



[Epicenter](#)[Mind Our Tech Business](#)[Previous post](#)[Next post](#)

# Speed Matters: How Ethernet Went From 3 Mbps to 100 Gbps ... and Beyond

By [Iljitsch Van Beijnum](#)   July 16, 2011 | 9:30 am | Categories: [Broadband](#)



Although watching TV shows from the 1970s suggests otherwise, the era wasn't completely devoid of all things resembling modern communication systems. Sure, the 50 Kbps modems that the ARPANET ran on were the size of refrigerators, and the widely used Bell 103 modems only transferred 300 bits per second. But long distance digital communication was common enough, relative to the number of computers deployed. Terminals could also be hooked up to mainframe and minicomputers over relatively short distances with simple serial lines or with more complex [multidrop](#) systems.

This was all well known; what was new in the '70s was the local area network (LAN). But how to connect all these machines?

The point of a LAN is to connect many more than just two systems, so a simple cable back and forth doesn't get the job done. Connecting several thousands of computers to a LAN can in theory be done using a star, a ring, or a bus topology. A star is obvious enough: every computer is connected to some central point. A bus consists of a single, long cable that computers connect to along its run. With a ring, a cable runs from the first computer to the second, from there to the third and so on until all participating systems are connected, and then the last is connected to the first, completing the ring.

In practice, things aren't so simple. Token Ring is a LAN technology that uses a ring topology, but you wouldn't know it by looking at the network cabling, because computers are hooked up to concentrators (similar to today's Ethernet switches). However, the cable does in fact form a ring, and Token Ring uses a somewhat complex token passing system to determine which computer gets to send a packet at which time. A token circles the ring, and the system in possession of the token gets to transmit. Token Bus uses a physical bus topology, but also uses a token-passing scheme to arbitrate access to the bus. A token network's complexity makes it vulnerable to a number of [failure modes](#), but such networks do have the advantage that performance is deterministic; it can be calculated precisely in advance, which is important in certain applications.

But in the end it was Ethernet that won the battle for LAN standardization through a combination of standards body politics and a clever, minimalist — and thus cheap to implement — design. It went on to obliterate the competition by seeking out and assimilating higher bitrate protocols and adding their technological distinctiveness to its own. Decades later, it had become ubiquitous.

If you've ever looked at the network cable protruding from your computer and wondered how Ethernet got started, how it has lasted so long, and how it works, wonder no more: here's the story.

## Brought to you by Xerox PARC

Ethernet was invented by [Bob Metcalfe](#) and others at Xerox's [Palo Alto Research Center](#) in the mid-1970s. PARC's experimental Ethernet ran at 3Mbps, a "convenient data transfer rate [...] well below that of the computer's path to main memory," so packets wouldn't have to be buffered in Ethernet interfaces. The name comes from the [luminiferous ether](#) that was at one point thought to be the medium through which electromagnetic waves propagate, like sound waves propagate through air.



ars technica

Ethernet used its cabling as radio "ether" by simply broadcasting packets over a thick coaxial line. Computers were connected to the Ethernet cable through "taps," where a hole is punched through the coax cladding and the outer conductor so a connection can be made to the inner conductor. The two ends of the coax cable—branching is not allowed—are fitted with terminating resistors that regulate the electrical properties of the cable so signals propagate throughout the length of the cable but don't reflect back. All computers see all packets pass by, but the Ethernet interface ignores packets that aren't addressed to the local computer or the broadcast address, so the software only has to process packets targeted at the receiving computer.

Other LAN technologies use extensive mechanisms to arbitrate access to the shared communication medium. Not Ethernet. I'm tempted to use the expression "the lunatics run the asylum," but that would be unfair to the clever distributed control mechanism developed at PARC. I'm sure that the mainframe and minicomputer makers of the era thought the asylum analogy wasn't far off, though.

Ethernet's media access control (MAC) procedures, known as "Carrier Sense Multiple Access with Collision Detect" (CSMA/CD), are based on ALOHAnet. This was a radio network between several Hawaiian islands set up in the early 1970s, where all the remote transmitters used the same frequency. Stations transmitted whenever they liked. Obviously, two of them might transmit at the same time, interfering with each other so both transmissions were lost.

To fix the problem, the central location acknowledges a packet if it was received correctly. If the sender doesn't see an acknowledgment, it tries to send the same packet again a little later. When a collision occurs because two stations transmit at the same time, the retransmissions make sure that the data gets across eventually.

Ethernet improves on ALOHAnet in several ways. First of all, Ethernet stations check to see if the ether is idle (*carrier sense*) and wait if they sense a signal. Second, once transmitting over the shared medium (*multiple access*), Ethernet stations check for interference by comparing the signal on the wire to the signal they're trying to send. If the two don't match, there must be a collision (*collision detect*). In that case, the transmission is broken off. Just to make sure that the source of the interfering transmission also detects a collision, upon detecting a collision, a station sends a "jam" signal for 32 bit times.

Both sides now know their transmission failed, so they start retransmission attempts using an exponential backoff procedure. On the one hand, it would be nice to retransmit as soon as possible to avoid wasting valuable bandwidth, but on the other hand, immediately having another collision defeats the purpose. So each Ethernet station maintains a maximum backoff time, counted as an integer value that is multiplied by the time it takes to transmit 512 bits. When a packet is successfully transmitted, the maximum backoff time is set to one. When a collision occurs, the maximum backoff time is doubled until it reaches 1024. The Ethernet system then selects an actual backoff time that's a random number below the maximum backoff time.

For instance, after the first collision, the maximum backoff time is 2, making the choices for the actual backoff

time 0 and 1. Obviously, if two systems both select 0 or both select 1, which will happen 50 percent of the time, there is another collision. The maximum backoff then becomes 4 and the chances of another collision go down to 25 percent for two stations wanting to transmit. After 16 successive collisions, an Ethernet system gives up and throws away the packet.

There used to be a lot of fear, uncertainty, and doubt surrounding the performance impact of collisions. But in practice they're detected very quickly and the colliding transmissions are broken off. So collisions don't waste much time, and CSMA/CD Ethernet performance under load is actually quite good: in their paper from 1976 describing the experimental 3Mbps Ethernet, Bob Metcalfe and David Boggs showed that for packets of 500 bytes and larger, more than 95 percent of the network's capacity is used for successful transmissions, even if 256 computers all continuously have data to transmit. Pretty clever.

## Standardization

In the late 1970s, Ethernet was owned by Xerox. But Xerox preferred owning a small piece of a large pie rather than all of a small pie, and it got together with Digital and Intel. As the DIX consortium, they created an open (or at least multi-vendor) 10Mbps Ethernet specification and then quickly ironed out some bugs, producing the DIX Ethernet 2.0 specification.

Then the Institute of Electrical and Electronics Engineers (IEEE) got into the game. Eventually, it produced standard 802.3, which is now considered the official Ethernet standard—although the IEEE carefully avoids using the word “Ethernet” lest it be accused of endorsing any particular vendor. (DIX 2.0 and IEEE 802.3 are fully compatible, except for one thing: the layout and meaning of the Ethernet header fields.)

Even right at the beginning, engineers realized that having a single cable snaking through a building was limiting, to say the least. Simply branching the thick coaxial cable wasn't possible; that would do bad things to the data signals. The solution was having repeaters. These regenerate the signal and make it possible to connect two or more Ethernet cables or segments.

The 9.5mm thick coaxial cable also wasn't the easiest type of cabling to work with. For instance, I once saw a two telecom company guys hammer on a couple of thick coax cables that went through a wall in order to bend the cables downward. This took them the better part of an hour. Another one told me that he keeps a nice big piece of the stuff in his car: “If the police find a baseball bat in your car they call it a weapon, but a piece of coax works just as well in a fight and the police never give me any trouble.”

Although less thug-repellant, *thin* coax is much easier to use. These cables are half as thin as thick ethernet and look a lot like TV antenna cable. Thin coax does away with the “vampire taps” that allow new stations to attach anywhere to a thick coax segment. Instead, thin cables end in [BNC connectors](#) and computers are attached through T-connectors. The big disadvantage of thin coax Ethernet segments is that if the cable gets interrupted somewhere, the whole network segment goes down. This happens when a new system is connected to the network, but it also happens often by accident, as coax loops have to run past every computer. There had to be a better way.

In the late 1980s, a new specification was developed to allow Ethernet to run over unshielded twisted pair cabling—in other words, phone wiring. UTP cables for Ethernet come as four pairs of thin, twisted cables. The cables can be solid copper or made of thin strands. (The former has better electrical properties; the latter is easier to work with.) UTP cables are outfitted with the now-common RJ45 plastic snap-in connectors. 10Mbps (and 100Mbps) Ethernet over UTP uses only two of the twisted pairs: one for transmitting and one for receiving.

A slight complication to this setup is that every UTP cable is also its own Ethernet segment. So in order to build a LAN with more than two computers, it's necessary to use a *multiport repeater*, also known as a hub. The hub or repeater simply repeats an incoming signal on all ports and also sends the jam signal if there's a collision. Complex rules limit the topology and the use of hubs in Ethernet networks, but I'll skip those as I doubt anyone still has interest in building a large scale Ethernet network using repeater hubs.

This setup created its own cabling issues, and they're still with us. Computers use pins 1 and 2 to transmit and

pins 3 and 6 to receive, but for hubs and switches, this is the other way around. This means that a computer is connected to a hub using a regular cable, but two computers or two hubs must be connected using “crossover” cables that connect pins 1 and 2 on one side with 3 and 6 on the other side (and vice versa). Interestingly, FireWire, co-developed by Apple, managed to avoid this failure of userfriendliness by simply always requiring a crossover cable.

Still, the end result was a fast and flexible system—so fast, it’s still in use. But more speed was needed.

[Continue reading ...](#)

## The need for speed: Fast Ethernet

It’s hard to believe now, but in the early 1980s, 10Mbps Ethernet was *very* fast. Think about it: is there any other 30-year-old technology still present in current computers? 300 baud modems? 500 ns memory? Daisy wheel printers? But even today, 10Mbps is not an entirely unusable speed, and it’s still part of the 10/100/1000Mbps Ethernet interfaces in our computers.

Still, by the early 1990s, Ethernet didn’t feel as fast as it did a decade earlier. Consider the VAX-11/780, a machine released in 1977 by Digital Equipment Corporation. The 780 comes with some 2MB RAM and runs at 5MHz. Its speed is almost exactly one MIPS and it executes 1757 dhrystones per second. (Dhrystone is a CPU benchmark developed in 1984; the name is a play on the even older Whetstone benchmark.) A current Intel i7 machine may run at 3GHz and have 3GB RAM, executing nearly 17 million dhrystones per second. If network speeds had increased as fast as processor speeds, the i7 would today at least have a 10Gbps network interface, and perhaps a 100Gbps one.

But they haven’t increased as quickly. Fortunately, by the 1990s, another LAN technology was ten times faster than regular Ethernet: Fiber Distributed Data Interface (FDDI).

FDDI is a ring network running at 100Mbps. It supports a second, redundant ring for automatic failovers when the primary ring breaks somewhere, and an FDDI network can span no less than 200 kilometers. So FDDI is very useful as a high capacity backbone between different LANs. Even though Ethernet and FDDI are different in many ways, it’s possible to translate the packet formats, so Ethernet and FDDI networks can be interconnected through *bridges*.

Bridges are connected to multiple LAN segments and learn which addresses are used on which segment. They then retransmit packets from the source segment to the destination segment when necessary. This means that, unlike in the case of a repeater, communication (and collisions!) local to each segment remain local. So a bridge splits the network into separate *collision domains*, but all the packets still get to go everywhere, so the bridged network is still a single *broadcast domain*.

A network can be split into multiple broadcast domains using routers. Routers operate at the network layer in the network model, one step above Ethernet. This means that routers strip off the Ethernet header upon reception of a packet, and then add a new lower layer header—Ethernet or otherwise—when the packet is forwarded.

FDDI was useful to connect Ethernet segments and/or servers, but it suffered from the same “oops, didn’t mean to step on that cable!” problems as thin coax Ethernet, coupled with high cost. CDDI, a copper version of FDDI, was developed, but it didn’t go anywhere. So the IEEE created Fast Ethernet, a 100Mbps version of Ethernet.

10Mbps Ethernet uses “Manchester encoding” to put bits on the wire. Manchester encoding transforms each data bit into a low and a high voltage on the wire. Then, 0 is encoded as a low-high transition and a 1 as a high-low transition. This basically doubles the number of bits transmitted, but it avoids issues that can come up with long sequences of only zeros or only ones: transmission media typically can’t maintain “low” or “high” for extended periods—the signal starts to look too much like a DC potential. Also, clocks will drift: did I just see 93 zero bits or 94? Manchester encoding avoids both these problems by having a transition between high and low in the middle of each bit. And both coax and category 3 UTP can handle the additional bandwidth.

Not so much for 100Mbps, though. Transmitting at that speed using Manchester encoding would be problematic on UTP. So instead, 100BASE-TX borrows from CDDI a 4B/5B MLT-3 encoding. The 4B/5B part takes four bits and turns them into five. This way, it's possible to ensure there are always at least two transitions in every five-bit block. This also allows for some special symbols such as an idle symbol when there is no data to transmit.

The Multi-Level Transmit 3 encoding then cycles through the values -1, 0, +1, 0. If a bit in a 4B/5B block is one, a transition to the next value is made. If the bit is zero, the signal stays at the previous level this bit period. This limits the maximum frequency in the signal, allowing it to fit within the limitations of UTP cabling. However, the UTP wiring must conform to the tighter specifications of category 5, rather than category 3 for 10BASE-T. There are many other Fast Ethernet cabling specifications than 100BASE-TX over cat 5 UTP, but only 100BASE-TX became a mass market product.

## From bridges to switches

Fast Ethernet uses the same CDMA/CD as Ethernet, but the limitations on cable length and numbers of repeaters are much more stringent to allow collisions to be detected in a tenth of the time. Soon, 10/100Mbps hubs started to appear, where 10Mbps systems were connected to other 10Mbps systems, and 100Mbps systems to 100Mbps systems. Of course, it's helpful to have communication between both types of computers, so typically these hubs would have a bridge between the 10Mbps and 100Mbps hubs inside.

The next step was to simply bridge between *all* ports. These multiport bridges were called switching hubs or Ethernet switches. With a switch, if the computer on port 1 is sending to the computer on port 3, and the computer on port 2 to the one on port 4, there are no collisions—the packets are only sent to the port that leads to the packet's destination address. Switches learn which address is reachable over which port simply by observing the source addresses in packets flowing through the switch. If a packet is addressed to an unknown address, it's "flooded" to all ports, the same as broadcast packets.

One limitation that applies to hubs and switches alike is that an Ethernet network must be loop-free. Connecting port 1 on switch A to port 1 on switch B and then port 2 on switch B to port 2 on switch A leads to immediate catastrophic results. Packets start circling the network and broadcasts are multiplied as they are flooded. However, it's very useful to have backup links in a network so that when a primary connection goes down, traffic continues to flow over the backup.

This problem was solved (for switches) by creating a protocol that detects loops in an Ethernet network and prunes connections until the loops are gone. This makes the effective network topology look like what mathematicians call a *tree*: a graph where there's *no more* than one path between any two points. It's a *spanning tree* if there's also *at least* one path between any two points, i.e., no network nodes are left unconnected. If one of the active connections fails, the spanning tree protocol (STP) is executed again to create a new spanning tree so the network keeps running.

The spanning tree algorithm was created by Radia Perlman at DEC in 1985, who also immortalized the algorithm in the form of a poem:

### Algorithme

```
I think that I shall never see
a graph more lovely than a tree.
A tree whose crucial property
is loop-free connectivity.
A tree that must be sure to span
so packet can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least-cost paths from root are traced.
In the tree, these paths are placed.
```

A mesh is made by folks like me,  
then bridges find a spanning tree.

Radia Perlman

[Photograph by David Davies](#)

[Continue reading ...](#)

## Even more speed: Gigabit Ethernet

Fast Ethernet was standardized in 1995, but only three years later, the next iteration of Ethernet came around: Gigabit Ethernet. As before, speed was increased by a factor of ten and, as before, some technology was borrowed elsewhere to hit the ground running. In this case it was Fibre Channel (apparently of British descent), a technology mostly used for storage networks. Gigabit Ethernet is extensively used over different kinds and lengths of fiber, where it hews more closely to its Fibre Channel pedigree.

But for 1000BASE-T, the IEEE needed to open a new bag of tricks borrowed from 100BASE-T2 and 100BASE-T4, Fast Ethernet standards that never got any traction, as well as 100BASE-TX. For one thing, the UTP cabling requirements were upped again to category 5e, and 1000BASE-T uses all four twisted pairs—in both directions at the same time.

This requires some advanced digital signal processing, similar to what happens in dial-up modems but at some 10,000 times the speed. Each wire pair transmits two bits at a time using 4D-PAM5. The 4D means four data symbols (two bits), the PAM5 is Pulse Amplitude Modulation with five signal levels. This happens at a rate of 125 million symbols per second—the same rate as Fast Ethernet. There's also a complex bit scrambling procedure that makes sure that various properties, such as possible interference, are optimized.

The CSMA/CD mechanism depends on the first bit of a packet traveling all the way across a collision domain before a station transmits the last bit of a packet so that there is a shared notion of “transmitting at the same time.” With transmission times much reduced by the higher bitrate, the physical size of collision domains already had to be reduced for Fast Ethernet, but for Gigabit Ethernet this would have to shrink to maybe 20 meters—clearly unworkable. To avoid this, Gigabit Ethernet adds a “carrier extension” that more or less pads packets to 512 bytes so that aggregate cable lengths of 200 meters remain usable.

However, as far as I know, no vendors implement the above scheme; they assume the presence of switches instead. With a switch, or with a direct cable between two computers, CSMA/CD is unnecessary: the two sides can simply both transmit at the same time. This is called full duplex operation, as opposed to half duplex for traditional CSMA/CD operation. The UTP Ethernet variants support an additional autoconfiguration protocol that allows two Ethernet systems to negotiate which speed to use, in full or half duplex mode.

Before the autonegotiation protocol was widely used, people would sometimes manually configure one system to use full duplex, but the other would use half duplex. With little traffic, this causes few problems, but as traffic increases, more and more collisions occur. These will be ignored by the system that is in full duplex mode, leading to corrupted packets that aren't retransmitted. Autonegotiation works very reliably these days, so there is no longer any reason to turn it off and invite problems.

Ludicrous speed: 10 Gigabit Ethernet

A common way to create a LAN in a building or office these days is to have a series of relatively small switches, perhaps one per wiring closet where all the UTP cables come together. The small switches are then connected to a bigger and/or faster switch that functions as the backbone of the LAN. With users on multiple floors and servers concentrated in a server room, there's often a lot of bandwidth required between the switches, even if individual computers don't come close to saturating a Gigabit Ethernet connection. So even though computers with a 10 Gigabit Ethernet connection aren't common even today, 10GE was badly needed as a backbone

technology. The standard was published in 2002.

In the telecom world, a technology called SONET or SDH (Synchronous Optical Networking, Synchronous Digital Hierarchy) was/is used to transmit large numbers of phone calls and also data in digital form over fiber. SONET is available in speeds of 155Mbps, 622Mbps, 2.488Gbps... and 9.953Gbps! That was too perfect to resist, so one form of 10GE adopts a low level SONET/SDH framing. This is called the WAN (Wide Area Network) PHY (as in: physical layer). But there's also a LAN PHY, which runs at 10.3125Gbps. 10 Gigabit Ethernet no longer supports half duplex CSMA/CD operation; it's only full duplex operation at this speed.

Both the 10GE WAN PHY and most LAN PHY variants use fiber. Making Gigabit Ethernet run over UTP as well as it does wasn't easy. This is even more true for 10 Gigabit Ethernet; it works very well over fiber, even over fairly long distances, making it very popular with Internet Service Providers. But it required quite a bit of magic to make 10GE run over UTP—it took until 2006 for the 10GBASE-T standard to be published. 10GBASE-T needs even better cables than 1000BASE-T—category 6a to reach 100 meters. Cat 6a uses thicker insulation than Cat 5e, so it doesn't always physically fit where older cables went.

10GBASE-T also increases the number of symbols per second from 125 million for Fast and Gigabit Ethernet to 800 million and the PAM levels from 5 to 16, encoding 3.125 instead of 2 bits per symbol. It also soups up the echo and near end crosstalk cancellation and other signal processing that was introduced with Gigabit Ethernet over UTP and adds Forward Error Correction (FEC) to repair incidental transmission errors.

### Reaching for 100 Gigabit Ethernet

After 10 Gigabit Ethernet, 100Gbps was the obvious next step. However, transmitting at 100Gbps over fiber has numerous challenges, as the laser pulses that carry information through fiber become so short that they have a hard time maintaining their shape as they travel. The IEEE therefore kept open the option to make a smaller step towards 40Gbps instead of its customary tenfold boost in speeds.

Currently, there are a large set of 100GBASE-\* standards, but many of them use four parallel data paths to reach 40 or 100Gbps and/or only work over short distances. Work is still ongoing to create the one 100GBASE standard to rule them all.

### Ethernet's future

It's truly mindboggling that Ethernet managed to survive 30 years in production, increasing its speed by no less than four orders of magnitude. This means that a 100GE system sends an entire packet (well, if it's 1212 bytes long) in the time that the original 10Mbps Ethernet sends a single bit. In those 30 years, all aspects of Ethernet were changed: its MAC procedure, the bit encoding, the wiring... only the packet format has remained the same—which ironically is the part of the IEEE standard that's widely ignored in favor of the slightly different DIX 2.0 standard.

All this backward compatibility is actually a problem: at 10Mbps you can send some 14,000 46-byte packets per second, or 830 1500-byte packets. But even at GE speeds, the 1500-byte maximum is an issue. Many modern Gigabit Ethernet network cards actually let the TCP/IP stack transmit and receive much larger packets, which are then split into smaller ones or combined into larger ones to make life easier for the CPU, as most of the processing is per packet, independent from how large a packet is. And sending as many as 140 million 46-byte packets per second at 100GE is ridiculous. Unfortunately, allowing larger packets would break compatibility with older systems, and so far the IEEE has always punted on changing this.

LANs are now everywhere, if only to provide an onramp to the Internet. Ethernet in its various flavors has been spectacularly successful, pushing out all competing LAN technologies. The only reason Ethernet growth has slowed over the past decade is because wireless LANs (in the form of Wi-Fi) are so convenient. (And Wi-Fi is very compatible with wired Ethernet.) But wired and wireless are largely complimentary, so even though more and more computers go through life with an unoccupied Ethernet port—or even lack one altogether—Ethernet is always there to deliver the speed and reliability that the shared wireless ether keeps struggling to provide.

### Terabit Ethernet?

Will there ever be Terabit Ethernet, running at 1000Gbps? On the one hand, this seems unlikely, as transporting

100Gbps over fiber is already a big challenge. On the other hand, in 1975 few people would have guessed that today's students would go to class carrying affordable computers with 10Gbps ports.

CPU designers solved a similar problem by using multiple parallel cores. Gigabit Ethernet already uses parallelism by using all four wire pairs in a UTP cable, and many 40Gbps and 100Gbps Ethernet variants over fiber also use parallel datastreams, each using a slightly different wavelength laser light. Under-sea cables already transport multi-terabit aggregate bandwidths over a single fiber using dense wavelength division multiplexing (DWDM), so this seems an obvious opportunity for Ethernet to once again take existing technology, streamline it, and aggressively push the price down.

Or maybe it doesn't have to. When I e-mailed Radia Perlman to ask permission to use the Algorhyme poem, she mentioned a new technology called Transparent Interconnection of Lots of Links (TRILL), which should allow for building flexible, high-speed Ethernet networks using "lots of links" rather than a single fast link. In any event, it seems likely that the future of high speed Ethernet involves some form of parallelism.

I can't wait to see what the next 30 years bring for Ethernet.

*[Photograph by David Davies](#)*

**Pages:** [1](#) [2](#) [3](#) [View All](#)

Self-professed networking guru working on PhD; BGP and IPv6 connoisseur; sometime [Ars Technica](#) writer. Follow [@iljitsch](#) and [@arstechnica](#) on Twitter.

[Post Comment](#) | [Permalink](#)

