—the ability to recover quickly from a link or node failure—was a key design goal, to make the technology suitable for service provider networks. Historically this had been provided at lower layers (e.g., by SONET's protection mechanisms). Other design goals included bandwidth efficiency and quality of service (QoS) support, which had quite an impact on the protocol.

Like FDDI, RPR consists of two counterrotating optical fiber rings; unlike FDDI, it takes advantage of the bandwidth of both rings during normal operation. And unlike the previously described rings, an RPR frame is removed from the ring by the receiving node instead of leaving it to be removed by the sender, thereby freeing up some of the bandwidth on the ring in what is called *spatial reuse*.

Most strikingly, RPR does not use tokens. Instead, RPR uses a technique called *buffer insertion*. In a buffer insertion ring, a node can transmit its own frames whenever it has no other frames to forward. If a frame arrives while the node is transmitting its own frame, then the node temporarily buffers that frame. One of the major challenges for buffer insertion rings is how to avoid starvation and enforce QoS guarantees, since in its simplest form a buffer insertion ring could allow a station to hog the link indefinitely. RPR addresses this issue with fairly sophisticated QoS and fairness mechanisms.

RPR supports three QoS classes: class A provides low latency and low jitter (e.g., for phone calls), class B provides predictable latency and jitter (e.g., for prerecorded multimedia), and class C provides a best-effort transport.

To meet the resiliency goals, RPR uses two mechanisms to recover from the failure of a link or node. The first, *wrapping*, is similar to the approach described above for FDDI. The second, *steering*, is more sophisticated: nodes adjacent to the failure notify the other nodes, which are then able to direct packets in the correct (unbroken) direction around the ring toward any given destination, even if that is the "long" way around the ring—assuming the destination is not the node that just failed, of course.

A final interesting aspect of RPR is that it was designed to run over previously defined physical layers, including

Where Are They Now

The Future of Rings

The history of rings has seen them compete against Ethernet and ultimately lose on several occasions. 802.5 eventually lost out to 10-Mbit Ethernet for a variety of reasons, not least of which being the development of switched Ethernet, a topic we will discuss in the next chapter. FDDI was proposed as the faster alternative to Ethernet, but then Ethernet got faster too, and without the need for costly fiber optics, and FDDI never really caught on. The one ring technology that is still seeing some significant deployment is RPR, primarily in metropolitan area networks (MANs), although it seems likely that "metro Ethernet" will eventually come to dominate here just as Ethernet has done in LANs. There is, however, at least one reason RPR has had some success in MANs, which is the fact that rings are something of a natural fit for this kind of network, in a way that they are not in the LAN. Whereas it is cheap enough in a LAN

SONET and the physical layer specified for Ethernet. This saved the designers the time and effort of developing their own physical layer specs and hardware—a good example of the value of layered architectures.

2.8 Wireless

Wireless technologies differ in a variety of dimensions, most notably in how much bandwidth they provide and how far apart communicating nodes can be. Other important differences include which part of the electromagnetic spectrum they use (including whether it requires a license) and how much power they consume (important for mobile nodes). In this section we discuss four prominent wireless technologies: Bluetooth, Wi-Fi (more formally known as 802.11), WiMAX (802.16), and third-generation or 3G cellular wireless. In the following sections we present them in order from shortest range to longest. Table 2.6 gives an overview of these technologies and how they relate to each other.

The most widely used wireless links today are usually asymmetric, that is, the two endpoints are usually different kinds of nodes. One endpoint, sometimes described as the *base station*, usually has no mobility, but has a wired (or at least high bandwidth) connection to the Internet or other networks as in Figure 2.38. The node at the other end of the link—shown here as a "client node"—is often mobile, and relies on its link to the base station for all its communication with other nodes.

Observe that in Figure 2.38 we have used a wavy pair of lines to represent the wireless "link" abstraction provided between two devices (e.g., between a base station and one of its client nodes). One of the interesting aspects of wireless

Where Are They Now

to string cables in a hub-and-spoke manner from a central switch to each workstation, a ring actually provides a very cost-effective way to interconnect nodes in a MAN, where the cost of obtaining rights-of-way and laying fiber can be significant. The resiliency of a ring is also attractive in this environment—the fact that you have both a "clockwise" and an "counterclockwise" path between any two points ensures that a single fiber cut won't cut off a customer. RPR was also developed with some fairness mechanisms that ensure that a node's location on the ring doesn't put it at an unfair advantage or disadvantage to another node in another location when it comes to getting access to the bandwidth—this is harder to achieve with Ethernet. Thus, while there is certainly plenty of momentum behind Ethernet in the MAN, it is probably too soon to predict the demise of RPR in this environment.

	Bluetooth 802.15.1	Wi-Fi 802.11	WiMAX 802.16	3G Cellular
Typical link length	10 m	100 m	10 km	Tens of km
Typical bandwidth	2.1 Mbps (shared)	54 Mbps (shared)	70 Mbps (shared)	384+ Kbps (per connection)
Typical use	Link a peripheral to a notebook computer	Link a notebook computer to a wired base	Link a building to a wired tower	Link a cell phone to a wired tower
Wired technology analogy	USB	Ethernet	Coaxial cable	DSL

Table 2.6 Overview of leading wireless technologies.

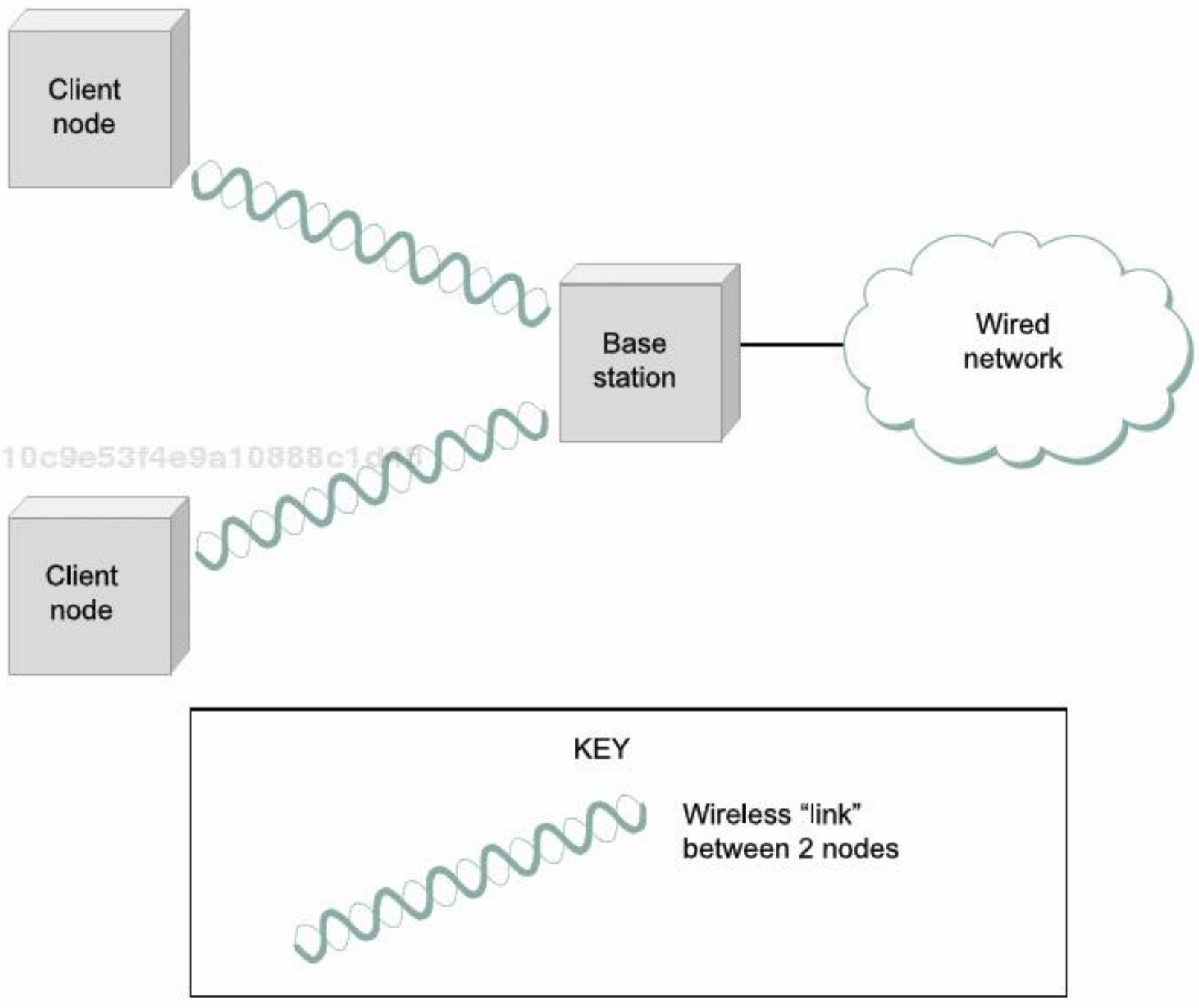

Figure 2.38 A wireless network using a base station.

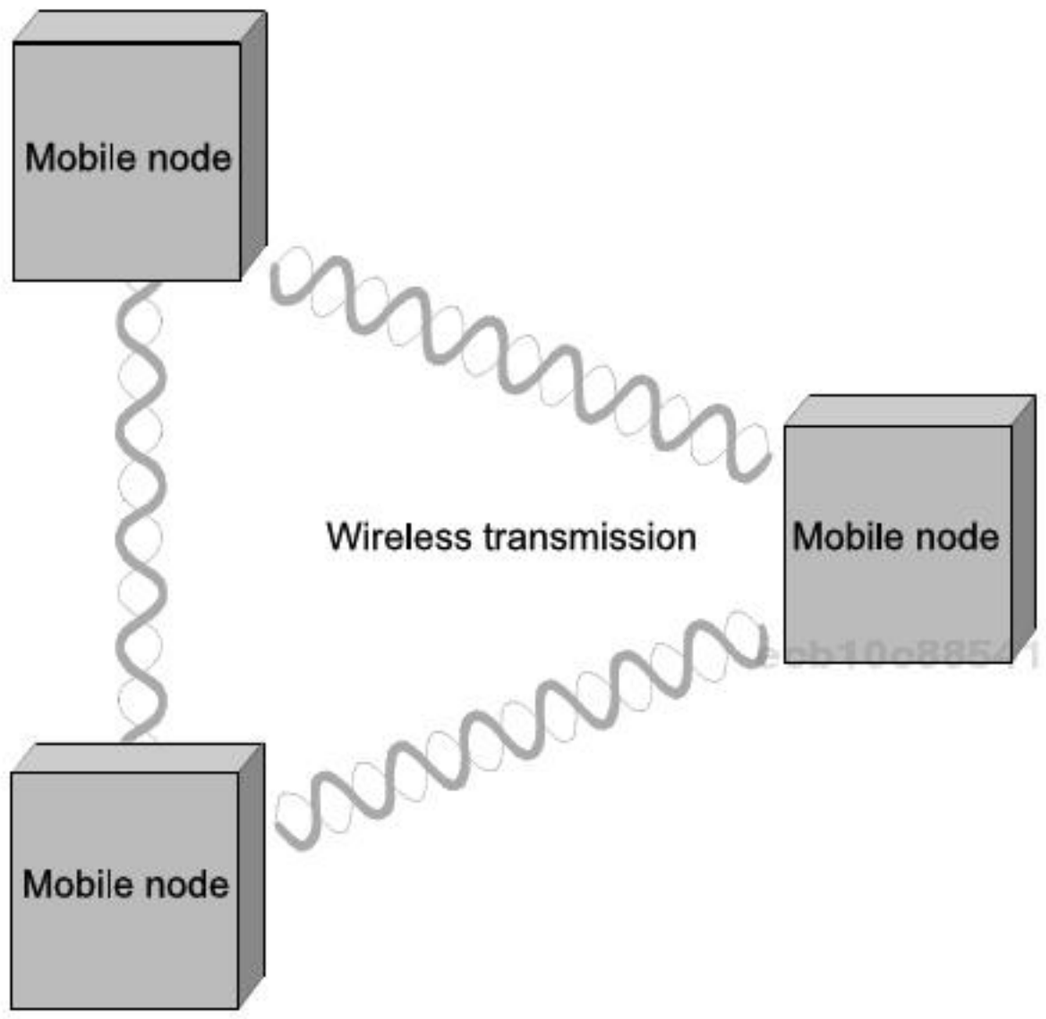

Figure 2.39 A wireless ad hoc or mesh network.

communication is that it naturally supports point-to-multipoint communication, because radio waves sent by one device can be simultaneously received by many devices. However, it is often useful to create a point-to-point link abstraction for higher-layer protocols, and we will see examples of how this works later in this section.

Note that in Figure 2.38, communication between nonbase (client) nodes is routed via the base station. This is in spite of the fact that radio waves emitted by one client node may well be received by other client nodes—the common base station model does not permit direct communication between the client nodes.

This topology implies three qualitatively different levels of mobility. The first level is no mobility, such as when a receiver must be in a fixed location to receive a directional transmission from the base station, as is the case with the initial version of WiMAX. The second level is mobility within the range of a base, as is the case with Bluetooth. The third level is mobility between bases, as is the case with cell phones and Wi-Fi.

An alternative topology that is seeing increasing interest is the *mesh* or *ad hoc* network. In a wireless mesh, nodes are peers (i.e., there is no special base station node). Messages may be forwarded via a chain of peer nodes as long as each node is within range of the preceding node. This is illustrated in Figure 2.39. This allows the wireless portion of a network to extend beyond the limited range of a single radio. From the point of view of competition between technologies, this allows a shorter-range technology to

extend its range and potentially compete with a longer-range technology. Meshes also offer fault tolerance by providing multiple routes for a message to get from point A to point B. A mesh network can be extended incrementally, with incremental costs. On the other hand, a mesh requires nonbase nodes to have a certain level of sophistication in their hardware and software, potentially increasing per-unit costs—and power consumption, a critical consideration for battery-powered devices. Wireless mesh networks are of considerable research interest, but they are still in their relative infancy compared to networks with base stations, and thus we do not cover them further here.

We now turn our attention to the details of the four wireless technologies mentioned above, beginning with the most short-range technology, Bluetooth.

2.8.1 Bluetooth (802.15.1)

Bluetooth fills the niche of very short-range communication between mobile phones, PDAs, notebook computers, and other personal or peripheral devices. For example, Bluetooth can be used to connect a mobile phone to a headset, or a notebook computer to a printer. Roughly speaking, Bluetooth is a more convenient alternative to connecting two devices with a wire. In such applications, it is not necessary to provide much range or bandwidth. This is fortunate for some of the target battery-powered devices, since it is important that they not consume much power.

Bluetooth operates in the license-exempt band at 2.45 GHz. It has a range of only about 10 m. For this reason, and because the communicating devices typically belong to one individual or group, Bluetooth is sometimes categorized as a personal area network (PAN). Version 2.0 provides speeds up to 2.1 Mbps. Power consumption is low.

Bluetooth is specified by an industry consortium called the Bluetooth Special Interest Group. It specifies an entire suite of protocols, going beyond the link layer to define application protocols, which it calls *profiles*, for a range of applications. For example, there is a profile for synchronizing a PDA with a personal computer. Another profile gives a mobile computer access to a wired LAN in the manner of 802.11, although this was not Bluetooth's original goal. The IEEE 802.15.1 standard is based on Bluetooth but excludes the application protocols.

The basic Bluetooth network configuration, called a *piconet*, consists of a master device and up to seven slave devices, as in Figure 2.40. Any communication is between the master and a slave; the slaves do not communicate directly with each other. Because slaves have a simpler role, their Bluetooth hardware and software can be simpler and cheaper.

Since Bluetooth operates in an license-exempt band, it is required to use a spread spectrum technique (as discussed in Section 2.1.2) to deal with possible interference in the band. It uses frequency hopping with 79 *channels* (frequencies), using each for 625 μm at a time. This provides a natural time slot for Bluetooth to use for synchronous time division multiplexing. A frame takes up 1, 3, or 5 consecutive time slots. Only

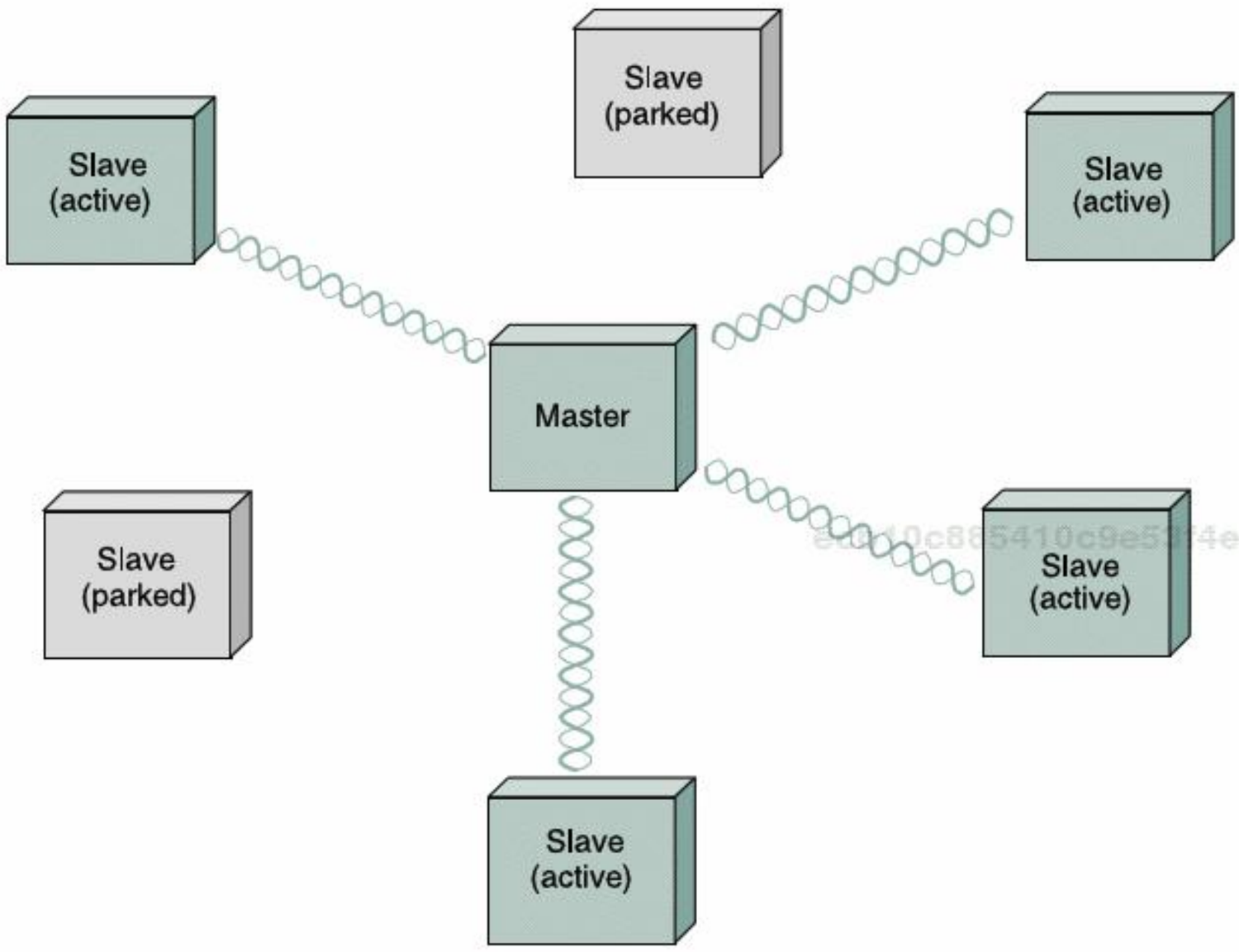

Figure 2.40 A Bluetooth piconet.

the master can start to transmit in odd-numbered slots. A slave can start to transmit in an even-numbered slot, but only in response to a request from the master during the previous slot, thereby preventing any contention between the slave devices.

A slave device can be *parked*: set to an inactive, low-power state. A parked device cannot communicate on the piconet; it can only be reactivated by the master. A piconet can have up to 255 parked devices in addition to its active slave devices.

ZigBee is a newer technology that competes with Bluetooth to some extent. Devised by the ZigBee alliance and standardized as IEEE 802.15.4, it is designed for situations where the bandwidth requirements are low and power consumption must be very low to give very long battery life. It is also intended to be simpler and cheaper than Bluetooth, making it financially feasible to incorporate in cheaper devices such as a wall switch that wirelessly communicates with a ceiling-mounted fan.

2.8.2 Wi-Fi (802.11)

This section takes a closer look at a specific technology centered around the emerging IEEE 802.11 standard, also known as *Wi-Fi*.[6] Wi-Fi is technically a trademark, owned by

[6] There is some debate over whether Wi-Fi stands for "wireless fidelity," by analogy to Hi-Fi, or whether it is just a catchy name that doesn't stand for anything other than 802.11.

a trade group called the Wi-Fi alliance, that certifies product compliance with 802.11. Like its Ethernet and token ring siblings, 802.11 is designed for use in a limited geographical area (homes, office buildings, campuses), and its primary challenge is to mediate access to a shared communication medium—in this case, signals propagating through space. 802.11 supports additional features (e.g., time-bounded services, power management, and security mechanisms), but we focus our discussion on its base functionality.

Physical Properties

802.11 runs over six different physical layer protocols (so far). Five are based on spread spectrum radio, and one on diffused infrared (and is of historical interest only at this point). The fastest runs at a maximum of 54 Mbps.

The original 802.11 standard defined two radio-based physical layers standards, one using frequency hopping (over 79 1-MHz-wide frequency bandwidths) and the other using direct sequence (with an 11-bit chipping sequence). Both provide up to 2 Mbps. Then physical layer standard 802.11b was added. Using a variant of direct sequence, 802.11b provides up to 11 Mbps. These three standards run in the license-exempt 2.4 GHz frequency band of the electromagnetic spectrum. Then came 802.11a, which delivers up to 54 Mbps using a variant of FDM called *orthogonal frequency division multiplexing (OFDM)*. 802.11a runs in the license-exempt 5-GHz band. On one hand, this band is less used, so there is less interference. On the other hand, there is more absorption of the signal and it is limited to almost line of sight. The most recent standard is 802.11g, which is backward compatible with 802.11b (and returns to the 2.4-GHz band). 802.11g uses OFDM and delivers up to 54 Mbps. It is common for commercial products to support all three of 802.11a, 802.11b, and 802.11g, which not only ensures compatibility with any device that supports any one of the standards, but also makes it possible for two such products to choose the highest bandwidth option for a particular environment.

Collision Avoidance

At first glance, it might seem that a wireless protocol would follow the same algorithm as the Ethernet—wait until the link becomes idle before transmitting and back off should a collision occur—and to a first approximation, this is what 802.11 does. The additional complication for wireless is that, while a node on an Ethernet receives every other node's transmissions, a node on an 802.11 network may be too far from certain other nodes to receive their transmissions (and vice versa).

Consider the situation depicted in Figure 2.41, where A and C are both within range of B but not each other. Suppose both A and C want to communicate with B and so they each send it a frame. A and C are unaware of each other since their signals do not carry that far. These two frames collide with each other at B, but unlike an Ethernet,

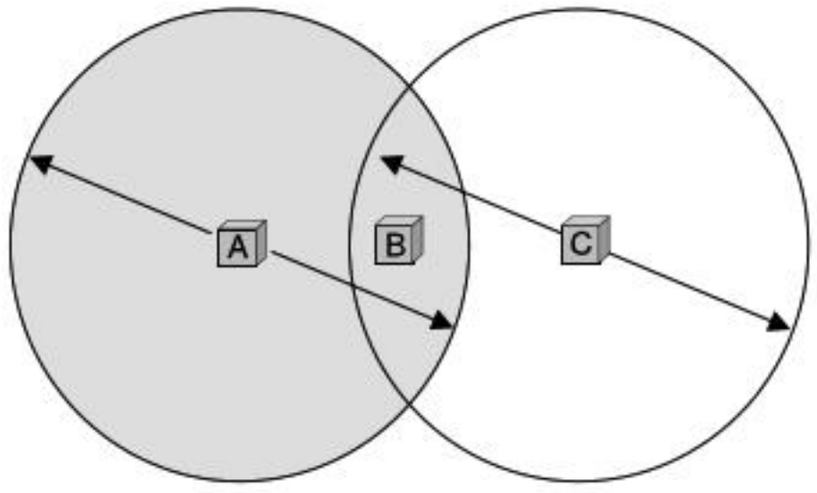

Figure 2.41 The hidden node problem. Although A and C are hidden from each other, their signals can collide at B. (B's reach is not shown.)

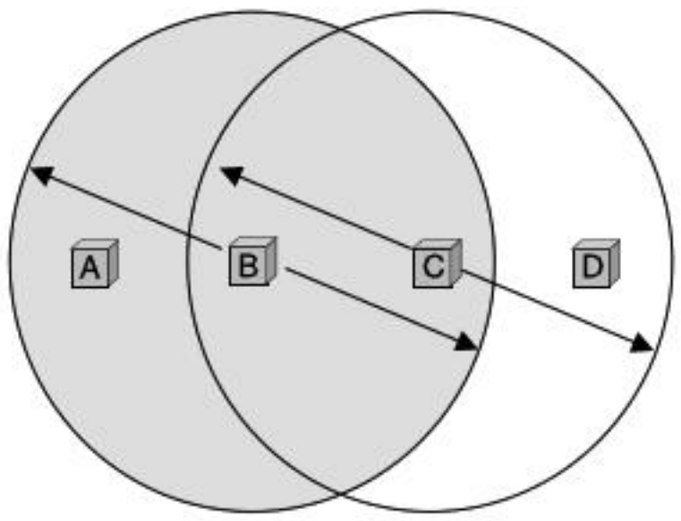

Figure 2.42 The exposed node problem. Although B and C are exposed to each other's signals, there is no interference if B transmits to A while C transmits to D. (A's and D's reaches are not shown.)

neither A nor C is aware of this collision. A and C are said to be *hidden nodes* with respect to each other.

A related problem, called the *exposed node problem*, occurs under the circumstances illustrated in Figure 2.42, where each of the four nodes is able to send and receive signals that reach just the nodes to its immediate left and right. For example, B can exchange frames with A and C but it cannot reach D, while C can reach B and D but not A. Suppose B is sending to A. Node C is aware of this communication because it hears B's transmission. It would be a mistake, however, for C to conclude that it cannot transmit to anyone just because it can hear B's transmission. For example, suppose C wants to transmit to node D. This is not a problem since C's transmission to D will not interfere with A's ability to receive from B. (It would interfere with A sending to B, but B is transmitting in our example.)

802.11 addresses these two problems with an algorithm called *multiple access with collision avoidance (MACA)*. The idea is for the sender and receiver to exchange control frames with each other before the sender actually transmits any data. This exchange informs all nearby nodes that a transmission is about to begin. Specifically, the sender transmits a *Request to Send (RTS)* frame to the receiver; the RTS frame includes a field that indicates how long the sender wants to hold the medium (i.e., it specifies the length of the data frame to be transmitted). The receiver then replies with a *Clear to Send (CTS)* frame; this frame echoes this length field back to the sender. Any node that sees the CTS frame knows that it is close to the receiver, and therefore cannot transmit for the period of time it takes to send a frame of the specified length. Any node that sees the RTS frame but not the CTS frame is not close enough to the receiver to interfere with it, and so is free to transmit.

There are two more details to complete the picture. First, the receiver sends an ACK to the sender after successfully receiving a frame. All nodes must wait for this ACK before trying to transmit.[7] Second, should two or more nodes detect an idle link and try to transmit an RTS frame at the same time, their RTS frames will collide with each other. 802.11 does not support collision detection, but instead the senders realize the collision has happened when they do not receive the CTS frame after a period of time, in which case they each wait a random amount of time before trying again. The amount of time a given node delays is defined by the same exponential backoff algorithm used on the Ethernet (see Section 2.6.2).

Distribution System

As described so far, 802.11 would be suitable for a network with a mesh (ad hoc) topology, and development of an 802.11s standard for mesh networks is nearing completion. At the current time, however, nearly all 802.11 networks use a base-station-oriented topology.

Instead of all nodes being created equal, some nodes are allowed to roam (e.g., your laptop) and some are connected to a wired network infrastructure. 802.11 calls these base stations *access points (AP)*, and they are connected to each other by a so-called *distribution system*. Figure 2.43 illustrates a distribution system that connects three access points, each of which services the nodes in some region. The details of the distribution system are not important to this discussion—it could be an Ethernet or a token ring, for example. The only important point is that the distribution network runs at layer 2 of the ISO architecture (the link layer), that is, it operates at the same protocol layer as the wireless links. In other words, it does not depend on any higher-level protocols (such as the network layer).

[7] This ACK was not part of the original MACA algorithm, but was instead proposed in an extended version called MACAW: MACA for *Wireless* LANs.

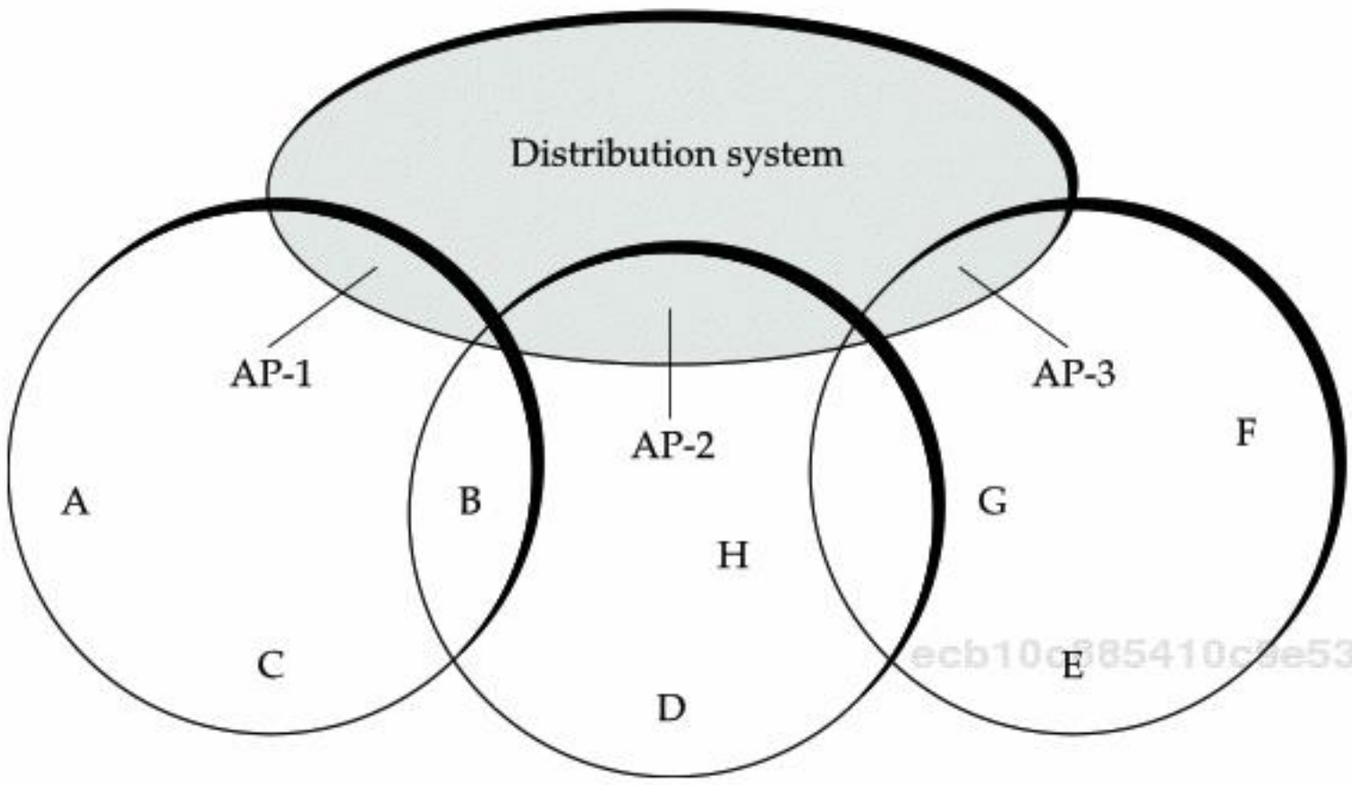

Figure 2.43 Access points connected to a distribution network.

Although two nodes can communicate directly with each other if they are within reach of each other, the idea behind this configuration is that each node associates itself with one access point. For node A to communicate with node E, for example, A first sends a frame to its access point (AP-1), which forwards the frame across the distribution system to AP-3, which finally transmits the frame to E. How AP-1 knew to forward the message to AP-3 is beyond the scope of 802.11; it may have used the bridging protocol described in the next chapter (Section 3.2). What 802.11 does specify is how nodes select their access points and, more interestingly, how this algorithm works in light of nodes moving from one cell to another.

The technique for selecting an AP is called *scanning* and involves the following four steps:

1. The node sends a **Probe** frame;
2. All APs within reach reply with a **Probe Response** frame;
3. The node selects one of the access points, and sends that AP an **Association Request** frame;
4. The AP replies with an **Association Response** frame.

A node engages this protocol whenever it joins the network, as well as when it becomes unhappy with its current AP. This might happen, for example, because the signal from its current AP has weakened due to the node moving away from it. Whenever a node

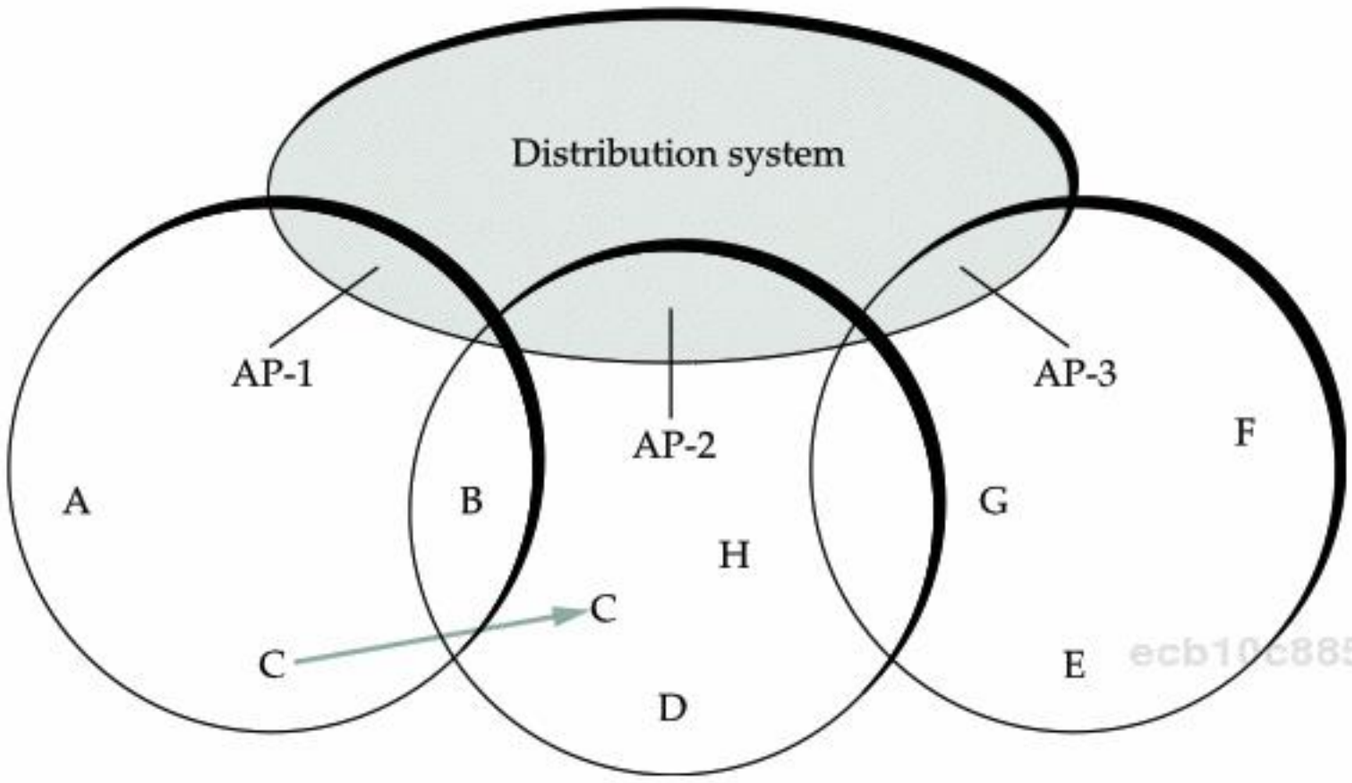

Figure 2.44 Node mobility.

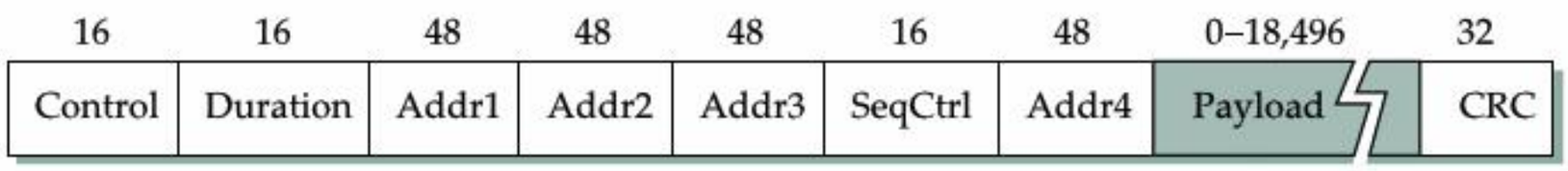

Figure 2.45 802.11 frame format.

acquires a new AP, the new AP notifies the old AP of the change (this happens in step 4) via the distribution system.

Consider the situation shown in Figure 2.44, where node C moves from the cell serviced by AP-1 to the cell serviced by AP-2. As it moves, it sends Probe frames, which eventually result in Probe Response frames from AP-2. At some point, C prefers AP-2 over AP-1, and so it associates itself with that access point.

The mechanism just described is called active scanning since the node is actively searching for an access point. APs also periodically send a Beacon frame that advertises the capabilities of the access point; these include the transmission rates supported by the AP. This is called passive scanning, and a node can change to this AP based on the Beacon frame simply by sending an Association Request frame back to the access point.

Frame Format

Most of the 802.11 frame format, which is depicted in Figure 2.45, is exactly what we would expect. The frame contains the source and destination node addresses, each of which are 48 bits long, up to 2,312 bytes of data, and a 32-bit CRC. The Control

field contains three subfields of interest (not shown): a 6-bit Type field that indicates whether the frame carries data, is an RTS or CTS frame, or is being used by the scanning algorithm; and a pair of 1-bit fields—called ToDS and FromDS—that are described below.

The peculiar thing about the 802.11 frame format is that it contains four, rather than two, addresses. How these addresses are interpreted depends on the settings of the ToDS and FromDS bits in the frame's Control field. This is to account for the possibility that the frame had to be forwarded across the distribution system, which would mean that the original sender is not necessarily the same as the most recent transmitting node. Similar reasoning applies to the destination address. In the simplest case, when one node is sending directly to another, both the DS bits are 0, Addr1 identifies the target node, and Addr2 identifies the source node. In the most complex case, both DS bits are set to 1, indicating that the message went from a wireless node onto the distribution system, and then from the distribution system to another wireless node. With both bits set, Addr1 identifies the ultimate destination, Addr2 identifies the immediate sender (the one that forwarded the frame from the distribution system to the ultimate destination), Addr3 identifies the intermediate destination (the one that accepted the frame from a wireless node and forwarded it across the distribution system), and Addr4 identifies the original source. In terms of the example given in Figure 2.43, Addr1 corresponds to E, Addr2 identifies AP-3, Addr3 corresponds to AP-1, and Addr4 identifies A.

2.8.3 WiMAX (802.16)

WiMAX, which stands for Worldwide Interoperability for Microwave Access, was designed by the WiMAX Forum and standardized as IEEE 802.16. It was originally conceived as a last-mile technology (Section 2.1.2). In WiMAX's case that "mile" is typically 1 to 6 miles, with a maximum of about 30 miles, leading to WiMAX being classified as a metropolitan area network (MAN). In keeping with a last-mile role, WiMAX does not incorporate mobility at the time of this writing, although efforts to add mobility are nearing completion as IEEE 802.16e. Also in keeping with the last-mile niche, WiMAX's client systems, called *subscriber stations*, are assumed to be not end-user computing devices, but rather systems that multiplex all the communication of the computing devices being used in a particular building. WiMAX provides up to 70 Mbps to a single subscriber station.

In order to adapt to different frequency bands and different conditions, WiMAX defines several physical layer protocols. The original WiMAX physical layer protocol is designed to use frequencies in the 10- to 66-GHz range. In this range waves travel in straight lines, so communication is limited to line-of-sight (LOS). A WiMAX base station uses multiple antennas pointed in different directions; the area covered by one

antenna's signal is a *sector*. To extend WiMAX to near-line-of-sight and nonline-of-sight situations, several physical layer protocols were added that use the frequencies below 11 GHz (in the 10- to 11-GHz range, WiMAX can use either the original physical layer or one of the newer ones). Since this range includes both licensed and license-exempt frequencies, each of these physical layer protocols defines a variant better adapted to the additional interference and the regulatory constraints of the license-exempt frequencies.

The physical layer protocols provide two ways to divide the bandwidth between upstream (i.e., from subscribers to base station) and downstream traffic: time division duplexing (TDD) and frequency division duplexing (FDD). TDD is simply STDM of the two streams; they take turns using the same frequency, and the proportion of upstream to downstream time can be varied dynamically, *adaptively*, by the base station. FDD is simply FDM of the two streams; one frequency is used for upstream and another for downstream. In license-exempt bands, the protocols use only TDD.

Both channels, upstream and downstream, must be shared not just among the many subscriber stations in a given sector, but also among the many WiMAX *connections* that each subscriber can have with the base station. WiMAX—unlike 802.11 and Ethernet—is connection oriented. One reason for this is to be able to offer a variety of QoS guarantees regarding properties such as latency and jitter, with the aim of supporting high-quality telephony and high-volume multimedia in addition to bursty data traffic. This is conceptually similar to some of the wired last mile technologies (such as DSL) with which WiMAX is intended to compete.

Sharing of the upstream and downstream channels is based on dividing them into equal-sized time slots. A WiMAX frame generally takes up multiple slots, with different frames taking different numbers of slots. The downstream channel (from base to subscribers) is relatively easy to subdivide into connections since only the base station sends on that channel. The base station simply sends addressed frames, one after the other. Each subscriber station in the sector receives all the frames, but ignores those not addressed to one of its connections.

In the upstream direction, how a connection gets handled depends on its QoS parameters. Some connections get slots at a fixed rate, some get polled to determine how many slots they need currently, and some must request slots whenever they need them. Connections in this last category must contend to place their requests in a limited number of upstream slots set aside for contention. They use an exponential backoff algorithm to minimize the chance of a collision, even on the first attempt.

A European alternative to WiMAX is HIPERMAN, which stands for high-performance radio metropolitan area network and uses the 2- to 11-GHz range. South Korea's WiBro (for wireless broadband) technology operates at 2.3 GHz, and is being brought into line with the emerging IEEE 802.16e standard for mobile WiMAX.

2.8.4 Cell Phone Technologies

Cell phone technology seems an obvious approach to mobile computer communication, and indeed data services based on cellular standards are commercially available. One drawback is the cost to users, due in part to cellular's use of licensed spectrum (which has historically been sold off to cellular phone operators for astronomical sums). The frequency bands that are used for cellular telephones (and now for cellular data) vary around the world. In Europe, for example, the main bands for cellular phones are at 900 and 1,800 MHz. In North America, 850- and 1,900-MHz bands are used. This global variation in spectrum usage creates problems for users who want to travel from one part of the world to another, and has created a market for phones that can operate at multiple frequencies (e.g., a tri-band phone can operate at three of the four frequency bands mentioned above). That problem, however, pales in comparison to the proliferation of incompatible standards that have plagued the cellular communication business. Only recently have some signs of convergence on a small set of standards appeared. And finally, there is the problem that most cellular technology was designed for voice communication, and is only now starting to support moderately high-bandwidth data communication.

Like 802.11 and WiMAX, cellular technology relies on the use of base stations that are part of a wired network. The geographic area served by a base station's antenna is called a *cell*. A base station could serve a single cell, or use multiple directional antennas to serve multiple cells. Cells don't have crisp boundaries, and they overlap. Where they overlap, a mobile phone could potentially communicate with multiple base stations. This is somewhat similar to the 802.11 picture shown in Figure 2.43. At any time, however, the phone is in communication with, and under the control of, just one base station. As the phone begins to leave a cell, it moves into an area of overlap with one or more other cells. The current base station senses the weakening signal from the phone, and gives control of the phone to whichever base station is receiving the strongest signal from it. If the phone is involved in a call at the time, the call must be transferred to the new base station in what is called a *handoff*.

As we noted above, there is not one unique standard for cellular, but rather a collection of competing technologies that support data traffic in different ways and deliver different speeds. These technologies are loosely categorized by "generation." The first generation (1G) was analog, and thus of limited interest from a data communications perspective. Most of the cell phone technology currently deployed is considered second generation (2G) or "2.5G" (not quite worthy of being called 3G, but more advanced than 2G). The 2G and later technologies are digital. The most widely deployed 2G technology is referred to as GSM—the Global System for Mobile Communications, which is used in more than 200 countries. North America, however, is a late adopter of GSM, which helped prolong the proliferation of competing standards.

Most 2G technologies use one of two approaches to sharing a limited amount of spectrum between simultaneous calls. One way is a combination of FDM and TDM. The spectrum available is divided into disjoint frequency bands, and each band is subdivided into time slots. A given call is allocated every *n*th slot in one of the bands. The other approach is code division multiple access (CDMA). CDMA does not divide the channel in either time or frequency, but rather uses different chipping codes to distinguish the transmissions of different cellphone users. (See Section 2.1.2 for a discussion of chipping codes.)

The 2G and later cell phone technologies use compression algorithms tailored to human speech to compress voice data to about 8 Kbps without losing quality. Since 2G technologies focus on voice communication, they provide connections with just enough bandwidth for that compressed speech—not enough for a decent data link. One of the first cellular data standards to gain widespread adoption is the General Packet Radio Service (GPRS), which is part of the GSM set of standards and is often referred to as a 2.5G technology.

GSM networks make use of a multiplexing technique called time division multiple access (TDMA). (Confusingly, there is also a particular cellular *standard* that is sometimes called TDMA, but is known formally as IS-136.) You can think of TDMA as being like TDM (time division multiplexing)—traditionally used for telephone services—with the additional feature that the timeslots can be dynamically allocated to users or devices that need them (and deallocated from devices that no longer need them). The number of timeslots that are available for GPRS at a given frequency depends on how many cellular voice calls are currently in progress, since voice calls also consume timeslots. As a result, GPRS data rates tend to be lower in busy cells. In practice, users often get between 30 and 70 Kbps—coincidentally, just about the same as a user of a dial-up modem on a landline. Nevertheless, GPRS has proven quite useful and popular in some parts of the world as a way to communicate wirelessly when faster connection methods (such as 802.11) are not available. Other 2.5G data standards have also become available and some manage to be quite a bit higher in bandwidth than GPRS.

The concept of a third generation (3G) was established before there was any implementation of 3G technologies, with the aim of shaping a single international standard that would provide much higher data bandwidth. Unfortunately, at the time of writing, several mutually incompatible 3G standards are emerging. Thus, the possibility that cellular standards will continue to diverge seems quite realistic. Interestingly, all the 3G standards are based on variants of CDMA. For example, the Universal Mobile Telecommunications System (UMTS) is based on wideband CDMA (W-CDMA). UMTS appears poised to be the successor to GSM, and in fact is sometimes referred to as 3GSM (i.e., the third generation version of GSM). UMTS is intended to support data transfer rates of up to 1.92 Mbps, although

real network conditions will probably result in lower rates in practice. Nevertheless, it should represent a significant performance improvement over GPRS. And like GSM, it should have quite widespread (if not actually universal) adoption around the world.

There are a number of commercial UMTS networks in operation at the time of writing with many more announced or planned. And to make it quite clear that work in this field is far from complete, we note that 3.5G and 4G standards are also in the works.

Finally, it should be noted that there is a class of mobile phones that are not cellular phones but satellite phones, or *satphones*. Satphones use communication satellites as base stations, communicating on frequency bands that have been reserved internationally for satellite use. Consequently, service is available even where there are no cellular base stations. Satphones are rarely used where cellular is available, since service is typically much more expensive. Satphones are also larger and heavier than modern cell phones because of the need to transmit and receive over much longer distances, to reach satellites rather than cellphone towers. Satellite communication is more extensively used in television and radio broadcasting, taking advantage of the fact that the signal is broadcast, not point-to-point. High-bandwidth data communication via satellite is commercially available, but its relatively high price (for both equipment and service) limits its use to regions where no alternative is available.

2.9 Summary

This chapter introduced the hardware building blocks of a computer network—nodes and links—and discussed the five key problems that must be solved so that two or more nodes that are directly connected by a physical link can exchange messages with each other. In practice, most of the algorithms that address these five problems are implemented on the adaptor that connects the host to the link. It turns out that the design of this adaptor, and how the rest of the host interacts with it, is of critical importance in how well the network performs overall.

The first problem is to encode the bits that make up a binary message into the signal at the source node and then to recover the bits from the signal at the receiving node. This is the encoding problem, and it is made challenging by the need to keep the sender's and receiver's clocks synchronized. We discussed four different encoding techniques—NRZ, NRZI, Manchester, and 4B/5B—which differ largely in how they encode clock information along with the data being transmitted. One of the key attributes of an encoding scheme is its efficiency, that is, the ratio of signal pulses to encoded bits.

Once it is possible to transmit bits between nodes, the next step is to figure out how to package these bits into frames. This is the framing problem, and it boils down

to being able to recognize the beginning and end of each frame. Again, we looked at several different techniques, including byte-oriented protocols, bit-oriented protocols, and clock-based protocols.

Assuming that each node is able to recognize the collection of bits that make up a frame, the third problem is to determine if those bits are in fact correct, or if they have possibly been corrupted in transit. This is the error detection problem, and we looked at three different approaches: cyclic redundancy check, two-dimensional parity, and checksums. Of these, the CRC approach gives the strongest guarantees and is the most widely used at the link level.

Given that some frames will arrive at the destination node containing errors and thus will have to be discarded, the next problem is how to recover from such losses. The goal is to make the link appear reliable. The general approach to this problem is called ARQ and involves using a combination of acknowledgments and timeouts. We looked at three specific ARQ algorithms: stop-and-wait, sliding window, and concurrent channels. What makes these algorithms interesting is how effectively they use the link, with the goal being to keep the pipe full.

The final problem is not relevant to point-to-point links, but it is the central issue in multiple-access links: how to mediate access to a shared link so that all nodes eventually have a chance to transmit their data. In this case, we looked at a variety of media access protocols—Ethernet, token ring, and several wireless protocols—which have been put to practical use in building local area networks. The Ethernet and token ring media access protocols are notable for their distributed nature—there is no central arbitrator of access. Media access in wireless networks is made more complicated by the fact that some nodes may be "hidden" from each other due to range limitations of radio transmission. Most of the common wireless protocols today designate some nodes as "wired" or "base-station" nodes, while the other "mobile" nodes communicate with a base station. Wireless standards and technologies are rapidly evolving, with mesh networks, in which all nodes communicate as peers, now starting to emerge.

OPEN ISSUE
Sensor Networks

A sensor network is a wireless network of many nodes—up to tens of thousands—whose purpose is to monitor some aspect of the geographic area over which it is spread. The nodes are equipped with one or more types of sensor that allow them to detect, for example, sound, motion, radiation, or chemicals. Some example applications of sensor networks are monitoring a battleground to detect the locations of enemy forces, monitor-

ing a natural environment for pollutants or seismic activity, and monitoring temperature throughout a building to optimize climate control.

The nodes in a sensor network must be low cost because of the quantity involved, and must use very little power because they are generally battery powered. These minimal nodes are perhaps better described as devices rather than computers. There is an open-source operating system called TinyOS designed specifically for the constraints of these devices. Researchers are pursuing development of *smart dust*—sensor network nodes called *motes* whose size is on the order of millimeters.

Although the sensor information is ultimately routed to a base station, most of the nodes are not directly linked to the base station. Instead they communicate only with their nearest neighbors, who forward the data to their neighbors until it reaches the base station. This uses less power than transmitting over a longer distance, and allows the sensor network to extend beyond the range of a single link. One of the open questions about sensor networks is how a node should determine which node to transmit or forward data to. In one technique, the nodes form clusters. Each cluster designates one node as *cluster head*, and all data is routed via cluster heads. In a technique that blurs the line between network and application, nodes called *aggregation points* collect and process the data they receive from neighboring nodes, then transmit the processed data. By processing the data incrementally, instead of forwarding all the raw data to the base station, the amount of traffic in the network (and the power consumed) is reduced. Further complicating the issue of how to organize the network is the possibility of nodes failing, perhaps because of battery exhaustion, and the possibility of nodes being dynamically added to an existing network.

Another open issue for sensor networks is *localization* or *location discovery*—determining the locations of nodes. Suppose the nodes are deployed by dropping them from an aircraft, as might be the case for, say, military or environmental monitoring. Then neither the node nor the base station would know where a node is. And yet that geographical information is crucial; we need to know the location of that seismic activity or enemy tank. GPS is considered too expensive and consumes too much power for the majority of nodes. A typical solution requires a few nodes called *beacons* to determine their own absolute locations based on GPS or manual configuration. The majority of nodes can then derive their absolute location by combining an estimate of their position relative to the beacons with the absolute location information provided by the beacons.

FURTHER READING

One of the most important contributions in computer networking over the last 20 years is the original paper by Metcalf and Boggs (1976) introducing the Ethernet. Many years later, Boggs, Mogul, and Kent (1988) reported their practical experiences with Ethernet,

debunking many of the myths that had found their way into the literature over the years. Both papers are must reading. The third and fourth papers discuss the issues involved in integrating high-speed network adaptors with system software.

- Metcalf, R., and D. Boggs. "Ethernet: Distributed Packet Switching for Local Computer Networks." *Communications of the ACM* 19(7):395–403, July 1976.
- Boggs, D., J. Mogul, and C. Kent. "Measured Capacity of an Ethernet." *Proceedings of the SIGCOMM '88 Symposium*, pp. 222–234, August 1988.
- Metcalf, R. "Computer/Network Interface Design Lessons from Arpanet and Ethernet." *IEEE Journal of Selected Areas in Communication (JSAC)* 11(2):173–180, February 1993.
- Druschel, P., M. Abbott, M. Pagels, and L. L. Peterson. "Network Subsystem Design." *IEEE Network (Special Issue on End-System Support for High-Speed Networks)* 7(4):8–17, July 1993.

There are countless textbooks with a heavy emphasis on the lower levels of the network hierarchy, with a particular focus on *telecommunications*—networking from the phone company's perspective. Books by Spragins et al. [SHP91] and Minoli [Min93] are two good examples. Several other books concentrate on various local area network technologies. Of these, Stallings's book is the most comprehensive [Sta00], while Jain gives a thorough description of FDDI [Jai94]. Jain's book also gives a good introduction to the low-level details of optical communication. Also, a comprehensive overview of FDDI can be found in Ross's article [Ros86].

For an introduction to information theory, Blahut's book is a good place to start [Bla87], along with Shannon's seminal paper on link capacity [Sha48].

For a general introduction to the mathematics behind error codes, Rao and Fujiwara [RF89] is recommended. For a detailed discussion of the mathematics of CRCs in particular, along with some more information about the hardware used to calculate them, see Peterson and Brown [PB61].

On the topic of network adaptor design, much work was done in the early 1990s by researchers trying to connect hosts to networks running at higher and higher rates. In addition to the two examples given in the reading list, see Traw and Smith [TS93], Ramakrishnan [Ram93], Edwards et al. [EWL+94], Druschel et al. [DPD94], Kanakia and Cheriton [KC88], Cohen et al. [CFFD93], and Steenkiste [Ste94a]. Recently, a new generation of interface cards, ones that utilize *network processors*, are coming onto the market. Spalink et al. demonstrate how these new processors can be programmed to implement various network functionality [SKPG01].