Multiple access protocols

The point–to–point protocols are simple because only one sender is involved.

Broadcast links allow several nodes to attempt to transmit at the same time. There are three basic approaches to controlling access to a shared link:

- Random access (collision protocols).
- Reservation–based access.
- Channel division protocols (multiplexing)

Traditional terminology calls a NIC attached to a multiple access link a station.

Random access

In this set of protocols, a station is allowed to attempt to transmit whenever it sees the link as free.

The main issue is that there is a difference between a link **being free** and a link **seen as free**. this difference is caused by timing differences (the state of the Universe is different for each observer).

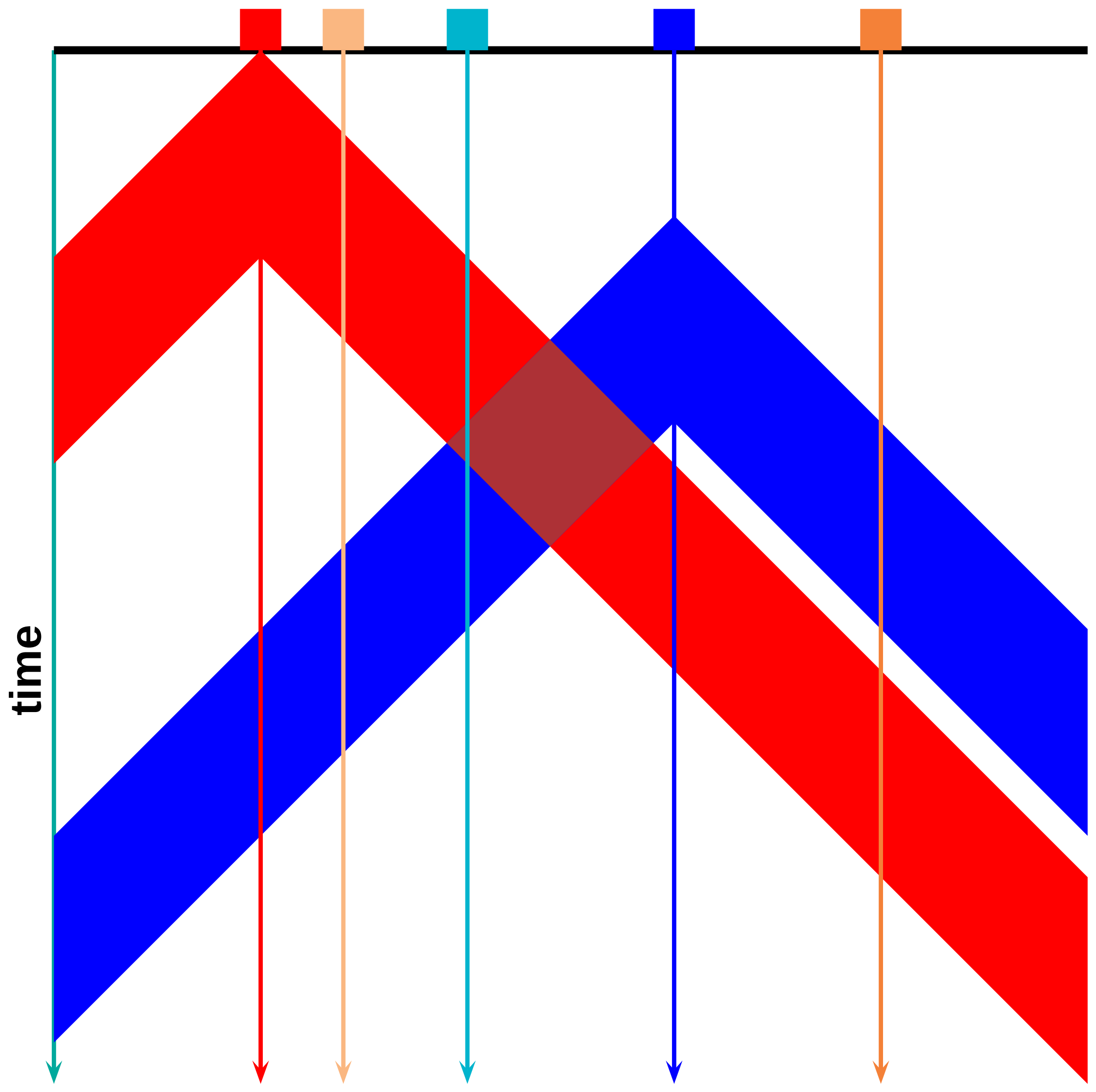
The inescapable result of relying on information that is not universally correct is the occurrence of conflicts, called **contention**: several stations compete for the right to transmit over a single link.

Contention is discovered when it happens; its result is a **collision** when two (or more) frames are sensed by the same receiver at the same time.

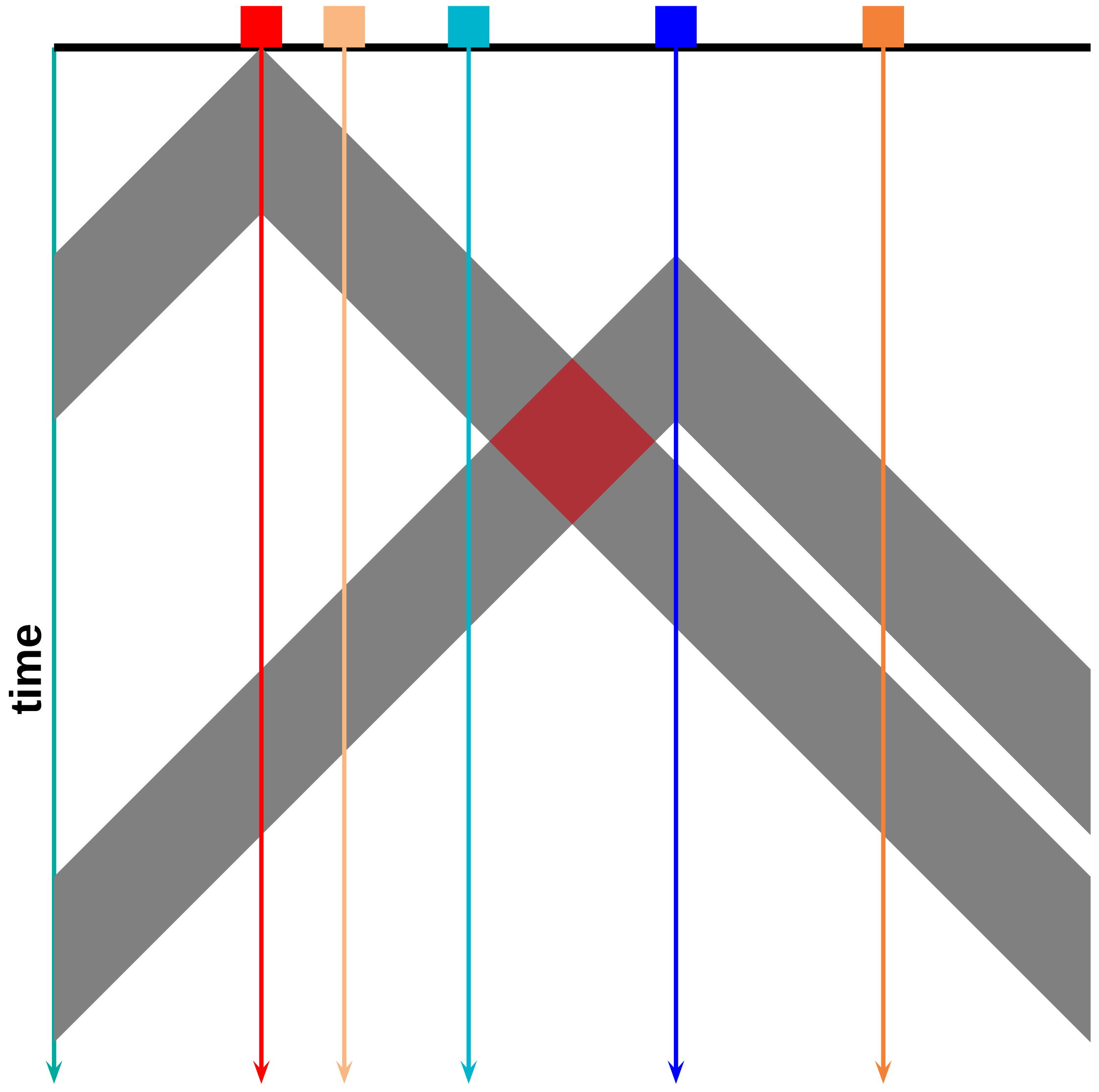
In most cases a collision makes proper reception of the frames impossible, de facto destroying them,

Note that a collision is a local event: frames collide at a particular receiver(s); the same frames may show up at different times at another receiver.

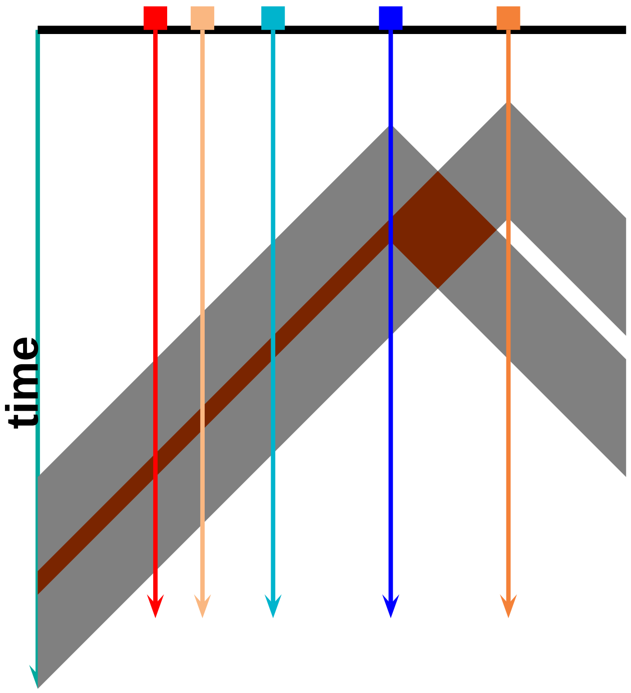
A “small” collision



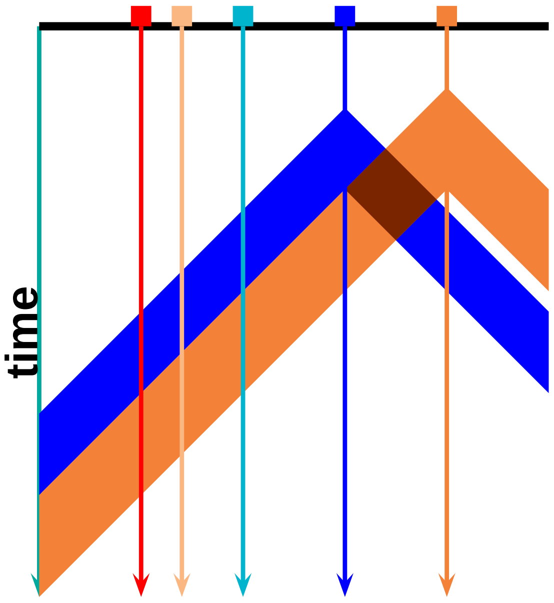
An “unnoticed” collision



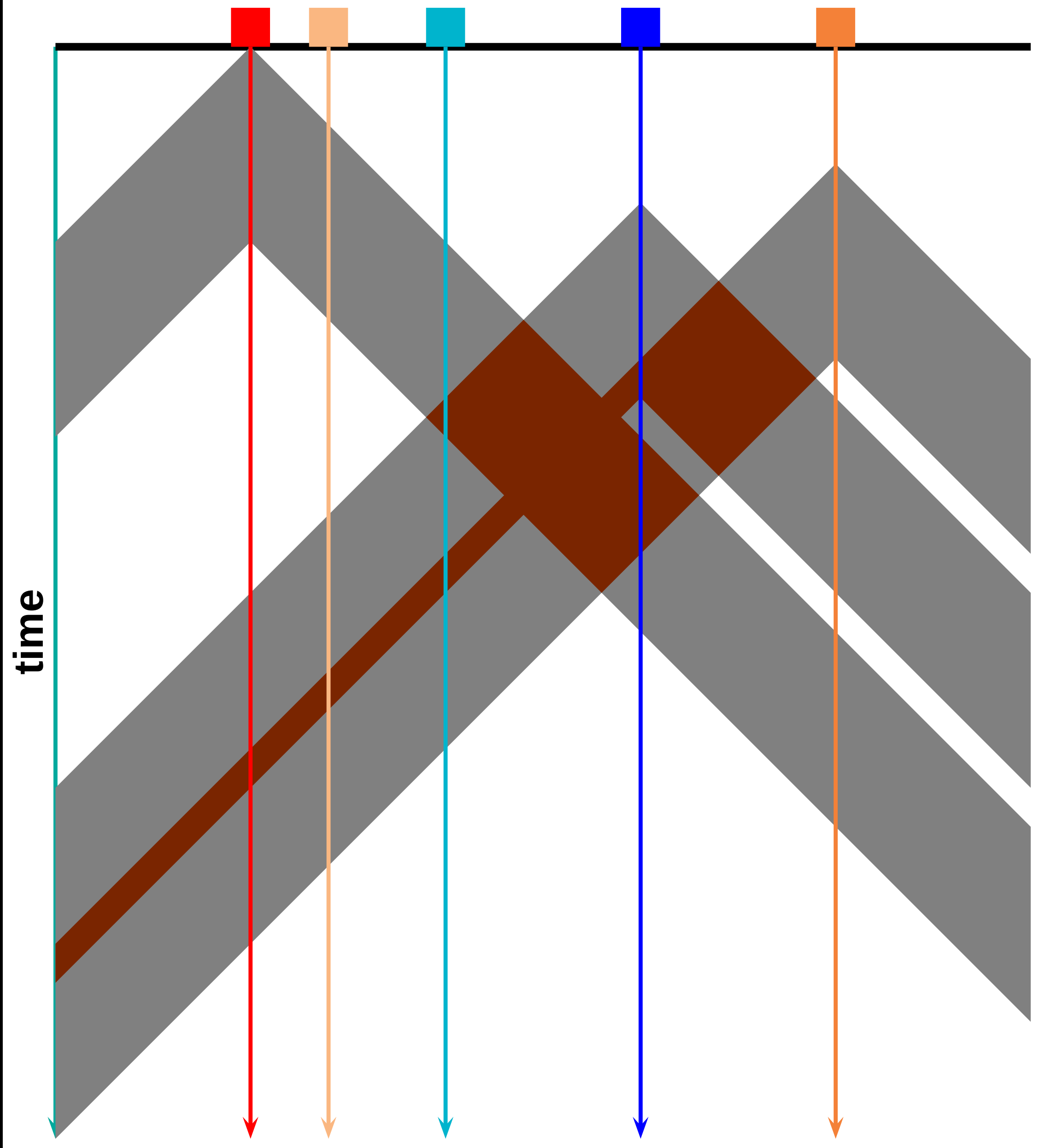
A “nasty” collision



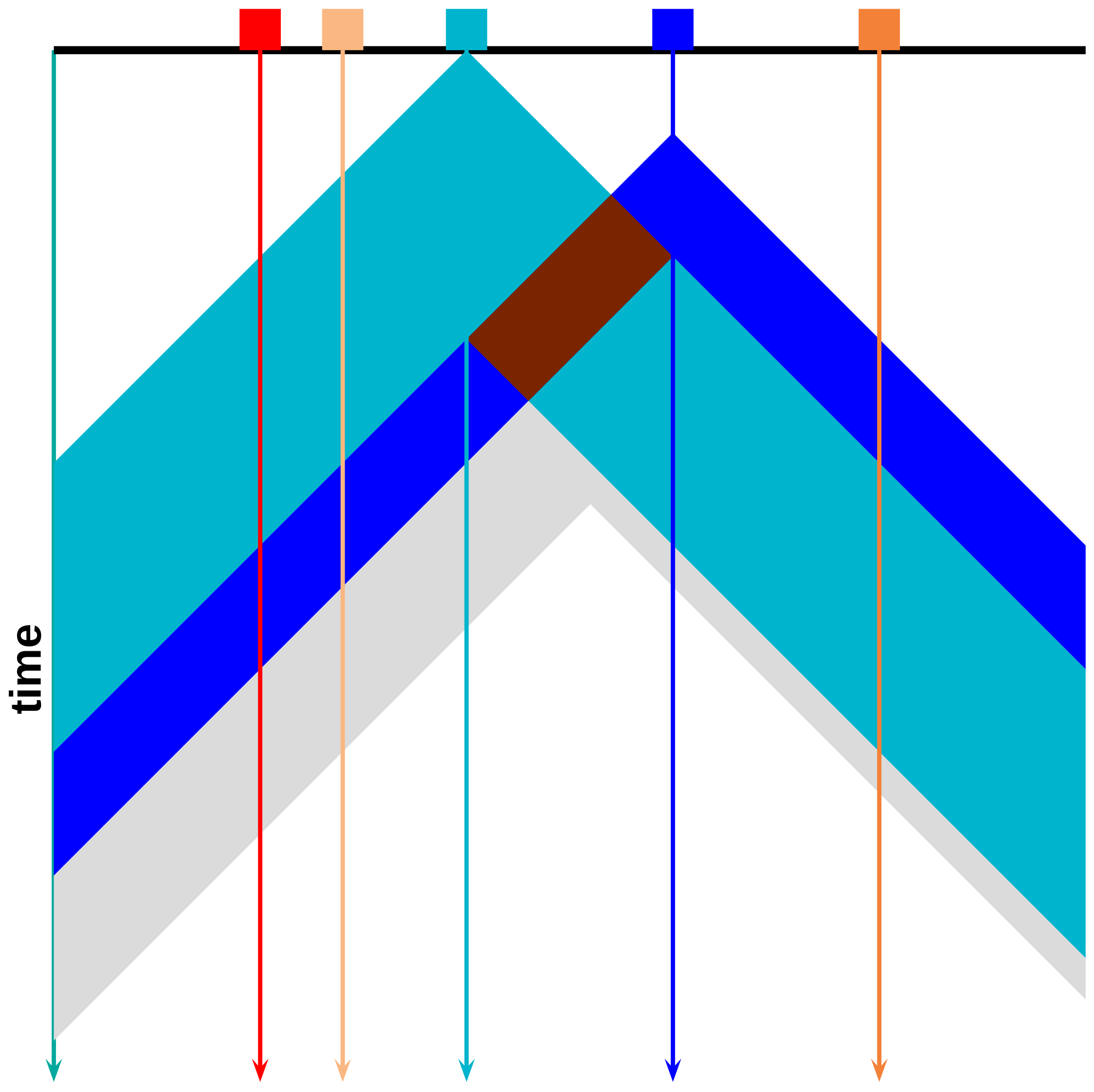
In this case the later transmission could be salvaged by an extremely fast reaction of the blue station:



A massive collision



Another massive collision



Collision conclusions

- A collision is a local phenomenon: it may be noticed by none or by some stations only.
- It does not help much if a station stops immediately upon detecting a collision.
- The fact that a sender did not notice a collision does not imply that no collision occurred. As a consequence, the absence of a collision is not a tacit acknowledgment.
- If a sender does not know whether a collision occurred or not, the sender cannot deduce whether a frame arrived undamaged.

A protocol must be able to guarantee the delivery of a frame (possibly after a long period of retransmissions).

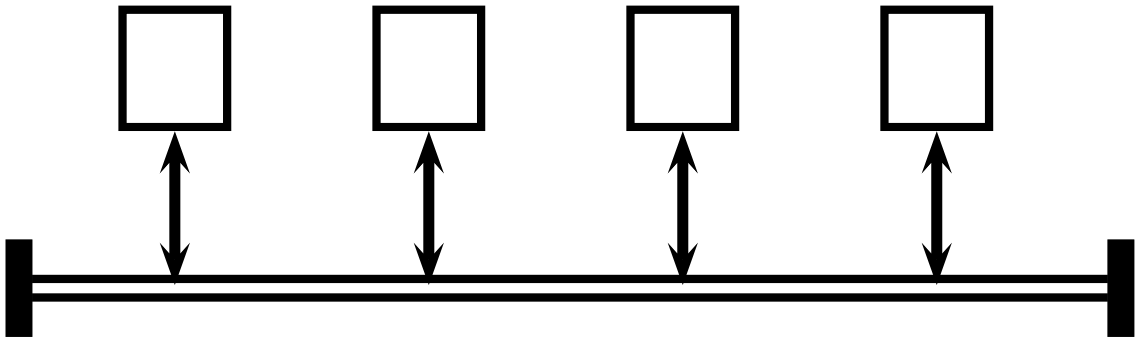
One way to guarantee delivery is through the use of acknowledgments and retransmissions.

In random access protocols, collisions may destroy frames without the sender knowing about it. One may, however, introduce rules that will make a sender aware that a collision happened anywhere in the link. This will allow to resolve collisions through retransmissions.

A protocol that wants to guarantee frame arrivals must either use acknowledgments or resolve all collisions that took place.

Ethernet

Ethernet is the oldest modern local area network (1970s). In its oldest variation it had a **bus** configuration running with the single link operating at 10Mbps. Although the topology was changed drastically, the original protocol was retained, with all its aspects shaped by the bus topology, some of them absurd in today’s, star–based, Ethernet.



The link was designed as a single bi–directional channel, typically a coax cable. Soon two unidirectional links were used instead (see twisted pair) and the bus architecture was replaced by hubs.

What is the Ethernet?

The term Ethernet is now used to describe a number of unrelated entities:

- A local–area–network that uses a 8P8C plug and a twisted–pair cable bundle (a mistaken view that the Ethernet is a PL concept).
- A DLL protocol called CSMA/CD or 802.3.
- Any DLL that uses MAC frames.
- (Most common:) any local area network.

CSMA/CD protocol

The DLL protocol used by Ethernet is called **Carrier Sense Multiple Access** with **Collision Detection**.

It does not use acknowledgments and relies on detecting all the collisions that occurred anywhere in the network.

All the frames sent and not involved in collisions are assumed successful because one of the following must have happened:

- The receiver received the frame. Note that corrupted frames are not considered invalid in this scheme although they could signal failure by jamming the link.
- The receiver is dead. No protocol is smart enough to deliver this frame.
- The receiver could have picked the frame but chose not to do so for some reason (lack of buffer space, etc.). It is the receiver's problem.

Based on these principles, Ethernet assumes that a sender that managed to send a whole frame successfully (i.e. without a collision being detected) has the right to assume that the frame was correctly received and can therefore discard the frame (no sliding window here).

Clearly, Ethernet is not really reliable and another protocol must be combined with CSMA/CD if one really wants frames delivered for sure.

Since Ethernet senders do not keep sliding windows, the additional protocol must be part of a higher layer. Normally TCP does the work; if UDP is used, there is no need for concern, since UDP is not reliable either.

Collision detection

CSMA/CD requires that all stations detecting a collision report it to every other station by sending a jamming signal.

Additionally, a minimum length of a frame is required so that there are no “unnoticed” collisions. This requirement made sense when the Ethernet medium was a shared bus.

The clock drifting problem (and other considerations) imposes a maximum frame length which is set to 1500B.

Minimum frame length

The idea is to guarantee that a sender detects a collision of its own frame. To achieve that, the frame must be transmitted for a time long enough to interfere with any other frame being transmitted somewhere in the bus.

If sender $\mathcal{A}$ is located at the near end of a bus of length L, its frame will take L/v to reach the far end of the bus. In the worst possible case, station $\mathcal{B}$ located at the far end starts to transmits just as it is about to hear $\mathcal{A}$'s frame. $\mathcal{B}$'s frame will take another L/v to reach $\mathcal{A}$.

Adding the two together, $\mathcal{A}$ will become aware that its frame participated in a collision after a time of $2 \times L/v$. The minimum length frame must therefore last at least this long.

Taking $L = 5120$m and $v = 2 \times 10^8$ m/s we get the minimum transmission time to be:

$$T_{min} = \frac{2 \times 5120}{2 \times 10^8} s = 51.2 \mu s$$

At 10Mb/s ($1b = 0.1 \mu s$) this gives a minimum frame size of 512 bits or 64B.

The actual maximum distance of an Ethernet segment was limited not to 5120m but to 2500m because the actual propagation speed is greatly reduced by repeaters.

Note that newer versions of Ethernet have much higher transmission rates. Consequently, the choice (for faster rates) is either to increase the minimum frame size or to decrease the maximum link length. the former is not really a choice for compatibility reasons, so the lengths were decreased appropriately.

CSMA/CD

The NIC is divided (as usual) into two parts:

Controller: managing the actions of the PL.

Transceiver: (made of a transmitter and a receiver) performing the function of the PL.

A CSMA/CD controller operates in two modes:

- **Normal** mode.
- **Backoff** or **Collision resolution** mode.

1. In normal mode, the controller makes the transmitter start transmitting as soon as the transceiver reports 96 bits of silence.
2. Upon an interrupt from the transceiver, the controller (temporarily) stops the transmitter and enters the **backoff** mode and stays in it until it manages to transmit a frame.

Note that when a single node is interested in transmitting a large file, CSMA/CD is always in the normal mode and its efficiency approaches $\frac{1500}{1500+26+96/8} = 97\%$

When the link is congested, Ethernet is often in backoff mode and its efficiency drops to about 30% but practically never below.

Receiver

It senses the link at a normal reading rate (usually twice the transmission rate) and reads bits from it, decoding them appropriately (method depends on the medium used (light or electricity).

It detects transmission errors (crooked bits) and ignores the remainder of a frame if comes damaged.

It is possible that a receiver sensing an error does send a jam signal (pretending that a collision occurred). This saves a lot, but it is not required because the original Ethernet used Manchester encoding in which crooked bits are interpreted as a collision (so there was no need to require anything extra).

Receiver’s extra role

The main hardware requirement is that the receiver in the Transceiver (also called “adapter”) must also sense any change in the signal energy level (note that it must be clever enough to recognise signal from its own transmitter).

Several possibilities exist:

1. Energy same as during the previous bit. Nothing special happens.
2. Energy level drops to silence (end of frame). The Transceiver starts a 96 bit period of silence. If the link remains silent during these 96 bits, it informs the Controller and waits for a signal to start transmitting.
3. The energy level goes up from silence to standard level (somebody’s frame showed up). The Transceiver does nothing.
4. The energy level goes up from standard to abnormally high. This implies that more than one signal is being received. (Note that when two unidirectional channels are used, this is detected differently.)

The energy level goes up from standard to abnormally high. The receiver passes this information on to the transmitter which performs one of two actions based on its previous state:

- The local transmitter was silent: a **collision** between two (or more) frames transmitted by other nodes. Nothing special: the transceiver waits for event (2).
- The local transmitter was transmitting. It aborts immediately, sends a 48 bit **jam signal** and informs the Controller which enters the **backoff** mode. This is a **collision** involving this station.

Exponential backoff

When CSMA/CD enters the backoff mode, it performs a complex algorithm called **collision resolution**. The node stays in backoff mode until it successfully transmits a frame.

While in backoff mode, the Controller:

1. Waits for a random period of time. Typically, **exponential backoff** is used which requires that the wait be a 2^r where $r = random(1, n)$ where n is the number of collisions experienced while in this instance of backoff. When $n = 11$, the protocol resets itself to an initial state.
2. Makes the Transceiver start to retransmit the frame that collided.

Reservation–based access

A shared medium can be shared by a form of time sharing. Three basic forms were proposed under numerous names (and classifications):

- Distributed reservation in the form of random–access contest of channel reservations.
- Central reservation protocols.
- Token passing.

Distributed reservation

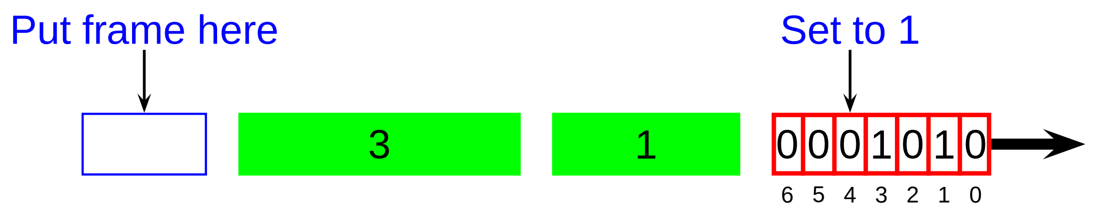
Several schemes were proposed mostly for unidirectional channels. Two examples:

- Reservations using the reverse channel (requires two unidirectional channels). Used in DQDB, this scheme allows stations to reserve fixed–size slots on the channel going in the opposite direction.
- Piggybacking to a train.

Piggybacking

A special reservation miniframe is sent by the most upstream station. It contains a bit for every station attached to the channel. If a station wants to send a frame, it sets its bit in the reservation frame and appends its frame after all the frames that follow it.

Example: station #4 (i.e. fifth from the head of the channel) wants to transmit a frame.



Station #5 will have a chance to piggyback its frame after the newly written frame (if it wants to).

Central reservation

The basic idea is to have one selected station which is responsible for accepting reservations. Reservations may involve single frames or whole sessions (streams of frames).

These methods are commonly used in cellular telephony (and elsewhere) where the selected station also happens to be the **access point**.

- Single frames: a station sends a request to the access point asking for the right to transmit. If the access point responds, the station transmits one frame.

- Whole sessions: a station sends a request to the access point asking for a sub–channel. If there are free sub–channels, the access point replies with the number of the sub–channel, which can from now on be used exclusively by the asking station. A sub–channel may be in the form of a frequency band (**FDMA**), a regular–interval time slot (**TDMA**) or a code (**CDMA**).

xDMA methods

TDMA and FDMA are simple. The only subtlety is the need for a **guard** separators:

TDMA: The channel is divided into rounds made a fixed number of time slots of fixed duration. Each station is given the right to use one of the same slot in each round. Clock drifting (etc.) requires that the slots be separated by **guard times**, small periods of silence (similar to CSMA/CD's 96 bits of silence).

FDMA: The available bandwidth is divided into frequency bands, each treated as a sub–channel allocated to one station. The sub–channels are separated by **guard bands**, narrow frequency ranges that are unused (because pass filters do not attenuate sharply but gradually).

Although TDMA and FDMA appear similar to TDM and FDM (Physical Layer) they are different in the sense that the PL of DLL using either of them has no multiplexing at all (it uses what it thinks is a complete point–to–point channel).

CDMA

All the stations are allowed to broadcast all the time, each of them encoding its transmission using a different code than the other stations.

The receiver must know the code of the sender; with this code the original message can be extracted from the otherwise garbled message that is seen by the receiver (this message is the sum of all the simultaneous transmissions).

Orthogonal code

Consider vectors on length $\mathcal{N}$. One operation defined on such vectors is the **inner product** (**dot** or $\circ$) defined as:

$$\vec{a} \circ \vec{b} = \sum_{i=0}^{\mathcal{N}} a_i b_i$$

Two vectors $\vec{v_1}$ and $\vec{v_2}$ are called orthogonal if $\vec{v_1} \circ \vec{v_2} = 0$.

In the case of CDMA we are interested solely in orthogonal vectors made of +1 and -1 elements. Note that for any such vector $\vec{c}$, $\vec{c} \circ \vec{c} = \mathcal{N}$.

Suppose that the multiple–access channel is to be divided into $\mathcal{N}$ sub-channels using CDMA.

Each potential sender applies for a code. If a sub–channel is available, the access point replies by sending to the potential sender an orthogonal vector of length $\mathcal{N}$ (orthogonal to the other vectors distributed).

Now the station can start transmitting. If the channel is a centralised one (wired) its signal is merged with the signal sent by other stations; otherwise (wireless) it is merged by being received simultaneously with other signals.

Decoding

Four stations are sending one bit each, each station using its own code.

the resulting signal $\vec{s}$ is:

$$\vec{s} = b_1\vec{c_1} + b_2\vec{c_2} + b_3\vec{c_3} + b_4\vec{c_4}$$

A receiver wants to receive a frame from a station using code $\vec{c_2}$. This is done by multiplying $\vec{s}$ by $\vec{c_2}$:

$$\vec{s} \circ \vec{c_2} = 4b_2$$

In reality it is not feasible to give truly orthogonal vectors (the different components of the signal will show at the receiver at slightly different times). So, a certain degree of noise will show as a result of the multiplication done by the receiver.

Note that in ideal circumstances CDMA is capable of filling the channel completely with a performance similar to the best effort approach.

Token passing

A special frame called **token** determines which station has the right to transmit.

The token frame is sent by a station to its upstream neighbour. When a station receives a token, it performs these steps:

- If it has a frame to transmit, it transmits it.
- It sends the token frame to its upstream neighbour (the most upstream station sends the token to the most downstream station).

A station not holding a token is not allowed to transmit.

Special care is needed in the case when the token frame is lost.