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.

Figure 4-13. The most common kinds of Ethernet cabling.

Name	Cable	Max. seg.	Nodes/seg.	Advantages
10Base5	Thick coax	500 m	100	Original cable; now obsolete
10Base2	Thin coax	185 m	30	No hub needed
10Base-T	Twisted pair	100 m	1024	Cheapest system
10Base-F	Fiber optics	2000 m	1024	Best between buildings

Historically, **10Base5** cabling, popularly called **thick Ethernet,** came first. It resembles a yellow garden hose, with markings every 2.5 meters to show where the taps go. (The 802.3 standard does not actually *require* the cable to be yellow, but it does *suggest* it.) Connections to it are generally made using **vampire taps**, in which a pin is *very* carefully forced halfway into the coaxial cable's core. The notation 10Base5 means that it operates at 10 Mbps, uses baseband signaling, and can support segments of up to 500 meters. 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.''

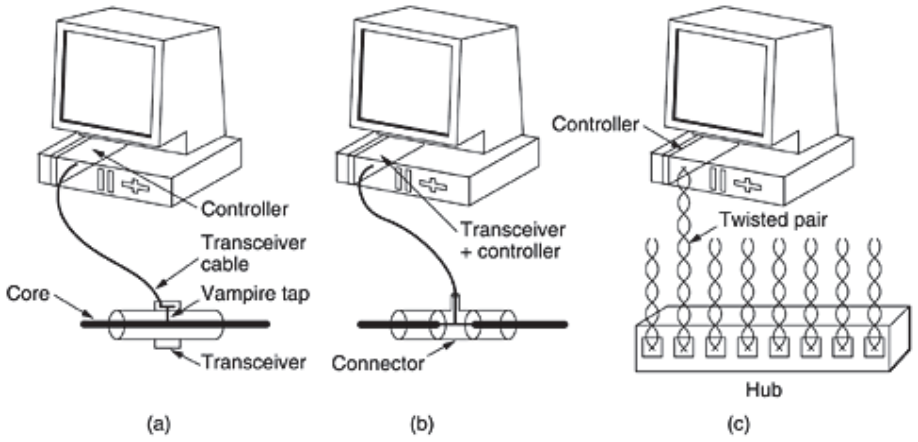
Historically, the second cable type was **10Base2**, or **thin Ethernet,** which, in contrast to the garden-hose-like thick Ethernet, bends easily. Connections to it are made using industry-standard BNC connectors to form T junctions, rather than using vampire taps. BNC connectors are easier to use and 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.

Figure 4-14. Three kinds of Ethernet cabling. (a) 10Base5. (b) 10Base2. (c) 10Base-T.



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.

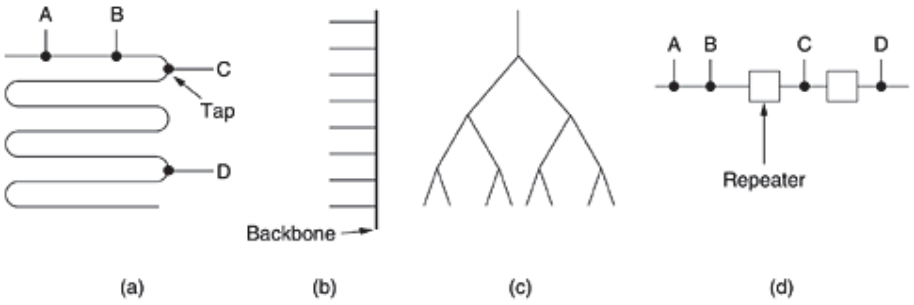
With 10Base2, the connection to the cable is just a passive BNC T-junction connector. The transceiver electronics are on the controller board, and each station always has its own transceiver.

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.

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



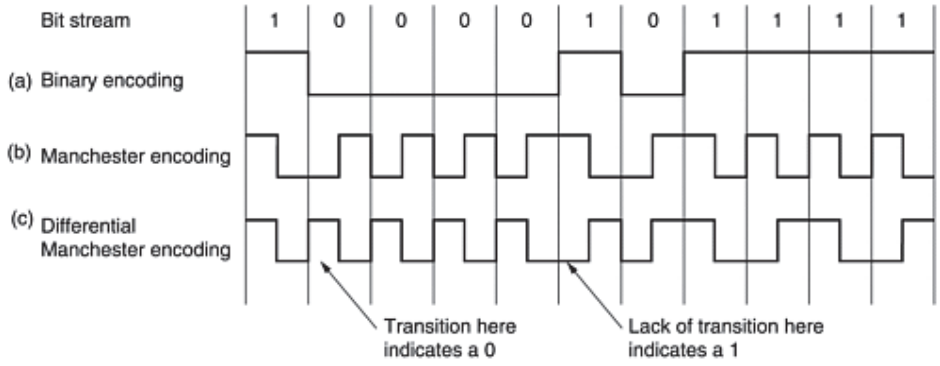
Each version of Ethernet has a maximum cable length per segment. To allow larger networks, multiple cables can be connected by **repeaters**, as shown in Fig. 4-15(d). A repeater is a physical layer device. It receives, amplifies (regenerates), and retransmits signals in both directions. As far as the 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).

Figure 4-16. (a) Binary encoding. (b) Manchester encoding. (c) Differential Manchester encoding.

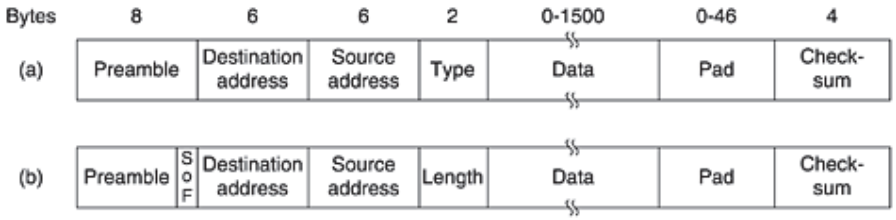


Differential Manchester encoding, shown in Fig. 4-16(c), is a variation of basic Manchester encoding. In it, a 1 bit is indicated by the absence of a transition at the start of the interval. A 0 bit is indicated by the presence of a transition at the start of the interval. In both cases, there is a transition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

4.3.3 The Ethernet MAC Sublayer Protocol

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

Figure 4-17. Frame formats. (a) DIX Ethernet. (b) IEEE 802.3.



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 7×10^{13} global addresses. The idea is that any station can uniquely address any other station by just giving the right 48-bit number. It is up to the network layer to figure out how to locate the destination.

Next comes the *Type* field, which tells the receiver what to do with the frame. 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.

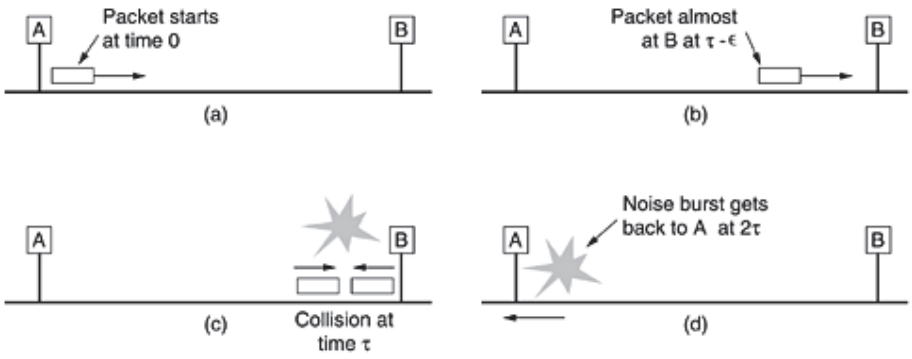
Next come the data, up to 1500 bytes. This limit was chosen somewhat arbitrarily at the time the DIX standard was cast in stone, mostly based on the fact that a transceiver needs enough RAM to hold an entire frame and RAM was expensive in 1978. A larger upper limit would have meant more RAM, hence a 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 τ. Just before the frame gets to the other end (i.e., at time τ-ε), 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 2τ,

the sender sees the noise burst and aborts its transmission, too. It then waits a random time before trying again.

Figure 4-18. Collision detection can take as long as 2τ.



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 2τ. The sender will then incorrectly conclude that the frame was successfully sent. To prevent this situation from occurring, all frames must take more than 2τ 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 μsec 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 (2τ). To accommodate the longest path allowed by Ethernet, the slot time has been set to 512 bit times, or 51.2 μsec as mentioned above.

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

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

This algorithm, called **binary exponential backoff**, was chosen to dynamically adapt to the number of stations trying to send. If the randomization interval for all collisions was 1023, the chance of two stations colliding for a second time would be negligible, but the average wait after a collision would be hundreds of slot times, introducing significant delay. On the other hand, if each station always delayed for either zero or one slots, then if 100 stations ever tried to send at once, they would collide over and over until 99 of them picked 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

Equation 4

$$A = kp(1-p)^{k-1}$$