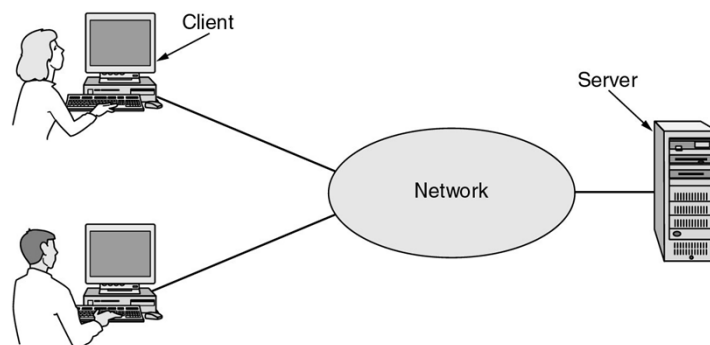# Chapter 1

Introduction

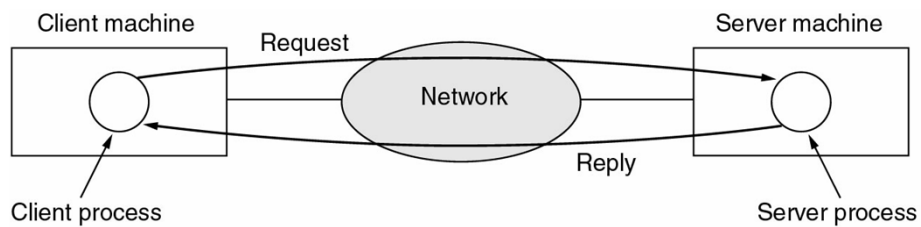---

Uses of Computer Networks

- Business Applications
- Home Applications
- Mobile Users
- Social Issues

Business Applications of Networks

A network with two clients and one server.
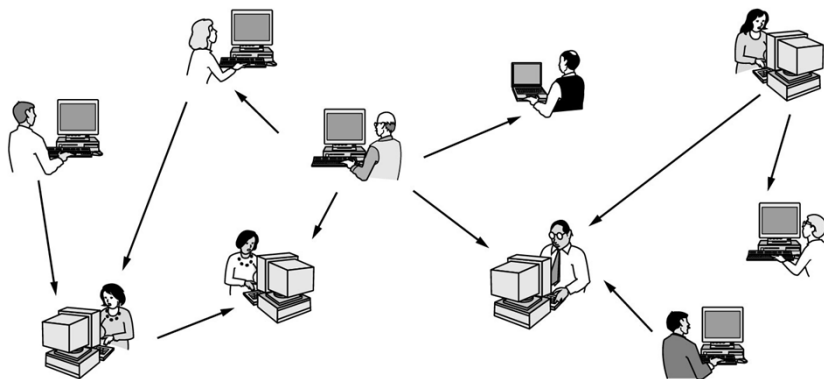
Business Applications of Networks (2)

The client-server model involves requests and replies.

Home Network Applications

- Access to remote information
- Person-to-person communication
- Interactive entertainment
- Electronic commerce

Home Network Applications (2)

In peer-to-peer system there are no fixed clients and servers.

Home Network Applications (3)

Tag	Full name	Example
B2C	Business-to-consumer	Ordering books on-line
B2B	Business-to-business	Car manufacturer ordering tires from supplier
G2C	Government-to-consumer	Government distributing tax forms electronically
C2C	Consumer-to-consumer	Auctioning second-hand products on-line
P2P	Peer-to-peer	File sharing

Some forms of e-commerce.

Mobile Network Users

Wireless	Mobile	Applications
No	No	Desktop computers in offices
No	Yes	A notebook computer used in a hotel room
Yes	No	Networks in older, unwired buildings
Yes	Yes	Portable office; PDA for store inventory

Combinations of wireless networks and mobile computing.

Network Hardware

- Local Area Networks
- Metropolitan Area Networks
- Wide Area Networks
- Wireless Networks
- Home Networks
- Internetworks
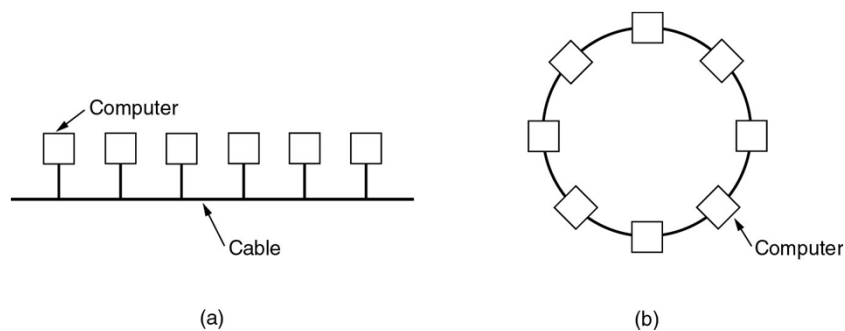
Broadcast Networks

Types of transmission technology
- Broadcast links
- Point-to-point links

Broadcast Networks (2)

Interprocessor distance	Processors located in same	Example
1 m	Square meter	Personal area network
10 m	Room	Local area network
100 m	Building	Local area network
1 km	Campus	Local area network
10 km	City	Metropolitan area network
100 km	Country	Wide area network
1000 km	Continent	Wide area network
10,000 km	Planet	The Internet

Classification of interconnected processors by scale.
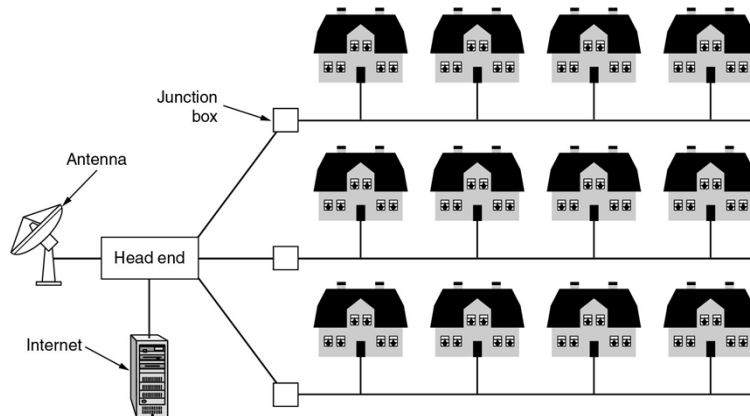
Local Area Networks

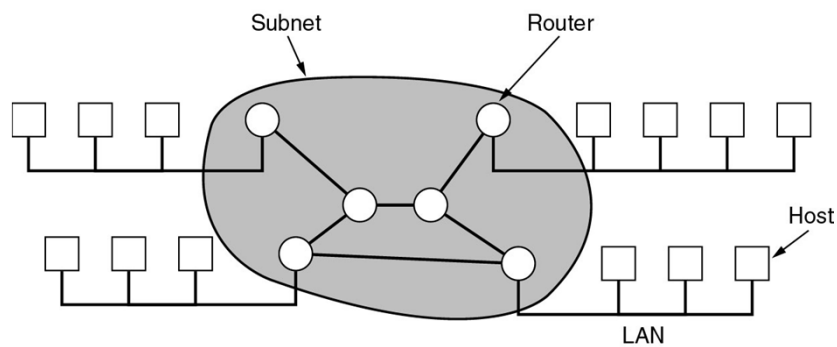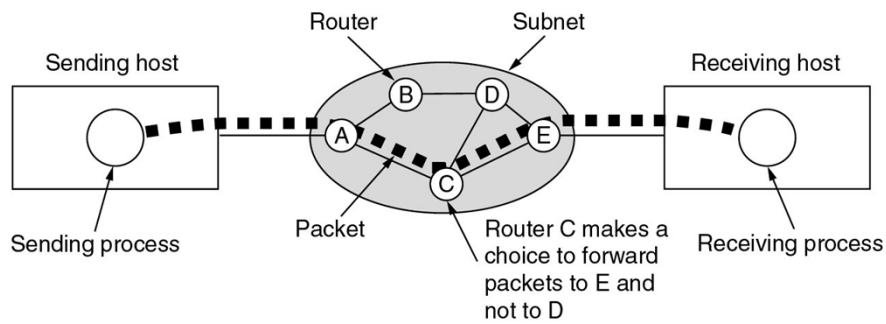Two broadcast networks
(a) Bus
(b) Ring

Metropolitan Area Networks

A metropolitan area network based on cable TV.

Wide Area Networks

Relation between hosts on LANs and the subnet.

Wide Area Networks (2)

Router   Subnet

Sending host   Receiving host

B   D

A   E

C

Sending process

Packet   Router C makes a   Receiving process
choice to forward
packets to E and
not to D

A stream of packets from sender to receiver.
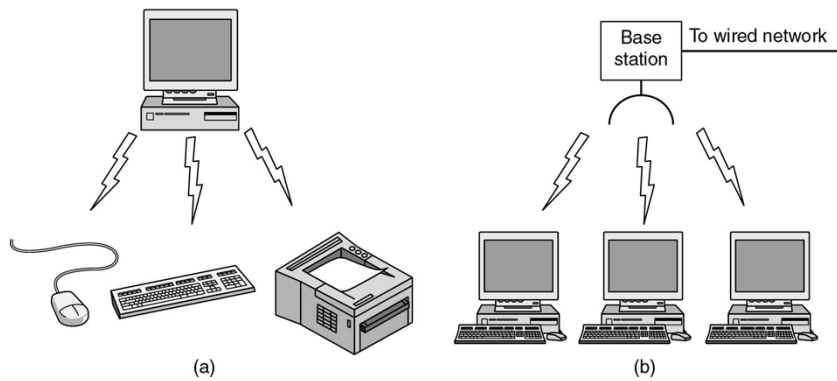
Wireless Networks

Categories of wireless networks:
- System interconnection
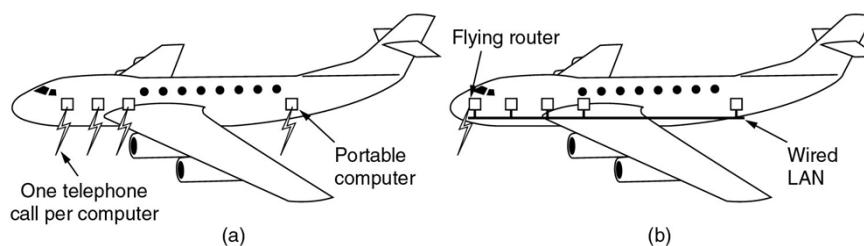- Wireless LANs
- Wireless WANs

Wireless Networks (2)

(a) Bluetooth configuration
(b) Wireless LAN

Wireless Networks (3)


(a) Individual mobile computers
(b) A flying LAN

Home Network Categories

- Computers (desktop PC, PDA, shared peripherals
- Entertainment (TV, DVD, VCR, camera, stereo, MP3)
- Telecomm (telephone, cell phone, intercom, fax)
- Appliances (microwave, fridge, clock, furnace, airco)
- Telemetry (utility meter, burglar alarm, babycam).
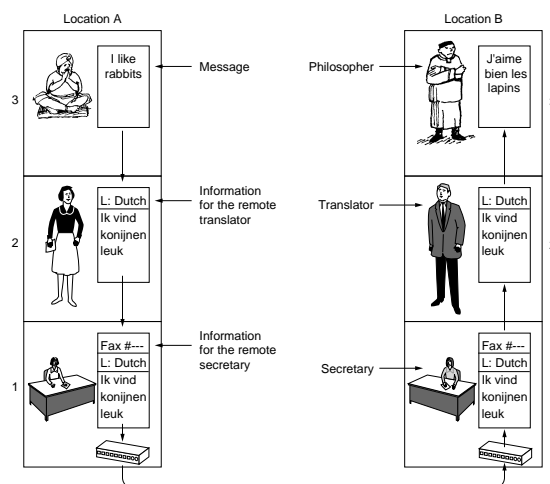
Network Software

- Protocol Hierarchies
- Design Issues for the Layers
- Connection-Oriented and Connectionless Services
- Service Primitives
- The Relationship of Services to Protocols
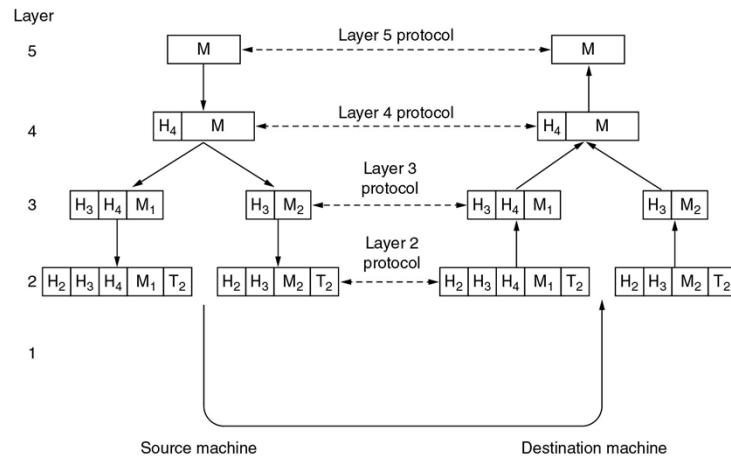
Network Software
Protocol Hierarchies
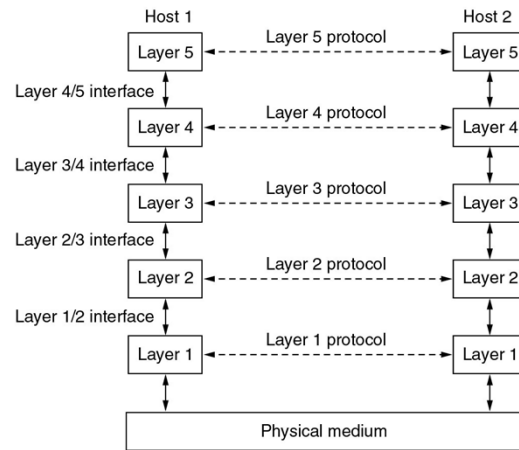
Layers, protocols, and interfaces.

Protocol Hierarchies (2)

The philosopher-translator-secretary architecture.

Protocol Hierarchies (3)

Example information flow supporting virtual communication in layer 5.

Design Issues for the Layers

- Addressing
- Error Control
- Flow Control
- Multiplexing
- Routing

Connection-Oriented and Connectionless Services

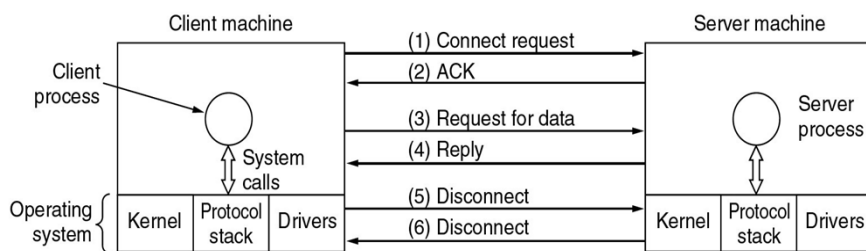|	Service	Example
Connection-oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Remote login
	Unreliable connection	Digitized voice
Connection-less	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Registered mail
	Request-reply	Database query

Six different types of service.

Service Primitives

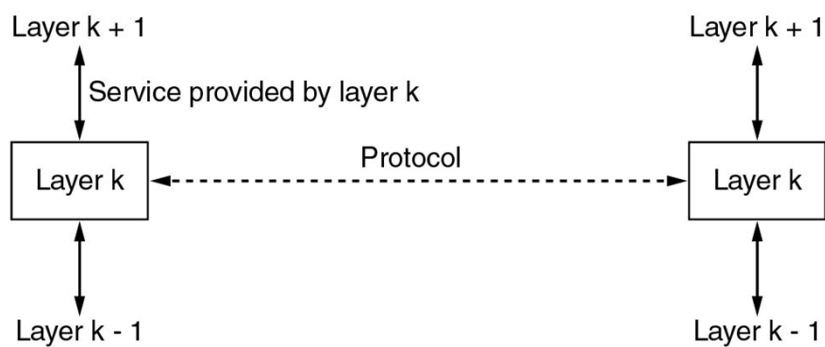| Primitive	Meaning
LISTEN	Block waiting for an incoming connection
CONNECT	Establish a connection with a waiting peer
RECEIVE	Block waiting for an incoming message
SEND	Send a message to the peer
DISCONNECT	Terminate a connection

Five service primitives for implementing a simple connection-oriented service.

Service Primitives (2)

Client machine		Server machine	

Client process

System calls

Operating system { Kernel | Protocol stack | Drivers

(1) Connect request
(2) ACK
(3) Request for data
(4) Reply
(5) Disconnect
(6) Disconnect

Server process

Kernel | Protocol stack | Drivers

Packets sent in a simple client-server interaction on a connection-oriented network.

Services to Protocols Relationship

Layer k + 1

Service provided by layer k

Layer k ‹- - - - - - - - - Protocol - - - - - - - - -› Layer k
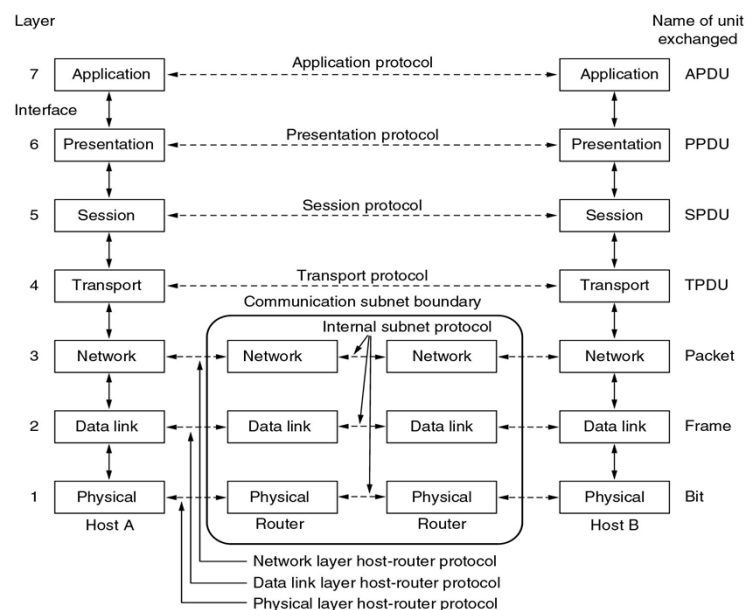
Layer k - 1

Layer k + 1

Layer k - 1

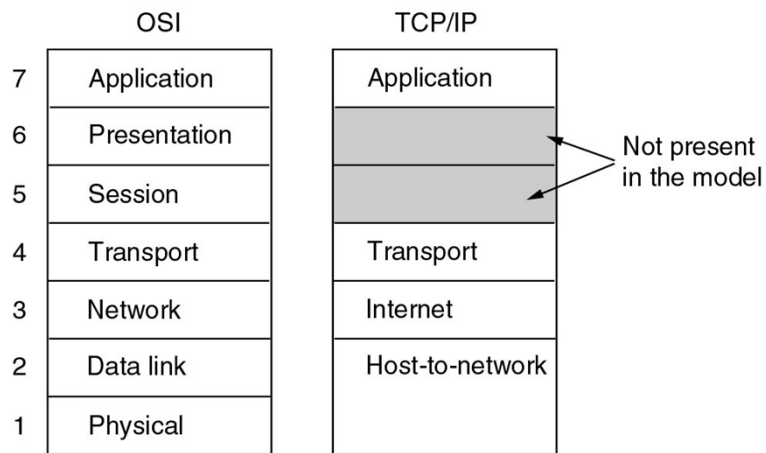The relationship between a service and a protocol.

Reference Models

- The OSI Reference Model
- The TCP/IP Reference Model
- A Comparison of OSI and TCP/IP
- A Critique of the OSI Model and Protocols
- A Critique of the TCP/IP Reference Model
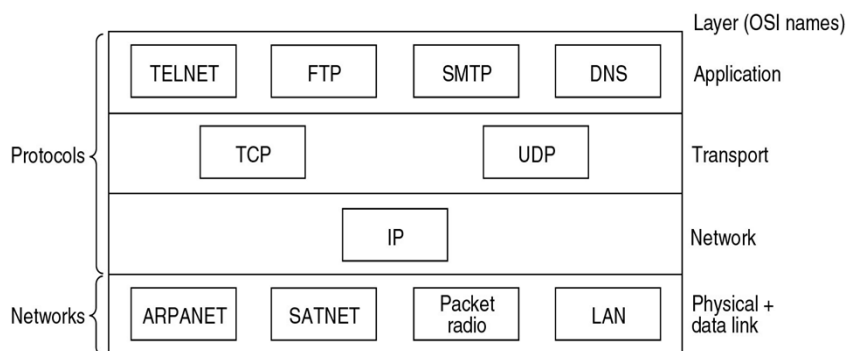
Reference Models

The OSI reference model.

Reference Models (2)

The TCP/IP reference model.

Reference Models (3)

Protocols and networks in the TCP/IP model initially.

Comparing OSI and TCP/IP Models

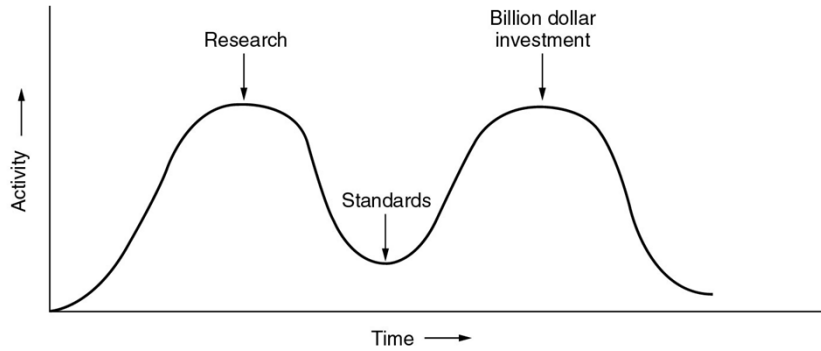Concepts central to the OSI model

- Services
- Interfaces
- Protocols

A Critique of the OSI Model and Protocols

Why OSI did not take over the world

- Bad timing
- Bad technology
- Bad implementations
- Bad politics

Bad Timing

The apocalypse of the two elephants.

A Critique of the TCP/IP Reference Model

Problems:
- Service, interface, and protocol not distinguished
- Not a general model
- Host-to-network "layer" not really a layer
- No mention of physical and data link layers
- Minor protocols deeply entrenched, hard to replace

Hybrid Model

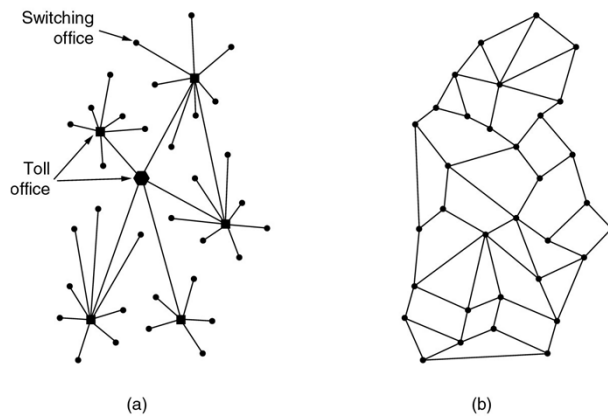|	
5	Application layer
4	Transport layer
3	Network layer
2	Data link layer
1	Physical layer

The hybrid reference model to be used in this book.

Example Networks

- The Internet

- Connection-Oriented Networks:
    X.25, Frame Relay, and ATM

- Ethernet

- Wireless LANs: 802:11

The ARPANET

(a) Structure of the telephone system.
(b) Baran's proposed distributed switching system.

The ARPANET (2)
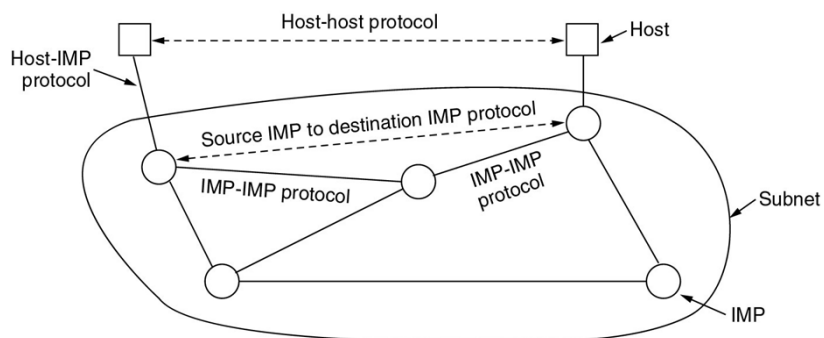
The original ARPANET design.

The ARPANET (3)
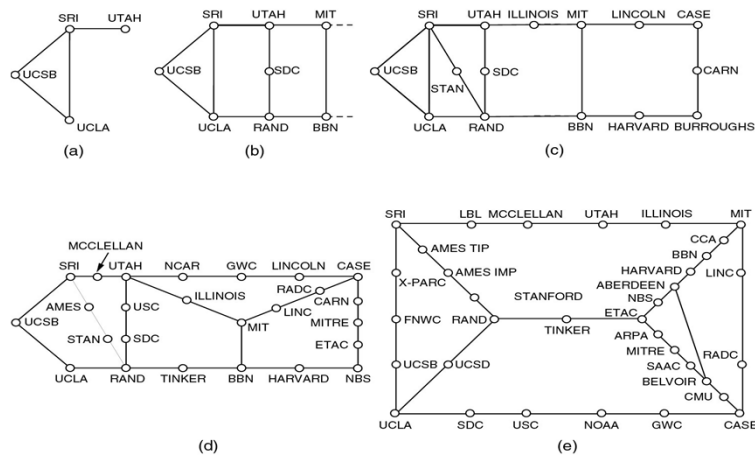
Growth of the ARPANET (a) December 1969.   (b) July 1970.
(c) March 1971.   (d) April 1972.   (e) September 1972.

NSFNET

○ NSF Supercomputer center
⊗ NSF Midlevel network
● Both
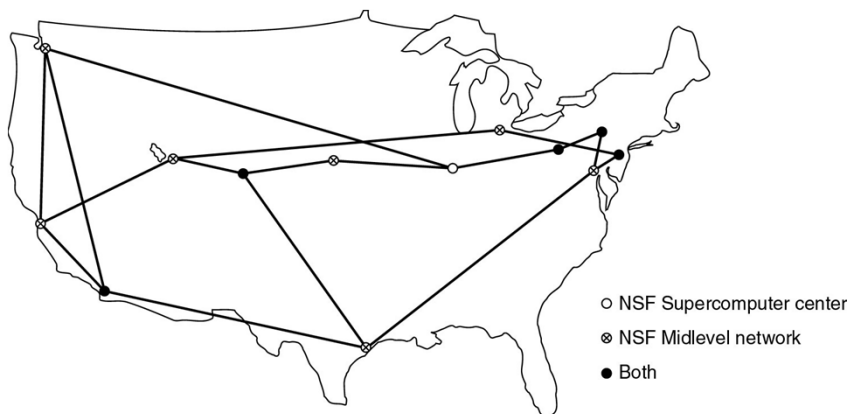
The NSFNET backbone in 1988.

Internet Usage

Traditional applications (1970 – 1990)

- E-mail
- News
- Remote login
- File transfer

Architecture of the Internet

Overview of the Internet.

ATM Virtual Circuits

A virtual circuit.

ATM Virtual Circuits (2)

Bytes	5	48
	Header	User data
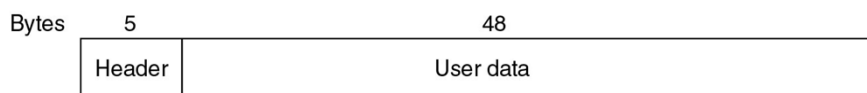
An ATM cell.

The ATM Reference Model

CS: Convergence sublayer
SAR: Segmentation and reassembly sublayer
TC: Transmission convergence sublayer
PMD: Physical medium dependent sublayer

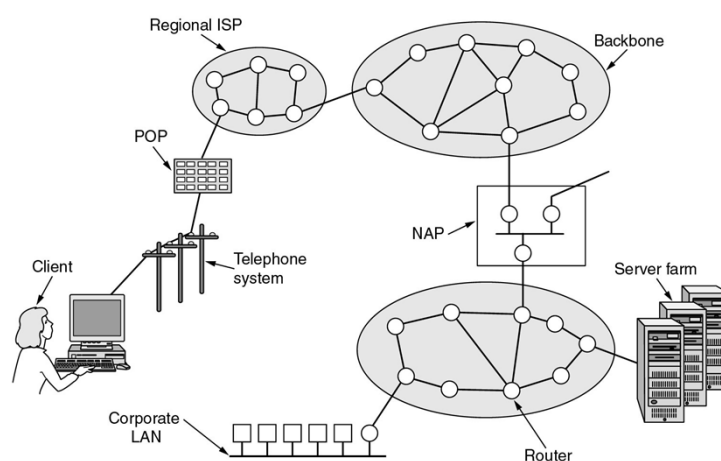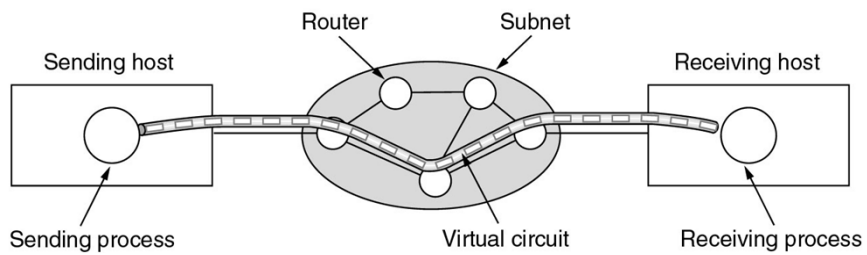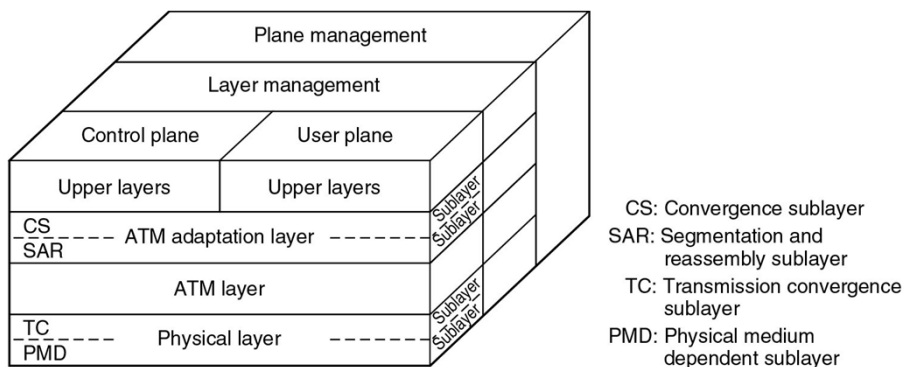The ATM reference model.

The ATM Reference Model (2)

OSI layer	ATM layer	ATM sublayer	Functionality
3/4	AAL	CS	Providing the standard interface (convergence)
		SAR	Segmentation and reassembly
2/3	ATM		Flow control Cell header generation/extraction Virtual circuit/path management Cell multiplexing/demultiplexing
2	Physical	TC	Cell rate decoupling Header checksum generation and verification Cell generation Packing/unpacking cells from the enclosing envelope Frame generation
1		PMD	Bit timing Physical network access

The ATM layers and sublayers and their functions.

Ethernet
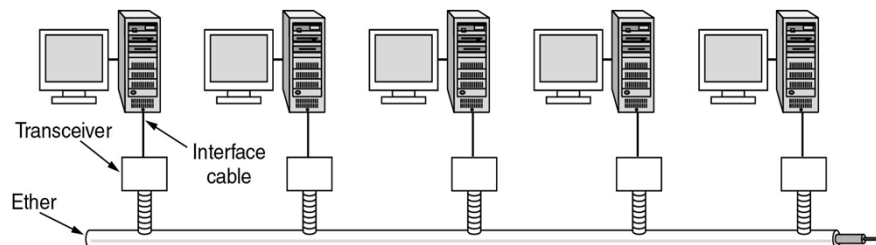
Architecture of the original Ethernet.

Wireless LANs
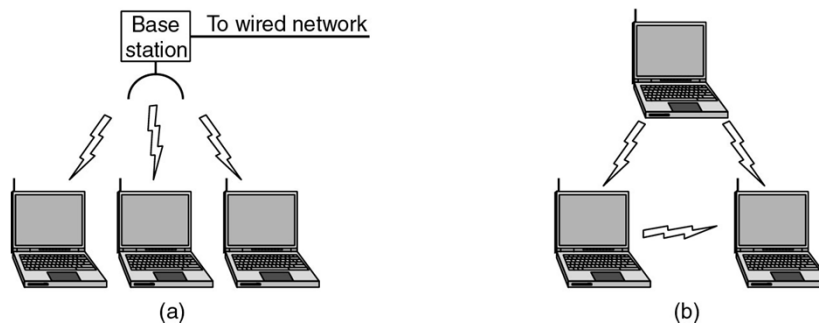
(a) Wireless networking with a base station.
(b) Ad hoc networking.

Wireless LANs (2)
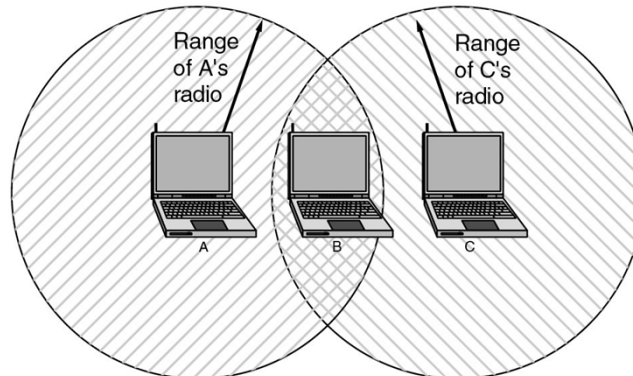
The range of a single radio may not cover the entire system.

Wireless LANs (3)
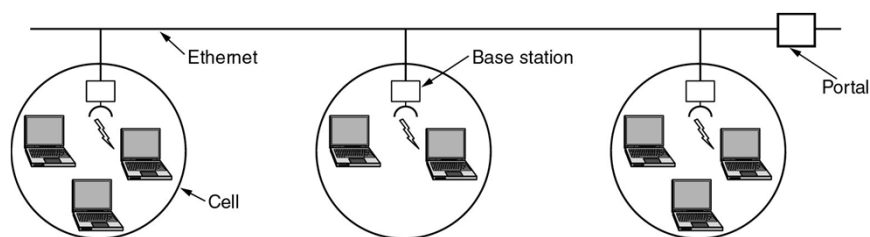
A multicell 802.11 network.

Network Standardization

- Who's Who in the Telecommunications World
- Who's Who in the International Standards World
- Who's Who in the Internet Standards World

ITU

- Main sectors
  - Radiocommunications
  - Telecommunications Standardization
  - Development
- Classes of Members
  - National governments
  - Sector members
  - Associate members
  - Regulatory agencies

IEEE 802 Standards

Number	Topic
802.1	Overview and architecture of LANs
802.2 ↓	Logical link control
802.3 *	Ethernet
802.4 ↓	Token bus (was briefly used in manufacturing plants)
802.5	Token ring (IBM's entry into the LAN world)
802.6 ↓	Dual queue dual bus (early metropolitan area network)
802.7 ↓	Technical advisory group on broadband technologies
802.8 †	Technical advisory group on fiber optic technologies
802.9 ↓	Isochronous LANs (for real-time applications)
802.10 ↓	Virtual LANs and security
802.11 *	Wireless LANs
802.12 ↓	Demand priority (Hewlett-Packard's AnyLAN)
802.13	Unlucky number. Nobody wanted it
802.14 ↓	Cable modems (defunct: an industry consortium got there first)
802.15 *	Personal area networks (Bluetooth)
802.16 *	Broadband wireless
802.17	Resilient packet ring

The 802 working groups. The important ones are marked with *. The ones marked with ↓ are hibernating. The one marked with † gave up.

Metric Units

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

The principal metric prefixes.

Chapter 2

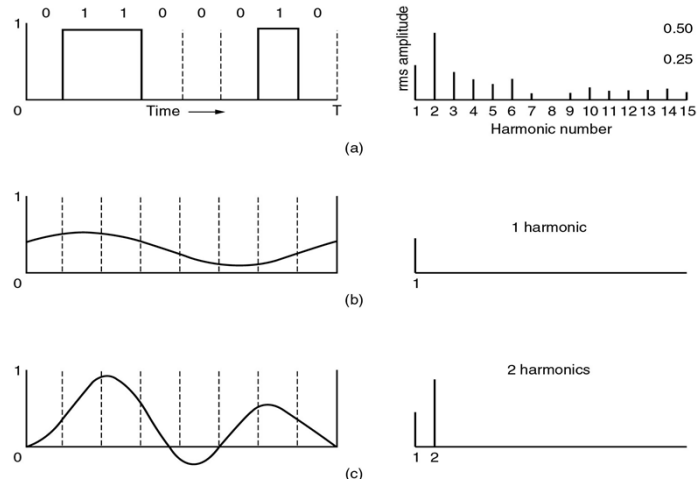## The Physical Layer

The Theoretical Basis for Data Communication

- Fourier Analysis
- Bandwidth-Limited Signals
- Maximum Data Rate of a Channel

Bandwidth-Limited Signals

A binary signal and its root-mean-square Fourier amplitudes.

(b) – (c) Successive approximations to the original signal.

Bandwidth-Limited Signals (2)
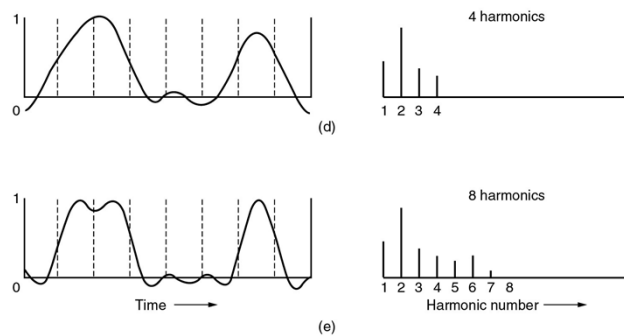
(d) – (e) Successive approximations to the original signal.

Bandwidth-Limited Signals (3)

Bps	T (msec)	First harmonic (Hz)	# Harmonics sent
300	26.67	37.5	80
600	13.33	75	40
1200	6.67	150	20
2400	3.33	300	10
4800	1.67	600	5
9600	0.83	1200	2
19200	0.42	2400	1
38400	0.21	4800	0

Relation between data rate and harmonics.

Guided Transmission Data

- Magnetic Media
- Twisted Pair
- Coaxial Cable
- Fiber Optics

Twisted Pair

(a)

(b)

(a) Category 3 UTP.
(b) Category 5 UTP.

Coaxial Cable

Copper core   Insulating material   Braided outer conductor   Protective plastic covering
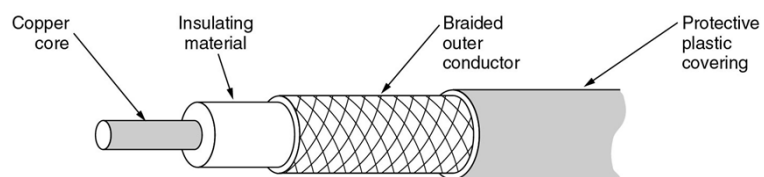
A coaxial cable.

Fiber Optics

(a) Three examples of a light ray from inside a silica fiber impinging on the air/silica boundary at different angles.

(b) Light trapped by total internal reflection.

Transmission of Light through Fiber
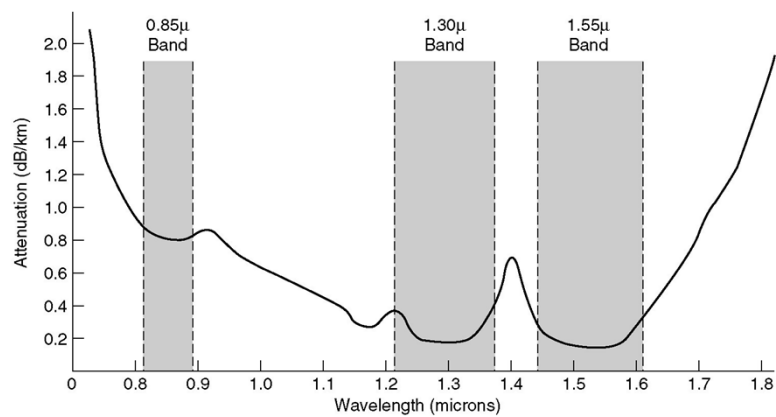
Attenuation of light through fiber in the infrared region.

Fiber Cables
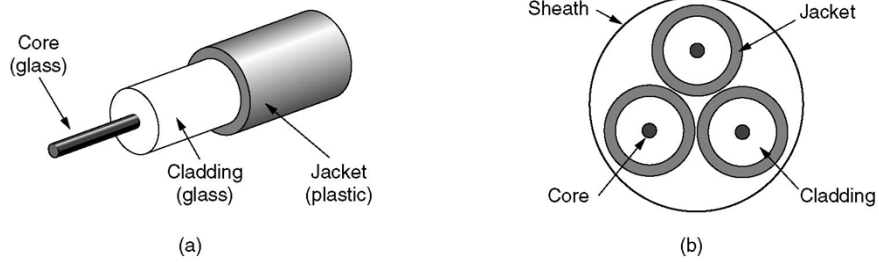
(a) Side view of a single fiber.
(b) End view of a sheath with three fibers.

Fiber Cables (2)

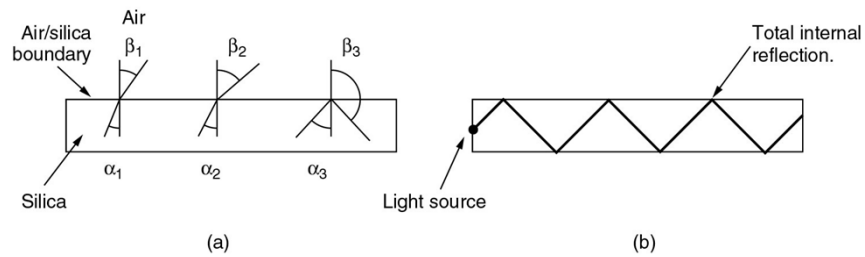| Item	LED	Semiconductor laser
Data rate	Low	High
Fiber type	Multimode	Multimode or single mode
Distance	Short	Long
Lifetime	Long life	Short life
Temperature sensitivity	Minor	Substantial
Cost	Low cost	Expensive

A comparison of semiconductor diodes and LEDs as light sources.

Fiber Optic Networks
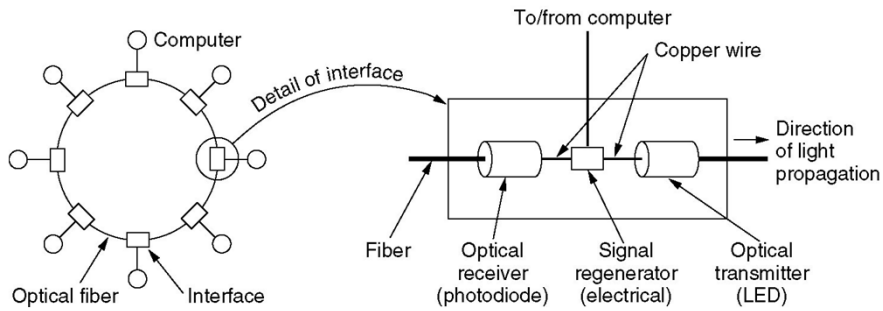
A fiber optic ring with active repeaters.

Fiber Optic Networks (2)
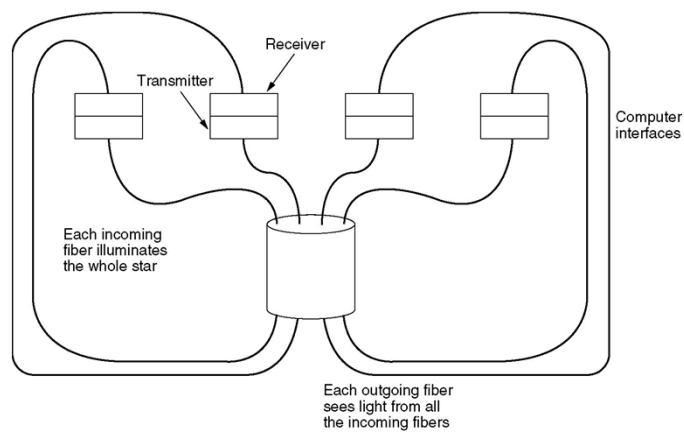
A passive star connection in a fiber optics network.

Wireless Transmission

- The Electromagnetic Spectrum
- Radio Transmission
- Microwave Transmission
- Infrared and Millimeter Waves
- Lightwave Transmission

The Electromagnetic Spectrum
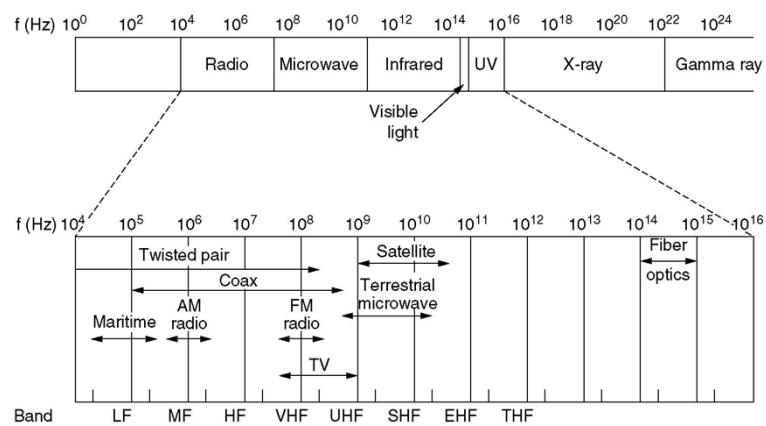
The electromagnetic spectrum and its uses for communication.

Radio Transmission
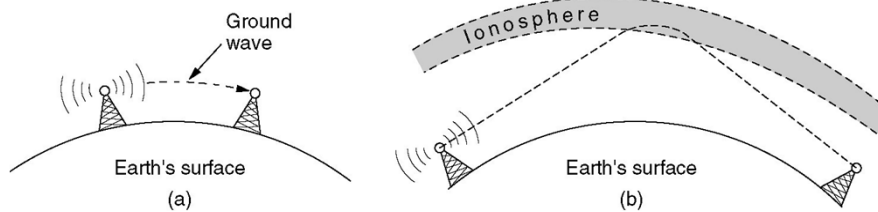
(a) In the VLF, LF, and MF bands, radio waves follow the curvature of the earth.

(b) In the HF band, they bounce off the ionosphere.

Politics of the Electromagnetic Spectrum
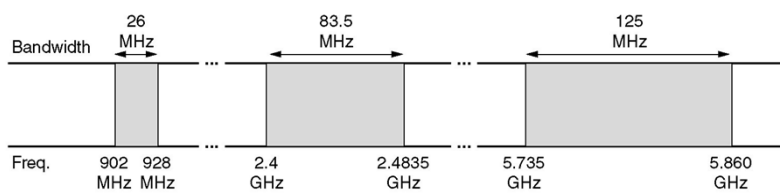
The ISM bands in the United States.

Lightwave Transmission
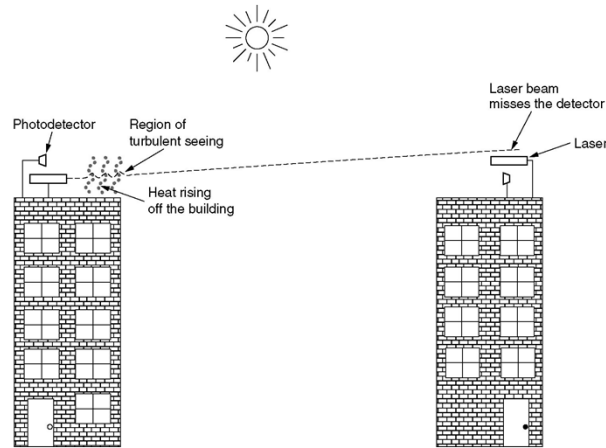
Convection currents can interfere with laser communication systems.
A bidirectional system with two lasers is pictured here.

Communication Satellites

- Geostationary Satellites
- Medium-Earth Orbit Satellites
- Low-Earth Orbit Satellites
- Satellites versus Fiber

Communication Satellites

Altitude (km)		Type	Latency (ms)	Sats needed
35,000		GEO	270	3
30,000				
25,000				
20,000				
	Upper Van Allen belt			
15,000				
10,000		MEO	35–85	10
5,000				
	Lower Van Allen belt		1–7	50
0		LEO		
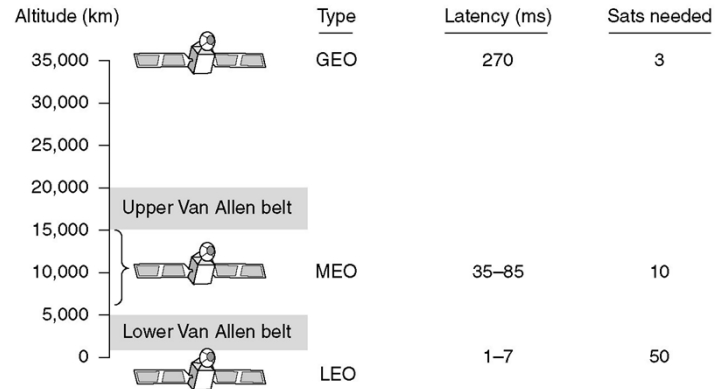
Communication satellites and some of their properties,
including altitude above the earth, round-trip delay time
and number of satellites needed for global coverage.

Communication Satellites (2)

Band	Downlink	Uplink	Bandwidth	Problems
L	1.5 GHz	1.6 GHz	15 MHz	Low bandwidth; crowded
S	1.9 GHz	2.2 GHz	70 MHz	Low bandwidth; crowded
C	4.0 GHz	6.0 GHz	500 MHz	Terrestrial interference
Ku	11 GHz	14 GHz	500 MHz	Rain
Ka	20 GHz	30 GHz	3500 MHz	Rain, equipment cost

The principal satellite bands.

Communication Satellites (3)
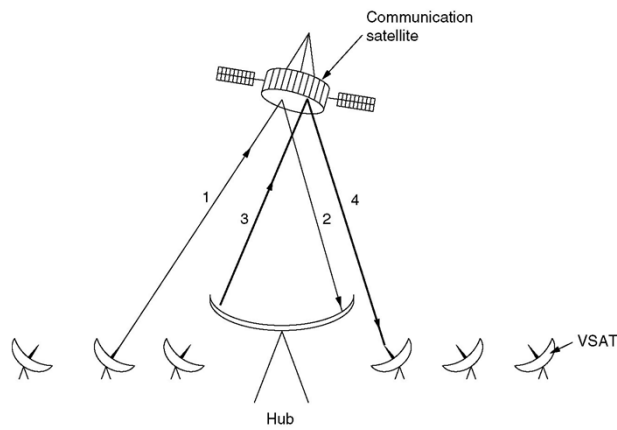
VSATs using a hub.

Low-Earth Orbit Satellites
Iridium
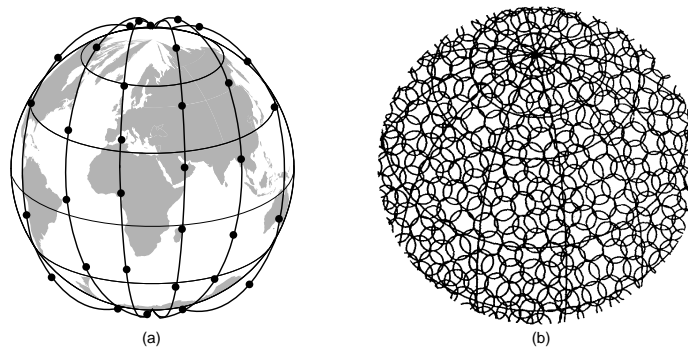
(a) The Iridium satellites from six necklaces around the earth.
(b) 1628 moving cells cover the earth.

Globalstar
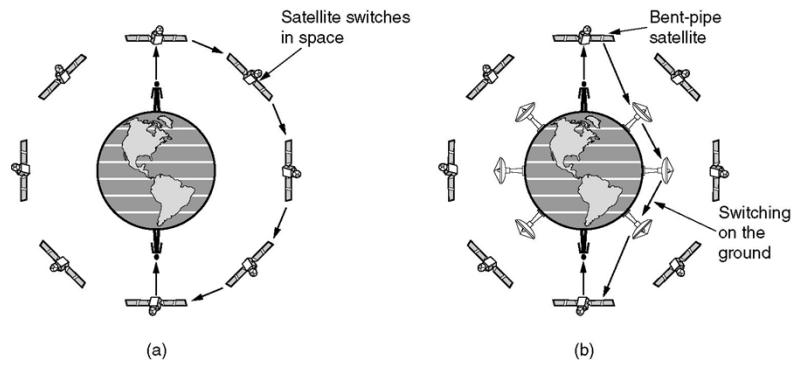
(a) Relaying in space.
(b) Relaying on the ground.

Public Switched Telephone System

- Structure of the Telephone System
- The Politics of Telephones
- The Local Loop: Modems, ADSL and Wireless
- Trunks and Multiplexing
- Switching

Structure of the Telephone System
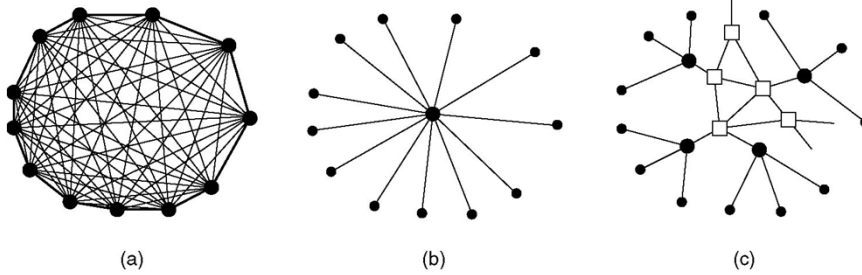
(a) Fully-interconnected network.
(b) Centralized switch.
(c) Two-level hierarchy.

Structure of the Telephone System (2)
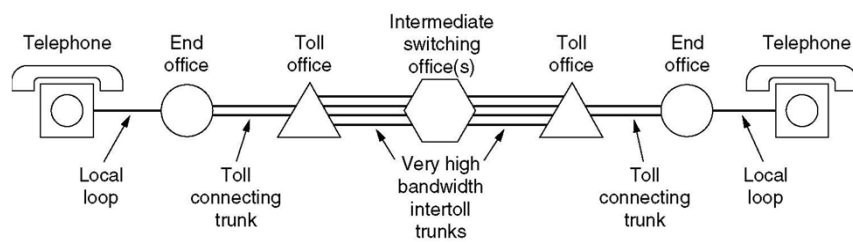
A typical circuit route for a medium-distance call.

Major Components of the Telephone System

- Local loops
  - Analog twisted pairs going to houses and businesses
- Trunks
  - Digital fiber optics connecting the switching offices
- Switching offices
  - Where calls are moved from one trunk to another

The Politics of Telephones
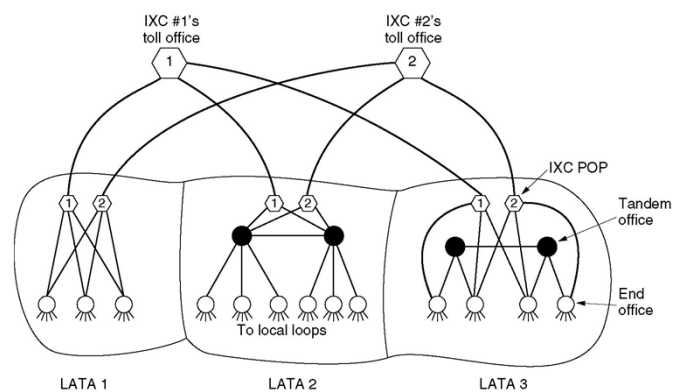
The relationship of LATAs, LECs, and IXCs. All the circles are LEC switching offices. Each hexagon belongs to the IXC whose number is on it.

The Local Loop: Modems, ADSL, and Wireless
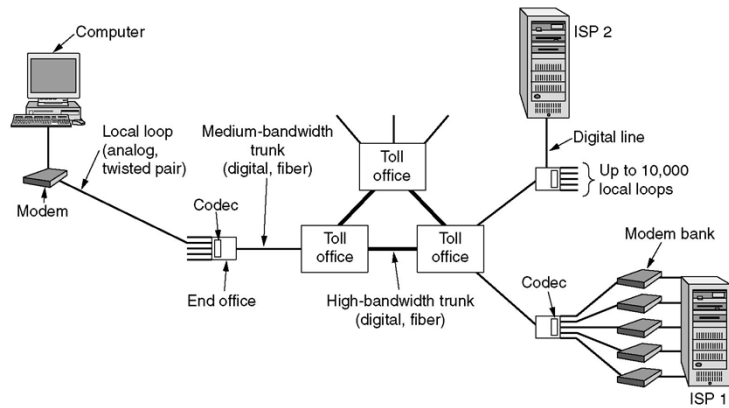
The use of both analog and digital transmissions for a computer to computer call. Conversion is done by the modems and codecs.

Modems
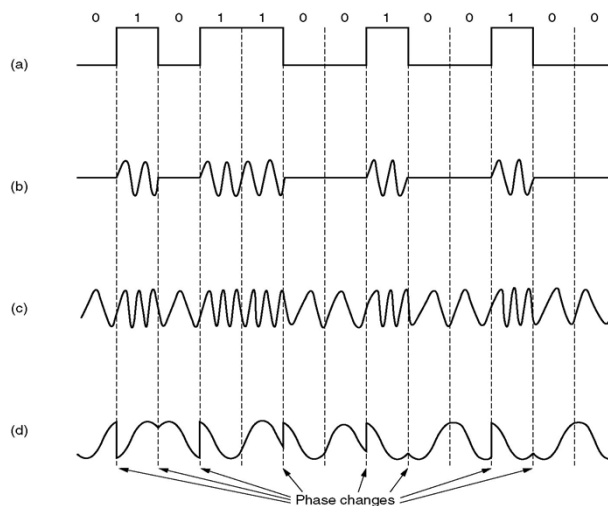
(a) A binary signal

(b) Amplitude modulation

(c) Frequency modulation

(d) Phase modulation

Modems (2)
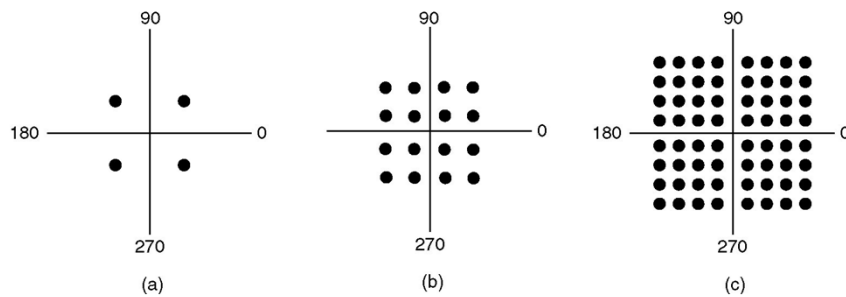
(a) QPSK.
(b) QAM-16.
(c) QAM-64.

Modems (3)
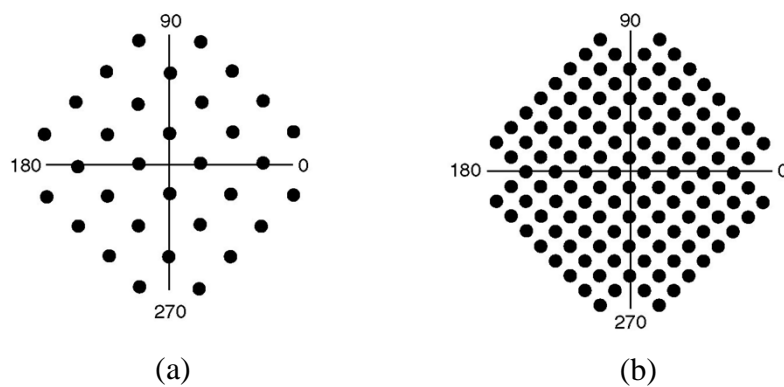
(a) V.32 for 9600 bps.
(b) V32 bis for 14,400 bps.

Digital Subscriber Lines
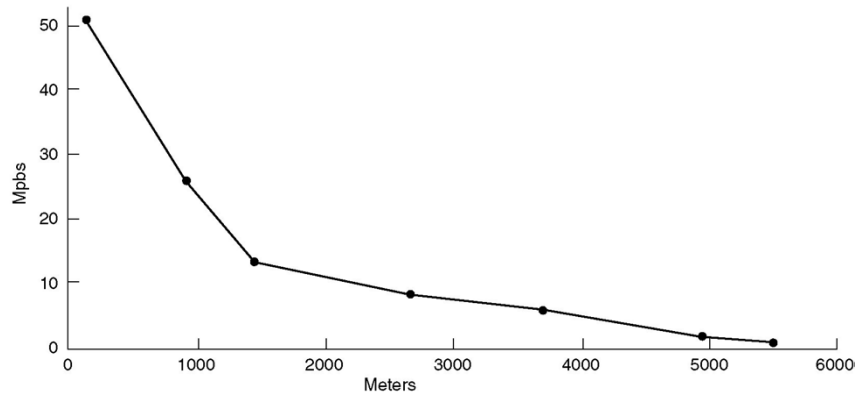
Bandwidth versus distanced over category 3 UTP for DSL.

Digital Subscriber Lines (2)
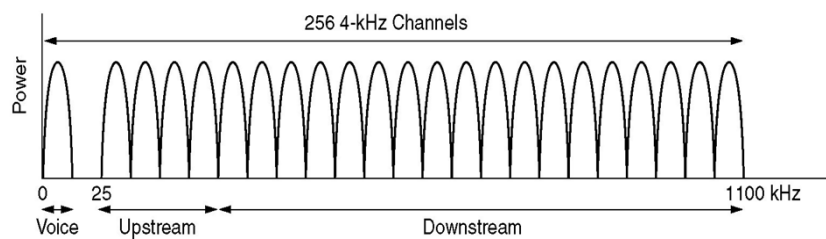
Operation of ADSL using discrete multitone modulation.

Digital Subscriber Lines (3)
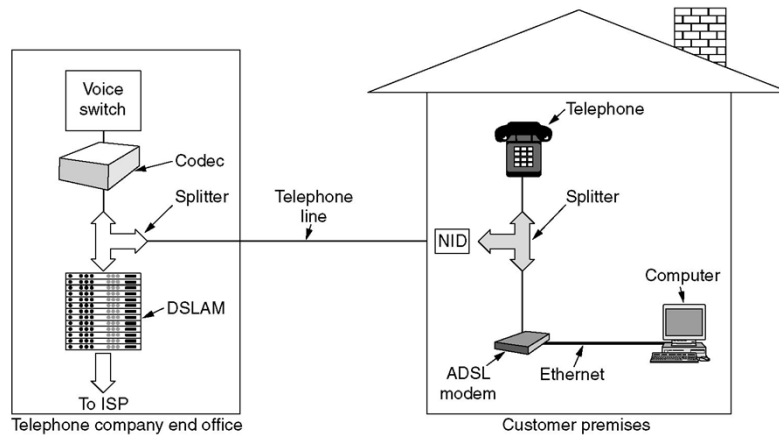
A typical ADSL equipment configuration.

Wireless Local Loops
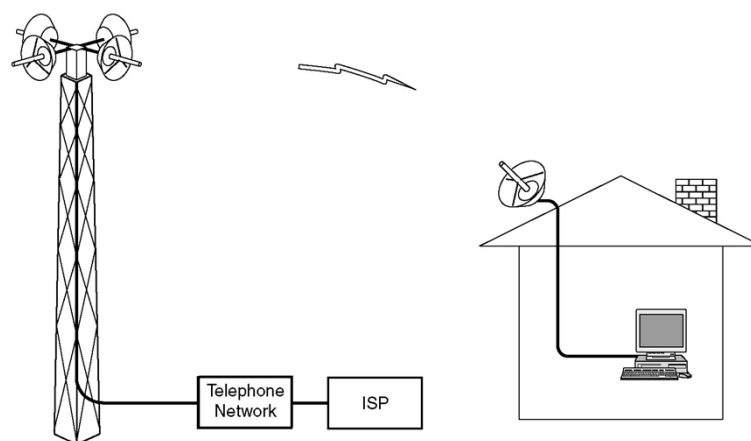
Architecture of an LMDS system.

Frequency Division Multiplexing
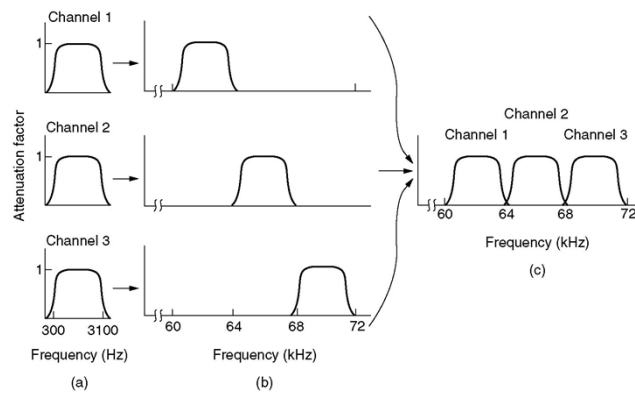
(a) The original bandwidths.
(b) The bandwidths raised in frequency.
(b) The multiplexed channel.

Wavelength Division Multiplexing
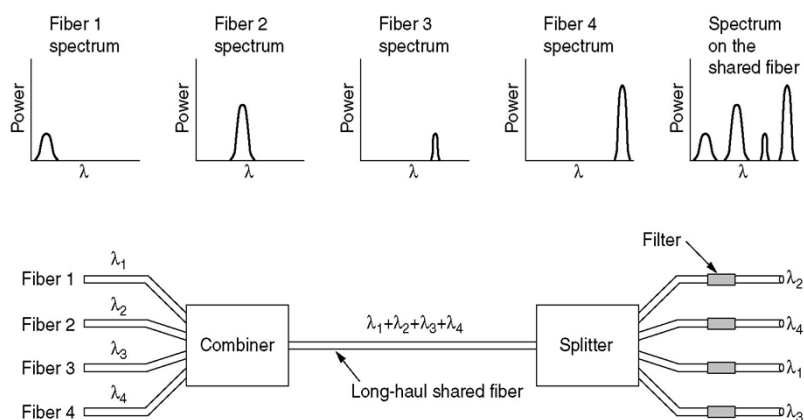
Wavelength division multiplexing.

Time Division Multiplexing
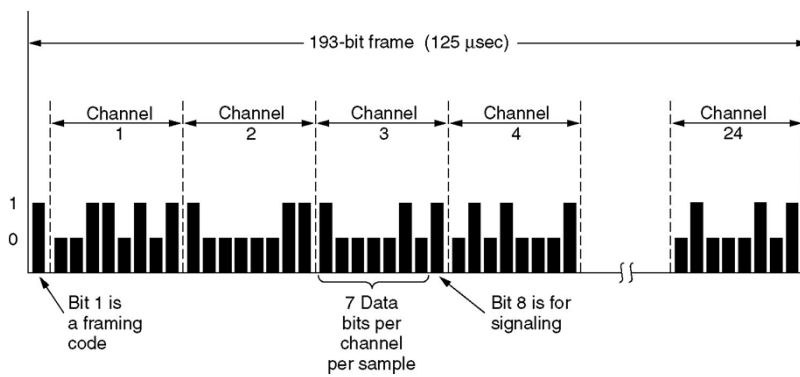
The T1 carrier (1.544 Mbps).

Time Division Multiplexing (2)
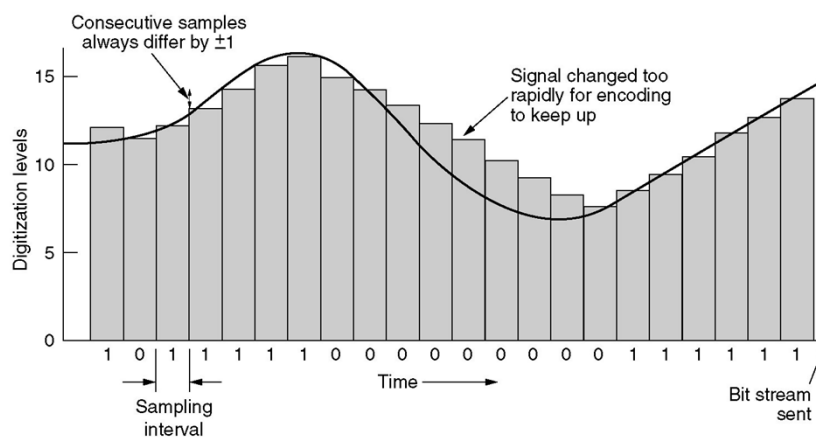
Delta modulation.

Time Division Multiplexing (3)

4 T1 streams in   7 T2 streams in   6 T3 streams in

1 T2 stream out

4 0
5 1   4:1   6 5 4 3 2 1 0   7:1   6:1
6 2
7 3

1.544 Mbps   6.312 Mbps   44.736 Mbps   274.176 Mbps

T1   T2   T3   T4
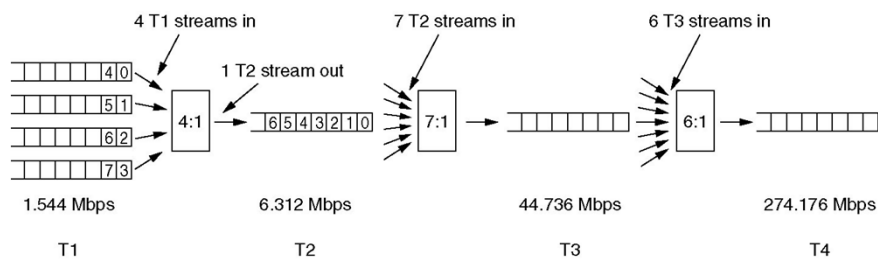
Multiplexing T1 streams into higher carriers.

Time Division Multiplexing (4)

3 Columns
for overhead

87 Columns

9
Rows

Sonet
frame
(125 μsec)

Sonet
frame
(125 μsec)

Section overhead   Line overhead   Path overhead   SPE
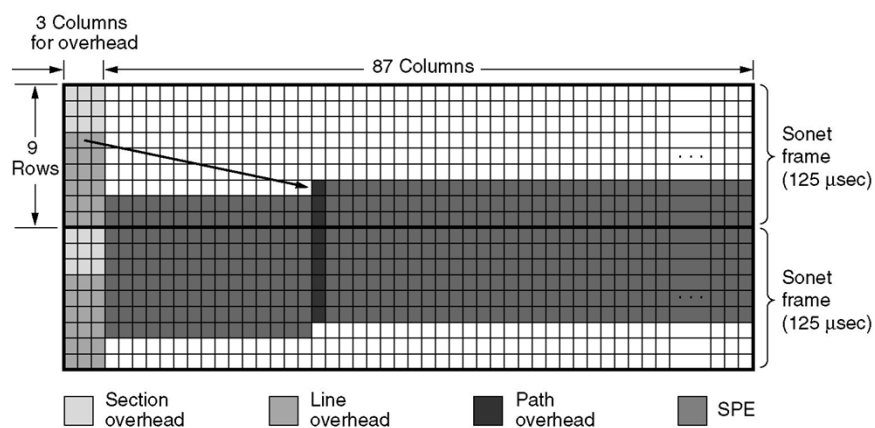
Two back-to-back SONET frames.

Time Division Multiplexing (5)

SONET		SDH	Data rate (Mbps)		
Electrical	Optical	Optical	Gross	SPE	User
STS-1	OC-1		51.84	50.112	49.536
STS-3	OC-3	STM-1	155.52	150.336	148.608
STS-9	OC-9	STM-3	466.56	451.008	445.824
STS-12	OC-12	STM-4	622.08	601.344	594.432
STS-18	OC-18	STM-6	933.12	902.016	891.648
STS-24	OC-24	STM-8	1244.16	1202.688	1188.864
STS-36	OC-36	STM-12	1866.24	1804.032	1783.296
STS-48	OC-48	STM-16	2488.32	2405.376	2377.728
STS-192	OC-192	STM-64	9953.28	9621.504	9510.912

SONET and SDH multiplex rates.

Circuit Switching
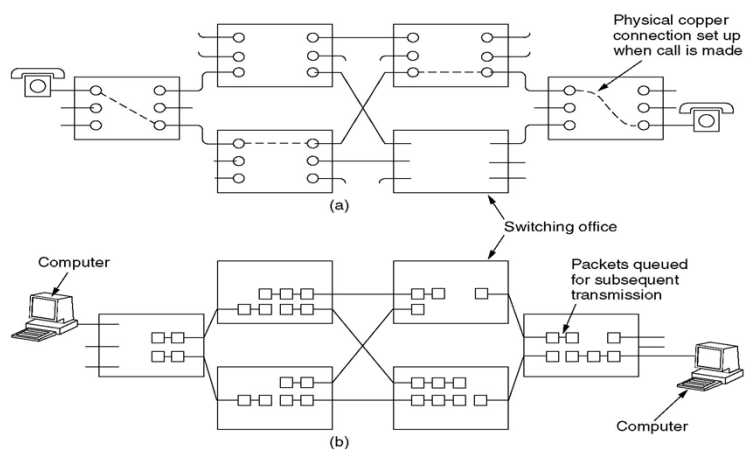
(a) Circuit switching.
(b) Packet switching.

Message Switching
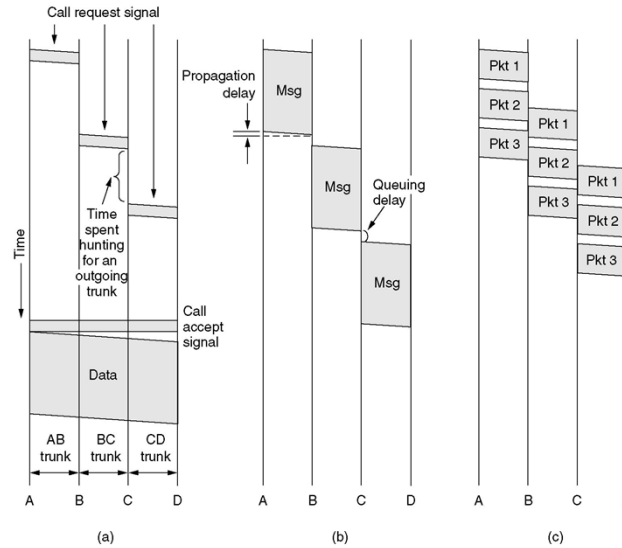
(a) Circuit switching   (b) Message switching   (c) Packet switching

Packet Switching

Item	Circuit-switched	Packet-switched
Call setup	Required	Not needed
Dedicated physical path	Yes	No
Each packet follows the same route	Yes	No
Packets arrive in order	Yes	No
Is a switch crash fatal	Yes	No
Bandwidth available	Fixed	Dynamic
When can congestion occur	At setup time	On every packet
Potentially wasted bandwidth	Yes	No
Store-and-forward transmission	No	Yes
Transparency	Yes	No
Charging	Per minute	Per packet

A comparison of circuit switched and packet-switched networks.

The Mobile Telephone System

- First-Generation Mobile Phones:
  Analog Voice

- Second-Generation Mobile Phones:
  Digital Voice

- Third-Generation Mobile Phones:
  Digital Voice and Data

Advanced Mobile Phone System

(a)   (b)
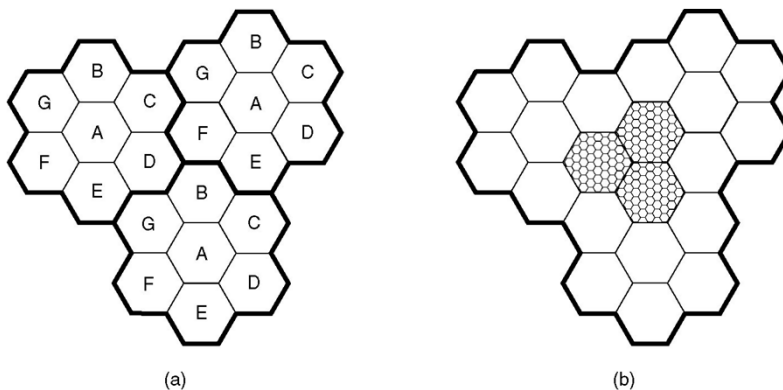
(a) Frequencies are not reused in adjacent cells.
(b) To add more users, smaller cells can be used.

Channel Categories

The 832 channels are divided into four categories:

- Control (base to mobile) to manage the system

- Paging (base to mobile) to alert users to calls for them

- Access (bidirectional) for call setup and channel assignment

- Data (bidirectional) for voice, fax, or data

D-AMPS
Digital Advanced Mobile Phone System
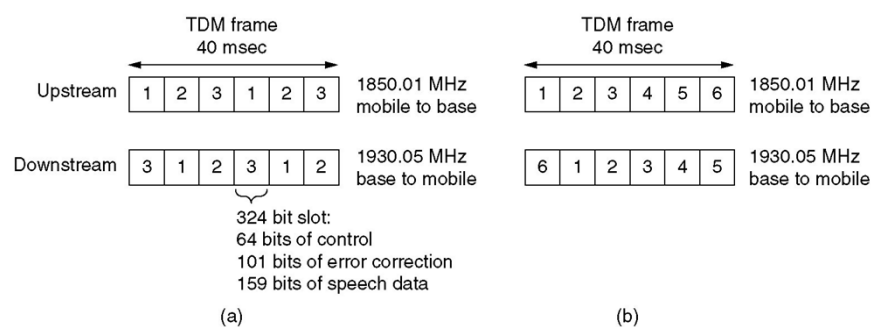
(a) A D-AMPS channel with three users.
(b) A D-AMPS channel with six users.

GSM
Global System for Mobile Communications

GSM uses 124 frequency channels, each of which
uses an eight-slot TDM system

GSM (2)

A portion of the GSM framing structure.
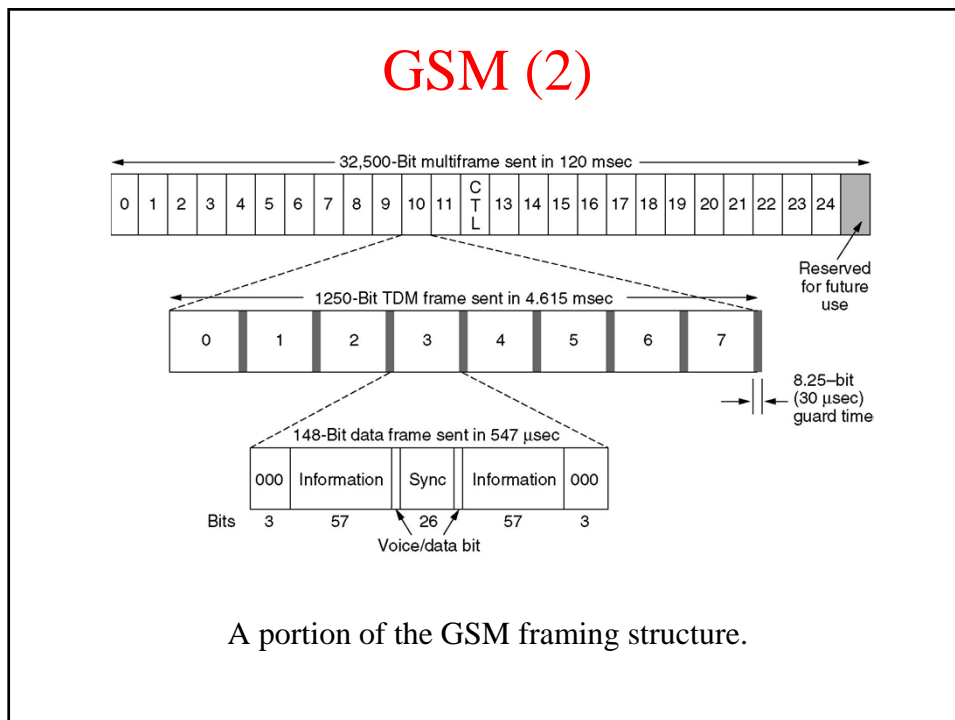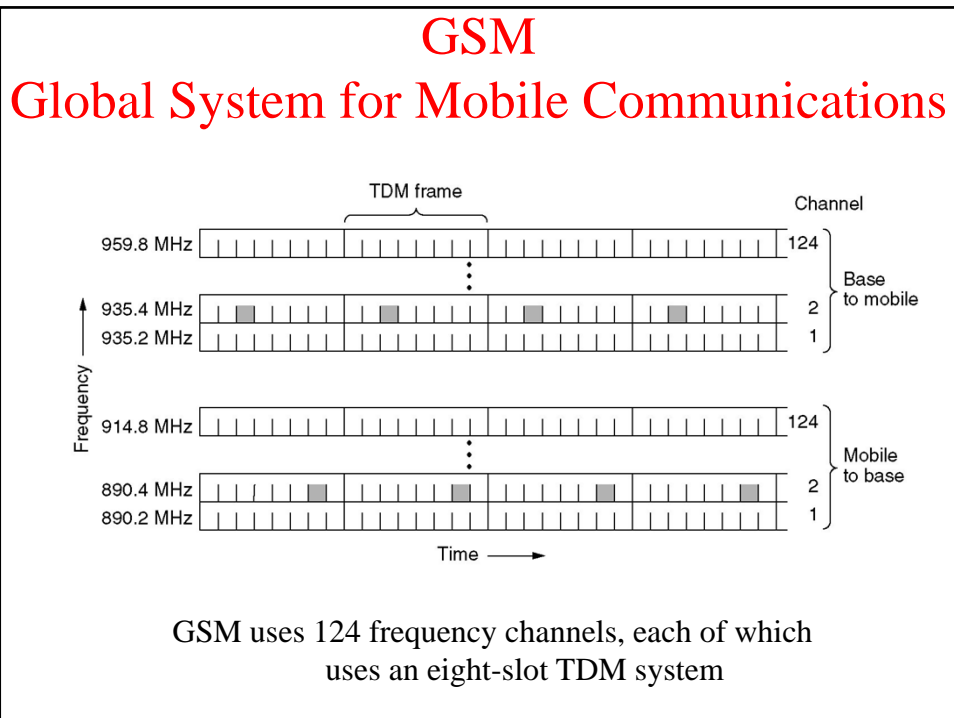
CDMA – Code Division Multiple Access

```
A: 0 0 0 1 1 0 1 1        A: (−1 −1 −1 +1 +1 −1 +1 +1)
B: 0 0 1 0 1 1 1 0        B: (−1 −1 +1 −1 +1 +1 +1 −1)
C: 0 1 0 1 1 1 0 0        C: (−1 +1 −1 +1 +1 +1 −1 −1)
D: 0 1 0 0 0 0 1 0        D: (−1 +1 −1 −1 −1 −1 +1 −1)
        (a)                               (b)
```

Six examples:

```
− − 1 −     C           S₁ = (−1 +1 −1 +1 +1 +1 −1 −1)
− 1 1 −     B + C       S₂ = (−2  0  0  0 +2 +2  0 −2)
1 0 − −     A + B       S₃ = ( 0  0 −2 +2  0 −2  0 +2)
1 0 1 −     A + B + C   S₄ = (−1 +1 −3 +3 +1 −1 −1 +1)
1 1 1 1     A + B + C + D   S₅ = (−4  0 −2  0 +2  0 +2 −2)
1 1 0 1     A + B + C̄ + D   S₆ = (−2 −2  0 −2  0 −2 +4  0)
                    (c)
```

$S_1 \bullet C = (1 +1 +1 +1 +1 +1 +1 +1)/8 = 1$
$S_2 \bullet C = (2 +0 +0 +0 +2 +2 +0 +2)/8 = 1$
$S_3 \bullet C = (0 +0 +2 +2 +0 −2 +0 −2)/8 = 0$
$S_4 \bullet C = (1 +1 +3 +3 +1 −1 +1 −1)/8 = 1$
$S_5 \bullet C = (4 +0 +2 +0 +2 +0 −2 +2)/8 = 1$
$S_6 \bullet C = (2 −2 +0 −2 +0 −2 −4 +0)/8 = −1$
(d)

(a) Binary chip sequences for four stations
(b) Bipolar chip sequences
(c) Six examples of transmissions
(d) Recovery of station C's signal

Third-Generation Mobile Phones: Digital Voice and Data

Basic services an IMT-2000 network should provide

- High-quality voice transmission
- Messaging (replace e-mail, fax, SMS, chat, etc.)
- Multimedia (music, videos, films, TV, etc.)
- Internet access (web surfing, w/multimedia.)

Cable Television

- Community Antenna Television
- Internet over Cable
- Spectrum Allocation
- Cable Modems
- ADSL versus Cable

Community Antenna Television
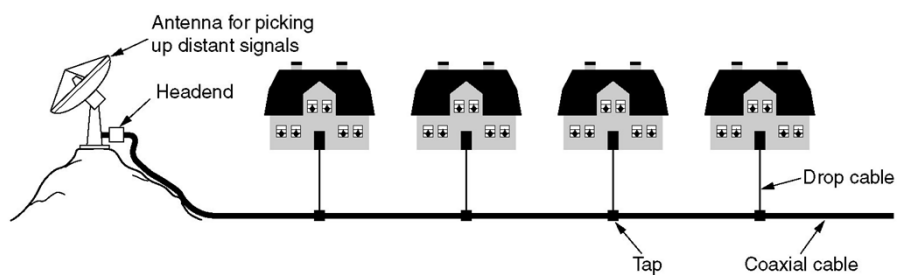
An early cable television system.

Internet over Cable
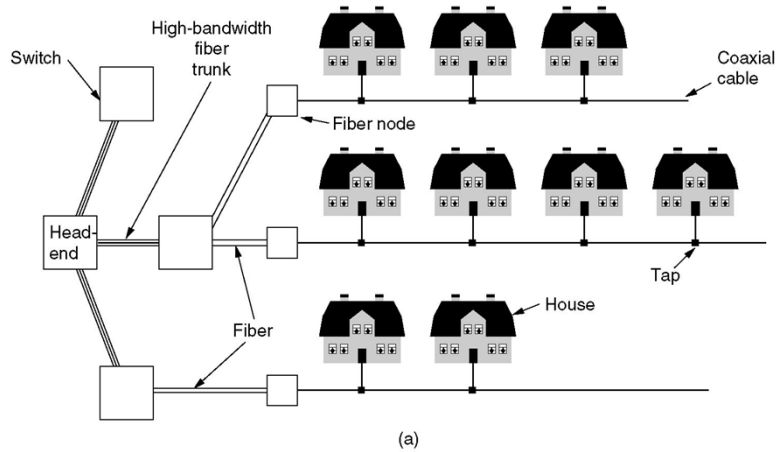
Cable television

Internet over Cable (2)
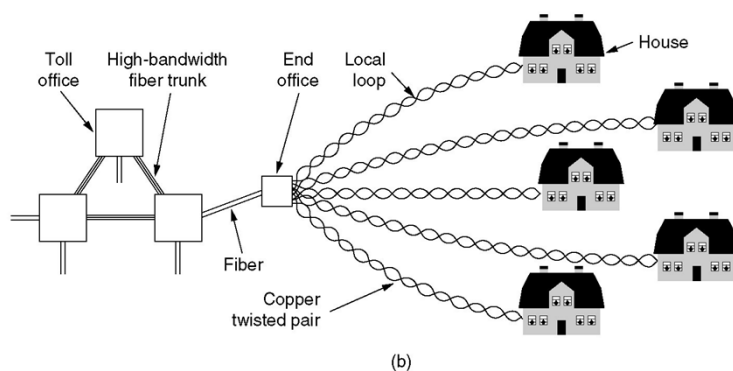
The fixed telephone system.

Spectrum Allocation
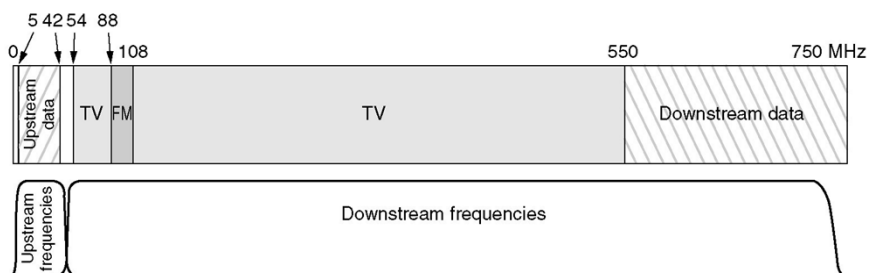
Frequency allocation in a typical cable TV system
used for Internet access

Cable Modems
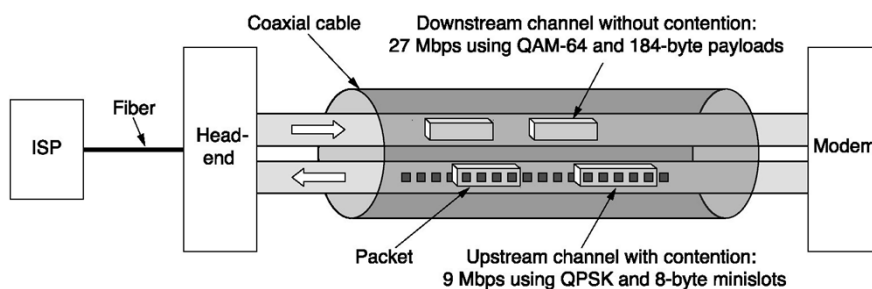
Typical details of the upstream and downstream
channels in North America.

Chapter 3

The Data Link Layer

Data Link Layer Design Issues

- Services Provided to the Network Layer
- Framing
- Error Control
- Flow Control

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)

	Sending machine			Receiving machine		
	Packet			Packet		

Frame

| Header | Payload field | Trailer | | Header | Payload field | Trailer |
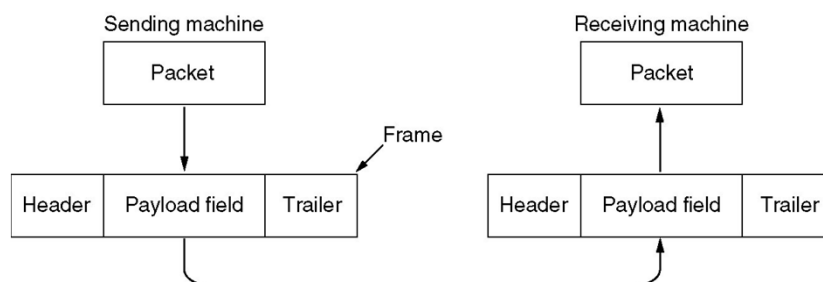
Relationship between packets and frames.

Services Provided to Network Layer
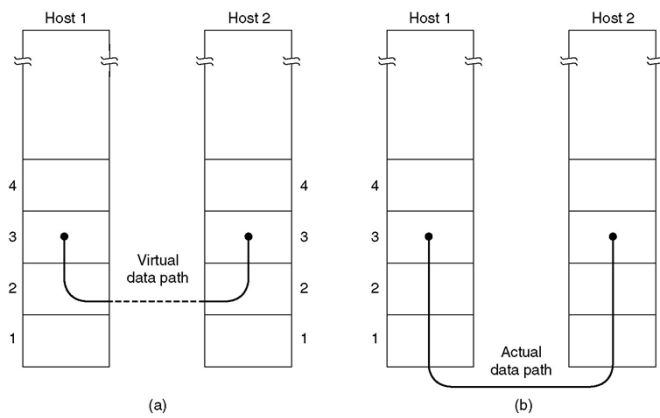
(a) Virtual communication.
(b) Actual communication.

Services Provided to Network Layer (2)
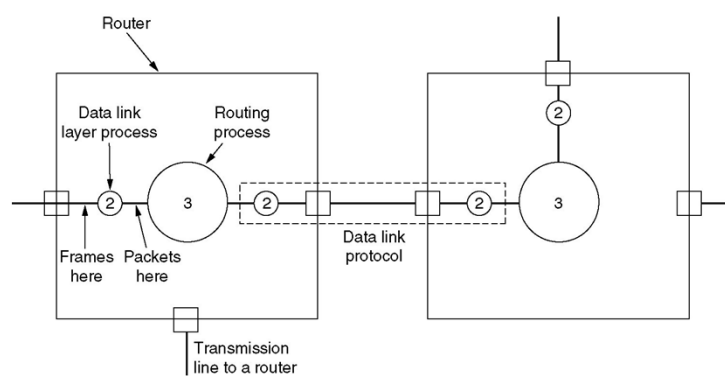
Placement of the data link protocol.

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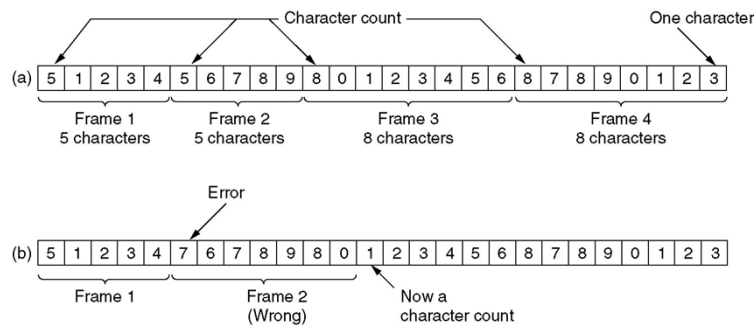
A character stream.   (a) Without errors.   (b) With one error.

Framing (2)
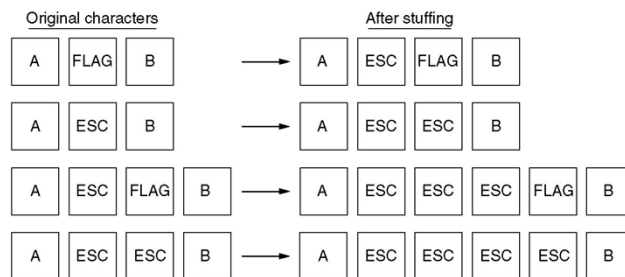
(a) A frame delimited by flag bytes.
(b) Four examples of byte sequences before and after stuffing.

Framing (3)

(a) 0 1 1 0 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 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-Correcting Codes

Char.	ASCII	Check bits
H	1001000	00110010000
a	1100001	10111001001
m	1101101	11101010101
m	1101101	11101010101
i	1101001	01101011001
n	1101110	01101010110
g	1100111	01111001111
	0100000	10011000000
c	1100011	11111000011
o	1101111	10101011111
d	1100100	11111001100
e	1100101	00111000101

Order of bit transmission

Use of a Hamming code to correct burst errors.

Error-Detecting Codes

```
Frame    : 1 1 0 1 0 1 1 0 1 1
Generator: 1 0 0 1 1
Message after 4 zero bits are appended:  1 1 0 1 0 1 1 0 1 1 0 0 0 0

                            1 1 0 0 0 0 1 0 1 0
              1 0 0 1 1 ) 1 1 0 1 0 1 1 0 1 1 0 0 0 0
                          1 0 0 1 1
                            1 0 0 1 1
                            1 0 0 1 1
                              0 0 0 0 1
                              0 0 0 0 0
                                0 0 0 1 0
                                0 0 0 0 0
                                  0 0 1 0 1
                                  0 0 0 0 0
                                    0 1 0 1 1
                                    0 0 0 0 0
                                      1 0 1 1 0
                                      1 0 0 1 1
                                        0 1 0 1 0
                                        0 0 0 0 0
                                          1 0 1 0 0
                                          1 0 0 1 1
                                            0 1 1 1 0
                                            0 0 0 0 0  Remainder
                                              1 1 1 0

Transmitted frame:  1 1 0 1 0 1 1 0 1 1 1 1 1 0
```
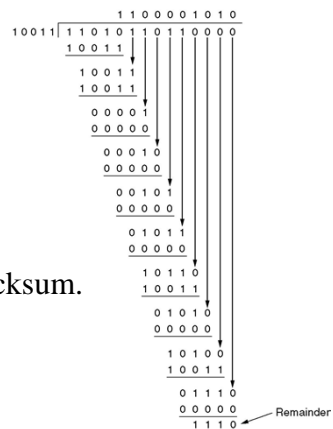
Calculation of the polynomial code checksum.

6

Elementary Data Link Protocols

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

Protocol Definitions

```
#define MAX_PKT 1024                               /* determines packet size in bytes */

typedef enum {false, true} boolean;                /* boolean type */
typedef unsigned int seq_nr;                       /* sequence or ack numbers */
typedef struct {unsigned char data[MAX_PKT];} packet;/*    packet definition */
typedef enum {data, ack, nak} frame_kind;          /* frame_kind definition */

typedef struct {                                   /* frames are transported in this layer */
  frame_kind kind;                                 /* what kind of a frame is it? */
  seq_nr seq;                                      /* sequence number */
  seq_nr ack;                                      /* acknowledgement number */
  packet info;                                     /* the network layer packet */
} frame;
```

Continued →

Some definitions needed in the protocols to follow.
These are located in the file protocol.h.

Protocol Definitions (ctd.)

Some definitions needed in the protocols to follow. These are located in the file protocol.h.

```
/* Wait for an event to happen; return its type in event. */
void wait_for_event(event_type *event);

/* Fetch a packet from the network layer for transmission on the channel. */
void from_network_layer(packet *p);

/* Deliver information from an inbound frame to the network layer. */
void to_network_layer(packet *p);

/* Go get an inbound frame from the physical layer and copy it to r. */
void from_physical_layer(frame *r);

/* Pass the frame to the physical layer for transmission. */
void to_physical_layer(frame *s);

/* Start the clock running and enable the timeout event. */
void start_timer(seq_nr k);

/* Stop the clock and disable the timeout event. */
void stop_timer(seq_nr k);

/* Start an auxiliary timer and enable the ack_timeout event. */
void start_ack_timer(void);

/* Stop the auxiliary timer and disable the ack_timeout event. */
void stop_ack_timer(void);

/* Allow the network layer to cause a network_layer_ready event. */
void enable_network_layer(void);

/* Forbid the network layer from causing a network_layer_ready event. */
void disable_network_layer(void);

/* Macro inc is expanded in-line: Increment k circularly. */
#define inc(k) if (k < MAX_SEQ) k = k + 1; else k = 0
```

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 */
     }
 }
}
```
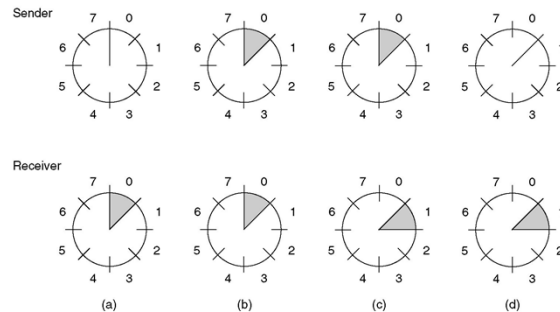
A positive acknowledgement with retransmission protocol.

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

```
/* Protocol 4 (sliding window) is bidirectional. */
#define MAX_SEQ 1                        /* must be 1 for protocol 4 */
typedef enum {frame_arrival, cksum_err, timeout} event_type;
#include "protocol.h"

void protocol4 (void)
{
  seq_nr next_frame_to_send;            /* 0 or 1 only */
  seq_nr frame_expected;                /* 0 or 1 only */
  frame r, s;                           /* scratch variables */
  packet buffer;                        /* current packet being sent */
  event_type event;

  next_frame_to_send = 0;               /* next frame on the outbound stream */
  frame_expected = 0;                   /* frame expected next */
  from_network_layer(&buffer);          /* fetch a packet from the network layer */
  s.info = buffer;                      /* prepare to send the initial frame */
  s.seq = next_frame_to_send;           /* insert sequence number into frame */
  s.ack = 1 − frame_expected;           /* piggybacked ack */
  to_physical_layer(&s);                /* transmit the frame */
  start_timer(s.seq);                   /* start the timer running */
```

Continued →

A One-Bit Sliding Window Protocol (ctd.)

```
while (true) {
    wait_for_event(&event);              /* frame_arrival, cksum_err, or timeout */
    if (event == frame_arrival) {        /* a frame has arrived undamaged. */
        from_physical_layer(&r);         /* go get it */

        if (r.seq == frame_expected) {   /* handle inbound frame stream. */
            to_network_layer(&r.info);   /* pass packet to network layer */
            inc(frame_expected);         /* invert seq number expected next */
        }

        if (r.ack == next_frame_to_send) { /* handle outbound frame stream. */
            stop_timer(r.ack);           /* turn the timer off */
            from_network_layer(&buffer); /* fetch new pkt from network layer */
            inc(next_frame_to_send);     /* invert sender's sequence number */
        }
    }
    s.info = buffer;                     /* construct outbound frame */
    s.seq = next_frame_to_send;          /* insert sequence number into it */
    s.ack = 1 – frame_expected;          /* seq number of last received frame */
    to_physical_layer(&s);               /* transmit a frame */
    start_timer(s.seq);                  /* start the timer running */
}
}
```
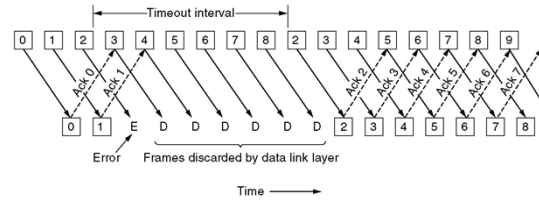
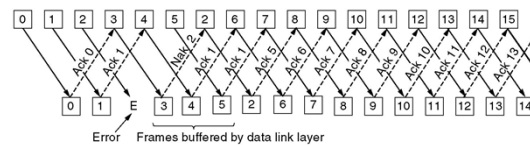# A One-Bit Sliding Window Protocol (2)
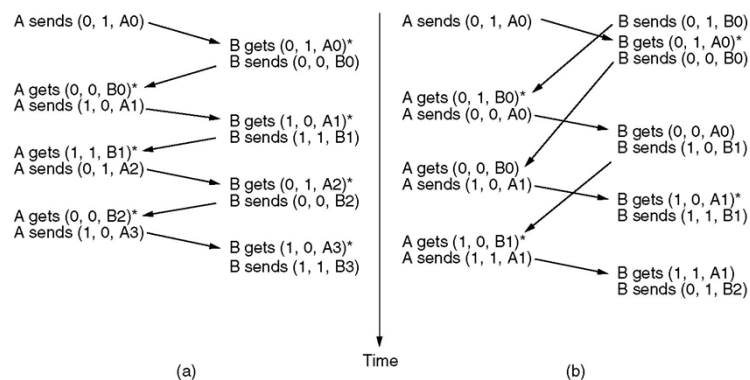
Two scenarios for protocol 4. (a) Normal case. (b) Abnormal case. The notation is (seq, ack, packet number). An asterisk indicates where a network layer accepts a packet.

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.

Sliding Window Protocol Using Go Back N

```
/* Protocol 5 (pipelining) allows multiple outstanding frames. The sender may transmit up
   to MAX_SEQ frames without waiting for an ack. In addition, unlike the previous protocols
   the network layer is not assumed to have a new packet all the time. Instead, the
   network layer causes a network_layer_ready event when there is a packet to send. */

#define MAX_SEQ 7                       /* should be 2^n – 1 */
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready} event_type;
#include "protocol.h"

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
/* Return true if a <=b < c circularly; false otherwise. */
  if (((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a)))
       return(true);
    else
       return(false);
}

static void send_data(seq_nr frame_nr, seq_nr frame_expected, packet buffer[ ])
{
/* Construct and send a data frame. */
  frame s;                              /* scratch variable */

  s.info = buffer[frame_nr];            /* insert packet into frame */
  s.seq = frame_nr;                     /* insert sequence number into frame */
  s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);/* piggyback ack */
  to_physical_layer(&s);                /* transmit the frame */
  start_timer(frame_nr);                /* start the timer running */
}
```

Sliding Window Protocol Using Go Back N

```
void protocol5(void)
{
  seq_nr next_frame_to_send;        /* MAX_SEQ > 1; used for outbound stream */
  seq_nr ack_expected;              /* oldest frame as yet unacknowledged */
  seq_nr frame_expected;            /* next frame expected on inbound stream */
  frame r;                          /* scratch variable */
  packet buffer[MAX_SEQ + 1];       /* buffers for the outbound stream */
  seq_nr nbuffered;                 /* # output buffers currently in use */
  seq_nr i;                         /* used to index into the buffer array */
  event_type event;

  enable_network_layer();           /* allow network_layer_ready events */
  ack_expected = 0;                 /* next ack expected inbound */
  next_frame_to_send = 0;           /* next frame going out */
  frame_expected = 0;               /* number of frame expected inbound */
  nbuffered = 0;                    /* initially no packets are buffered */
```

Continued →

Sliding Window Protocol Using Go Back N

```
while (true) {
  wait_for_event(&event);           /* four possibilities: see event_type above */

  switch(event) {
    case network_layer_ready:       /* the network layer has a packet to send */
        /* Accept, save, and transmit a new frame. */
        from_network_layer(&buffer[next_frame_to_send]); /* fetch new packet */
        nbuffered = nbuffered + 1;  /* expand the sender's window */
        send_data(next_frame_to_send, frame_expected, buffer);/* transmit the frame */
        inc(next_frame_to_send);    /* advance sender's upper window edge */
        break;

    case frame_arrival:             /* a data or control frame has arrived */
        from_physical_layer(&r);    /* get incoming frame from physical layer */

        if (r.seq == frame_expected) {
            /* Frames are accepted only in order. */
            to_network_layer(&r.info);  /* pass packet to network layer */
            inc(frame_expected);    /* advance lower edge of receiver's window */
        }
```

Continued →

14

Sliding Window Protocol Using Go Back N

```
            /* Ack n implies n – 1, n – 2, etc.  Check for this. */
            while (between(ack_expected, r.ack, next_frame_to_send)) {
                    /* Handle piggybacked ack. */
                    nbuffered = nbuffered    1; /* one frame fewer buffered */
                    stop_timer(ack_expected); /* frame arrived intact; stop timer */
                    inc(ack_expected);      /* contract sender's window */
            }
            break;

       case cksum_err: break;              /* just ignore bad frames */

       case timeout:                       /* trouble; retransmit all outstanding frames */
            next_frame_to_send = ack_expected;    /* start retransmitting here */
            for (i = 1; i <= nbuffered; i++) {
                    send_data(next_frame_to_send, frame_expected, buffer);/* resend 1 frame */
                    inc(next_frame_to_send);  /* prepare to send the next one */
            }

     }

     if (nbuffered < MAX_SEQ)
            enable_network_layer();
     else
            disable_network_layer();
 }
}
```
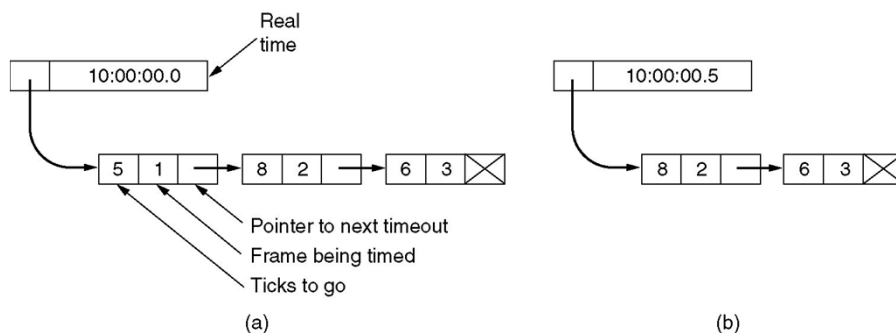
Sliding Window Protocol Using Go Back N (2)

Simulation of multiple timers in software.

A Sliding Window Protocol Using Selective Repeat

```
/* Protocol 6 (nonsequential receive) accepts frames out of order, but passes packets to the
   network layer in order. Associated with each outstanding frame is a timer. When the timer
   expires, only that frame is retransmitted, not all the outstanding frames, as in protocol 5. */

#define MAX_SEQ 7                           /* should be 2^n – 1 */
#define NR_BUFS ((MAX_SEQ + 1)/2)
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready, ack_timeout} event_type;
#include "protocol.h"
boolean no_nak = true;                      /* no nak has been sent yet */
seq_nr oldest_frame = MAX-SEQ + 1;          /* initial value is only for the simulator */

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{
/* Same as between in protocol5, but shorter and more obscure. */
  return ((a <= b) && (b < c)) || ((c < a) && (a <= b)) || ((b < c) && (c < a));
}

static void send_frame(frame_kind fk, seq_nr frame_nr, seq_nr frame_expected, packet buffer[])
{
/* Construct and send a data, ack, or nak frame. */
  frame s;                                  /* scratch variable */

  s.kind = fk;                              /* kind == data, ack, or nak */
  if (fk == data) s.info = buffer[frame_nr % NR_BUFS];
  s.seq = frame_nr;                         /* only meaningful for data frames */
  s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
  if (fk == nak) no_nak = false;            /* one nak per frame, please */
  to_physical_layer(&s);                    /* transmit the frame */
  if (fk == data) start_timer(frame_nr % NR_BUFS);
  stop_ack_timer();                         /* no need for separate ack frame */
}
```

Continued →

A Sliding Window Protocol Using Selective Repeat (2)

```
void protocol6(void)
{
  seq_nr ack_expected;                      /* lower edge of sender's window */
  seq_nr next_frame_to_send;                /* upper edge of sender's window + 1 */
  seq_nr frame_expected;                    /* lower edge of receiver's window */
  seq_nr too_far;                           /* upper edge of receiver's window + 1 */
  int i;                                    /* index into buffer pool */
  frame r;                                  /* scratch variable */
  packet out_buf[NR_BUFS];                  /* buffers for the outbound stream */
  packet in_buf[NR_BUFS];                   /* buffers for the inbound stream */
  boolean arrived[NR_BUFS];                 /* inbound bit map */
  seq_nr nbuffered;                         /* how many output buffers currently used */
  event_type event;

  enable_network_layer();                   /* initialize */
  ack_expected = 0;                         /* next ack expected on the inbound stream */
  next_frame_to_send = 0;                   /* number of next outgoing frame */
  frame_expected = 0;
  too_far = NR_BUFS;
  nbuffered = 0;                            /* initially no packets are buffered */
  for (i = 0; i < NR_BUFS; i++) arrived[i] = false;
```

Continued →

A Sliding Window Protocol Using Selective Repeat (3)

```
while (true) {
    wait_for_event(&event);                    /* five possibilities: see event_type above */
    switch(event) {
      case network_layer_ready:                /* accept, save, and transmit a new frame */
          nbuffered = nbuffered + 1;           /* expand the window */
          from_network_layer(&out buf[next_frame_to_send % NR_BUFS]); /* fetch new packet */
          send_frame(data, next_frame_to_send, frame_expected, out_buf);/* transmit the frame */
          inc(next_frame_to_send);             /* advance upper window edge */
          break;

      case frame_arrival:                      /* a data or control frame has arrived */
          from_physical_layer(&r);             /* fetch incoming frame from physical layer */
          if (r.kind == data) {
              /* An undamaged frame has arrived. */
              if ((r.seq != frame_expected) && no_nak)
                send_frame(nak, 0, frame_expected, out_buf); else start_ack_timer();
              if (between(frame_expected, r.seq, too_far) && (arrived[r.seq%NR_BUFS] == false)) {
                  /* Frames may be accepted in any order. */
                  arrived[r.seq % NR_BUFS] = true;     /* mark buffer as full */
                  in_buf[r.seq % NR_BUFS] = r.info;    /* insert data into buffer */
                  while (arrived[frame_expected % NR_BUFS]) {
                      /* Pass frames and advance window. */
                      to_network_layer(&in_buf[frame_expected % NR_BUFS]);
                      no_nak = true;
                      arrived[frame_expected % NR_BUFS] = false;
                      inc(frame_expected);   /* advance lower edge of receiver's window */
                      inc(too_far);          /* advance upper edge of receiver's window */
                      start_ack_timer();     /* to see if a separate ack is needed */
                  }
              }
          }
    }
```

Continued →

A Sliding Window Protocol Using Selective Repeat (4)

```
          if((r.kind==nak) && between(ack_expected,(r.ack+1)%(MAX_SEQ+1),next frame to send))
              send_frame(data, (r.ack+1) % (MAX_SEQ + 1), frame_expected, out_buf);

          while (between(ack_expected, r.ack, next_frame_to_send)) {
              nbuffered = nbuffered   1;        /* handle piggybacked ack */
              stop_timer(ack_expected % NR_BUFS);   /* frame arrived intact */
              inc(ack_expected);                /* advance lower edge of sender's window */
          }
          break;
      case cksum_err:
          if (no_nak) send_frame(nak, 0, frame_expected, out_buf);/* damaged frame */
          break;
      case timeout:
          send_frame(data, oldest_frame, frame_expected, out_buf);/* we timed out */
          break;
      case ack_timeout:
          send_frame(ack,0,frame_expected, out_buf);    /* ack timer expired; send ack */
  }
  if (nbuffered < NR_BUFS) enable_network_layer(); else disable_network_layer();
  }
}
```
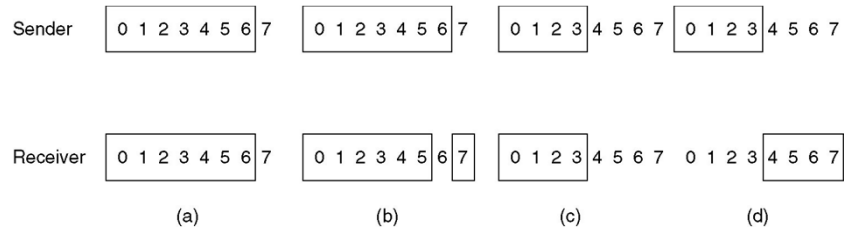
A Sliding Window Protocol Using Selective Repeat (5)

Sender:
- (a) `0 1 2 3 4 5 6 | 7`
- (b) `0 1 2 3 4 5 6 | 7`
- (c) `0 1 2 3 | 4 5 6 7`
- (d) `0 1 2 3 | 4 5 6 7`

Receiver:
- (a) `0 1 2 3 4 5 6 | 7`
- (b) `0 1 2 3 4 5 | 6 | 7`
- (c) `0 1 2 3 | 4 5 6 7`
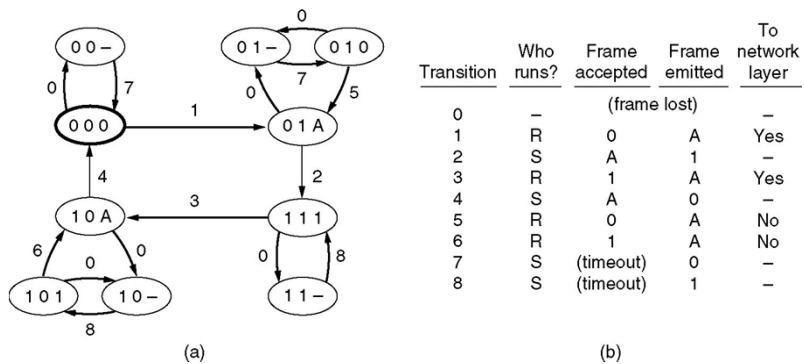- (d) `0 1 2 3 | 4 5 6 7`

(a)   (b)   (c)   (d)

(a) Initial situation with a window size seven.
(b) After seven frames sent and received, but not acknowledged.
(c) Initial situation with a window size of four.
(d) After four frames sent and received, but not acknowledged.

Protocol Verification

- Finite State Machined Models
- Petri Net Models

Finite State Machined Models

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	–

(a)   (b)

(a) State diagram for protocol 3. (b) Transmissions.

Petri Net Models
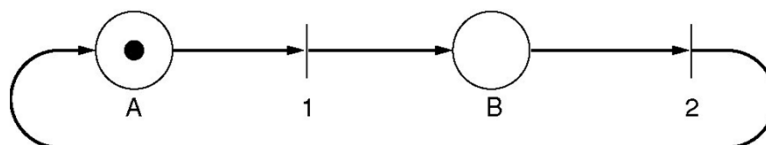
A Petri net with two places and two transitions.
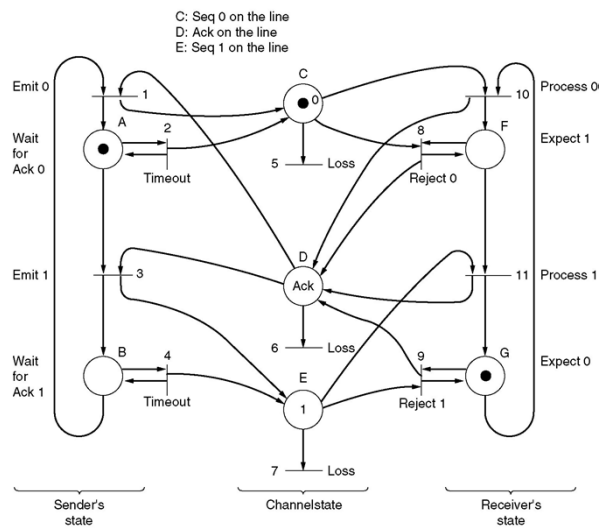
A Petri net model for protocol 3.

Example Data Link Protocols

- HDLC – High-Level Data Link Control
- The Data Link Layer in the Internet

High-Level Data Link Control

Bits	8	8	8	≥ 0	16	8
	0 1 1 1 1 1 1 0	Address	Control	Data	Checksum	0 1 1 1 1 1 1 0
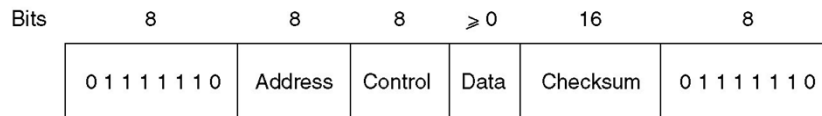
Frame format for bit-oriented protocols.

High-Level Data Link Control (2)

Bits	1		3	1	3
(a)	0		Seq	P/F	Next

(b)	1	0	Type	P/F	Next

(c)	1	1	Type	P/F	Modifier
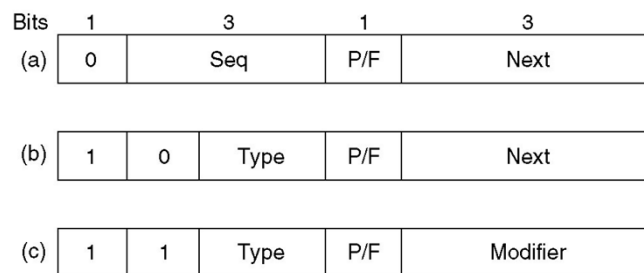
Control field of
(a) An information frame.
(b) A supervisory frame.
(c) An unnumbered frame.

The Data Link Layer in the Internet
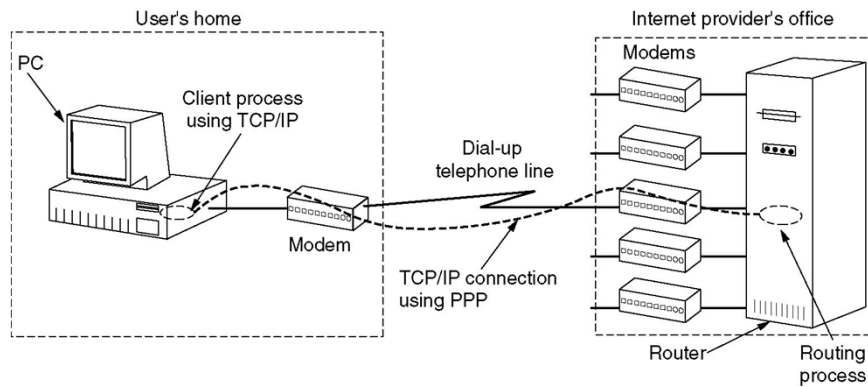
A home personal computer acting as an internet host.

PPP – Point to Point Protocol

Bytes	1	1	1	1 or 2	Variable	2 or 4	1
	Flag 01111110	Address 11111111	Control 00000011	Protocol	Payload	Checksum	Flag 01111110

The PPP full frame format for unnumbered mode operation.

PPP – Point to Point Protocol (2)
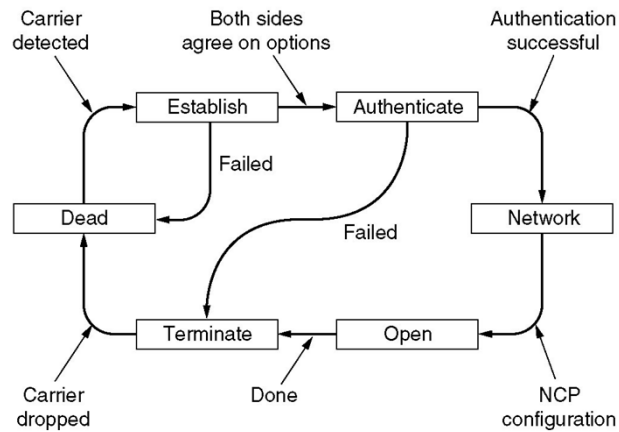
A simplified phase diagram for bring a line up and down.

PPP – Point to Point Protocol (3)

Name	Direction	Description
Configure-request	I → R	List of proposed options and values
Configure-ack	I ← R	All options are accepted
Configure-nak	I ← R	Some options are not accepted
Configure-reject	I ← R	Some options are not negotiable
Terminate-request	I → R	Request to shut the line down
Terminate-ack	I ← R	OK, line shut down
Code-reject	I ← R	Unknown request received
Protocol-reject	I ← R	Unknown protocol requested
Echo-request	I → R	Please send this frame back
Echo-reply	I ← R	Here is the frame back
Discard-request	I → R	Just discard this frame (for testing)

The LCP frame types.

Chapter 4

The Medium Access Control Sublayer

The Channel Allocation Problem

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

Dynamic Channel Allocation in LANs and MANs

1. Station Model.

2. Single Channel Assumption.

3. Collision Assumption.

4. (a) Continuous Time.
   (b) Slotted Time.

5. (a) Carrier Sense.
   (b) No Carrier Sense.

Multiple Access Protocols

- ALOHA
- Carrier Sense Multiple Access Protocols
- Collision-Free Protocols
- Limited-Contention Protocols
- Wavelength Division Multiple Access Protocols
- Wireless LAN Protocols

Pure ALOHA
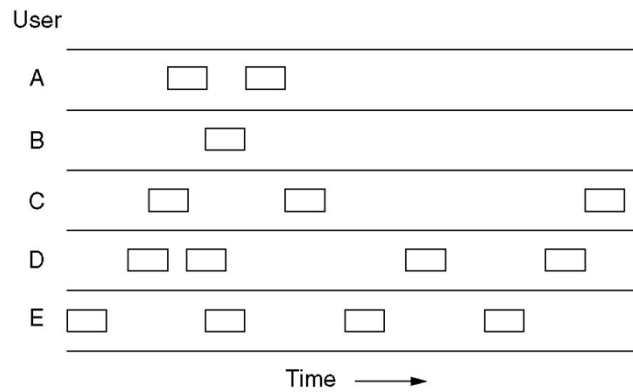
In pure ALOHA, frames are transmitted at completely arbitrary times.

Pure ALOHA (2)
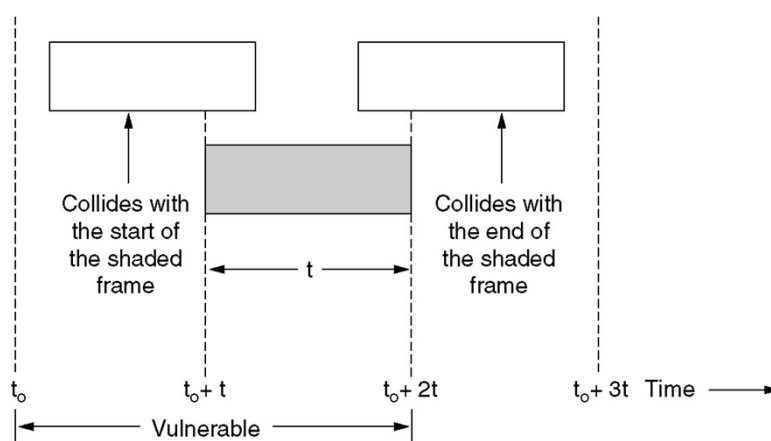
Vulnerable period for the shaded frame.

Pure ALOHA (3)
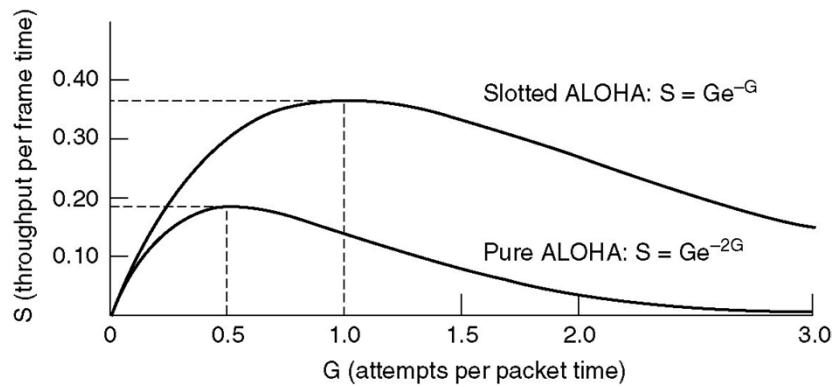
Throughput versus offered traffic for ALOHA systems.

Persistent and Nonpersistent CSMA
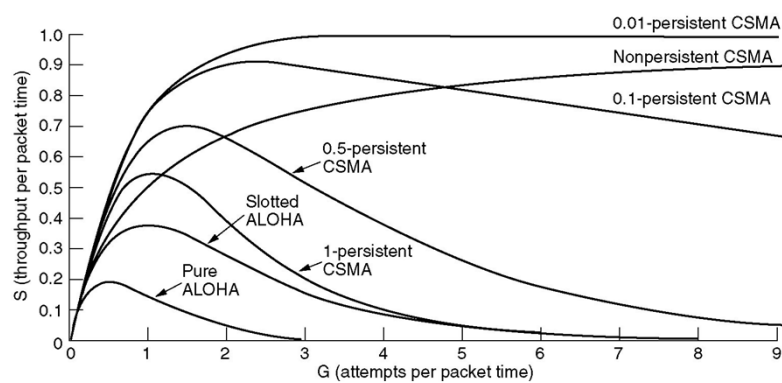
Comparison of the channel utilization versus load for various random access protocols.

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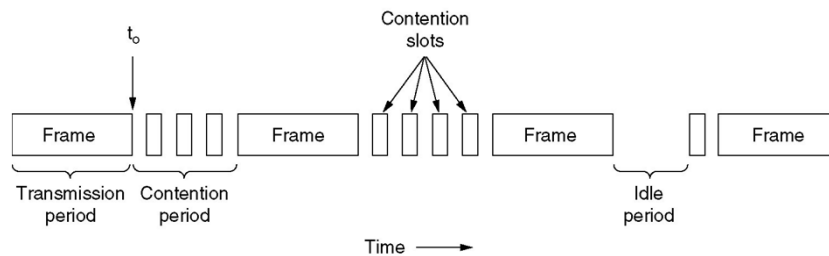
CSMA/CD can be in one of three states: contention, transmission, or idle.

Collision-Free Protocols
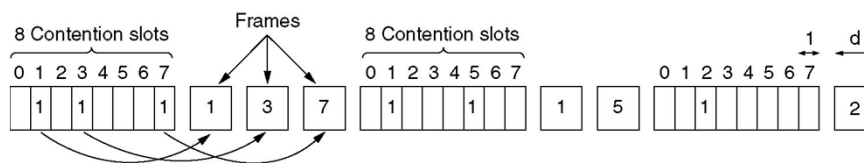
The basic bit-map protocol.

Collision-Free Protocols (2)

Bit time
0 1 2 3

```
0  0  1  0      0  –  –  –
0  1  0  0      0  –  –  –
1  0  0  1      1  0  0  –
1  0  1  0      1  0  1  0
```

Result   1 0 1 0

Stations 0010
and 0100 see this
1 and give up

Station 1001
sees this 1
and gives up
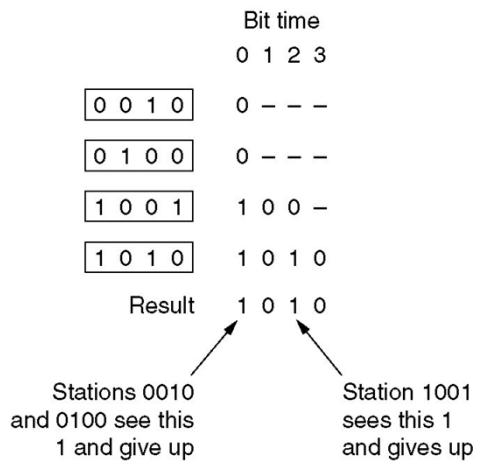
The binary countdown protocol. A dash indicates silence.

Limited-Contention Protocols
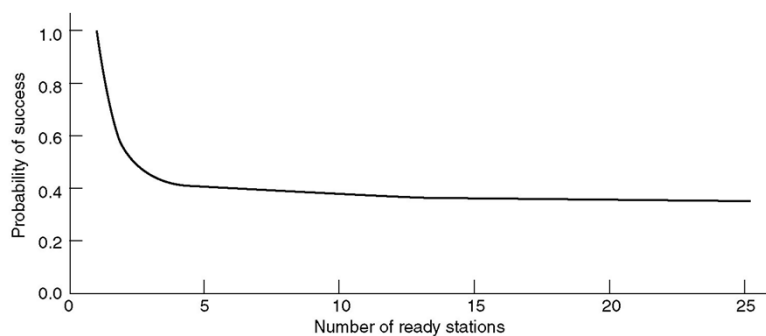
Acquisition probability for a symmetric contention channel.

Adaptive Tree Walk Protocol
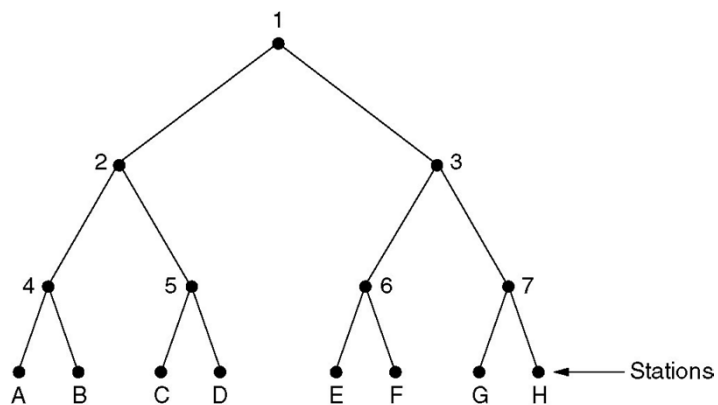
The tree for eight stations.

Wavelength Division Multiple Access Protocols
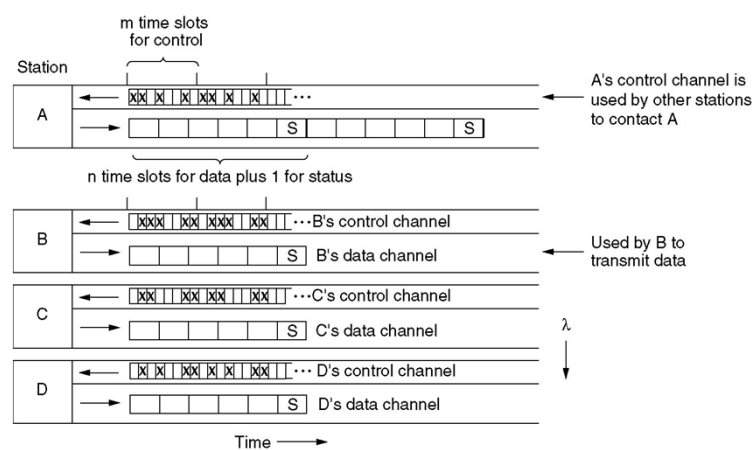
Wavelength division multiple access.

Wireless LAN Protocols

Radio range

(a)   (b)

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

Wireless LAN Protocols (2)

Range of A's transmitter   Range of B's transmitter

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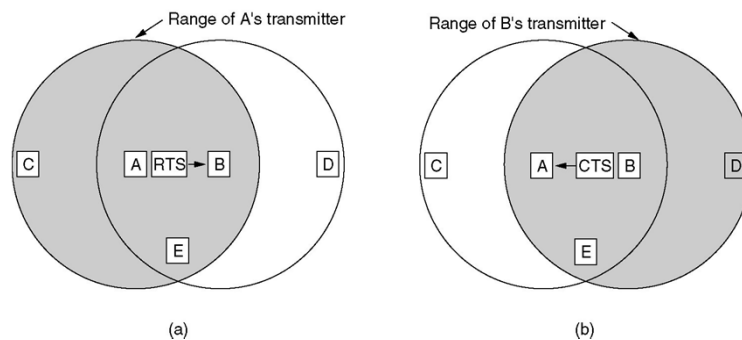
The MACA protocol. (a) A sending an RTS to B.
(b) B responding with a CTS to A.

Ethernet

- Ethernet Cabling
- Manchester Encoding
- The Ethernet MAC Sublayer Protocol
- The Binary Exponential Backoff Algorithm
- Ethernet Performance
- Switched Ethernet
- Fast Ethernet
- Gigabit Ethernet
- IEEE 802.2: Logical Link Control
- Retrospective on 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.

Ethernet Cabling (2)
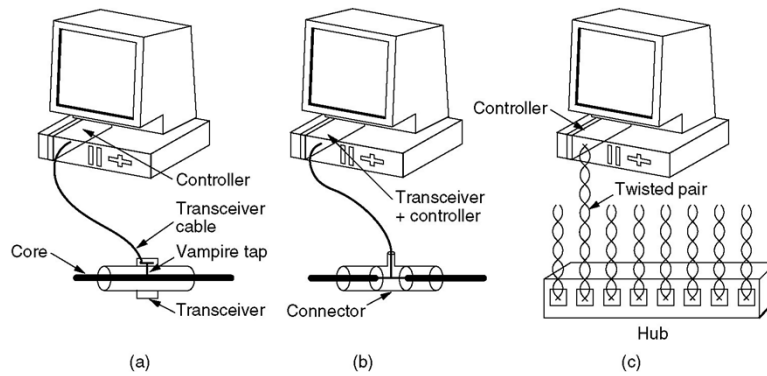
Three kinds of Ethernet cabling.
(a) 10Base5, (b) 10Base2, (c) 10Base-T.

Ethernet Cabling (3)
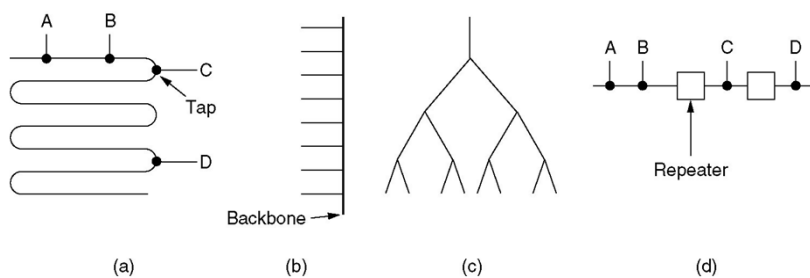
Cable topologies. (a) Linear, (b) Spine, (c) Tree, (d) Segmented.

Ethernet Cabling (4)
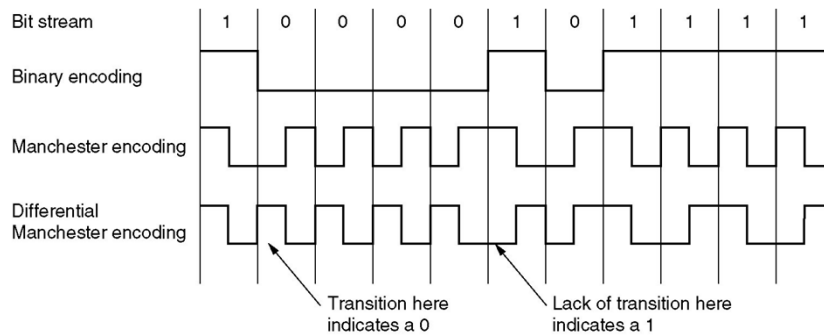
(a) Binary encoding, (b) Manchester encoding,
(c) Differential Manchester encoding.

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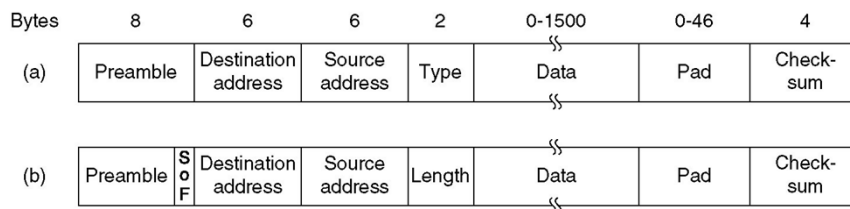
Frame formats. (a) DIX Ethernet, (b) IEEE 802.3.

11

Ethernet MAC Sublayer Protocol (2)
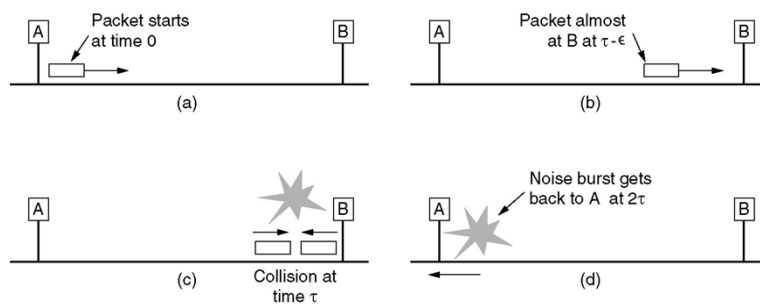
Collision detection can take as long as 2τ.

Ethernet Performance
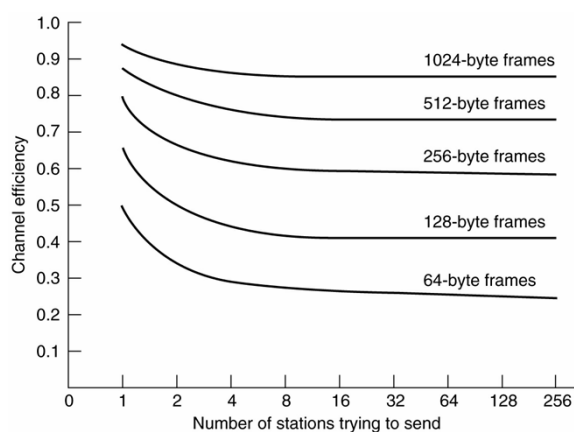
Efficiency of Ethernet at 10 Mbps with 512-bit slot times.

Switched Ethernet
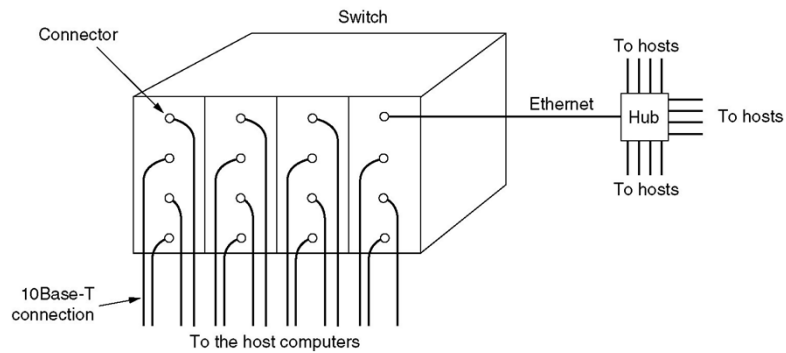
A simple example of switched Ethernet.

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
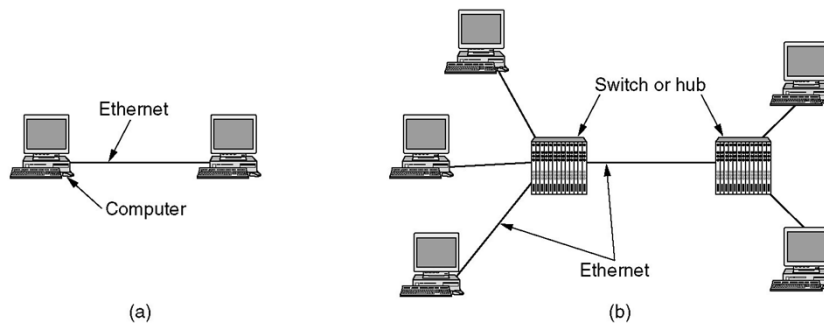
(a) A two-station Ethernet. (b) A multistation Ethernet.

Gigabit Ethernet (2)

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.

IEEE 802.2: Logical Link Control
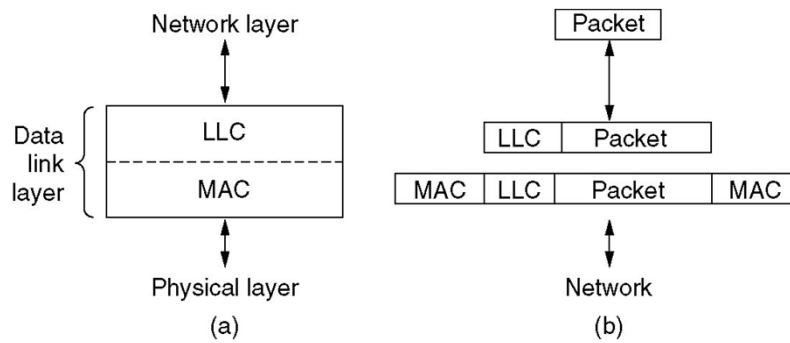
(a) Position of LLC. (b) Protocol formats.

Wireless LANs

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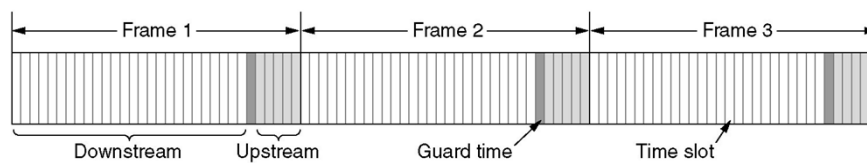- Services

The 802.11 Protocol Stack
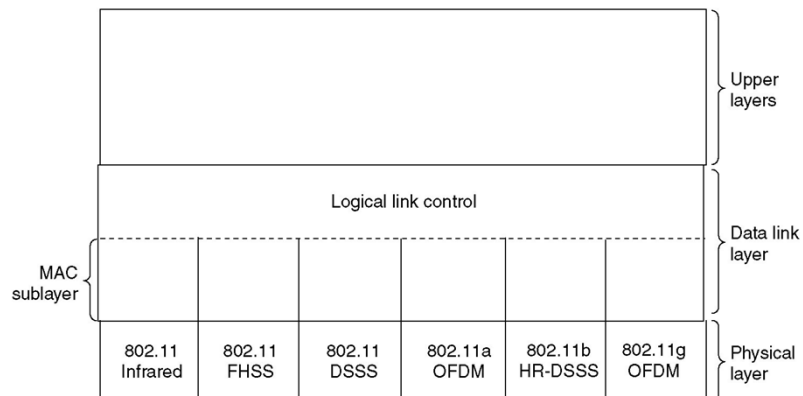
Part of the 802.11 protocol stack.

The 802.11 MAC Sublayer Protocol
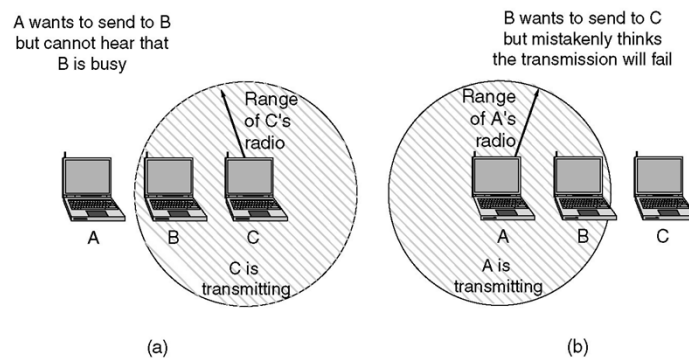
(a) The hidden station problem.
(b) The exposed station problem.

The 802.11 MAC Sublayer Protocol (2)
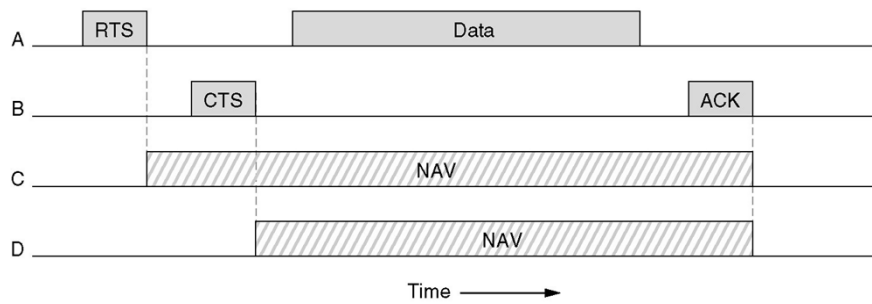
The use of virtual channel sensing using CSMA/CA.

The 802.11 MAC Sublayer Protocol (3)
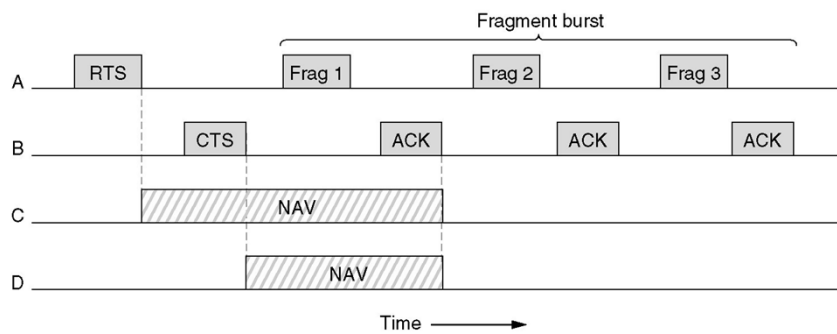
A fragment burst.

The 802.11 MAC Sublayer Protocol (4)
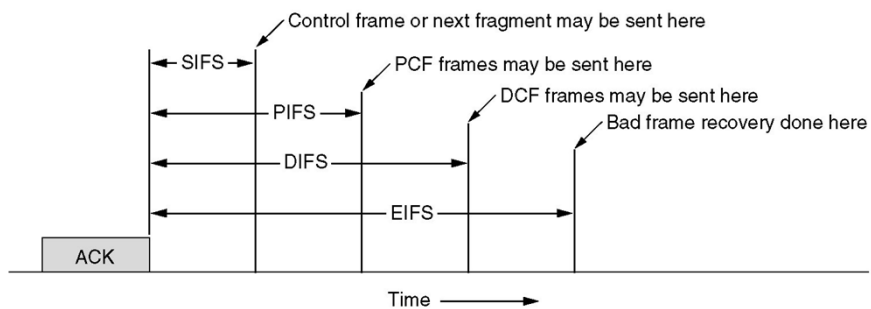
Interframe spacing in 802.11.

The 802.11 Frame Structure

The 802.11 data frame.

802.11 Services

Distribution Services

- Association
- Disassociation
- Reassociation
- Distribution
- Integration

802.11 Services

Intracell Services

- Authentication
- Deauthentication
- Privacy
- Data Delivery

Broadband Wireless

- Comparison of 802.11 and 802.16
- The 802.16 Protocol Stack
- The 802.16 Physical Layer
- The 802.16 MAC Sublayer Protocol
- The 802.16 Frame Structure

The 802.16 Protocol Stack

The 802.16 Protocol Stack.

The 802.16 Physical Layer

QAM-64 (6 bits/baud)

QAM-16 (4 bits/baud)

QPSK (2 bits/baud)
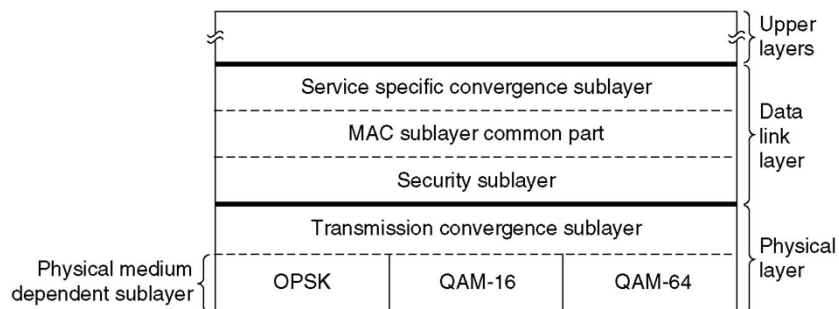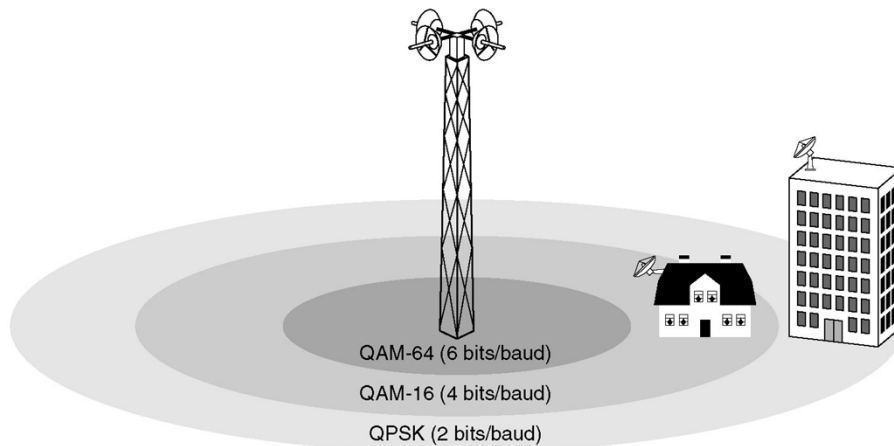
The 802.16 transmission environment.

The 802.16 Physical Layer (2)

| Frame 1 | Frame 2 | Frame 3 |

Downstream   Upstream   Guard time   Time slot

Frames and time slots for time division duplexing.

The 802.16 MAC Sublayer Protocol

Service Classes
- Constant bit rate service
- Real-time variable bit rate service
- Non-real-time variable bit rate service
- Best efforts service

The 802.16 Frame Structure

Bits1 1	6	1 1 2 1	11	16	8		4
(a) 0 EC	Type	C I EK	Length	Connection ID	Header CRC	Data	CRC

Bits1 1	6	16	16	8
(b) 1 0	Type	Bytes needed	Connection ID	Header CRC
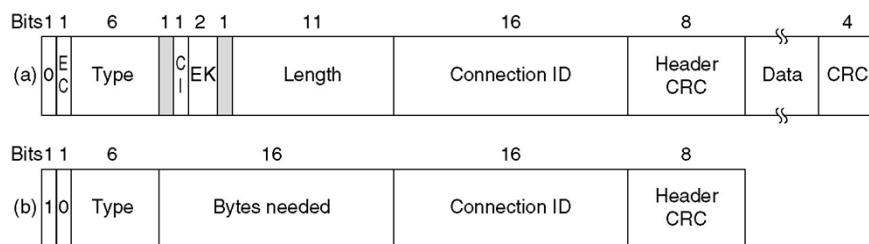
(a) A generic frame.   (b) A bandwidth request frame.

Bluetooth

- Bluetooth Architecture
- Bluetooth Applications
- The Bluetooth Protocol Stack
- The Bluetooth Radio Layer
- The Bluetooth Baseband Layer
- The Bluetooth L2CAP Layer
- The Bluetooth Frame Structure

Bluetooth Architecture
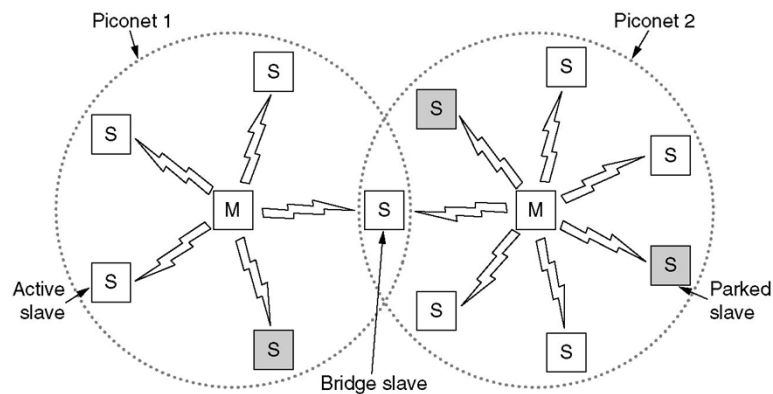
Two piconets can be connected to form a scatternet.

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
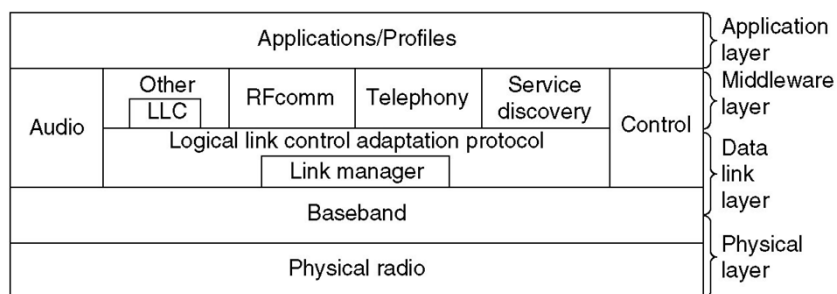
The 802.15 version of the Bluetooth protocol architecture.
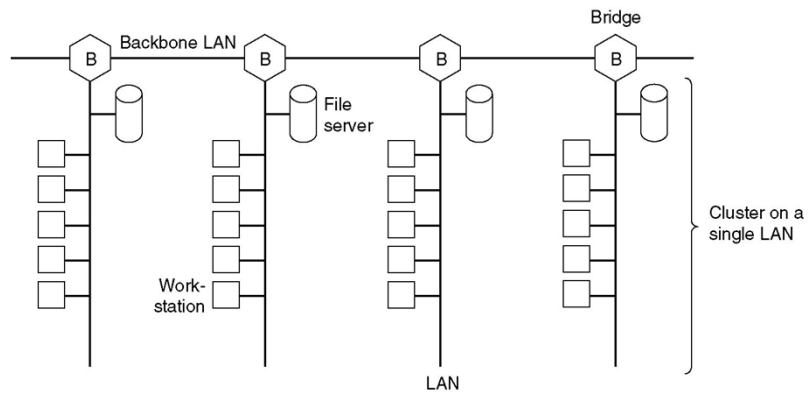
The Bluetooth Frame Structure
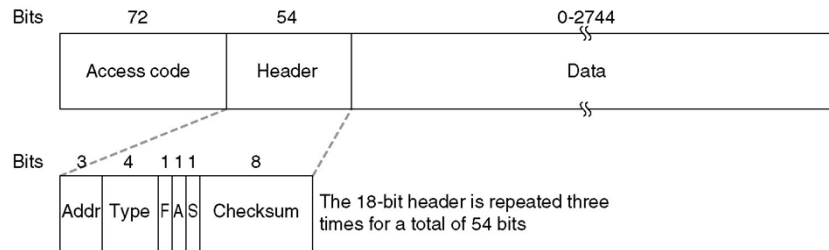
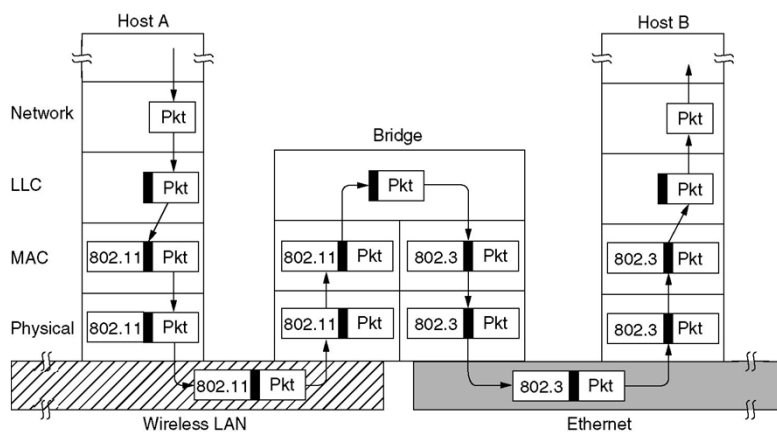A typical Bluetooth data frame.

Data Link Layer Switching

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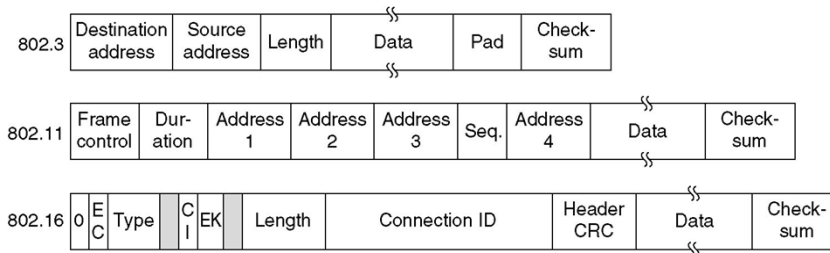- Virtual LANs

Data Link Layer Switching

Multiple LANs connected by a backbone to handle a total load higher than the capacity of a single LAN.
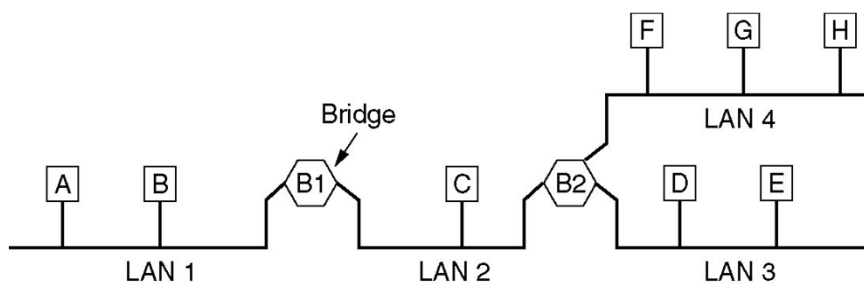
Bridges from 802.x to 802.y

Operation of a LAN bridge from 802.11 to 802.3.
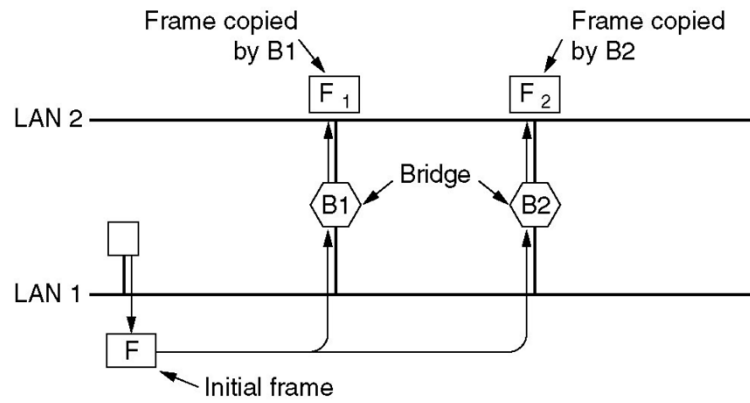
Bridges from 802.x to 802.y (2)

The IEEE 802 frame formats. The drawing is not to scale.
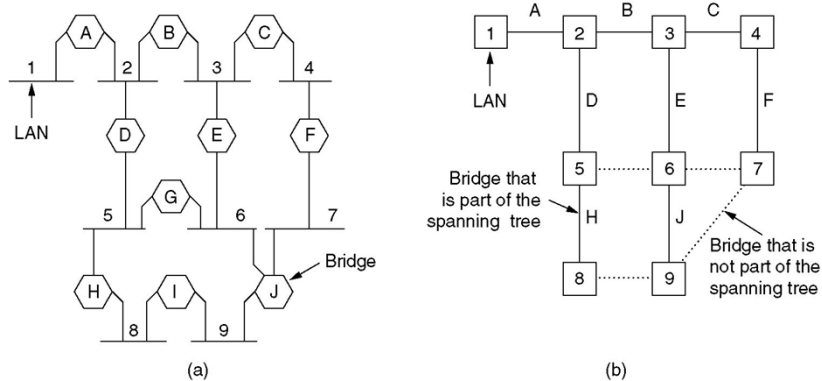
Local Internetworking

A configuration with four LANs and two bridges.
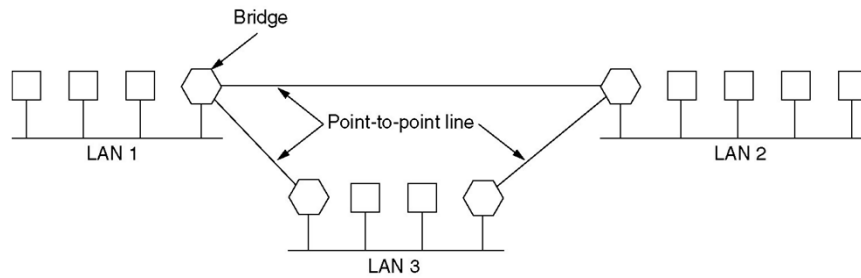
Spanning Tree Bridges

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.
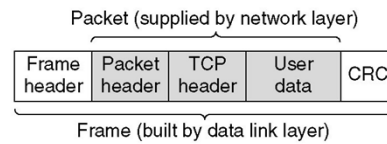
Remote Bridges

Bridge

Point-to-point line

LAN 1

LAN 2

LAN 3

Remote bridges can be used to interconnect distant LANs.

Repeaters, Hubs, Bridges, Switches, Routers and Gateways

Application layer	Application gateway
Transport layer	Transport gateway
Network layer	Router
Data link layer	Bridge, switch
Physical layer	Repeater, hub

(a)

Packet (supplied by network layer)

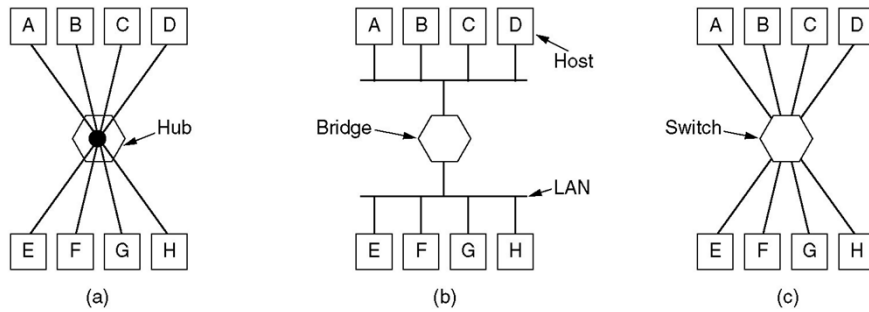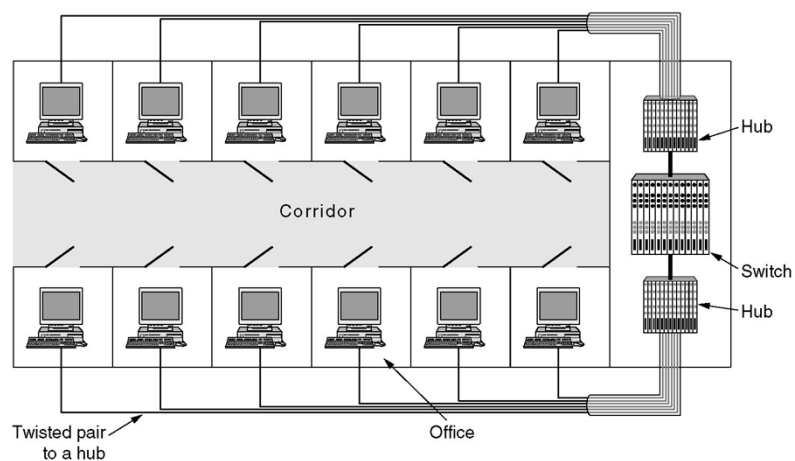| Frame header | Packet header | TCP header | User data | CRC |

Frame (built by data link layer)

(b)

(a) Which device is in which layer.

(b) Frames, packets, and headers.

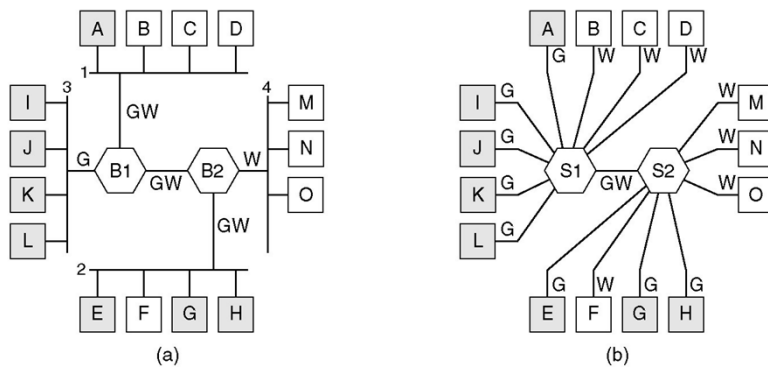# Repeaters, Hubs, Bridges, Switches, Routers and Gateways (2)

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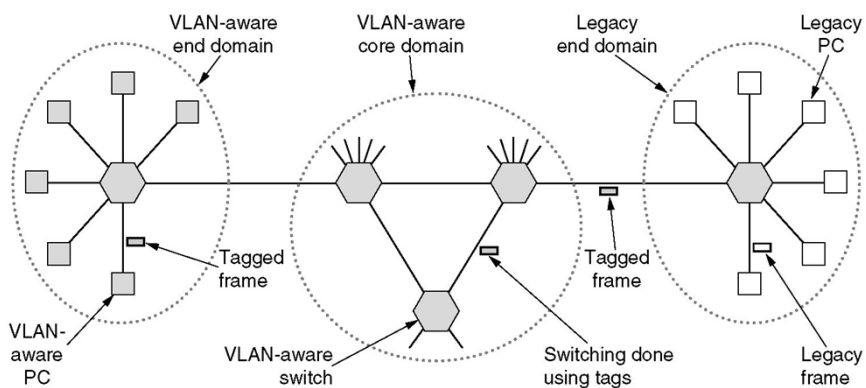
Virtual LANs

A building with centralized wiring using hubs and a switch.
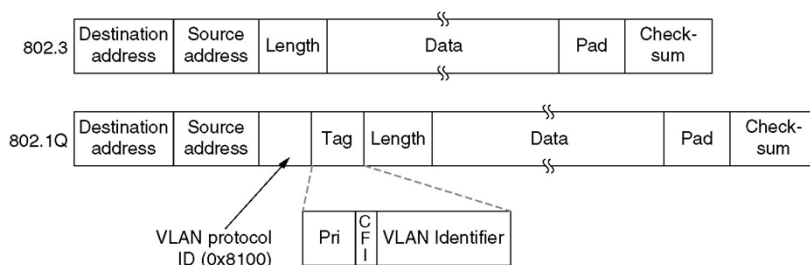
Virtual LANs (2)

(a)

(b)

(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.

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.

Summary

Method	Description
FDM	Dedicate a frequency band to each station
WDM	A dynamic FDM scheme for fiber
TDM	Dedicate a time slot to each station
Pure ALOHA	Unsynchronized transmission at any instant
Slotted ALOHA	Random transmission in well-defined time slots
1-persistent CSMA	Standard carrier sense multiple access
Nonpersistent CSMA	Random delay when channel is sensed busy
P-persistent CSMA	CSMA, but with a probability of p of persisting
CSMA/CD	CSMA, but abort on detecting a collision
Bit map	Round robin scheduling using a bit map
Binary countdown	Highest numbered ready station goes next
Tree walk	Reduced contention by selective enabling
MACA, MACAW	Wireless LAN protocols
Ethernet	CSMA/CD with binary exponential backoff
FHSS	Frequency hopping spread spectrum
DSSS	Direct sequence spread spectrum
CSMA/CA	Carrier sense multiple access with collision avoidance

Channel allocation methods and systems for a common channel.