Chapter 5 LINK LAYER

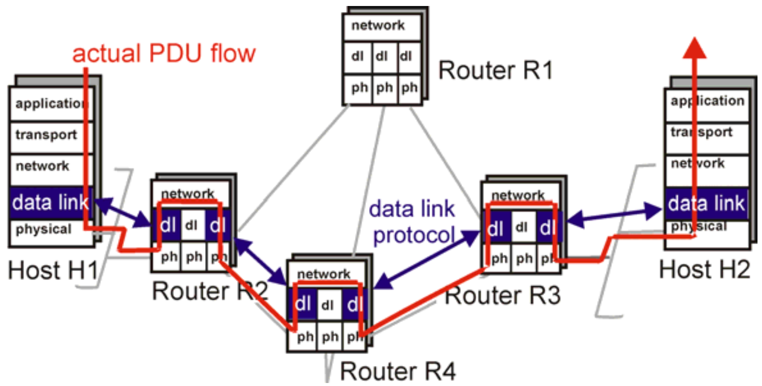
Computer Networks
Winter 2003 / 2004

Distributed Computing Group

Overview

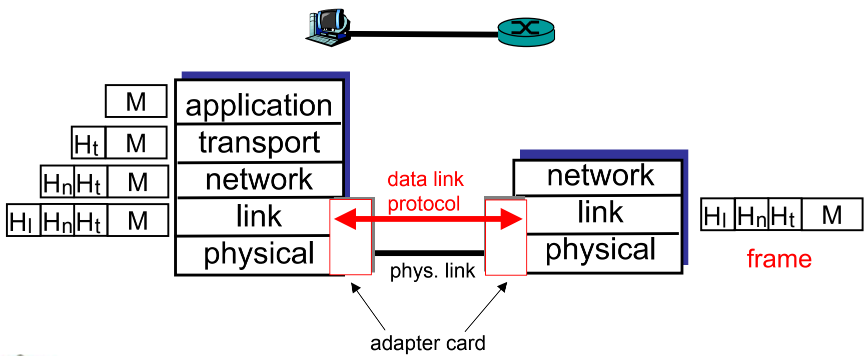
- Link layer services
- Error detection and correction
- Multiple access protocols and LANs
- Link layer addressing, ARP
- Specific link layer technologies
  - Ethernet
  - hubs, bridges, switches
  - PPP
  - IEEE 802.11 LANs

Link Layer: setting the context



Link Layer: setting the context

- *two* physically connected devices
  - host-router, router-router, host-host
- unit of data is called frame



Link Layer Services

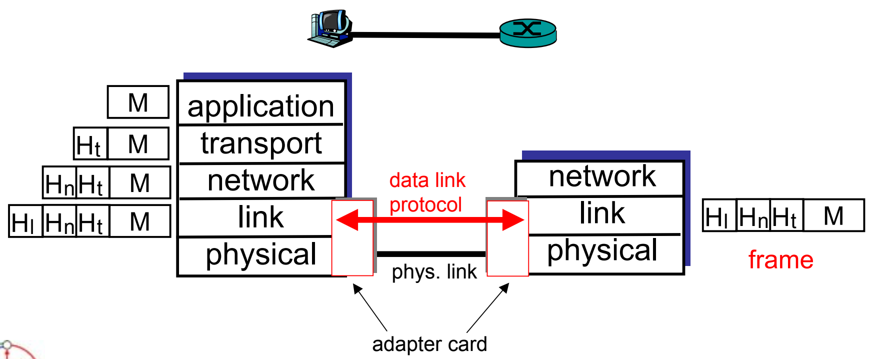
- Framing, link access
  - encapsulate datagram into frame, adding header, trailer
  - implement channel access if shared medium
  - ‘physical addresses’ used in frame headers to identify source, destination
    - different from IP address!
- Reliable delivery between two physically connected devices:
  - we learned how to do this in chapter 3
  - seldom used on low error link (fiber, some twisted pair)
  - wireless links: high error rates
    - Q: why both link-level and end-end reliability?

Link Layer Services (more)

- Flow Control
  - pacing between sender and receiver
- Error Detection
  - errors caused by signal attenuation, noise
  - receiver detects presence of errors:
    - receiver signals sender for retransmission or drops frame
- Error Correction
  - receiver identifies *and corrects* bit error(s) without resorting to retransmission

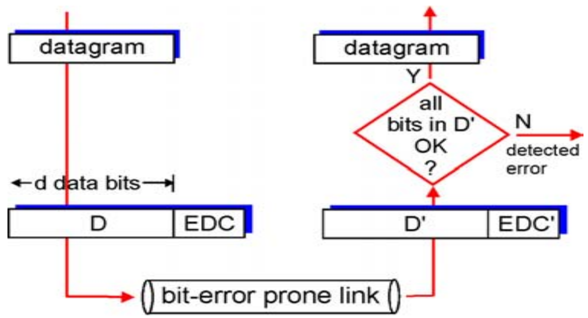
Link Layer: Implementation

- Link layer implemented in “adapter” (a.k.a. NIC)
  - e.g., PCMCIA card, Ethernet card
  - typically includes: RAM, DSP chips, host bus interface, and link interface



Error Detection

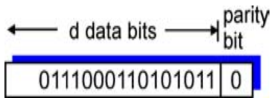
- EDC = Error Detection and Correction bits (redundancy)
- D = Data protected by error checking, may include header fields
- Error detection not 100% reliable!
  - protocol may miss some errors, but rarely
  - larger EDC field yields better detection and correction



Parity Checking

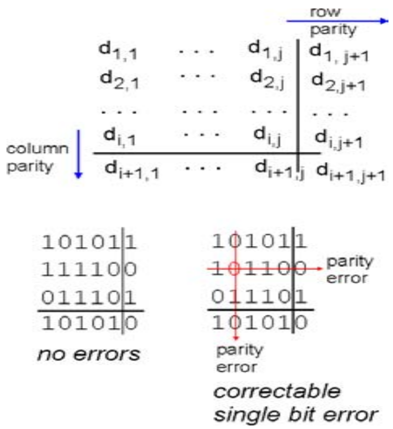
Single Bit Parity

Detect single bit errors



Two Dimensional Bit Parity

Detect *and correct* single bit errors



Internet checksum

Goal

- detect "errors" (e.g., flipped bits) in transmitted segment
- note: used at transport layer only

Sender

- treat segment content as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment content
- sender puts checksum value into UDP checksum field

Receiver

- compute checksum of received segment
- check if computed checksum equals checksum field value
  - NO → error detected
  - YES → no error detected. *But maybe errors nonetheless?* More later…

Cyclic Redundancy Code (CRC): Ring

- Polynomials with binary coefficients $b_k x^k + b_{k-1} x^{k-1} + \ldots + b_0 x^0$
- Order of polynomial: max i with $b_i \neq 0$
- Binary coefficients b_i (0 or 1) form a field with operations "+" (XOR) and "·" (AND).
- The polynomials form a ring R with operations "+" and "·": (R,+) is an abelian group, (R, ·) is an associative set, and the distributive law does hold, that is, $a\cdot(b+c) = a\cdot b+a\cdot c$ respectively $(b+c)\cdot a = b\cdot a+c\cdot a$ with $a,b,c \in R$.
- Example:

$(x^3+1)\cdot(x^4+x+1)$	$1001\cdot10011$
$= x^3\cdot(x^4+x+1) + 1\cdot(x^4+x+1)$	$=$ 10011
$= (x^7+x^4+x^3) + (x^4+x+1)$	+ 10011000
$= x^7+x^3+x+1$	= 10001011

Cyclic Redundancy Code (CRC): Division

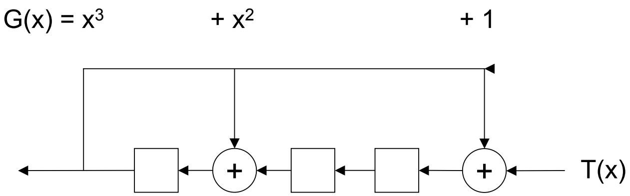
- Generator polynomial $G(x) = x^{16}+x^{12}+x^5+1$
- Let the whole frame (D+EDC) be polynomial T(x)
- Idea: fill EDC (CRC) field such that T(x) mod G(x) = 0
- How to divide with polynomials? Example with $G(x) = x^2+1$ (=101)

  11101100 / 101 = 110110, Remainder 10
  100
  011
  111
  100
  010

- Idea: Fill EDC with remainder when dividing T(x) with EDC=00…0 by G(x). Then calculating and testing CRC is the same operation.

Cyclic Redundancy Code (CRC): Division in Hardware

- Use cyclic shift register with r registers, where r is the order of G(x)
- Example



- Finally the remainder of the division is in the registers

Cyclic Redundancy Code (CRC): How to chose G(x)?

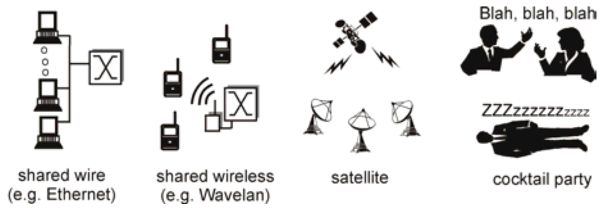
- Typical generator polynomial $G(x) = x^{16}+x^{12}+x^5+1$
- Why does G(x) look like this?

- Let E(x) be the transmission errors, that is T(x) = M(x) + E(x)
- T(x) mod G(x) = (M(x) + E(x)) mod G(x)
  = M(x) mod G(x) + E(x) mod G(x)
- Since M(x) mod G(x) = 0 we can detect all transmission errors as long as E(x) is not divisible by G(x) without remainder

- One can show that G(x) of order r can detect
  – all single bit errors as long as G(x) has 2 or more coefficients
  – all bursty errors (burst of length k is a k-bit long 1xxxx1 string) with $k \leq r$ (note: needs G(x) to include the term 1)
  – Any error with probability 2^{-r}

Multiple Access Links and Protocols

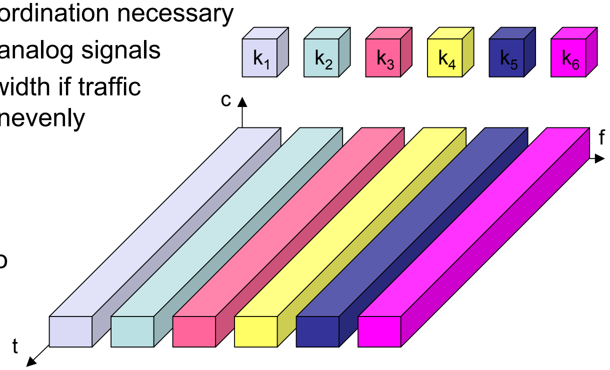
Three types of “links”
- point-to-point (single wire; e.g. PPP, SLIP)
- broadcast (shared wire or medium; e.g. Ethernet, WLAN)
- switched (e.g. switched Ethernet, ATM)



Channel Partitioning: Frequency Division Multiplex (FDM)

- Separation of the whole spectrum into smaller frequency bands
- A channel gets a certain band of the spectrum for the whole time
+ no dynamic coordination necessary
+ works also for analog signals
– waste of bandwidth if traffic is distributed unevenly
– inflexible

- Example:
  broadcast radio



Channel Partitioning: Time Division Multiplex (TDM)

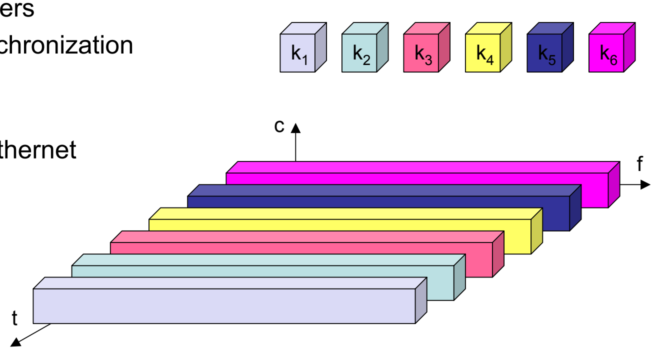
- A channel gets the whole spectrum for a certain amount of time

\+ only one carrier in the medium at any time

\+ throughput high even for many users

– precise synchronization necessary

- Example: Ethernet



Channel Partitioning: Time/Frequency Division Multiplex

- Combination of both methods
- A channel gets a certain frequency band for some time

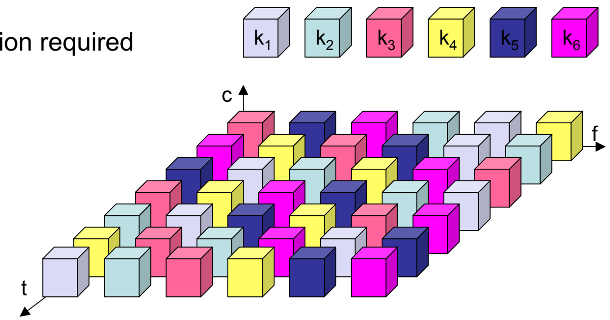
\+ protection against frequency selective interference

\+ protection against tapping

\+ adaptive

– precise coordination required

- Example: GSM



Channel Partitioning: Code Division Multiplex (CDM)

- Each channel has a unique code
- All channels use the same spectrum at the same time

\+ bandwidth efficient

\+ no coordination or synchronization

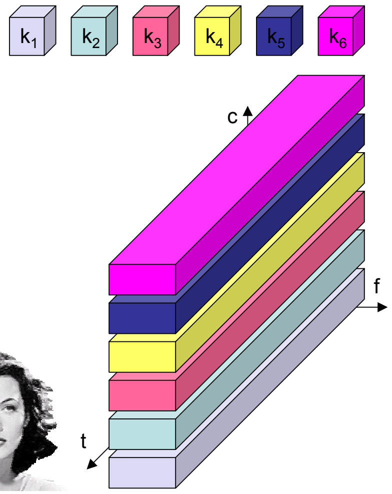
\+ hard to tap

\+ almost impossible to jam

– lower user data rates

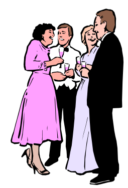
– more complex signal regeneration

- Example: UMTS
- Spread spectrum
- U. S. Patent 2‘292‘387, Hedy K. Markey (a.k.a. Lamarr or Kiesler) and George Antheil (1942)



Cocktail party as analogy for multiplexing

- Space multiplex: Communicate in different rooms
- Frequency multiplex: Use soprano, alto, tenor, or bass voices to define the communication channels
- Time multiplex: Let other speaker finish
- Code multiplex: Use different languages and hone in on your language. The “farther apart” the languages the better you can filter the “noise”: German/Japanese better than German/Dutch. Can we have orthogonal languages?



Multiple Access Protocols

- Single shared communication channel
- Two or more simultaneous transmissions by nodes: interference
  - only one node can send successfully at a time
- Multiple Access Control (MAC) Protocol
  - distributed algorithm that determines how stations share channel, i.e., determine when station can transmit
  - communication about channel sharing must use channel itself!
  - what to look for in multiple access protocols
    - synchronous or asynchronous
    - information needed about other stations
    - robustness (e.g. to channel errors)
    - performance

MAC Protocols: a taxonomy

Three broad classes

- Channel Partitioning
  - divide channel into smaller “pieces” (time slots, frequency)
  - allocate piece to node for exclusive use
- “Taking turns”
  - tightly coordinate shared access to avoid collisions
- Random Access
  - allow collisions
  - “recover” from collisions

- Goals: decentralized, efficient, simple, fair

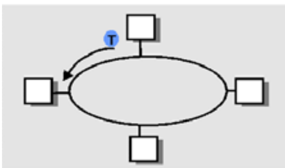
“Taking Turns” MAC protocols

Polling

- master node “invites” slave nodes to transmit in turn
- Request to Send, Clear to Send messages
- concerns
  - polling overhead
  - latency
  - single point of failure (master)

Token passing (Token Ring)

- control token passed from one node to next sequentially
- token message
- concerns
  - token overhead
  - latency
  - single point of failure (token)



“Taking Turns” Protocols: Round Robin

- Round robin protocol: station k sends after station k–1 (mod n)
- If a station does not need to transmit data, then it sends “ε”
- There is a maximum message size m that can be transmitted
- Is this different from token ring protocol?

- Questions
  - How efficient is round robin?
  - What if a station breaks or leaves?
  - Can a new station join?

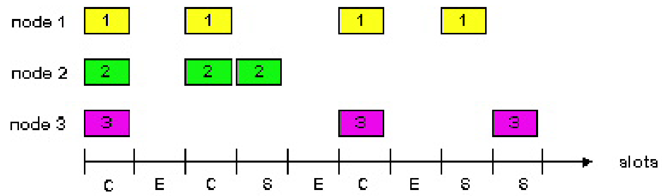
- All deterministic protocols have these (or worse) problems
  - Try randomized protocols instead!

Random Access protocols

- When node has packet to send
  - transmit at full channel data rate R
  - no *a priori* coordination among nodes
- Two or more transmitting nodes → "collision"
- Random access MAC protocol specifies
  - how to detect collisions
  - how to recover from collisions
    - via delayed retransmissions
- Examples of random access MAC protocols:
  - ALOHA and variants (slotted ALOHA, adaptive ALOHA)
  - Backoff protocols (CSMA, CSMA/CD, CSMA/CA)

Slotted Aloha

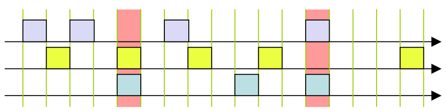
- Time is divided into equal size slots
  - A slot is equal to the packet transmission time
- Node with new arriving packet: transmit at beginning of next slot
- If collision: retransmit packet in future slots with probability p, until successful



Success (S), Collision (C), Empty (E) slots

Slotted Aloha (slightly simplified version for analysis)

- We assume that the stations are perfectly synchronous
- In each time slot each station transmits with probability *p*



$$P_1 = \Pr[\text{Station 1 succeeds}] = p(1-p)^{n-1}$$

$$P = \Pr[\text{any Station succeeds}] = nP_1$$

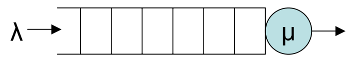
$$\text{maximize } P: \frac{dP}{dp} = n(1-p)^{n-2}(1-pn) \overset{!}{=} 0 \quad \Rightarrow \quad pn = 1$$

$$\text{then, } P = (1-\frac{1}{n})^{n-1} \geq \frac{1}{e}$$

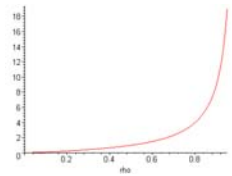
- In slotted aloha, a station can transmit successfully with probability at least 1/*e*. How quickly can an application send packets to the radio transmission unit? Queuing Theory!

Queuing Theory (Remember Chapter 3?)

- Simplest M/M/1 queuing model (M=Markov):
- Poisson arrival rate λ, exponential service time with mean $1/\mu$



- In our time slot model, this means that the probability that a new packet is received by the buffer is λ; the probability that sending succeeds is $\mu = 1/e$, for any time slot. To keep the queue bounded we need $\rho = \lambda/\mu < 1$, thus $\lambda < 1/e$.
- In the equilibrium, the expected number of packets in the system is $N = \rho/(1-\rho)$, the average time in the system is $T = N/\lambda$.

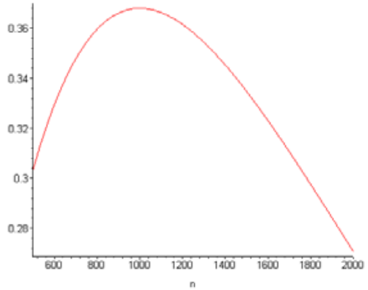


Slotted Aloha vs. Round Robin

– Slotted aloha uses not every slot of the channel; the round robin protocol is better in this respect.

+ What happens in round robin when a new station joins? What about more than one new station? Slotted aloha is more flexible.

- Example: If the actual number of stations is twice as high as expected, there is still a successful transmission with probability 27%. If it is only half, 30% of the slots are successful.



Adaptive slotted aloha

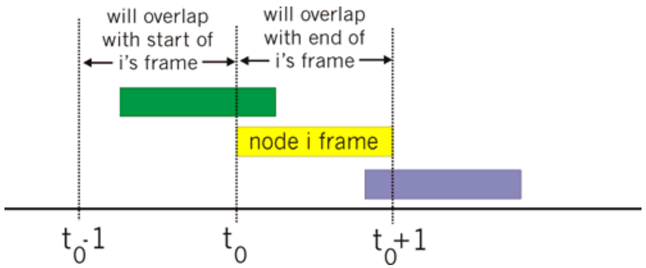
- Idea: Change the access probability with the number of stations
- How can we estimate the current number of stations in the system?
- Assume that stations can distinguish whether 0, 1, or more than 1 stations send in a time slot.

- Idea: Try to estimate the number of stations!
  – If you see that nobody sends, increase *p*.
  – If you see that more than one sends, decrease *p*.

- Analysis a little too tough for this course (unfortunately)

Pure (unslotted) ALOHA

- Unslotted Aloha: simpler, no synchronization
- Packet needs transmission
  – send without awaiting for beginning of slot
- Collision probability increases:
  – packet sent at t_0 collide with packets sent in (t_0-1, t_0+1)



Pure Aloha Analysis

- Partition each slot of size 1 into x "minislots" of size ε (then x = 1/ε)
- Probability to start transmission in minislot is p_ε for each station

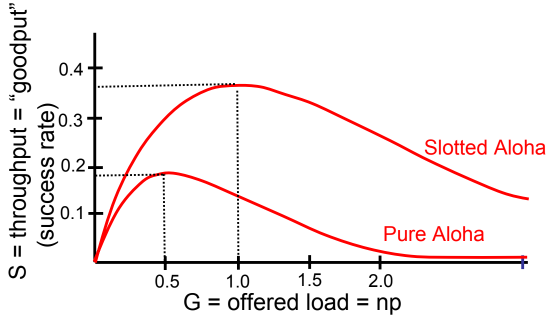
$$P_1 = \text{Pr[Station 1 succeeds]} = p_\epsilon(1-p_\epsilon)^{(2x-1)(n-1)}$$
$$P = \text{Pr[any station succeeds]} = n \cdot P_1$$

P can be maximized by choosing $p_\epsilon = \frac{\epsilon}{\epsilon+2n}$, which is $\frac{\epsilon}{2n}$ for $\epsilon \to 0^+$. Then

$$P = np_\epsilon(1-p_\epsilon)^{(2x-1)(n-1)} = \frac{\epsilon}{2}\left(1-\frac{\epsilon}{2n}\right)^{2n/\epsilon-\ldots} \geq \frac{\epsilon}{2e}$$

- Since there are x minislots in 1 slot, the success rate of a slot is about 1/2e, that is, half the rate of slotted aloha

Slotted Aloha vs. Pure Aloha



protocol constrains effective channel throughput!

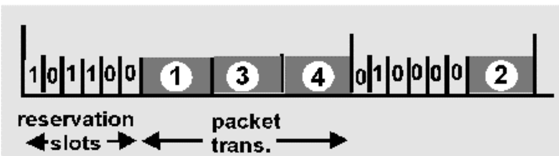
Demand Assigned Multiple Access (DAMA)

- Channel efficiency only 36% for Slotted Aloha, and even worse for Aloha.
- Practical systems therefore use reservation whenever possible. But: Every scalable system needs an Aloha style component.
- Reservation
  - a sender *reserves* a future time-slot
  - sending within this reserved time-slot is possible without collision
  - reservation also causes higher delays
  - typical scheme for satellite systems

Example for reservation-based protocol

Distributed Polling

- time divided into slots
- begins with N short *reservation slots*
  - reservation slot time equal to channel end-end propagation delay
  - station with message to send posts reservation
  - reservation seen by all stations
- after reservation slots, message transmissions ordered by known priority



Backoff Protocols

- Backoff protocols rely on acknowledgements only.
- Binary exponential backoff, for example, works as follows:
- If a packet has collided k times, we set $p = 2^{-k}$
  Or alternatively: wait from random number of slots in $[1..2^k]$
- It has been shown that binary exponential backoff is not stable for any λ > 0 (if there are infinitely many potential stations)
  [Proof sketch: with very small but positive probability you go to a bad situation with many waiting stations, and from there you get even worse with a potential function argument – sadly the proof is much too intricate to be shown in this course ☺]
- Interestingly when there are only finite stations, binary exponential backoff becomes unstable with λ > 0.568; Polynomial backoff however, remains stable for any λ < 1.

CSMA: Carrier Sense Multiple Access

Idea of CSMA: listen before transmit!

- If channel sensed idle: transmit entire packet
- If channel sensed busy, defer transmission. Two variants
  - *Persistent* CSMA
    - retry immediately with probability p when channel becomes idle (may cause instability)
  - *Non-persistent* CSMA
    - retry after random interval

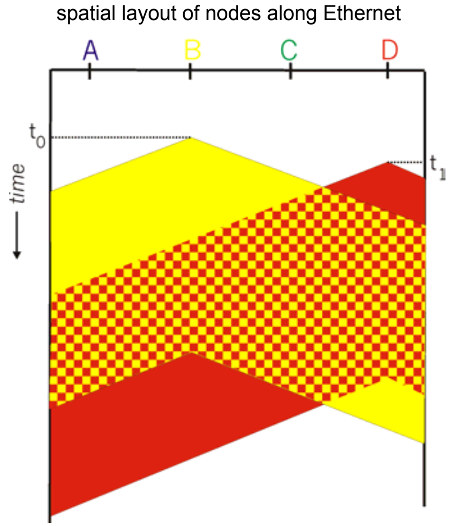
- Human analogy
  1. Don't interrupt anybody already speaking

CSMA collisions

collisions *can* occur
propagation delay: two nodes may not hear each other's transmission

collision
entire packet transmission time wasted

note role of distance and propagation delay in determining collision prob.



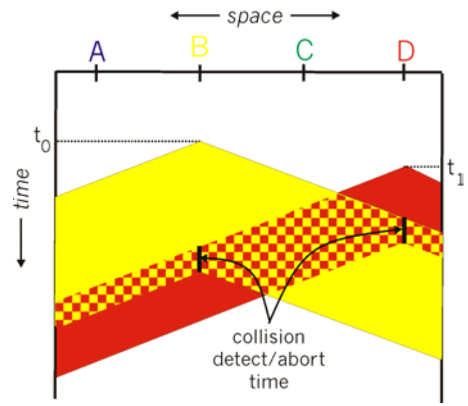
CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, as in CSMA
- collisions *detected* within short time
- colliding transmissions aborted, reducing channel wastage
- persistent or non-persistent retransmission

- collision detection
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: receiver shut off while transmitting

- Human analogy (the polite conversationalist)
  1. Don't interrupt anybody already speaking
  2. If another starts speaking with you, then back off.

CSMA/CD collision detection

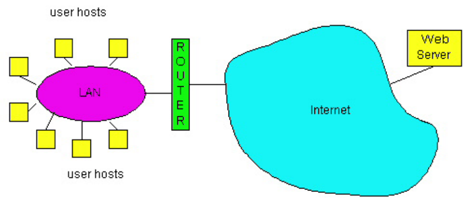


Summary of MAC protocols

- What do you do with a shared media?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Code Division, Frequency Division
  - Taking Turns
    - polling from a central site, token passing
  - Random partitioning (dynamic)
    - ALOHA, S-ALOHA, CSMA, CSMA/CD
    - carrier sensing
      - easy in some technologies (wire)
      - hard in others (wireless)
    - CSMA/CD used in Ethernet

LAN technologies

- Data link layer so far
  - services, error detection/correction, multiple access
- Next: LAN technologies
  - addressing
  - Ethernet
  - hubs, bridges, switches
  - 802.11
  - PPP



LAN Addresses and ARP

32-bit IP address

- *network-layer* address
- used to get datagram to destination network (recall IP network definition)

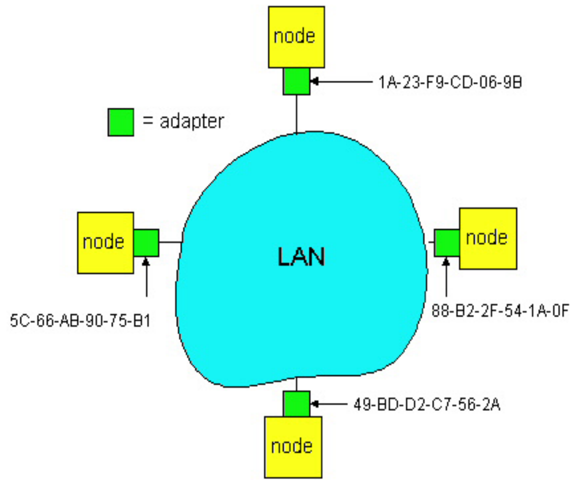
MAC (or LAN or physical) address

- used to get datagram from one interface to another physically-connected interface (same LAN)
- 48 bit MAC address (for most LANs) burned in the adapter ROM

ARP (Address Resolution Protocol)

- IP-Address → MAC-Address

LAN Addresses and ARP



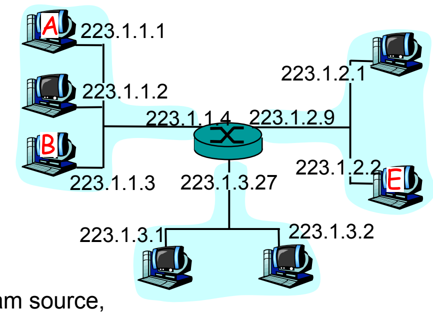
Each adapter on LAN has unique LAN address

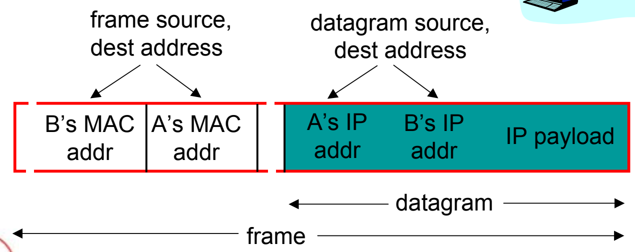
LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy
  - MAC address like Social Security (AHV) Number
  - IP address like postal address
- MAC flat address → portability
  - can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - depends on network to which one attaches

Recall earlier routing discussion

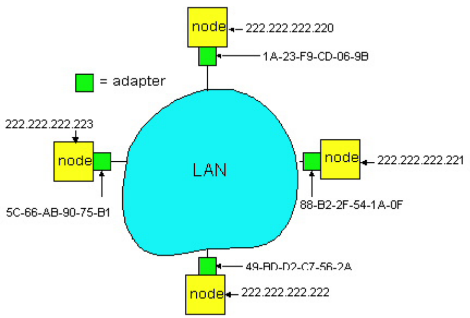
- Starting at A
- given IP datagram addressed to B
- look up network address of B
- find B on same net as A
- link layer send datagram to B inside link-layer frame





ARP: Address Resolution Protocol

Question: How to determine MAC address of B given B's IP address?



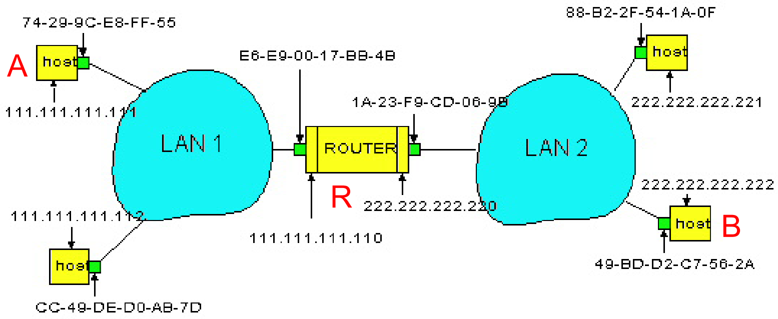
- Each IP node (Host, Router) on LAN has ARP table
- ARP Table: IP/MAC address mappings for some LAN nodes
  < IP addr; MAC addr; TTL >
- TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

ARP protocol

- A knows B's IP address, wants to learn physical address of B
- A broadcasts ARP query packet, containing B's IP address
  - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) physical layer address
- A caches (saves) IP-to-physical address pairs until information becomes old (times out)
- This is a so-called "soft state" protocol
  - information times out (goes away) unless refreshed

Routing to another LAN

walkthrough: routing from A to B via R



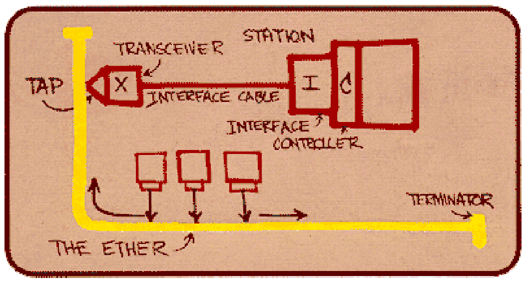
- In routing table at source host, find router 111.111.111.110
- In ARP table at source, find MAC address E6-E9-00-17-BB-4B, etc.

Continued…

- A creates IP packet with source A, destination B
- A uses ARP to get R's physical layer address for 111.111.111.110
- A creates Ethernet frame with R's physical address as destination, Ethernet frame contains A-to-B IP datagram
- A's data link layer sends Ethernet frame
- R's data link layer receives Ethernet frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B's physical layer address
- R creates frame containing A-to-B IP datagram sends to B

Ethernet

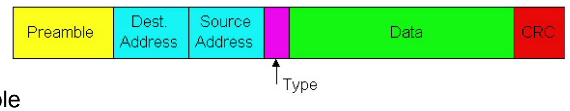
Currently predominant LAN technology
- cheap: $5 for 100Mbps!
- first widely used LAN technology
- Simpler/cheaper than token rings and ATM
- Keeps up with speed race: 10, 100, 1000 Mbps



Metcalfe's Ethernet sketch

Ethernet Frame Structure

- Sending adapter encapsulates IP datagram (or other network layer protocol packet) in *Ethernet frame*



- Preamble
  - 7 bytes with pattern 10101010
  - Followed by 1 byte with pattern 10101011
  - Used to synchronize receiver, sender clock rates
- Addresses
  - 6 bytes, frame is received by all adapters on a LAN and dropped if address does not match
- Type (2 bytes): indicates the higher layer protocol, mostly IP (0x0800) but others may be supported such as Novell IPX and AppleTalk)
- CRC (4 bytes): checked at receiver, if error is detected, the frame is simply dropped

Ethernet uses CSMA/CD (connectionless & unreliable)

- Connectionless
  - No handshaking between sending and receiving adapter.
- Unreliable
  - receiving adapter doesn't send ACKs or NAKs to sending adapter
  - stream of datagrams passed to network layer can have gaps, which will be filled if app is using TCP or seen by application
- No slots
- Carrier sense: adapter doesn't transmit if it senses that some other adapter is transmitting
- Collision detection: transmitting adapter aborts when it senses that another adapter is transmitting
- Random access: Before attempting a retransmission, adapter waits a random time

Ethernet CSMA/CD algorithm

1. Adapter gets datagram from network layer and creates frame
2. If adapter senses channel idle, it starts to transmit frame. If it senses channel busy, waits until channel idle and then transmits
3. If adapter transmits entire frame without detecting another transmission, the adapter is done with frame !
4. If adapter detects another transmission while transmitting, aborts and sends jam signal
5. After aborting, adapter enters exponential backoff: after the m^{th} collision, adapter chooses a K at random from $\{0,1,2,\ldots,2^m-1\}$. Adapter waits K·512 bit times and returns to Step 2

Ethernet's CSMA/CD (more)

Jam Signal

- make sure all other transmitters are aware of collision; 48 bits;

Bit time

- 0.1 microsec for 10 Mbps Ethernet
- for K=1023, wait time is about 50 msec

Exponential Backoff

- Goal: adapt retransmission attempts to estimated current load
  - heavy load: random wait will be longer
- first collision: choose K from {0,1}; delay is K·512 bit transmission times
- after second collision: choose K from {0,1,2,3}
- after ten collisions, choose K from {0,1,2,3,4,…,1023}

CSMA/CD efficiency

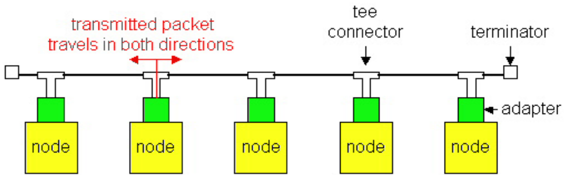
- t_{prop} = max. propagation time between any two nodes in LAN
- t_{trans} = time to transmit max-size frame

$$\text{utilization} \approx \frac{1}{1 + 6.2 \cdot t_{prop} / t_{trans}}$$

- Derivation of this formula is not trivial (not in this course)
- Remarks
  - Utilization goes to 1 as t_{prop} goes to 0
  - Utilization goes to 1 as t_{trans} goes to infinity
  - Much better than ALOHA, but still decentralized, simple, and cheap

Ethernet Technologies: 10Base2

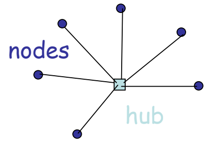
- 10: 10Mbps; 2: under 200 meters maximal cable length
- thin coaxial cable in a bus topology



- repeaters used to connect up to multiple segments
- repeater repeats bits it hears on one interface to its other interfaces: physical layer device only!
- has become a legacy technology

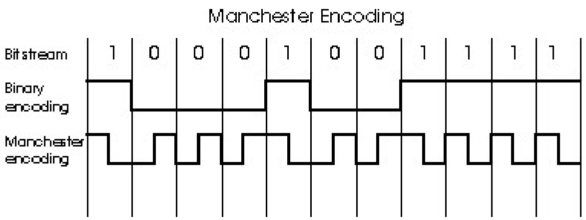
10BaseT and 100BaseT

- 10/100 Mbps rate; latter a.k.a. “fast ethernet”
- T stands for Twisted Pair
- Nodes connect to a hub: “star topology”; 100 m max distance between nodes and hub



- Hubs are essentially physical-layer repeaters
  - bits coming in on one link go out on all other links
  - no frame buffering
  - no CSMA/CD at hub: adapters detect collisions
  - provides net management functionality

Manchester encoding



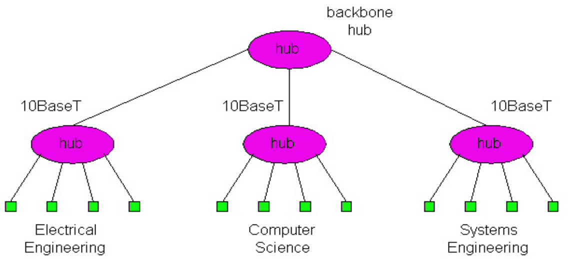
- Used in 10BaseT, 10Base2
- Each bit has a transition
- Allows clocks in sending and receiving nodes to synchronize to each other
  - no need for a centralized, global clock among nodes!
- Hey, this is physical-layer stuff!

Gbit Ethernet

- uses standard Ethernet frame format
- allows for point-to-point links and shared broadcast channels
- in shared mode, CSMA/CD is used; short distances between nodes to be efficient
- uses hubs, called here “Buffered Distributors”
- Full-Duplex at 1 Gbps for point-to-point links
- 10 Gbps now!

Interconnecting with Hubs

- Backbone hub interconnects LAN segments
- Extends max. distance between nodes
- But individual segment collision domains become one large collision domain
  - if a node in CS and a node in EE transmit at same time: collision
- Can't interconnect 10BaseT & 100BaseT

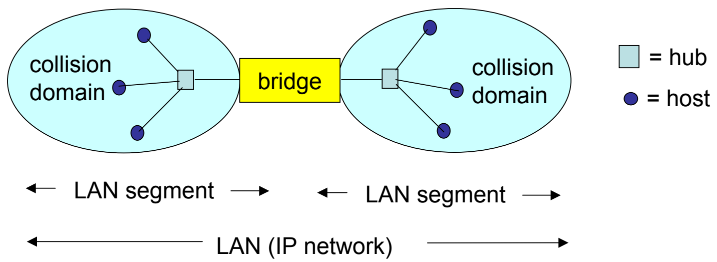


Interconnecting with Bridges

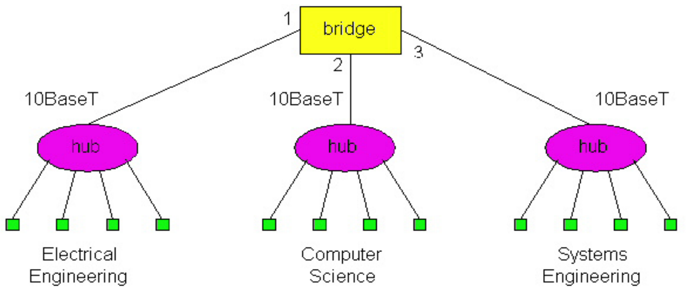
- A bridge is a link layer device
  - stores and forwards Ethernet frames
  - examines frame header and *selectively* forwards frame based on MAC destination address
  - when frame is to be forwarded on segment, uses CSMA/CD to access segment
- transparent
  - hosts are unaware of presence of bridges
- plug-and-play, self-learning
  - bridges do not need to be configured

Bridges: traffic isolation

- Bridge installation breaks LAN into LAN segments
- Bridges *filter* packets:
  - same-LAN-segment frames not usually forwarded onto other LAN segments
  - segments become separate *collision domains*



Forwarding



How do determine to which LAN segment to forward frame?

Looks like a routing problem...

Self learning

- A bridge has a *bridge table*
  - with entries (Node LAN Address, Bridge Interface, Time Stamp)
  - stale entries in table dropped (TTL can be 60 min)
- bridges *learn* which hosts can be reached through which interfaces
  - when frame received, bridge “learns” location of sender, incoming LAN segment
  - records sender/location pair in bridge table

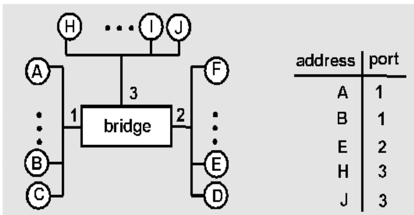
Filtering/Forwarding

When bridge receives a frame:

index bridge table using MAC destination address
if entry found for destination
  then
    if dest on segment from which frame arrived
      then drop the frame
      else forward the frame on interface indicated
  else
    forward on all but the interface on which frame arrived

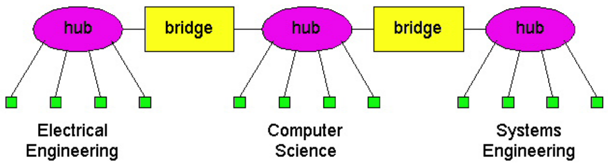
Bridge example

Suppose C sends frame to D and D replies back with frame to C.



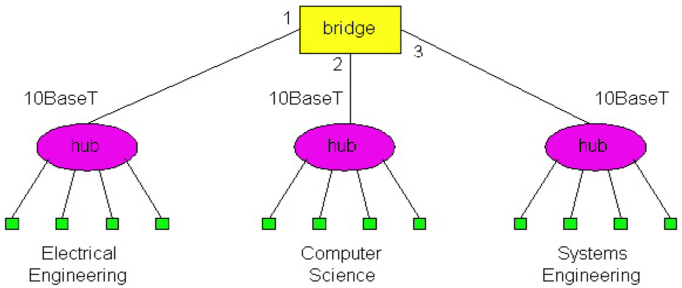
- Bridge receives frame from C
  - notes in bridge table that C is on interface 1
  - because D is not in table, bridge sends frame into interfaces 2 and 3
- frame received by D
- D generates frame for C, sends
- bridge receives frame
  - notes in bridge table that D is on interface 2
  - bridge knows C is on interface 1, so *selectively* forwards frame to interface 1

Interconnection without backbone



- Not recommended for two reasons:
  - single point of failure at Computer Science hub
  - all traffic between EE and SE must pass CS segment

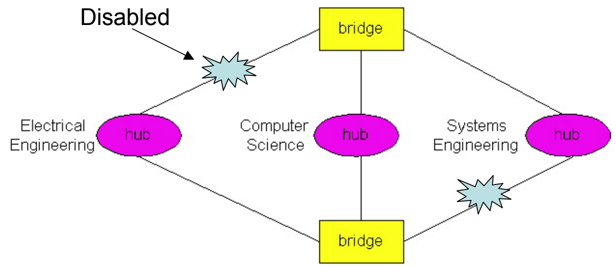
Backbone configuration



Recommended!

Bridges Spanning Tree

- for increased reliability, desirable to have redundant, alternative paths from source to destination
- with multiple paths, cycles result
  - bridges may multiply and forward frame forever
- solution: organize bridges in a spanning tree by disabling subset of interfaces



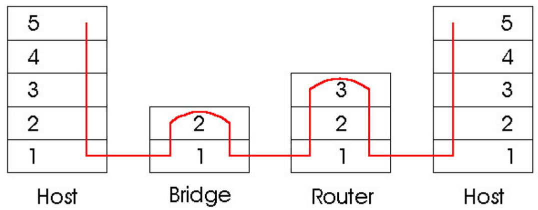
Some Bridge features

- Isolates collision domains resulting in higher total maximum throughput
- Limitless number of nodes and geographical coverage
- Can connect different Ethernet types
- Transparent (“plug-and-play”): no configuration necessary

- Switches
  - Remark: switches are essentially multi-port bridges.
  - What we say about bridges also holds for switches!

Bridges vs. Routers

- both store-and-forward devices
  - routers: network layer devices (examine network layer headers)
  - bridges are link layer devices
- routers maintain routing tables, implement routing algorithms
- bridges maintain bridge tables, implement filtering, learning and spanning tree algorithms



Bridges vs. Routers

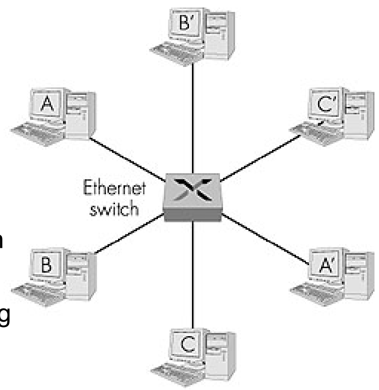
Bridges
+ Bridge operation is simpler requiring less packet processing
+ Bridge tables are self learning
– All traffic confined to spanning tree, even when alternative bandwidth is available
– Bridges do not offer protection from broadcast storms

Routers
+ arbitrary topologies can be supported, cycling is limited by TTL counters (and good routing protocols)
+ provide protection against broadcast storms
- require IP address configuration (not plug and play)
- require higher packet processing

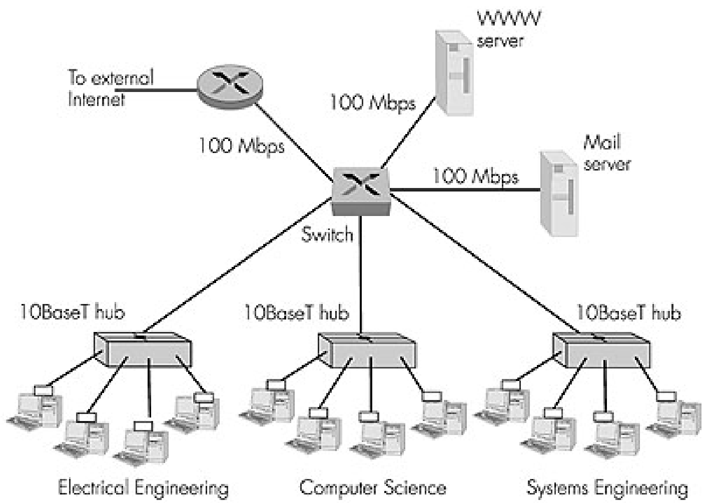
bridges do well in small (few hundred hosts) while routers used in large networks (thousands of hosts)

Ethernet Switch

- Essentially a multi-interface bridge
- Layer 2 (frame) forwarding, filtering using LAN addresses
- Switching: A-to-A’ and B-to-B’ simultaneously, no collisions
- Large number of interfaces
- Often
  – Hosts star-connected into switch
  – Ethernet, but no collisions!
  – cut-through switching: forwarding without waiting for entire frame
  – combinations of shared/dedicated, 10/100/1000 Mbps interfaces



Not an atypical LAN (IP network)



Summary comparison

	hubs	bridges	routers	switches
traffic isolation	no	yes	yes	yes
plug & play	yes	yes	no	yes
optimal routing	no	no	yes	no
cut through	yes	no	no	yes

Point to Point Data Link Control

- one sender, one receiver, one link: easier than broadcast link
  - no Media Access Control
  - no need for explicit MAC addressing
  - e.g., dialup link, ISDN line
- popular point-to-point DLC protocols
  - PPP (point-to-point protocol)
  - HDLC: High level data link control (Data link used to be considered “high layer” in protocol stack!)

PPP Design Requirements [RFC 1557]

- packet framing
  - encapsulation of network-layer datagram in data link frame
  - carry network layer data of any network layer protocol (not just IP) *at same time*
  - ability to demultiplex upwards
- bit transparency: must carry any bit pattern in the data field
- error detection (no correction)
- connection liveness: detect, signal link failure to network layer
- network layer address negotiation: endpoints can learn/configure each other’s network addresses
- No error correction/recovery, flow control, in-order delivery
  - all relegated to higher layers

PPP Data Frame

- Flag: delimiter (framing)
- Address: does nothing (only one option)
- Control: does nothing; in the future possible multiple control fields
- Protocol: upper layer protocol to which frame delivered (e.g. PPP-LCP, IP, IPCP, etc.)
- info: upper layer data being carried
- check: cyclic redundancy check for error detection

1	1	1	1 or 2	variable length	2 or 4	1
01111110	11111111	00000011	protocol	info	check	01111110
flag	address	control				flag

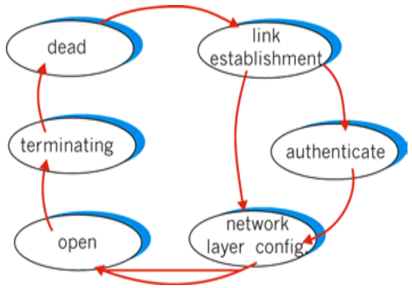
Byte Stuffing

- “data transparency” requirement:
  - data must be allowed to include flag pattern <01111110>
  - Question: is received <01111110> data or flag?
- Sender: adds (“stuffs”) extra <01111110> byte after each <01111110> *data* byte
- Receiver:
  - two 01111110 bytes in a row: discard first byte, continue data reception
  - single 01111110: that’s the flag byte

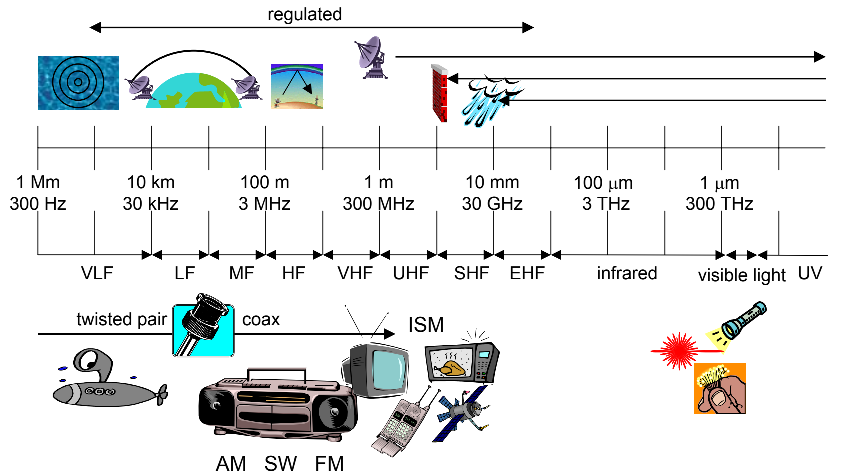
PPP Data Control Protocol

Before exchanging network-layer data, data link peers must

- configure PPP link
  - max. frame length
  - authentication
- learn/configure network layer information
  - for IP: carry IP Control Protocol (IPCP) msgs (protocol field: 8021) to configure/learn IP address



Physical Layer: Wireless Frequencies



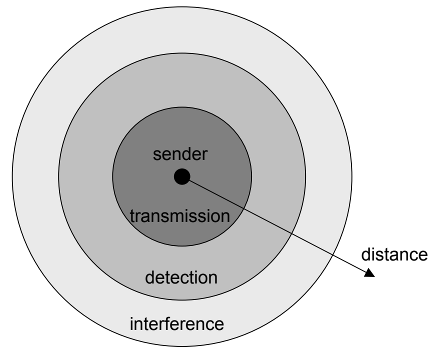
Frequencies and regulations

- ITU-R holds auctions for new frequencies, manages frequency bands worldwide (WRC, World Radio Conferences)

	Europe (CEPT/ETSI)	USA (FCC)	Japan
Mobile phones	**NMT** 453-457MHz, 463-467 MHz **GSM** 890-915 MHz, 935-960 MHz, 1710-1785 MHz, 1805-1880 MHz	**AMPS**, **TDMA**, **CDMA** 824-849 MHz, 869-894 MHz **TDMA**, **CDMA**, **GSM** 1850-1910 MHz, 1930-1990 MHz	**PDC** 810-826 MHz, 940-956 MHz, 1429-1465 MHz, 1477-1513 MHz
Cordless telephones	**CT1+** 885-887 MHz, 930-932 MHz **CT2** 864-868 MHz **DECT** 1880-1900 MHz	**PACS** 1850-1910 MHz, 1930-1990 MHz **PACS-UB** 1910-1930 MHz	**PHS** 1895-1918 MHz **JCT** 254-380 MHz
Wireless LANs	**IEEE 802.11** 2400-2483 MHz **HIPERLAN 1** 5176-5270 MHz	**IEEE 802.11** 2400-2483 MHz	**IEEE 802.11** 2471-2497 MHz

Signal propagation ranges

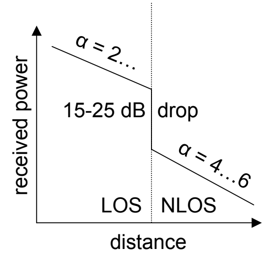
- Propagation in free space always like light (straight line)
- Transmission range
  - communication possible
  - low error rate
- Detection range
  - detection of the signal possible
  - no communication possible
- Interference range
  - signal may not be detected
  - signal adds to the background noise



Attenuation by distance

- Attenuation [dB] = 10 $\log_{10}$ (transmitted power / received power)
- Example: factor 2 loss = 10 $\log_{10} 2 \approx 3$ dB

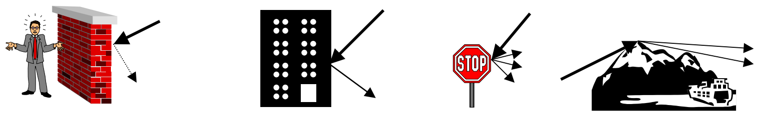
- In theory/vacuum (and for short distances), receiving power is proportional to $1/d^2$, where d is the distance.
- In practice (for long distances), receiving power is proportional to $1/d^\alpha$, $\alpha = 4\ldots6$. We call α the path loss exponent.



- Example: Short distance, what is the attenuation between 10 and 100 meters distance?
Factor 100 ($=100^2/10^2$) loss = 20 dB

Attenuation by objects

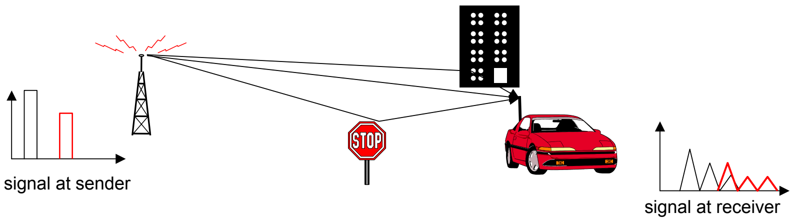
- Shadowing (3-30 dB):
  - textile (3 dB)
  - concrete walls (13-20 dB)
  - floors (20-30 dB)
- reflection at large obstacles
- scattering at small obstacles
- diffraction at edges
- fading (frequency dependent)



shadowing reflection scattering diffraction

Multipath propagation

- Signal can take many different paths between sender and receiver due to reflection, scattering, diffraction



- Time dispersion: signal is dispersed over time
- Interference with "neighbor" symbols: Inter Symbol Interference (ISI)
- The signal reaches a receiver directly and phase shifted
- Distorted signal depending on the phases of the different parts