INSA RENNES

NETWORKS

Chapter III
The Data Link Layer

Institut National des Sciences Appliquées
Ecole publique d'ingénieurs

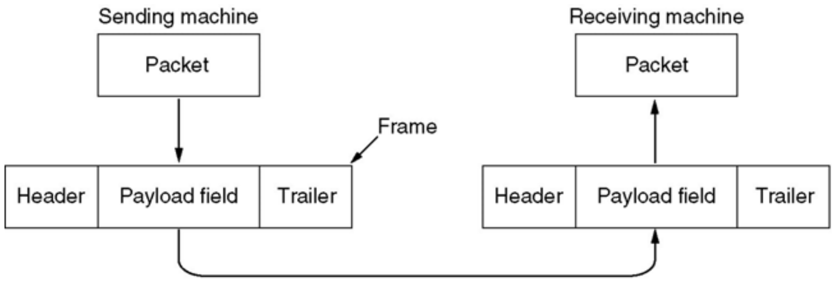
Chapter 3

The Data Link Layer

Functions of the Data Link Layer

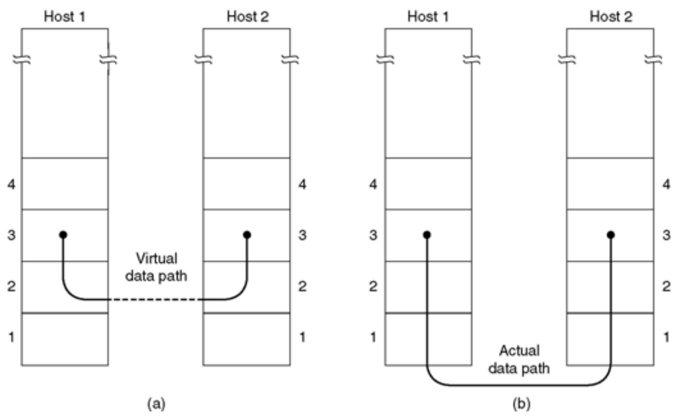
- Provide service interface to the network layer
- Dealing with transmission errors
- Regulating data flow
  - Slow receivers not swamped by fast senders

Functions of the Data Link Layer (2)



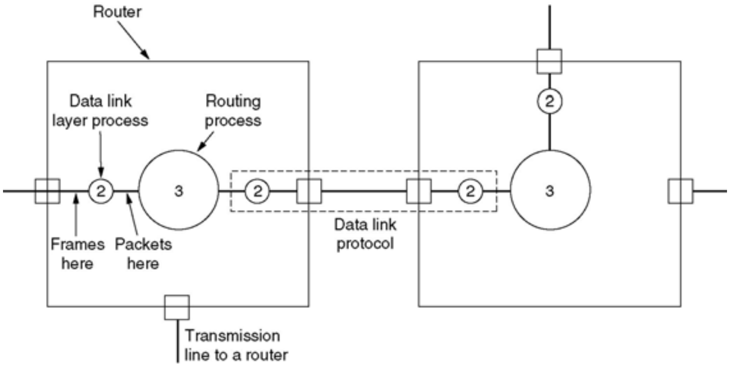
Relationship between packets and frames.

Services Provided to Network Layer



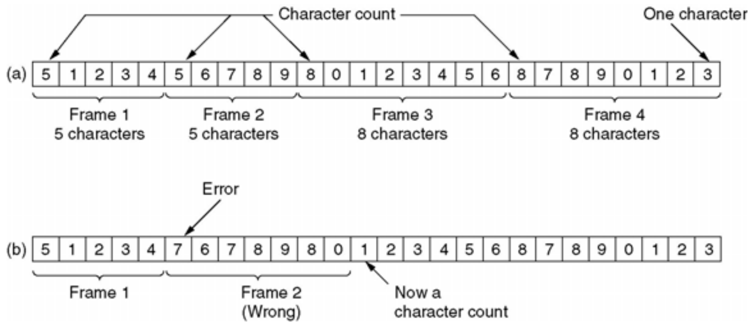
(a) Virtual communication.
(b) Actual communication.

Services Provided to Network Layer (2)



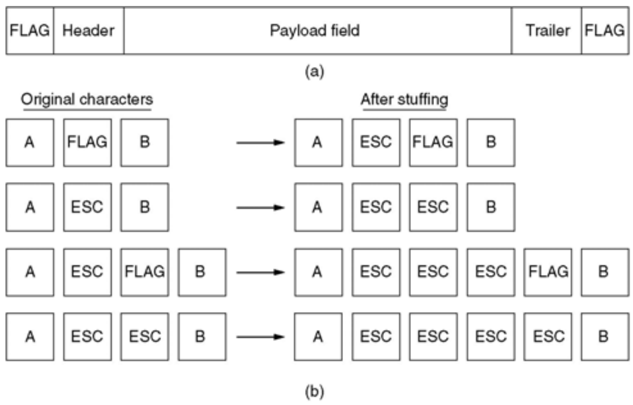
Placement of the data link protocol.

Framing



A character stream. (a) Without errors. (b) With one error.

Framing (2)



(a) A frame delimited by flag bytes.
(b) Insertion of special escape byte (ESC) just before each "accidental" flag byte in the data.

Framing (3)

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0

Stuffed bits

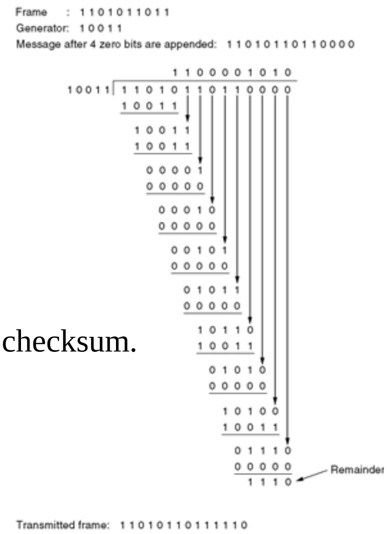
(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Bit stuffing
(a) The original data.
(b) The data as they appear on the line.
(c) The data as they are stored in receiver's memory after destuffing.

Error Detection and Correction

- Error-Correcting Codes
- Error-Detecting Codes

Error-Detecting Codes



Calculation of the polynomial code checksum.

Elementary Data Link Protocols

- An Unrestricted Simplex Protocol
- A Simplex Stop-and-Wait Protocol
- A Simplex Protocol for a Noisy Channel

Unrestricted Simplex Protocol

```
/* Protocol 1 (utopia) provides for data transmission in one direction only, from
   sender to receiver.  The communication channel is assumed to be error free,
   and the receiver is assumed to be able to process all the input infinitely quickly.
   Consequently, the sender just sits in a loop pumping data out onto the line as
   fast as it can. */

typedef enum {frame  arrival} event  type;
#include "protocol.h"

void sender1(void)
{
  frame s;                             /* buffer for an outbound frame */
  packet buffer;                       /* buffer for an outbound packet */

  while (true) {
      from_network_layer(&buffer);     /* go get something to send */
      s.info = buffer;                 /* copy it into s for transmission */
      to_physical_layer(&s);           /* send it on its way */
  }                          /          * Tomorrow, and tomorrow, and tomorrow,
                                         Creeps in this petty pace from day to day
                                         To the last syllable of recorded time
                                            - Macbeth, V, v */
}

void receiver1(void)
{
  frame r;
  event_type event;                    /* filled in by wait, but not used here */

  while (true) {
      wait_for_event(&event);          /* only possibility is frame_arrival */
      from_physical_layer(&r);         /* go get the inbound frame */
      to_network_layer(&r.info);       /* pass the data to the network layer */
  }
}
```

Simplex Stop-and-Wait Protocol

```
/* Protocol 2 (stop-and-wait) also provides for a one-directional flow of data from
   sender to receiver. The communication channel is once again assumed to be error
   free, as in protocol 1. However, this time, the receiver has only a finite buffer
   capacity and a finite processing speed, so the protocol must explicitly prevent
   the sender from flooding the receiver with data faster than it can be handled. */

typedef enum {frame_arrival} event_type;
#include "protocol.h"

void sender2(void)
{
 frame s;                           /* buffer for an outbound frame */
 packet buffer;                     /* buffer for an outbound packet */
 event_type event;                  /* frame_arrival is the only possibility */

 while (true) {
     from_network_layer(&buffer);   /* go get something to send */
     s.info = buffer;               /* copy it into s for transmission */
     to_physical_layer(&s);         /* bye bye little frame */
     wait_for_event(&event);        /* do not proceed until given the go ahead */
 }
}

void receiver2(void)
{
 frame r, s;                        /* buffers for frames */
 event_type event;                  /* frame_arrival is the only possibility */
 while (true) {
     wait_for_event(&event);        /* only possibility is frame_arrival */
     from_physical_layer(&r);       /* go get the inbound frame */
     to_network_layer(&r.info);     /* pass the data to the network layer */
     to_physical_layer(&s);         /* send a dummy frame to awaken sender */
 }
}
```

A Simplex Protocol for a Noisy Channel

```
/* Protocol 3 (par) allows unidirectional data flow over an unreliable channel. */

#define MAX_SEQ 1                           /* must be 1 for protocol 3 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"

void sender3(void)
{
 seq_nr next_frame_to_send;                 /* seq number of next outgoing frame */
 frame s;                                   /* scratch variable */
 packet buffer;                             /* buffer for an outbound packet */
 event_type event;

 next_frame_to_send = 0;                    /* initialize outbound sequence numbers */
 from_network_layer(&buffer);               /* fetch first packet */
 while (true) {
     s.info = buffer;                       /* construct a frame for transmission */
     s.seq = next_frame_to_send;            /* insert sequence number in frame */
     to_physical_layer(&s);                 /* send it on its way */
     start_timer(s.seq);                    /* if answer takes too long, time out */
     wait_for_event(&event);                /* frame_arrival, cksum_err, timeout */
     if (event == frame_arrival) {
          from_physical_layer(&s);          /* get the acknowledgement */
          if (s.ack == next_frame_to_send) {
               stop_timer(s.ack);           /* turn the timer off */
               from_network_layer(&buffer); /* get the next one to send */
               inc(next_frame_to_send);     /* invert next_frame_to_send */
          }
     }
 }
}
```

A positive acknowledgement with retransmission protocol.

Continued →

A Simplex Protocol for a Noisy Channel (ctd.)

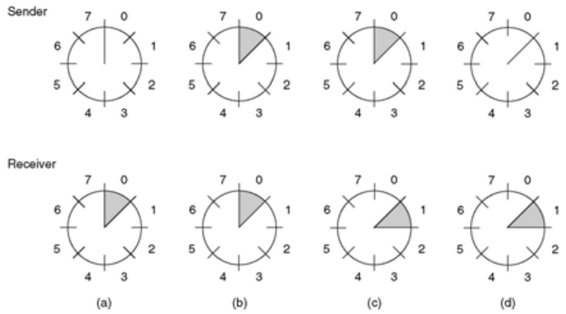
```
void receiver3(void)
{
  seq_nr frame_expected;
  frame r, s;
  event_type event;

  frame_expected = 0;
  while (true) {
      wait_for_event(&event);              /* possibilities: frame_arrival, cksum_err */
      if (event == frame_arrival) {        /* a valid frame has arrived. */
          from_physical_layer(&r);         /* go get the newly arrived frame */
          if (r.seq == frame_expected) {   /* this is what we have been waiting for. */
              to_network_layer(&r.info);   /* pass the data to the network layer */
              inc(frame_expected);         /* next time expect the other sequence nr */
          }
          s.ack = 1 – frame_expected;      /* tell which frame is being acked */
          to_physical_layer(&s);           /* send acknowledgement */
      }
  }
}
```

A positive acknowledgement with retransmission protocol.

Sliding Window Protocols

- A One-Bit Sliding Window Protocol
- A Protocol Using Go Back N
- A Protocol Using Selective Repeat

Sliding Window Protocols (2)



A sliding window of size 1, with a 3-bit sequence number.

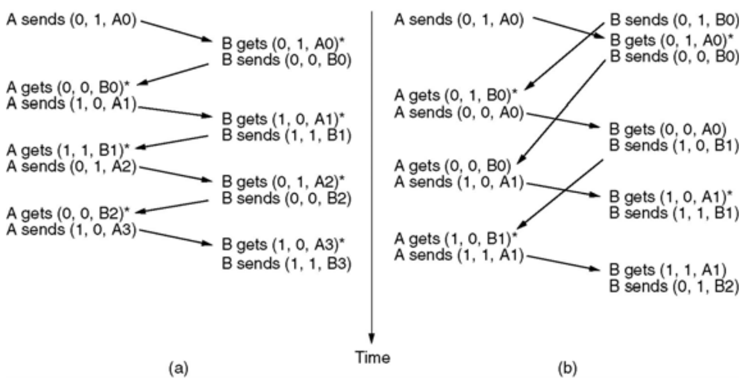
(a) Initially.

(b) After the first frame has been sent.

(c) After the first frame has been received.

(d) After the first acknowledgement has been received.

A One-Bit Sliding Window Protocol (2)



Two scenarios for protocol 4. (a) Normal case. (b) Abnormal case. The notation is (seq, ack, packet number). An asterisk indicates where a network layer accepts a packet.

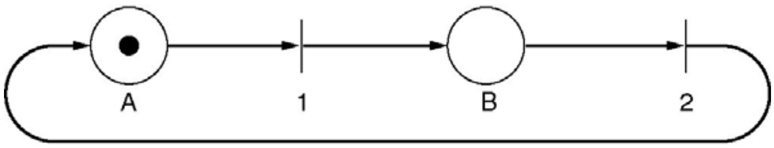
A Protocol Using Go Back N

Pipelining and error recovery. Effect on an error when
(a) Receiver's window size is 1.
(b) Receiver's window size is large.

Protocol Verification

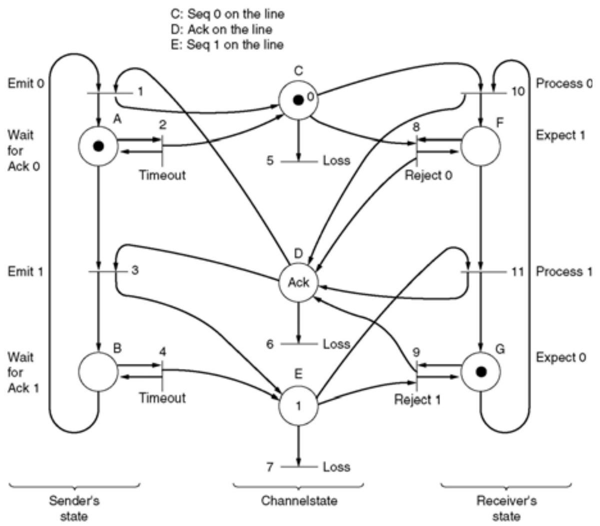
- Finite State Machined Models
- Petri Net Models

Petri Net Models



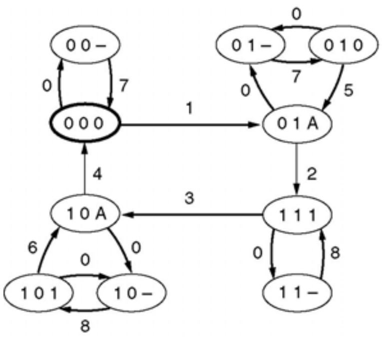
A Petri net with two places and two transitions.

Petri Net Models (2)



A Petri net model for protocol.

Finite State Machined Models



(a)

Transition	Who runs?	Frame accepted	Frame emitted	To network layer
0	–	(frame lost)		–
1	R	0	A	Yes
2	S	A	1	–
3	R	1	A	Yes
4	S	A	0	–
5	R	0	A	No
6	R	1	A	No
7	S	(timeout)	0	–
8	S	(timeout)	1	–

(b)

(a) State diagram. (b) Transmissions.

The Medium Access Control Sublayer

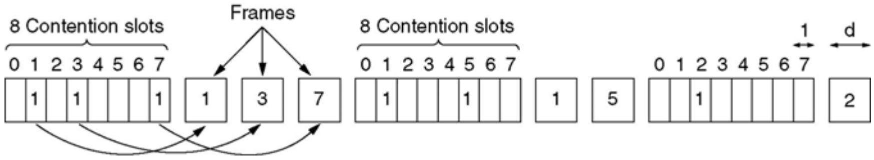
The Channel Allocation Problem

- Static Channel Allocation in LANs and MANs
- Dynamic Channel Allocation in LANs and MANs

Multiple Access Protocols

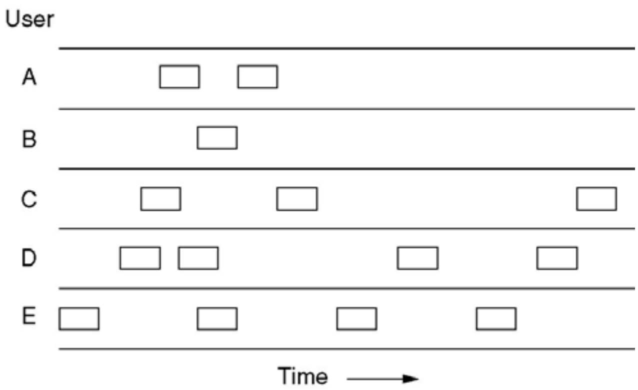
- Collision-Free Protocols
- ALOHA
- Carrier Sense Multiple Access Protocols
- Wireless LAN Protocols

Collision-Free Protocols



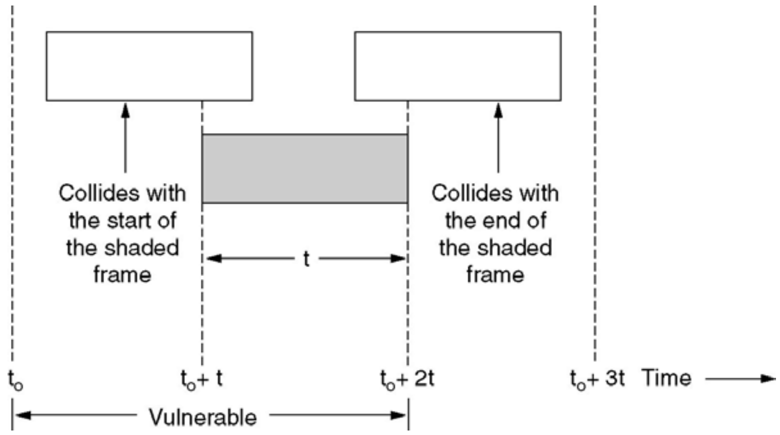
The basic bit-map protocol.

Pure ALOHA



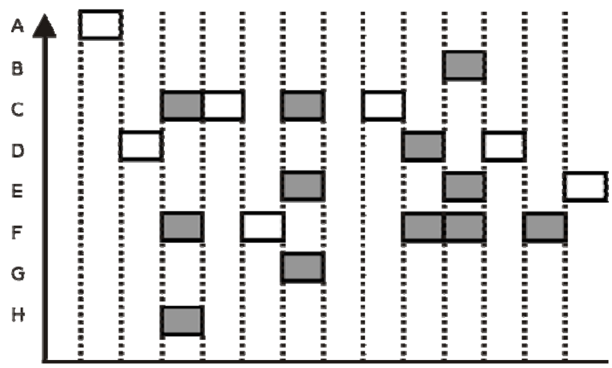
In pure ALOHA, frames are transmitted at completely arbitrary times (wireless data network)

Pure ALOHA (2)



Vulnerable period for the shaded frame.

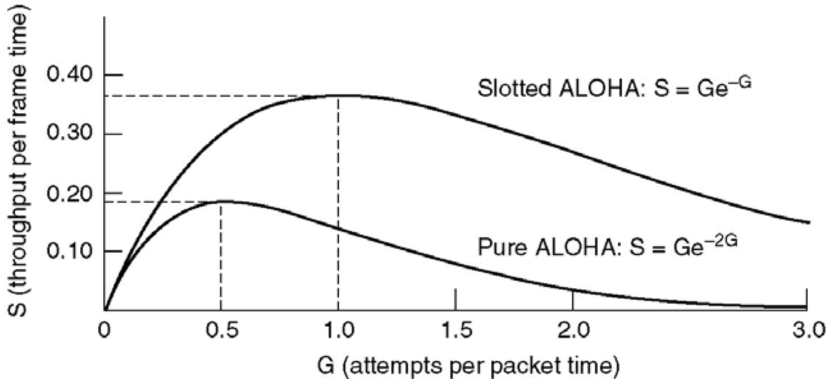
Slotted ALOHA



Slotted ALOHA protocol (shaded slots indicate collision)

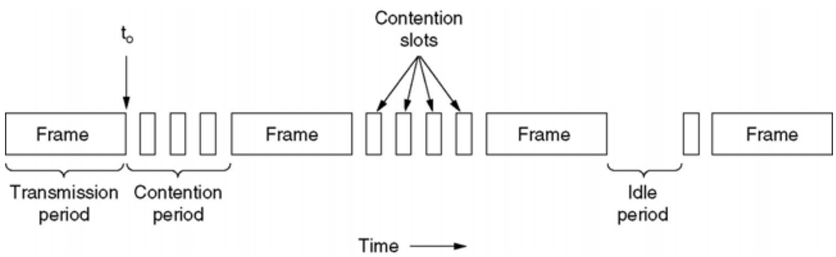
In Slotted ALOHA, introduction of discrete timeslots (wireless data network)

ALOHA



Throughput versus offered traffic for ALOHA systems.

CSMA with Collision Detection

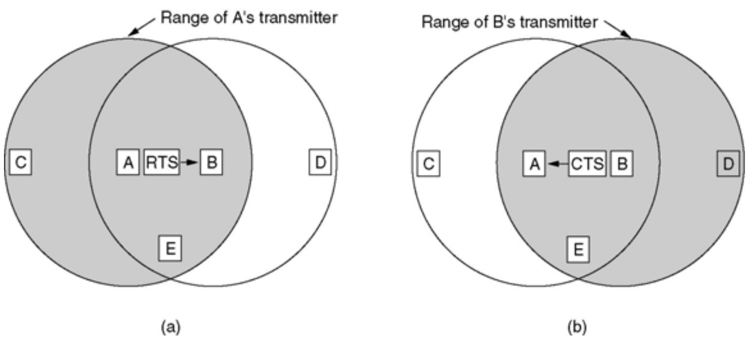


CSMA/CD can be in one of three states: contention, transmission, or idle.

Wireless LAN Protocols

A wireless LAN. (a) A transmitting. (b) B transmitting.

Wireless LAN Protocols (2)



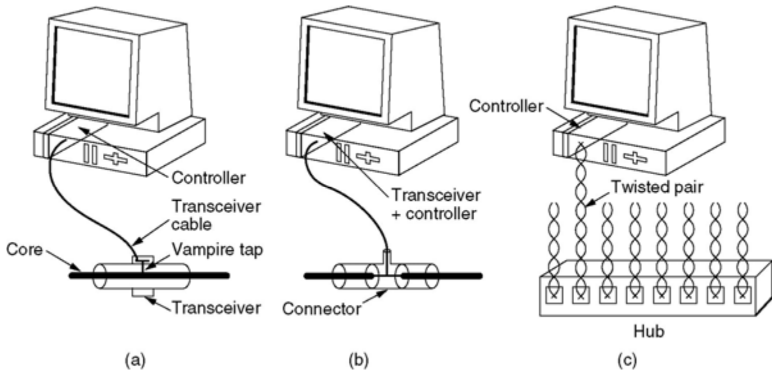
The MACA protocol. (a) A sending an RTS to B.
(b) B responding with a CTS to A.

Ethernet

Ethernet

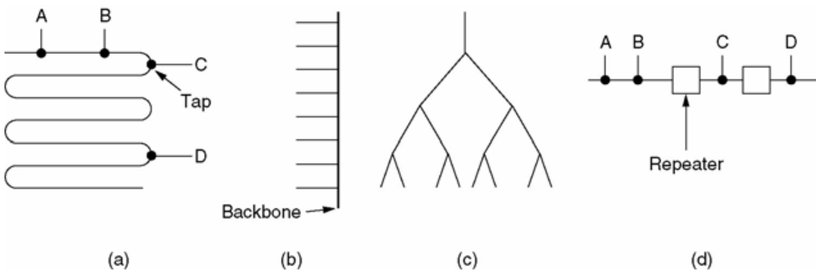
- **Ethernet Cabling**
- **The Ethernet MAC Sublayer Protocol**
- **IEEE 802.2: Logical Link Control**

Ethernet Cabling (1)



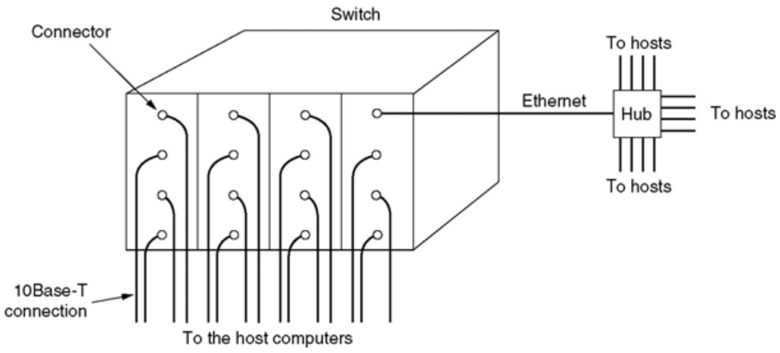
Three kinds of Ethernet cabling.
(a) 10Base5, (b) 10Base2, (c) 10Base-T.

Ethernet Cabling (2)



Cable topologies. (a) Linear, (b) Spine, (c) Tree, (d) Segmented.

Switched Ethernet



A simple example of switched Ethernet.

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

The most common kinds of Ethernet cabling.

Fast Ethernet

Name	Cable	Max. segment	Advantages
100Base-T4	Twisted pair	100 m	Uses category 3 UTP
100Base-TX	Twisted pair	100 m	Full duplex at 100 Mbps
100Base-FX	Fiber optics	2000 m	Full duplex at 100 Mbps; long runs

The original fast Ethernet cabling.

Gigabit Ethernet

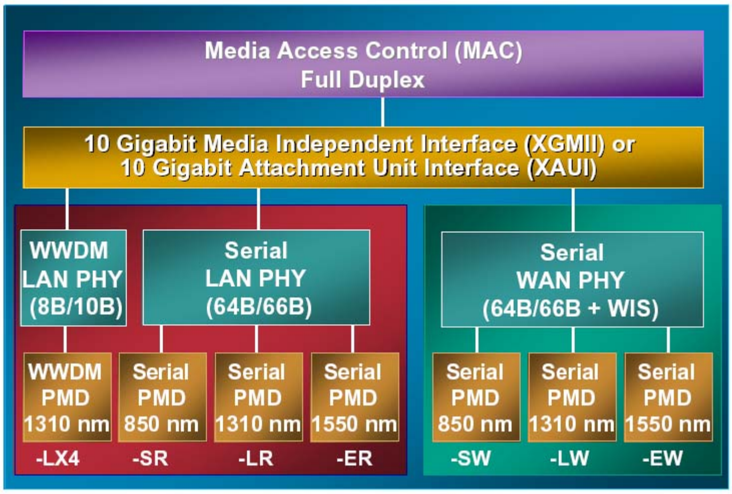
Name	Cable	Max. segment	Advantages
1000Base-SX	Fiber optics	550 m	Multimode fiber (50, 62.5 microns)
1000Base-LX	Fiber optics	5000 m	Single (10 µ) or multimode (50, 62.5 µ)
1000Base-CX	2 Pairs of STP	25 m	Shielded twisted pair
1000Base-T	4 Pairs of UTP	100 m	Standard category 5 UTP

Gigabit Ethernet cabling.

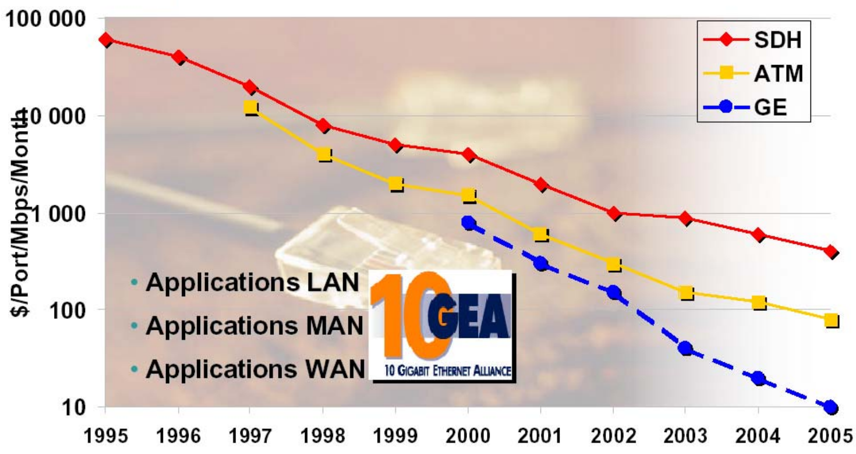
802.3ae 10 GbE Optical Transceivers

PMD	Fiber Supported	Diameter (Microns)	Bandwidth (MHz*km)	Distance (Meters)
850 nm serial	multimode	50*	400	66
1310 nm WWDM	multimode single mode	62.5 9.0**	160 N.A.	300 10K
1310 nm serial	single mode	9.0	N.A.	10 k
1550 nm serial	single mode	9.0	N.A.	40 k

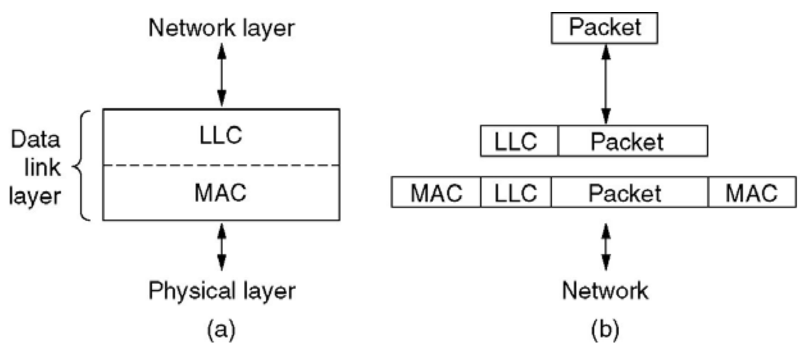
10 GbE Layer Diagram



economic facts

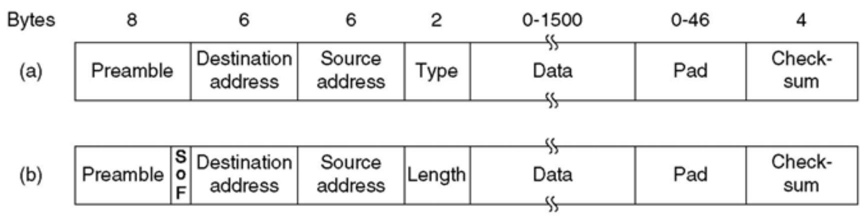


IEEE 802.2: Logical Link Control and Media Access Control



(a) Position of LLC & MAC sublayer.
(b) Protocol formats.

MAC Sublayer Protocol



Frame formats. (a) DIX Ethernet, (b) IEEE 802.3.

@ Ethernet

a) Address = 6 Bytes (48 bits)

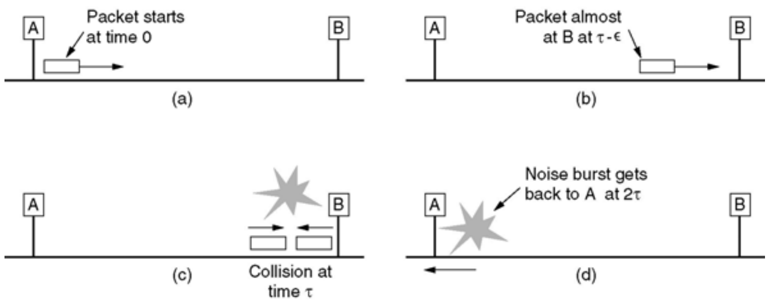
b) hex Notation :

- 08:00:20:06:D4:E8

c) @ Broadcast FF:FF:FF:FF:FF:FF

- All terminal

MAC Sublayer Protocol (2)



Collision detection can take as long as 2τ.

MAC Sublayer Protocol (3)

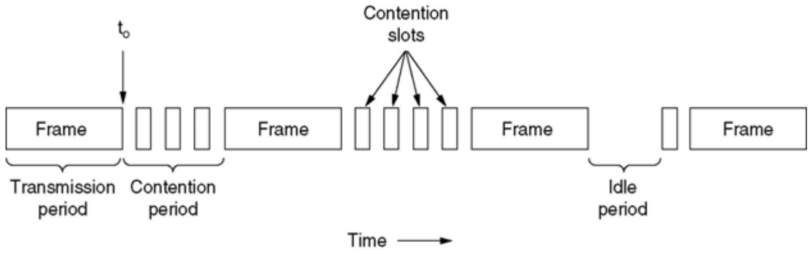
a) collision detection is an analog process. :
   - The station's hardware must listen to the cable while it is transmitting.
   - If what it reads back is different from what it is putting out
      - » a collision is occurring.

The implication is that the signal encoding must allow collisions to be detected (e.g., a collision of two 0-volt signals may well be impossible to detect).

MAC Sublayer Protocol (4)

a) After collision detection :

- waits a random period of time
  - $2^i - 1$ round-trip propagation time
  - i = min(10, collisions occurence)



LLC Sublayer Protocol

a) hides the differences between the various kinds of 802 networks by providing a single format and interface to the network layer

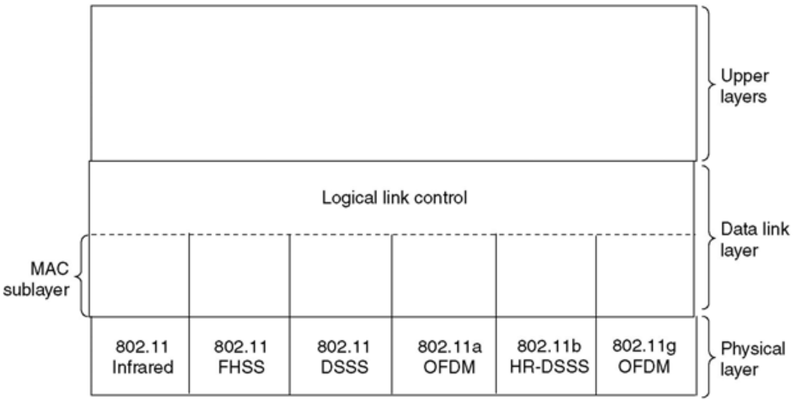
b) Typical usage of LLC is as follows :
   - The network layer on the sending machine passes a packet to LLC, using the LLC access primitives.
   - The LLC sublayer then adds an LLC header, containing sequence and acknowledgement numbers.

Wireless LANs

Wireless LANs

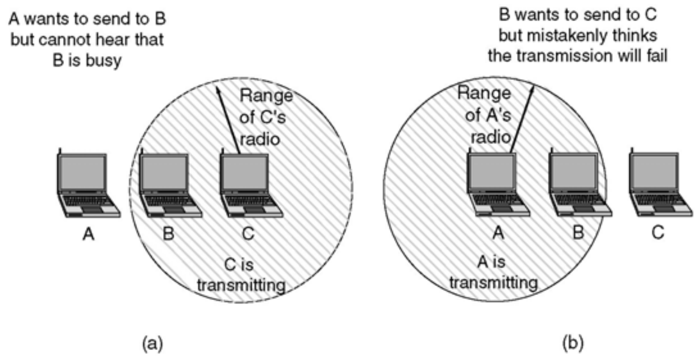
- The 802.11 Protocol Stack
- The 802.11 Physical Layer
- The 802.11 MAC Sublayer Protocol
- The 802.11 Frame Structure
- Services

The 802.11 Protocol Stack



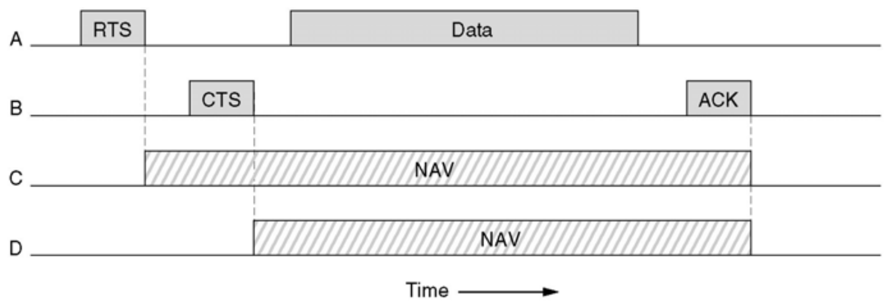
Part of the 802.11 protocol stack.

The 802.11 MAC Sublayer Protocol



(a) The hidden station problem.
(b) The exposed station problem.

The 802.11 MAC Sublayer Protocol (2)



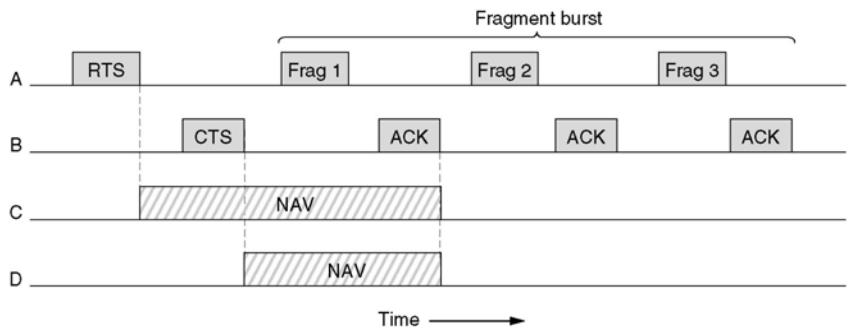
The use of virtual channel sensing using

Carrier **S**ense **M**ultiple **A**ccess with **C**ollision **A**voidance

RTS : Ready To Send

CTS :*Clear To Send*

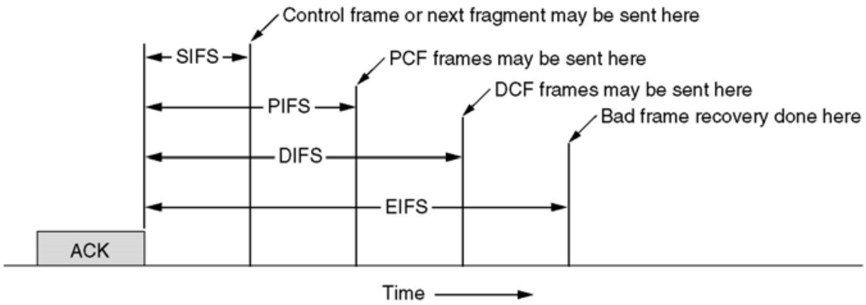
The 802.11 MAC Sublayer Protocol (3)



A fragment burst.

NAV : Network Allocation Vector

The 802.11 MAC Sublayer Protocol (4)



Interframe spacing in 802.11.

SIFS : Short Inter-Frame Spacing

PIFS : Point Coordination Function Inter-Frame Spacing

DIFS : Distributed Coordination Function Inter-Frame Spacing

EIFS : Extended Inter-Frame Spacing

The 802.11 MAC Sublayer Protocol (5)

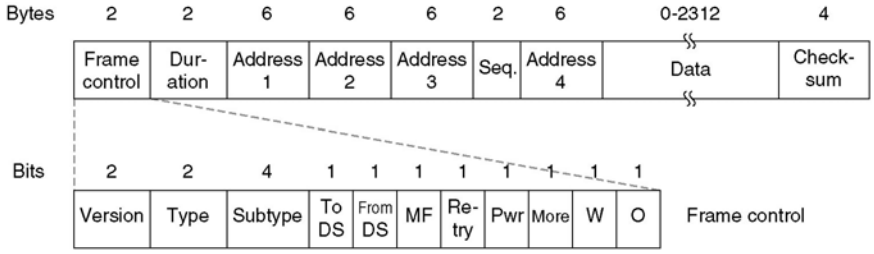
a) SIFS :
   - receiver can send a CTS to respond to an RTS,
   - receiver can send an ACK for a fragment or full data frame,
   - sender of a fragment burst can transmit the next fragment without having to send an RTS again.

b) PIFS
   - Reserved to base station to send a beacon frame or poll frame..

c) DIFS
   - any station may attempt to acquire the channel to send a new frame.

d) EIFS
   - a station that has just received a bad or unknown frame can report it

The 802.11 Frame Structure



The 802.11 data frame.

802.11 Services

Distribution Services

- Association
- Disassociation
- Reassociation (for mobility)
- Distribution
- Integration

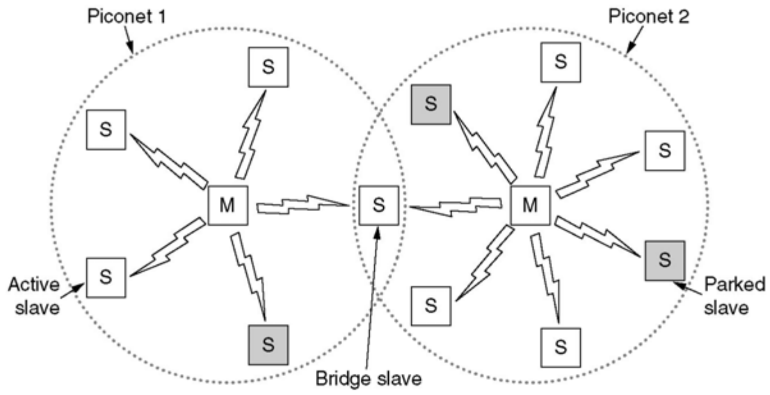
802.11 Services

Intracell Services

- Authentication
- Deauthentication
- Privacy
- Data Delivery

Bluetooth

Bluetooth Architecture



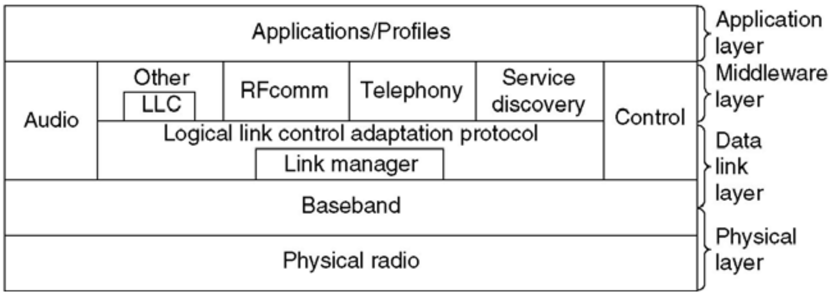
Two piconets can be connected to form a scatternet.

Bluetooth Applications

Name	Description
Generic access	Procedures for link management
Service discovery	Protocol for discovering offered services
Serial port	Replacement for a serial port cable
Generic object exchange	Defines client-server relationship for object movement
LAN access	Protocol between a mobile computer and a fixed LAN
Dial-up networking	Allows a notebook computer to call via a mobile phone
Fax	Allows a mobile fax machine to talk to a mobile phone
Cordless telephony	Connects a handset and its local base station
Intercom	Digital walkie-talkie
Headset	Intended for hands-free voice communication
Object push	Provides a way to exchange simple objects
File transfer	Provides a more general file transfer facility
Synchronization	Permits a PDA to synchronize with another computer

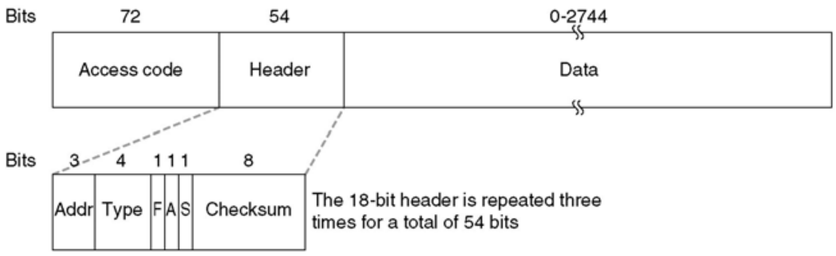
The Bluetooth profiles.

The Bluetooth Protocol Stack



The 802.15 version of the Bluetooth protocol architecture.

The Bluetooth Frame Structure



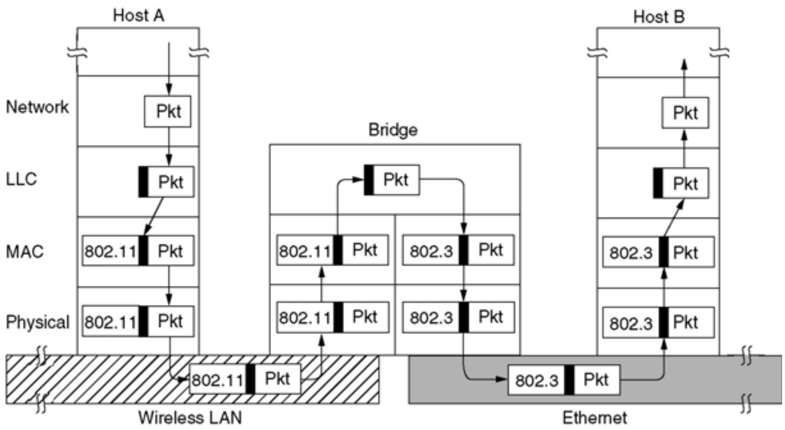
A typical Bluetooth data frame.

Data Link Layer Switching

Data Link Layer Switching

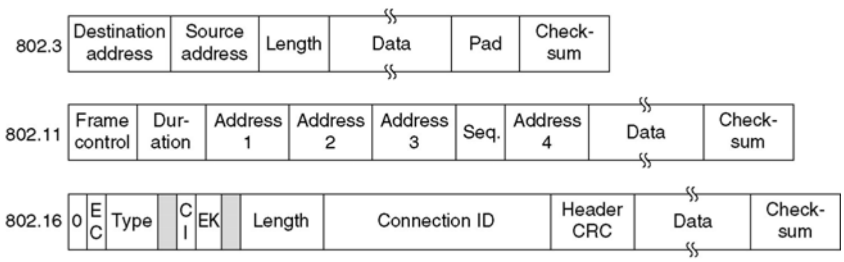
- Bridges from 802.x to 802.y
- Local Internetworking
- Spanning Tree Bridges
- Remote Bridges
- Repeaters, Hubs, Bridges, Switches, Routers, Gateways
- Virtual LANs

Bridges from 802.x to 802.y



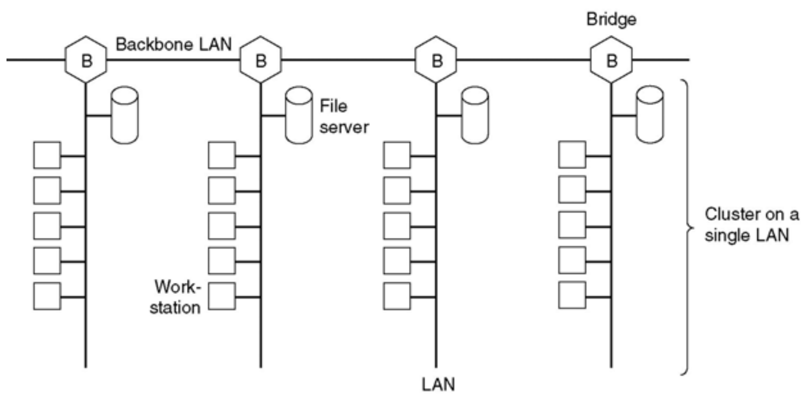
Operation of a LAN bridge from 802.11 to 802.3.

Bridges from 802.x to 802.y (2)



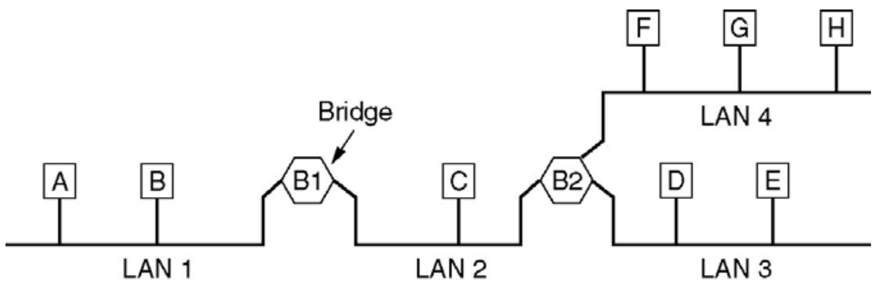
The IEEE 802 frame formats.

Data Link Layer Switching



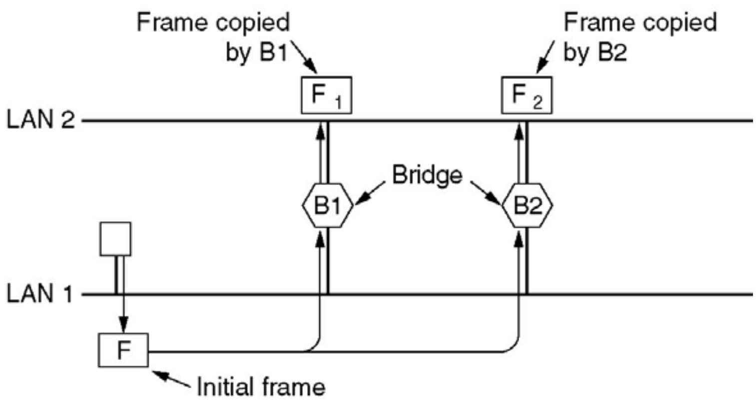
Multiple LANs connected by a backbone to handle a total load higher than the capacity of a single LAN.

Local Internetworking



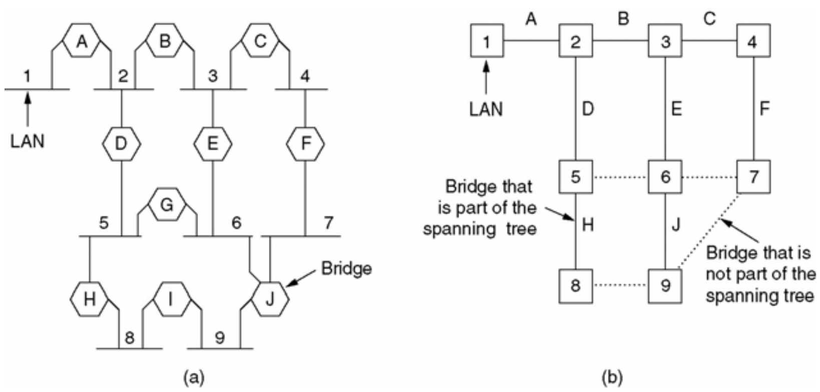
A configuration with four LANs and two bridges.

Spanning Tree Bridges



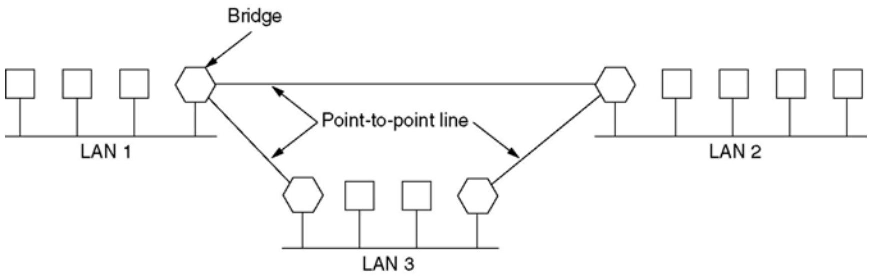
Two parallel transparent bridges.

Spanning Tree Bridges (2)



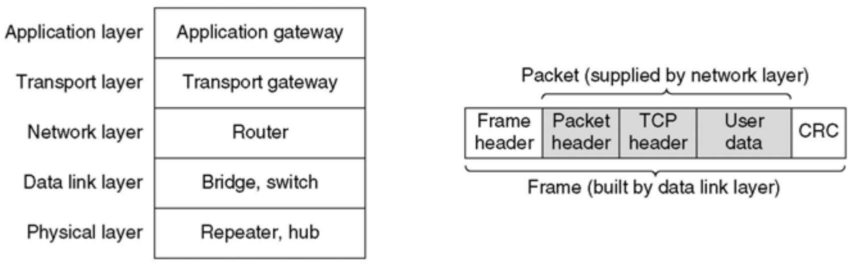
(a) Interconnected LANs. (b) A spanning tree covering the LANs. The dotted lines are not part of the spanning tree.

Remote Bridges



Remote bridges can be used to interconnect distant LANs.

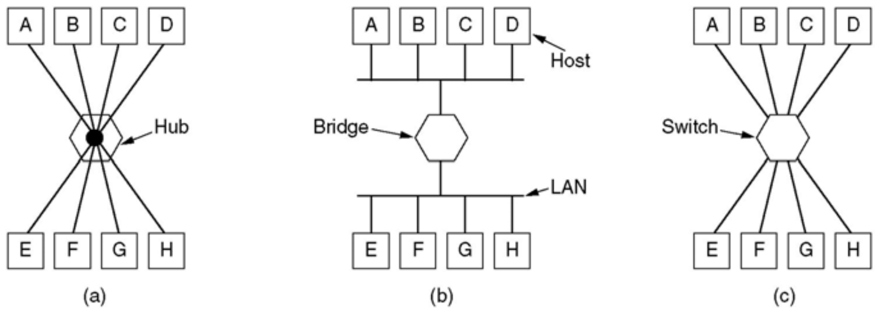
Repeaters, Hubs, Bridges, Switches, Routers and Gateways



(a) (b)

(a) Which device is in which layer.
(b) Frames, packets, and headers.

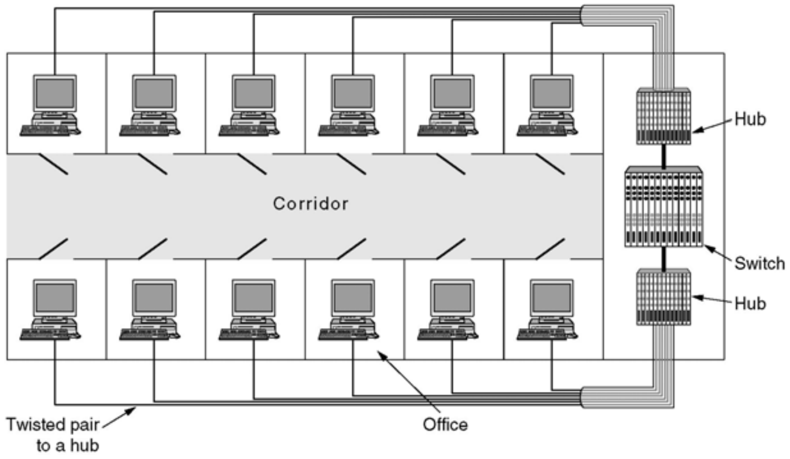
Repeaters, Hubs, Bridges, Switches, Routers and Gateways (2)



(a) A hub. (b) A bridge. (c) a switch.

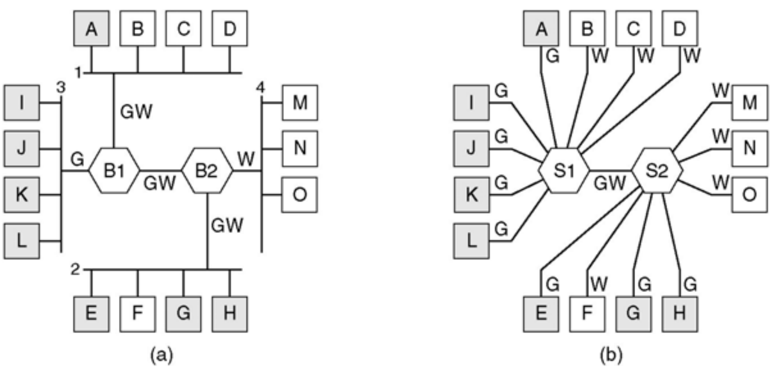
Virtual LANs

Virtual LANs



A building with centralized wiring using hubs and a switch.

Virtual LANs (2)



(a) Four physical LANs organized into two VLANs, gray and white, by two bridges. (b) The same 15 machines organized into two VLANs by switches.

Virtual LANs (3)

Three VLAN methods :

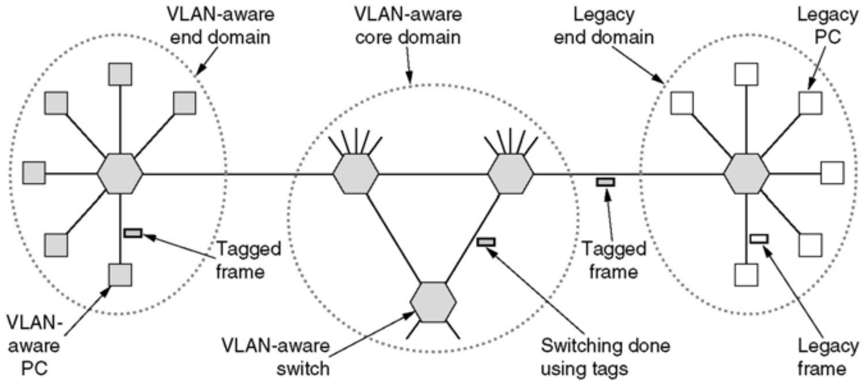
a) Every port is assigned to a VLAN.
b) Every MAC address is assigned to a VLAN.
c) Every layer 3 protocol or IP address is assigned to a VLAN.

Virtual LANs (4)

Three VLAN methods :

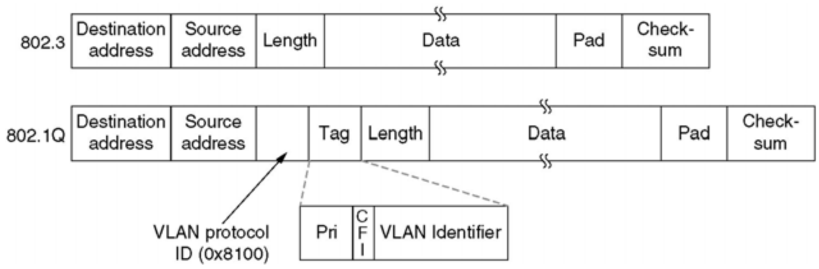
a) Every port is assigned to a VLAN.
b) Every MAC address is assigned to a VLAN.
c) Every layer 3 protocol or IP address is assigned to a VLAN.

The IEEE 802.1Q Standard



Transition from legacy Ethernet to VLAN-aware Ethernet. The shaded symbols are VLAN aware. The empty ones are not.

The IEEE 802.1Q Standard (2)



The 802.3 (legacy) and 802.1Q Ethernet frame formats.