Chapter 5: DataLink Layer

Course on Computer Communication and Networks, CTH/GU

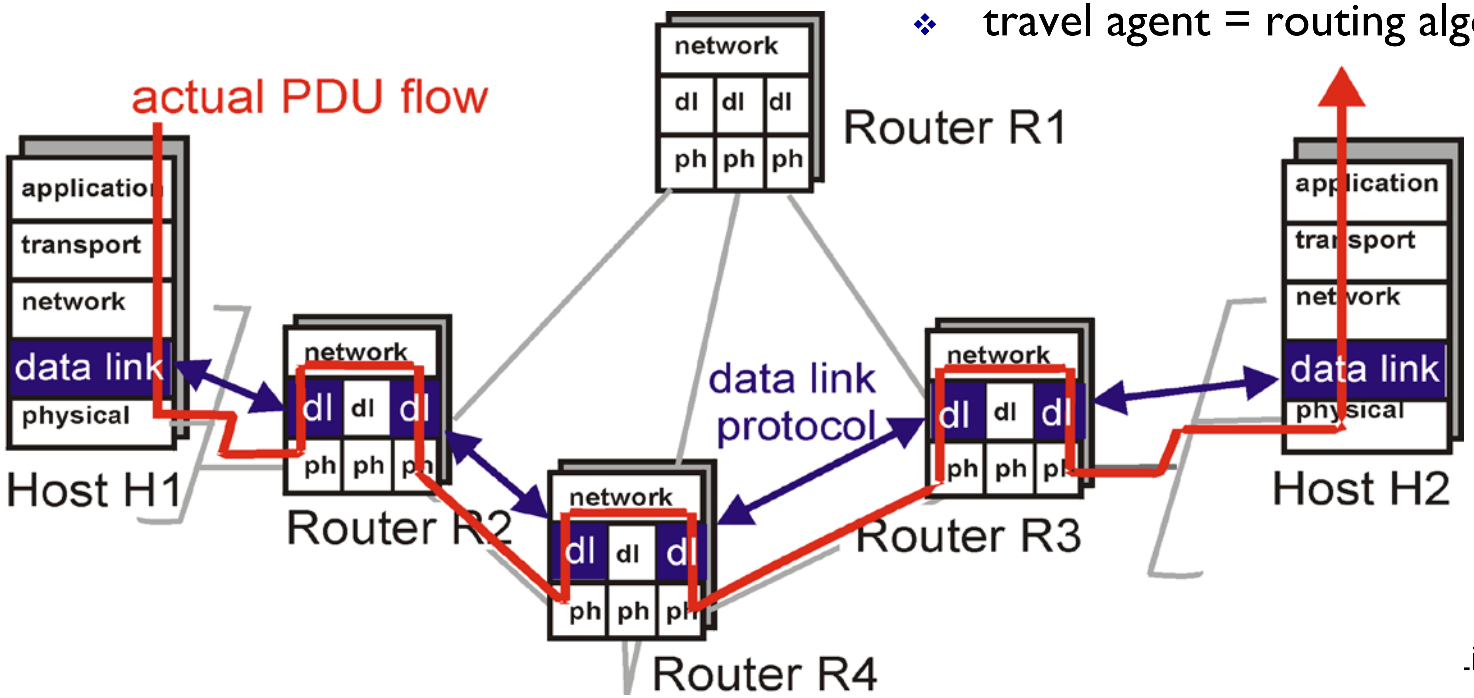
The slides are adaptation of the slides made available by the authors of the course's main textbook

Slides with darker background are for extra information or background/context

Link layer: context

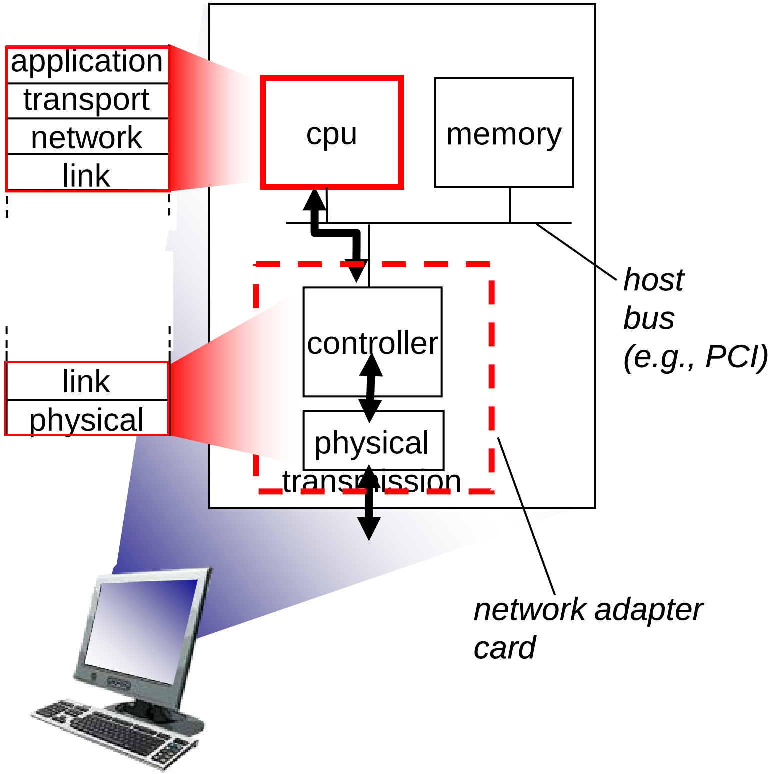
- Datagram transferred by different link protocols over different links:
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- Each link protocol provides different services
  - e.g., may or may not provide RDT over link

- <u>transportation analogy</u>
- trip from Princeton to Lausanne
  - limo: Princeton to JFK
  - plane: JFK to Geneva
  - train: Geneva to Lausanne
- tourist = datagram
- transport segment = communication link
- transportation mode = link layer protocol
- travel agent = routing algorithm

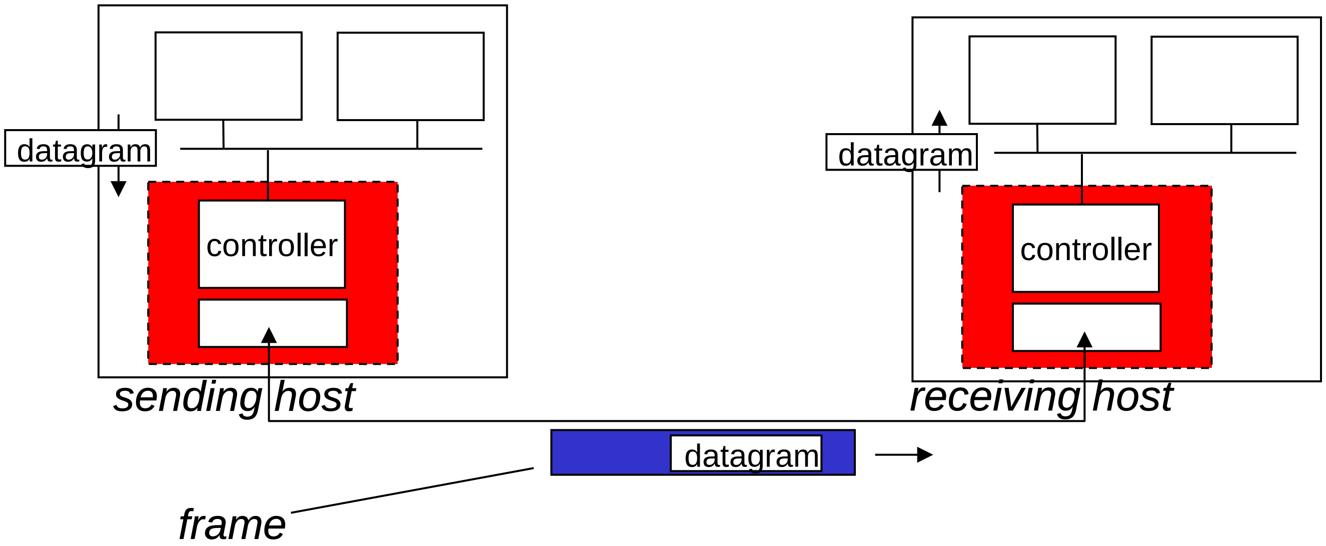


Where is the link layer implemented?

- in each and every host
- link layer implemented in "adapter" (aka *network interface card* NIC) or on a chip
  - Ethernet card, 802.11 card; Ethernet chipset
  - implements link, physical layer
- attaches into host's system buses
- combination of hardware, software, firmware



Adapters communicating



- sending side:
  - encapsulates datagram in frame
  - adds error checking bits, RDT, flow control, etc.
- receiving side
  - looks for errors, RDT, flow control, etc
  - extracts datagram, passes to upper layer at receiving side

Link layer services

- *framing, link access:*
  - encapsulate datagram into frame (header, trailer)
    - Link-layer addresses in frame headers to identify source, dest
      - different from IP address!
  - channel access if shared medium
- *reliable delivery between adjacent nodes*
  - we learned how to do this already (chapter 3)!
    - seldom used on low bit-error link (fiber, some twisted pair)
  - wireless links: high error rates; error detection and correction applicable
- *error detection:*
  - receiver detect errors caused by signal attenuation, noise.
- *error correction:*
  - receiver identifies *and corrects* bit error(s) without resorting to retransmission
- *flow control:*
  - pacing between adjacent sending and receiving nodes

Link Layer

- 5.1 Introduction and services
- 5.3 Multiple access protocols

- (5.2 Error detection and correction)

- *grey items will be treated as complement, in subsequent lecture

<u>LAN technology</u>

- 5.4.2 Ethernet
- 5.4.3 Interconnection
- 5.4.1 Link-Layer Addressing

- 5.7 A day in the life of a web request
- 5.5 Link Virtualization: ATM and MPLS)

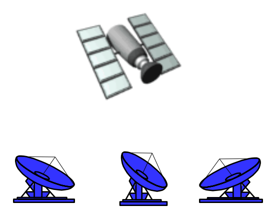
access links, protocols

two types of "links":

- point-to-point
  - PPP for dial-up access
  - point-to-point link between Ethernet switch, host
- *broadcast (shared wire or medium), eg*
  - old-fashioned Ethernet
  - 802.11 wireless LAN

shared wire (e.g., cabled Ethernet)

shared RF (e.g., 802.11 WiFi)



shared RF (satellite)

humans at a cocktail party (shared air, acoustical)

i.e. (Multiple access)

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
  - *collision* if node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
  - no out-of-band channel for coordination

An ideal multiple access protocol

given: broadcast channel of rate R bps

desired outcome:

1. when one node wants to transmit, it can send at rate R.
2. when M nodes want to transmit, each can send at average rate R/M
3. fully decentralized:
   - no special node to coordinate transmissions
   - no synchronization of clocks, slots
4. simple

MAC protocols: taxonomy

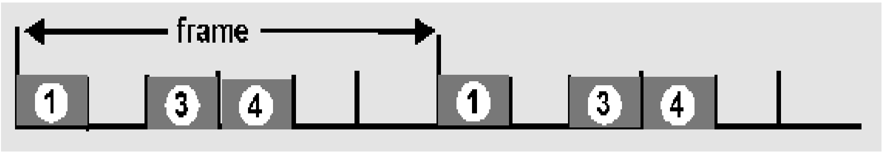
three broad classes:

- *channel partitioning*
  - divide channel into smaller "pieces" (time slots, frequency, code)
  - allocate piece to node for exclusive use
- *random access*
  - channel not divided, allow collisions
  - "recover" from collisions
- *"taking turns"*
  - nodes take turns, but nodes with more to send can take longer turns

Channel Partitioning MAC protocols: TDMA, FDMA

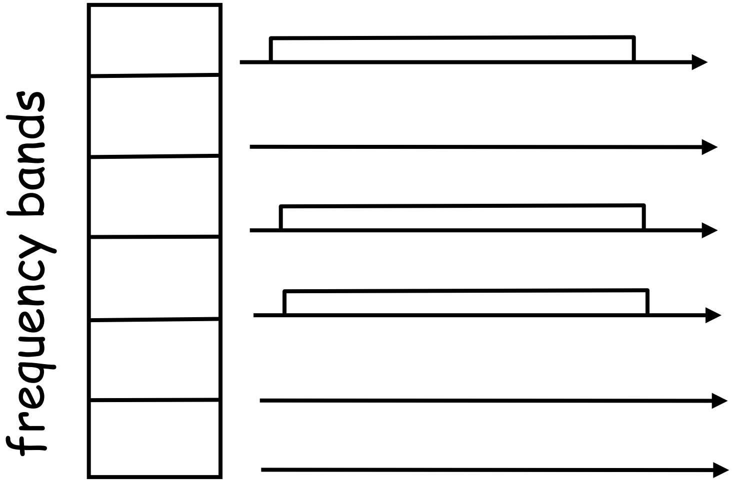
TDMA: time division multiple access

- access to channel in "rounds"
- each station gets fixed length slot (length = pkt trans time) in each round
- unused slots go idle
  - example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle



FDMA: frequency division multiple access

- each station assigned fixed frequency band
- unused transmission time in frequency bands goes idle
  - example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



Channel Partitioning CDMA

CDMA: Code Division Multiple Access

- allows each station to transmit over the entire frequency spectrum all the time.
- simultaneous transmissions are separated using coding theory.
- used mostly in wireless broadcast channels (cellular, satellite, etc) – we will study it in the wireless context
- has been ”traditionally” used in the military

Observe:

MUX = speak person-to-person in designated space

CDMA = ”shout” using different languages: the ones who know the language will get what you say

MAC protocols: taxonomy

three broad classes:

- *channel partitioning*
  - divide channel into smaller "pieces" (time slots, frequency, code)
  - allocate piece to node for exclusive use
- *random access*
  - channel not divided, allow collisions
  - "recover" from collisions
- *"taking turns"*
  - nodes take turns, but nodes with more to send can take longer turns

Random access protocols

- when node has packet to send
  - transmit at full channel data rate R.
  - no *a priori* coordination among nodes
- random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
  - slotted ALOHA
  - ALOHA
  - CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA

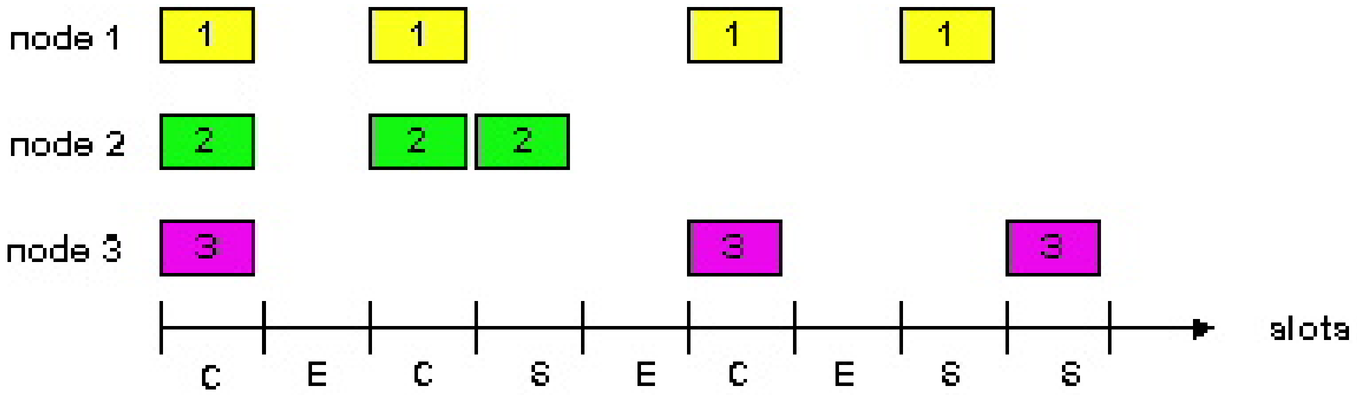
assumptions:

- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only at slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

operation:

- when node obtains fresh frame (from upper layer protocol), it transmits in next slot
  - *if no collision:* ok
  - *if collision:* node retransmits frame in each subsequent slot with prob. p until success

Slotted ALOHA



Pros

- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

Cons

- collisions, wasting slots
- idle slots
- clock synchronization

Slotted Aloha efficiency

Efficiency : long-run fraction of successful slots (many nodes, all with many frames to send)

Q: max fraction of successful transmissions?

A: Suppose N stations, each transmits in slot with probability p

- prob. successful transmission is:

P[specific node succeeds]= $p\,(1-p)^{(N-1)}$

P[any of N nodes succeeds]
$= N\,p\,(1-p)^{(N-1)}$

CSMA: Carrier Sense Multiple Access

CSMA: listen before transmit:

- If channel sensed busy, defer transmission
  - back-off, random interval
- If/when channel sensed idle:
  - p-persistent CSMA: transmit immediately with probability p; with probablility 1-p retry after random interval
  - non-persistent CSMA: transmit after random interval

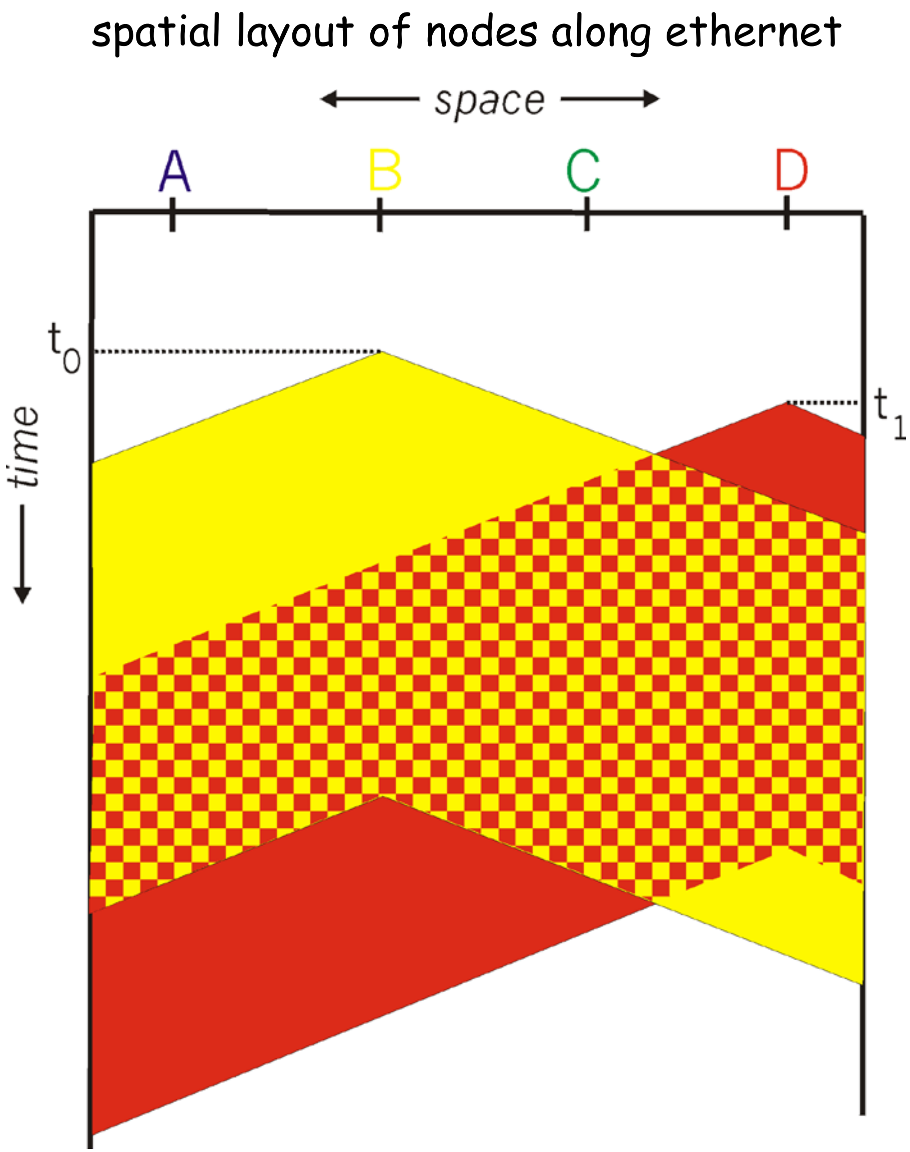
human analogy: don't interrupt others!

CSMA collisions

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

collision:
entire packet transmission time wasted

note:
role of distance and propagation delay (d)in determining collision (collision-detection delay <= 2d)



CSMA/CD (Collision Detection)

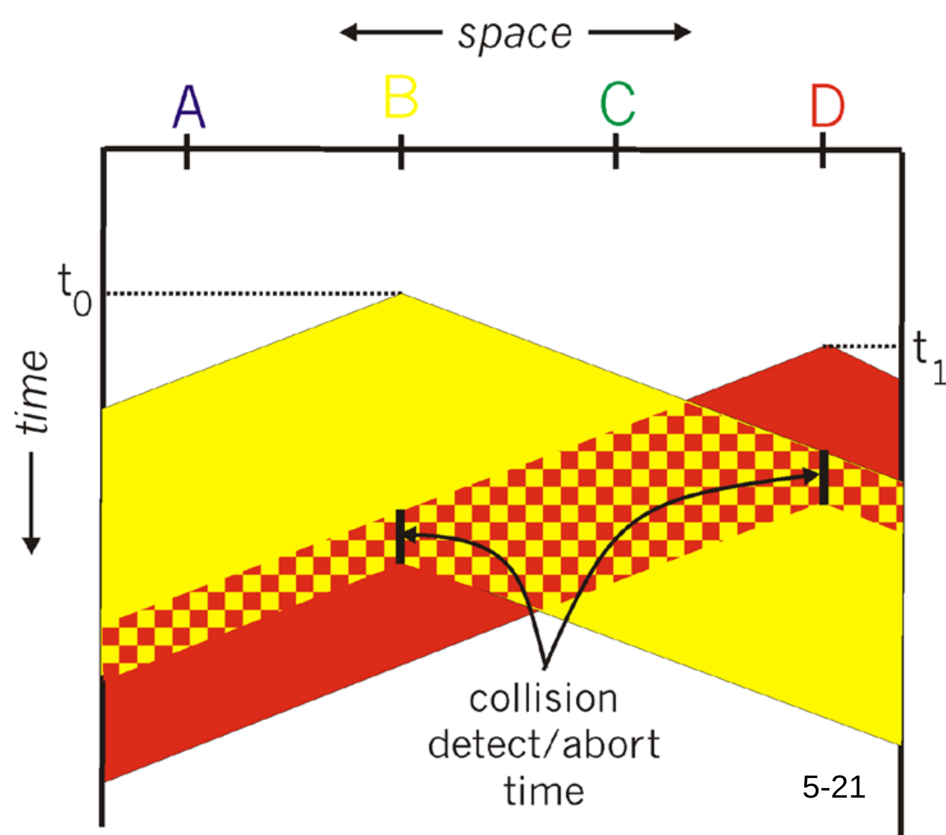
CSMA/CD: carrier sensing, deferral as in CSMA

- colliding transmissions aborted, reducing channel wastage
- persistent or non-persistent retransmission

collision detection:

- easy in wired LANs: measure signal strengths, compare transmitted, received signals
- different in wireless LANs: transmitter/receiver not "on" simultaneously; collision at the receiver matters, not the sender

human analogy: the polite conversationalist



MAC protocols: taxonomy

three broad classes:

- *channel partitioning*
  - divide channel into smaller "pieces" (time slots, frequency, code)
  - allocate piece to node for exclusive use
- *random access*
  - channel not divided, allow collisions
  - "recover" from collisions
- *"taking turns"*
  - nodes take turns, but nodes with more to send can take longer turns

Trade-off in MAC:

channel partitioning MAC protocols:

- share channel efficiently and fairly at high load
- inefficient at low load: delay in channel access, bandwidth allocated even if only 1 active node!

Random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

"taking turns" protocols

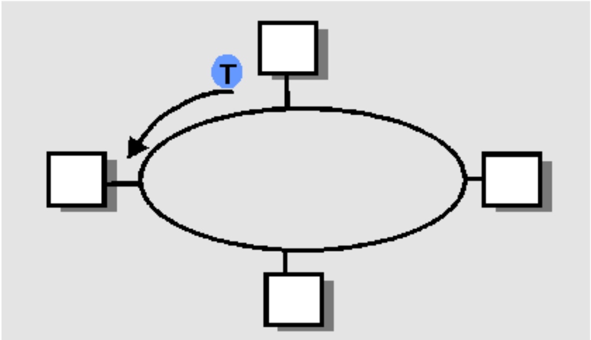
look for best of both worlds!

"Taking Turns" MAC protocols

Token passing:

- control **token-frame** passed from one node to next sequentially.
- not pure broadcast
- concerns:
  - token overhead
  - latency
  - single point of failure (token)

other: token bus, take-turns + reservation; see extra slides @ end of lecture



Summary of MAC protocols

- What do you do with a shared media?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Frequency Division
  - 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
    - *CSMA/CA used in 802.11 (to be studied in wireless)*
  - Taking Turns
    - polling, token passing
    - Bluetooth, FDDI, IBM Token Ring

Link Layer

- 5.1 Introduction and services
- 5.3Multiple access protocols

- (5.2 Error detection and correction)

- *grey items will be treated as complement, in subsequent lecture

<u>LAN technology</u>

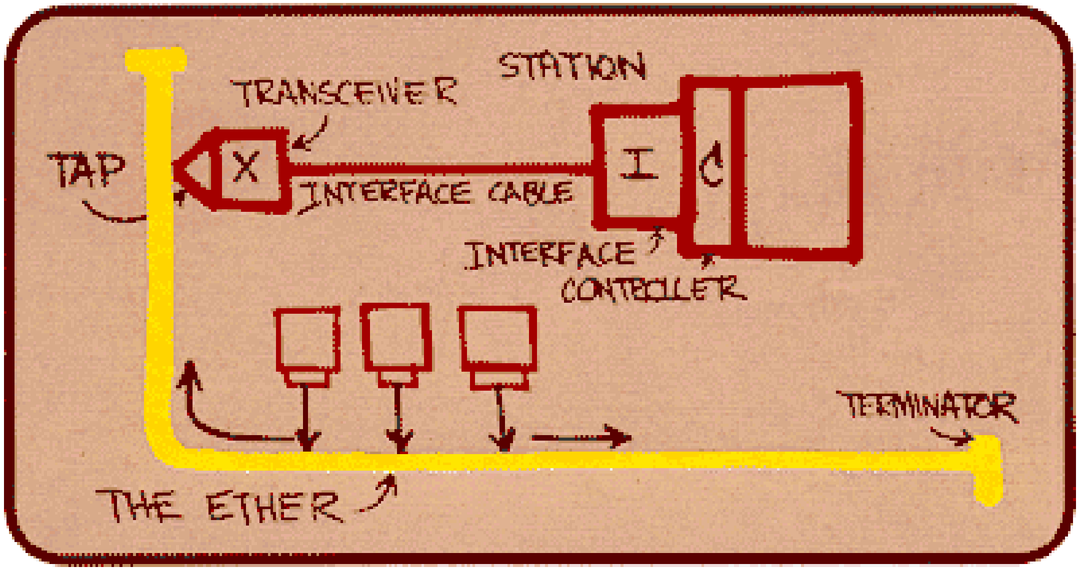
- 5.4.2 Ethernet
- 5.4.3 Interconnection
- 5.4.1 Link-Layer Addressing

- 5.7 A day in the life of a web request
- 5.5 Link Virtualization: ATM and MPLS)

Ethernet

"dominant" wired LAN technology:

- cheap $20 for 100Mbs!
- first widely used LAN technology
- Simpler, cheaper than token LANs and ATM
- Kept up with speed race: 10 Mbps – 100 Gbps



Metcalfe's Ethernet sketch

Ethernet: uses CSMA/CD

```
A: sense channel, if idle
    then {
            transmit and monitor the channel;
            If detect another transmission
              then {
                abort and send jam signal;
                update # collisions;
                delay as required by exponential backoff algorithm;
                goto A
                }
              else {done with the frame; set collisions to zero}
        }
    else {wait until ongoing transmission is over and goto A}
```

Ethernet's CSMA/CD (more)

Jam Signal: make sure all other transmitters are aware of collision; 48 bits;

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 x frame-transmission time)
- after m (<10) collisions: choose K from {0,…, 2^m}…
- after ten or more collisions, choose K from {0,1,2,3,4,…,1023}

Ethernet (CSMA/CD) Limitation

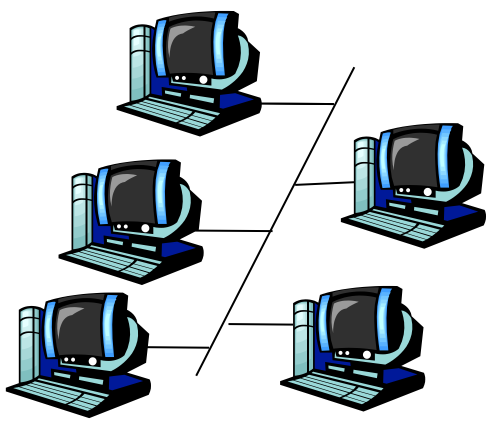
- Recall: collision detection interval = 2*Propagation delay along the LAN
- This implies a minimum frame size and/or a maximum wire length

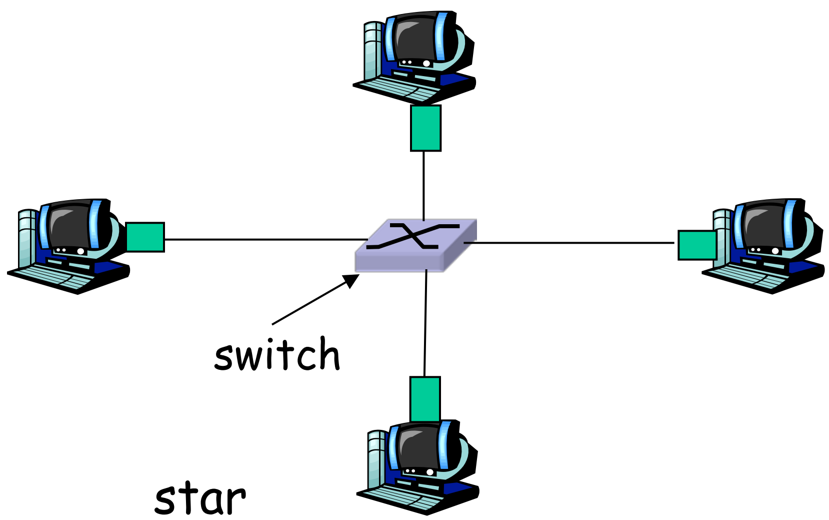
Critical factor:

$$a = 2 * propagation\_delay / frame\_transmission\_delay$$

Star topology

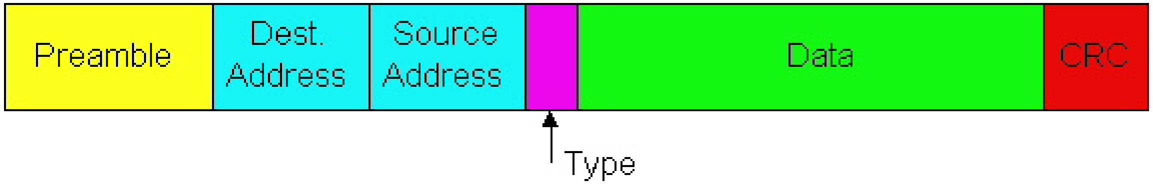
- bus topology popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- today: star topology prevails (more bps, shorter distances)
  - Hub or active *switch* in center
  - (more in a while)





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 one byte with pattern 10101011

- to synchronize receiver and sender clock rates

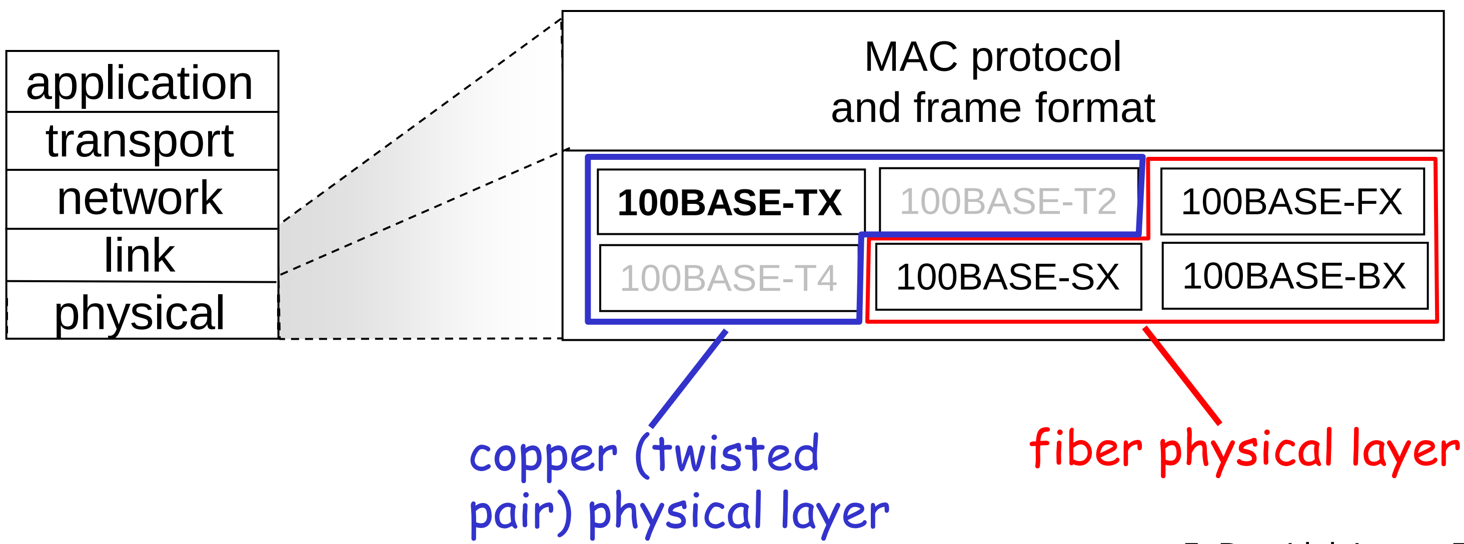
Addresses: 6 bytes, frame is received by all adapters on a LAN and dropped if address does not match

Type: indicates the higher layer protocol, mostly IP but others may be supported

CRC: checked at receiver, if error is detected, the frame is simply dropped

802.3 Ethernet Standards: Link & Physical Layers

- *many* different Ethernet standards
  - common MAC protocol and frame format
  - different speeds: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10G bps
  - different physical layer media: fiber, cable



Ethernet: Unreliable, connectionless

- connectionless: No handshaking between sending and receiving NICs
- unreliable: receiving NIC doesn't send acks or nacks to sending NIC
  - stream of datagrams passed to network layer can have gaps (missing datagrams)
  - gaps will be filled if app is using TCP
  - otherwise, app will see gaps

Link Layer

- 5.1 Introduction and services
- 5.3Multiple access protocols

LAN technology

- 5.4.2 Ethernet
- 5.4.3 Interconnection
- 5.4.2 Link-Layer Addressing

- (5.2 Error detection and correction)

- 5.7 A day in the life of a web request
- 5.5 Link Virtualization: ATM and MPLS)

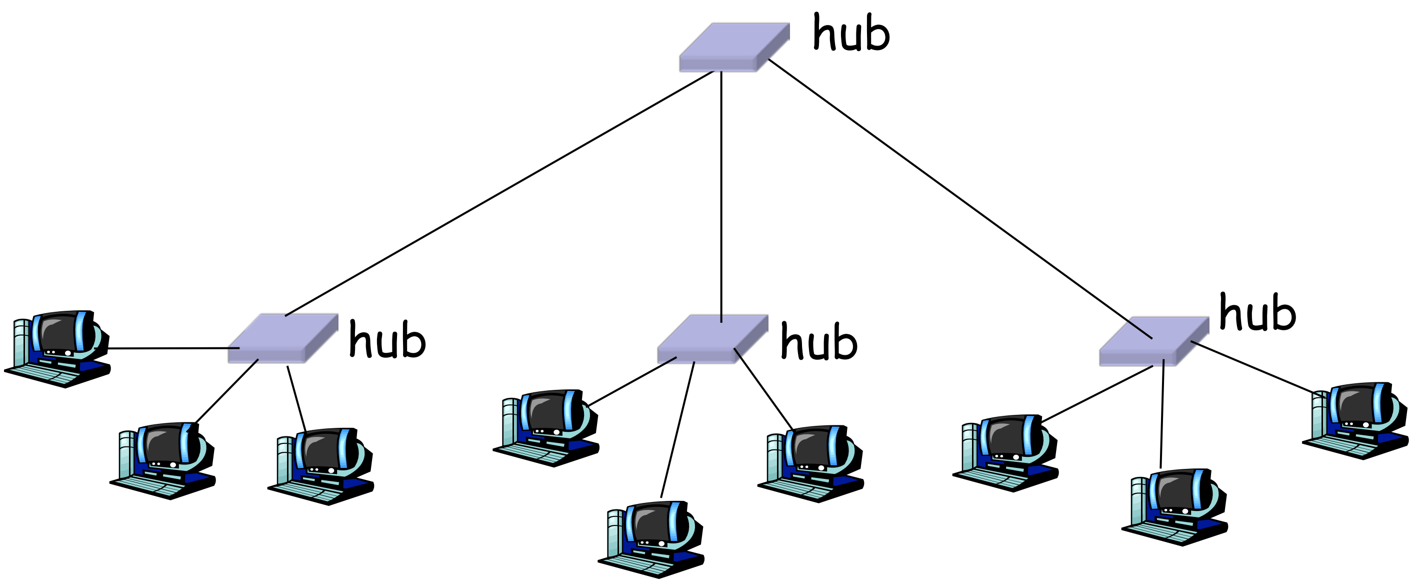
- *grey items will be treated as complement, in subsequent lecture

Interconnecting with hubs

Hubs are essentially physical-layer repeaters:

- bits coming from one link go out all other links
- at the same rate (no frame buffering)

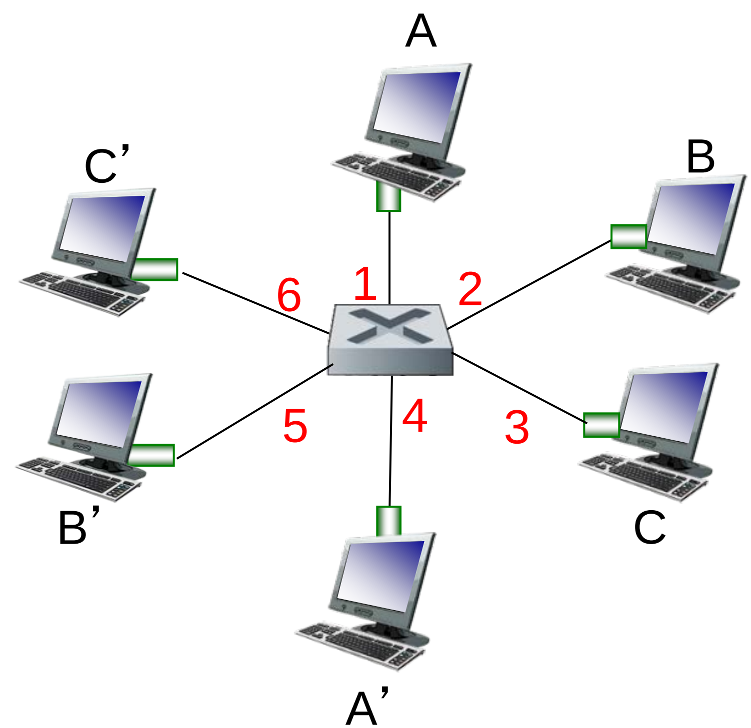
- no CSMA/CD at hub: adapters detect collisions (one large collision domain)
- Extends distance between nodes
- Can't interconnect different standards, e.g. 10BaseT & 100BaseT



- http://www.youtube.com/watch?v=reXS_e3fTAk&feature=related (video link)

Switch: *multiple* simultaneous transmissions

- switches buffer packets
- Ethernet protocol used on *each* incoming link, but no collisions; full duplex
  - each link is its own collision domain
- *switching:* A-to-A' and B-to-B' can transmit simultaneously, without collisions



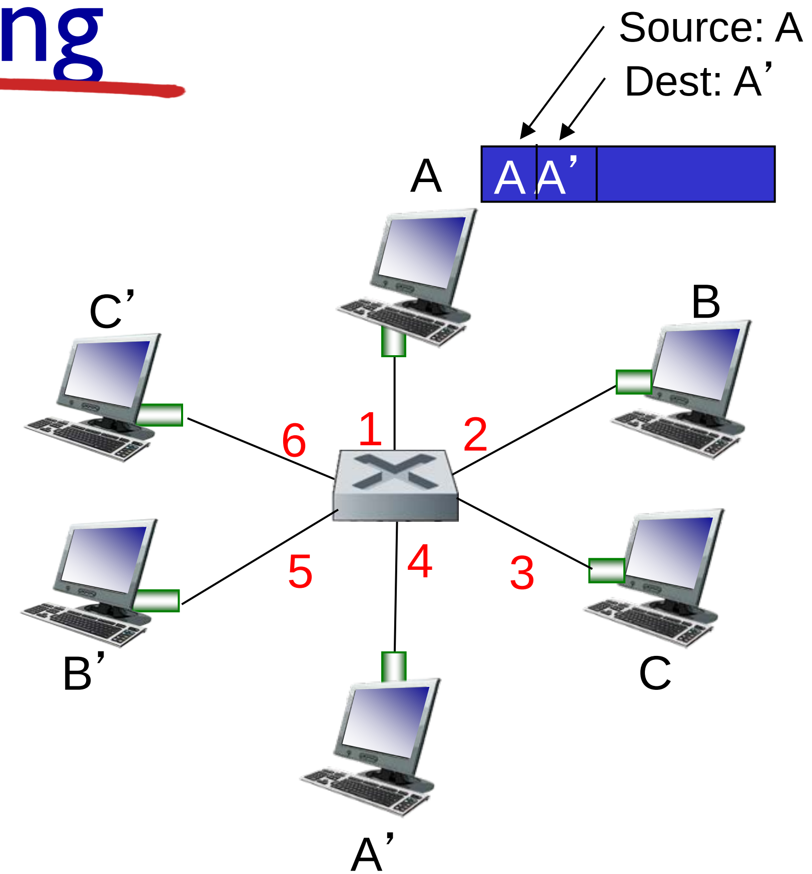
switch with six interfaces (1,2,3,4,5,6)

forwarding: how to know LAN segment on which to forward frame?

- looks like a routing problem…

Switch: self-learning

- switch *learns* which hosts can be reached through which interfaces
  - when frame received, switch "learns" location of sender: incoming LAN segment
  - records sender/location pair in switch table



MAC addr	interface	TTL
A	*1*	*60*

Switch table (initially empty)

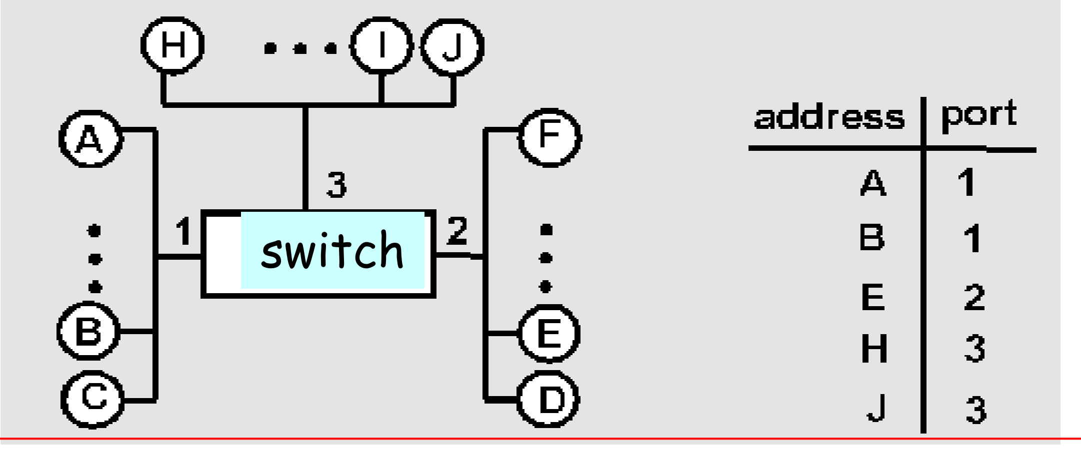
Switch: frame filtering/forwarding

when frame received at switch:

1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if entry found for destination
   then {
   if destination on segment from which frame arrived
   then drop frame
   else forward frame on interface indicated by entry
   }
   else flood /* forward on all interfaces except arriving interface */

Switch Learning: example

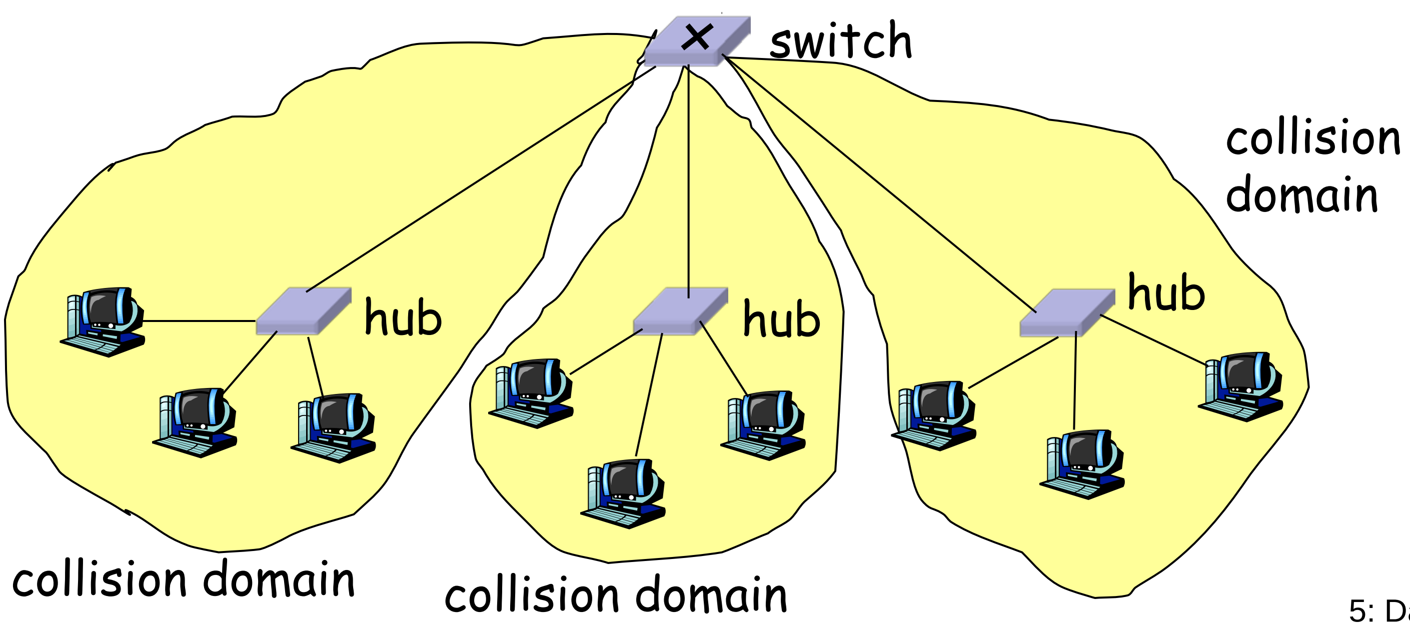
Suppose C sends a frame to D and D replies with a frame to C



- C sends frame, switch has no info about D, so floods
  - switch notes that C is on port 1
  - frame ignored on upper LAN
  - frame received by D
- D generates reply to C, sends
  - switch sees frame from D
  - switch notes that D is on interface 2
  - switch knows C on interface 1, so *selectively* forwards frame out via interface 1

Switch: traffic isolation

- switch installation breaks subnet into LAN segments
- switch filters packets:
  - same-LAN-segment frames not usually forwarded onto other LAN segments
  - segments become separate collision domains



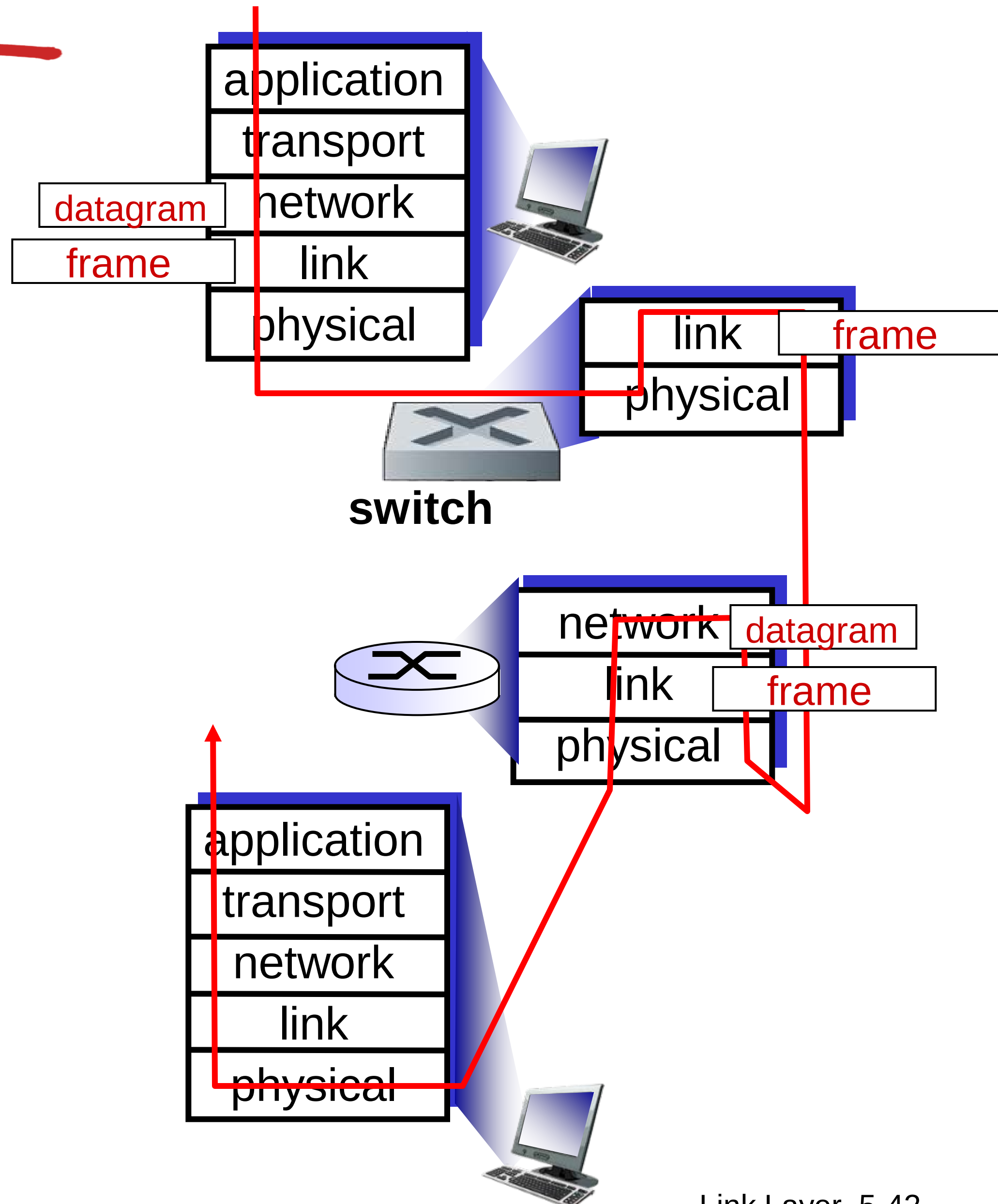
Switches vs. routers

both are store-and-forward:

- *routers:* network-layer devices (examine network-layer headers)
- *switches*: link-layer devices (examine link-layer headers)

both have forwarding tables:

- *routers:* compute tables using routing algorithms, IP addresses
- *switches:* learn forwarding table using flooding, learning, MAC addresses



Summary comparison

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

Link Layer

- 5.1 Introduction and services
- 5.3Multiple access protocols

LAN technology

- 5.4.2 Ethernet
- 5.4.3 Interconnection
- 5.4.1 Link-Layer Addressing

- (5.2 Error detection and correction)

- 5.7 A day in the life of a web request
- 5.5 Link Virtualization: ATM and MPLS)

- *grey items will be treated as complement, in subsequent lecture

LAN Addresses

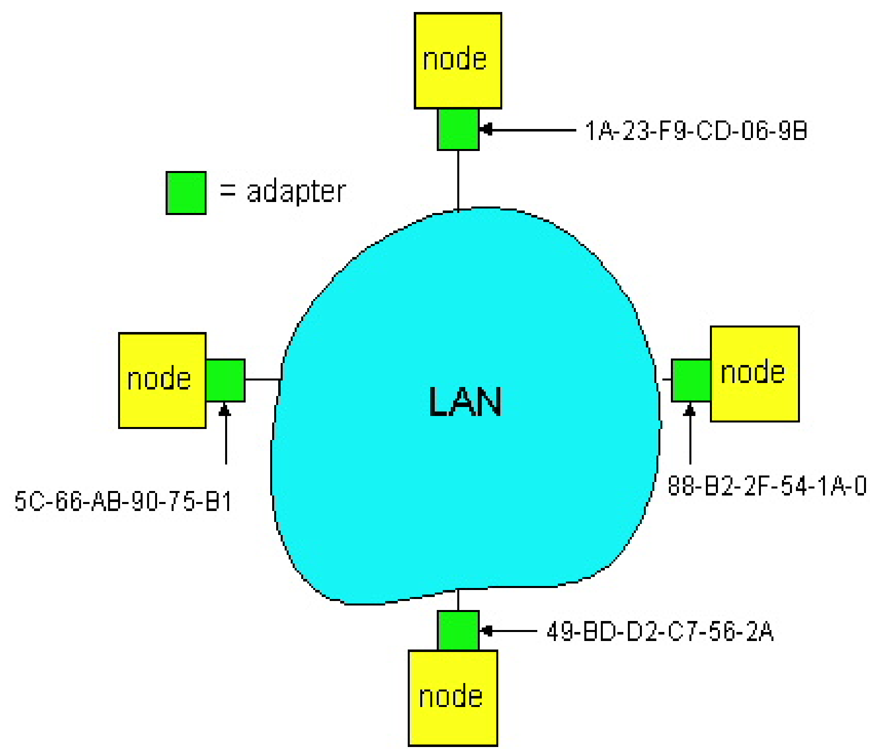
32-bit IP address:

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

LAN (or MAC or physical) address:

- to get datagram from one interface to another physically-connected interface (same network)
- 48 bit MAC address (for most LANs) burned in NIC's ROM (sometimes configurable)

Broadcast address = FF-FF-FF-FF-FF-FF



LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)

Analogy:

(a) MAC address: like People's Names or PersonalNum's

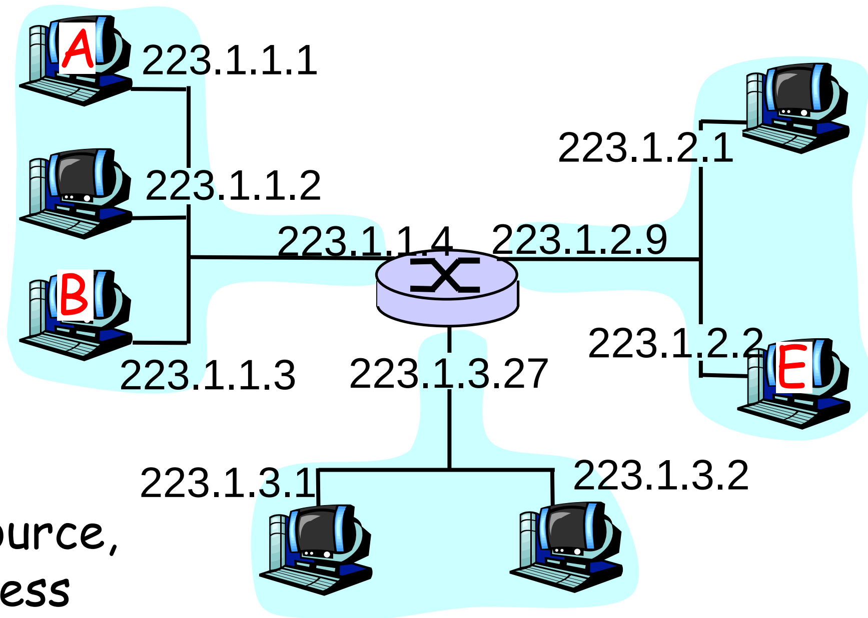
(b) IP address: like postal address

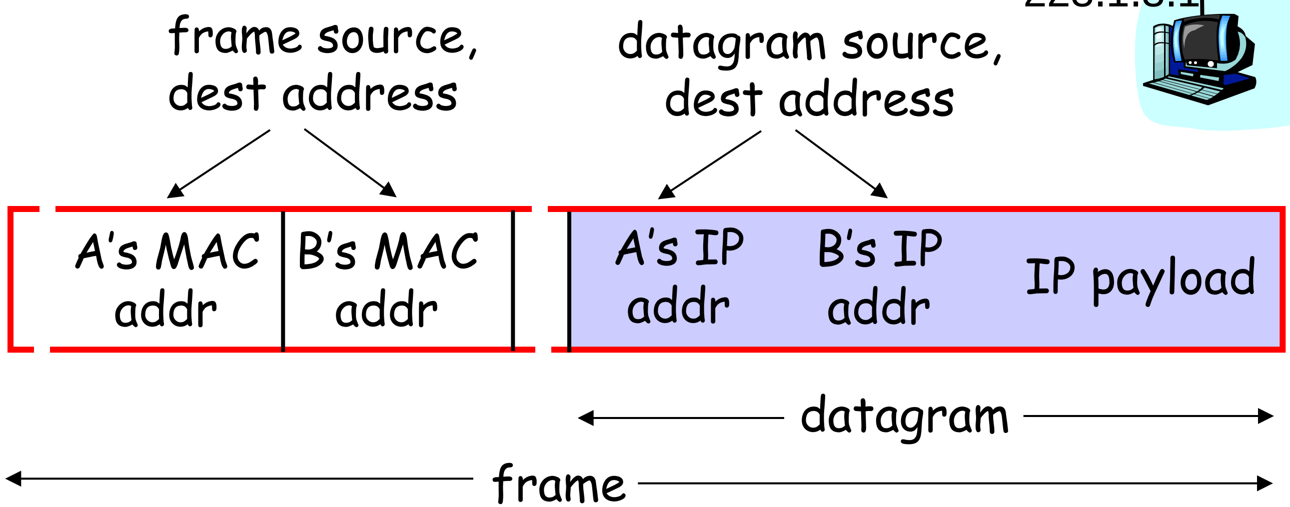
- MAC flat address => portability
  - can move LAN card from one LAN to another
- IP hierarchical address NOT portable
  - depends on network to which one attaches

Recall earlier routing discussion

Starting at A, given IP datagram addressed to B:

- look up net. address of B, find B on same net. as A
- link layer sends datagram to B inside link-layer frame

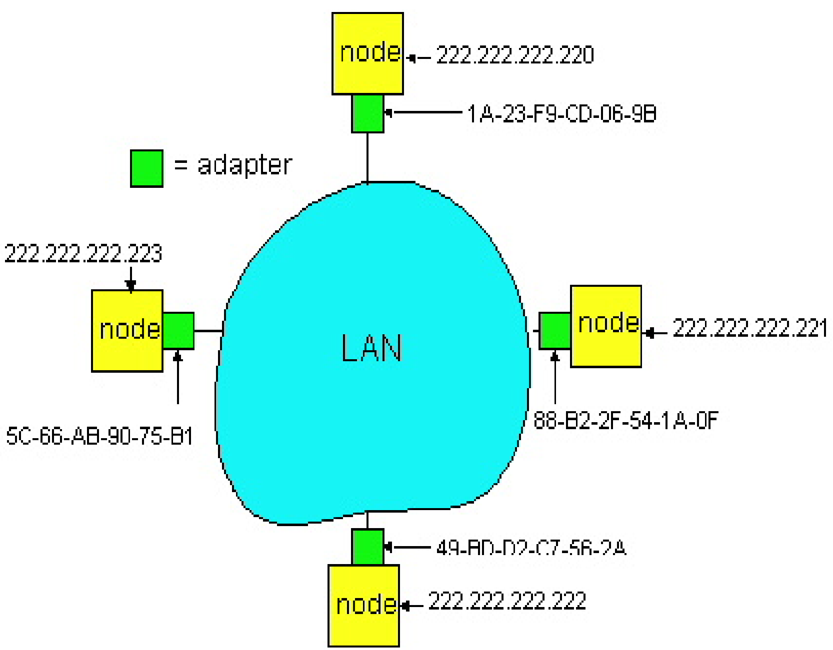




ARP: Address Resolution Protocol

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

Broadcast address = FF-FF-FF-FF-FF-FF



- Each IP node (Host, Router) on LAN has ARP table
  - ARP Table: IP/MAC address mappings

    < IP address; MAC address; TTL>

    < ……………………….. >

    - TTL (Time To Live): time to cache (typically 20 min); afterwards:

A broadcasts ARP query pkt, containing B's IP address

B receives ARP packet, replies to A with its (B's) physical layer address

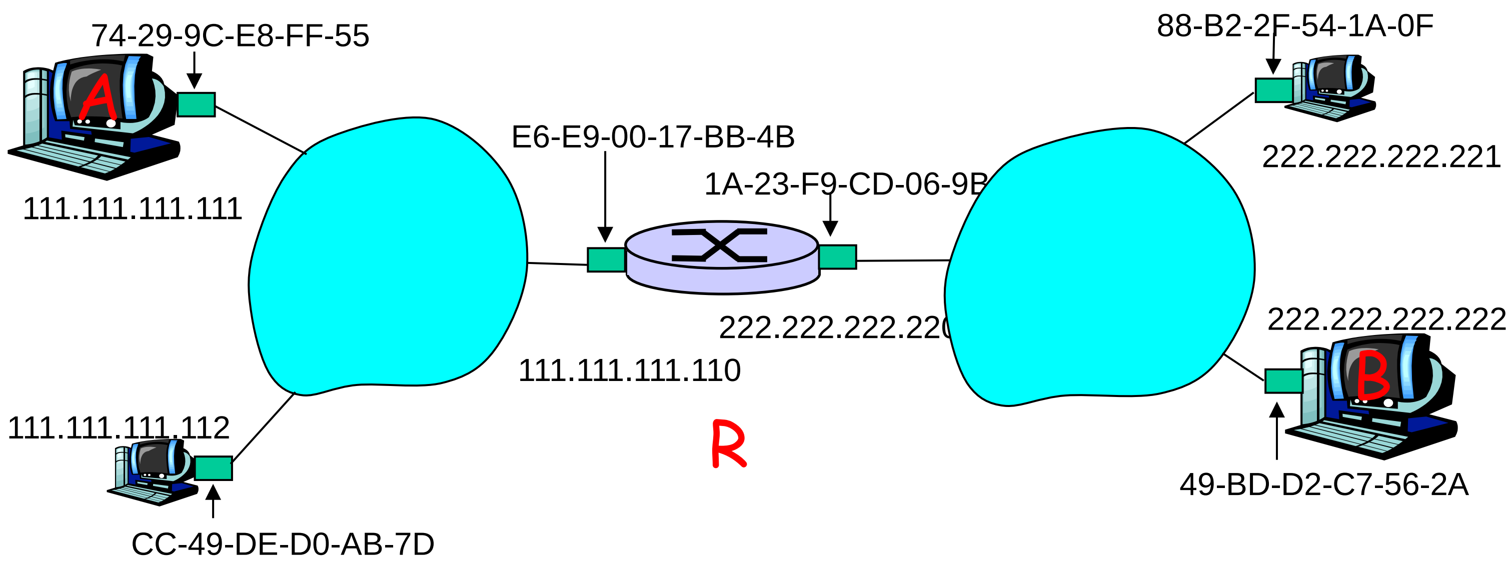
A caches (saves) IP-to-physical address pairs until they time out

- soft state: information that times out (goes away) unless refreshed

Addressing: routing to another LAN

walkthrough: send datagram from A to B via R

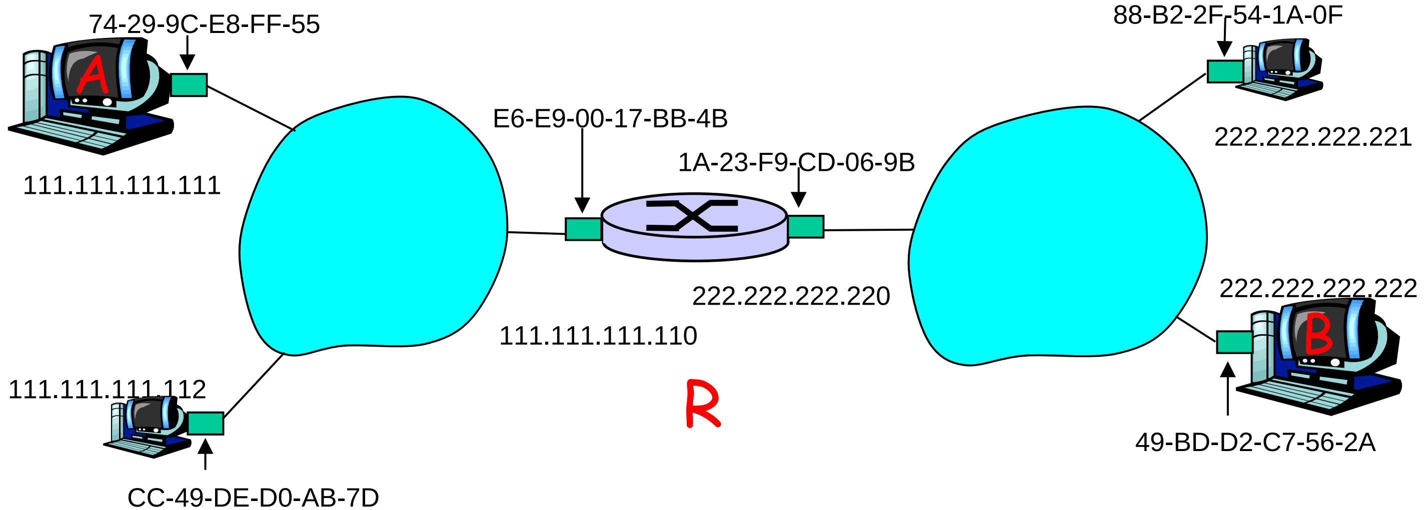
assume A knows B's IP address



- two ARP tables in router R, one for each IP network (LAN)

- A creates IP datagram with source A, destination B
  - Network layer finds out I should be forwarded to R
- A uses ARP to get R's MAC address for 111.111.111.110
- A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram
- A's NIC sends frame
- R's NIC receives frame
- R removes IP datagram from Ethernet frame, sees its destined to B
- R uses ARP to get B's MAC address
- R creates frame containing A-to-B IP datagram; sends to B

This is a really important example – make sure you understand!



Link Layer

- 5.1 Introduction and services
- 5.3Multiple access protocols
- (5.2 Error detection and correction)
- *grey items will be treated as complement, in subsequent lecture

<u>LAN technology</u>

- 5.4.2 Ethernet
- 5.4.3 Interconnection
- 5.4.1 Link-Layer Addressing
- 5.7 A day in the life of a web request
- 5.5 Link Virtualization: ATM and MPLS)

Review questions for this part

- Why both link-level and end-end reliability?
- Medium access methods: how they work, pros and cons
  - Partitioning
  - Random access
  - Reservation
- Aloha vs CSMA/CD
- Ethernet: protocol, management of collisions, connections
- Switches vs routers
- Addressing in link layer

EXTRA SLIDES/TOPICS

IEEE 802.4 Standard
(General Motors Token Bus)

(not in must-study material)

Contention systems limitation: worst-case delay until successful transmission is unlimited => not suitable for real-time traffic

Solution: token-passing, round robin

- *token* = special control frame; only the holding station can transmit; then it passes it to another station, i.e. for token bus, the next in the logical ring
- 4 priority classes of traffic, using timers
- Logical ring-maintenance: distributed strategy
  - Robust, somehow complicated though

IEEE Standard 802.5 (Token Ring)

(not in must-study material)

Motivation: instead of complicated token-bus, have a physical ring

Principle: Each bit arriving at an interface is copied into a 1-bit buffer (inspected and/or modified); then copied out to the ring again.

- copying step introduces a 1-bit delay at each interface.

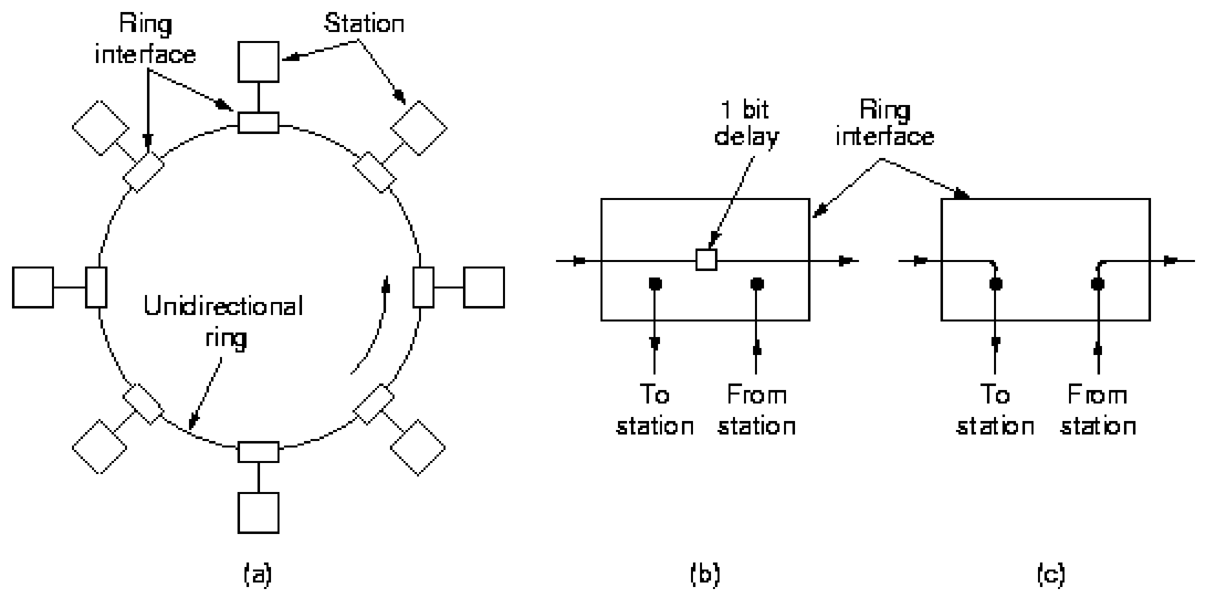


Fig. 4-28. (a) A ring network. (b) Listen mode. (c) Transmit mode.

Token Ring operation

- to transmit a frame, a station is required to seize the token and remove it from the ring before transmitting.
- bits that have propagated around the ring are removed from the ring by the sender (the receiver in FDDI).
- After a station has finished transmitting the last bit of its frame, it must regenerate the token.

IEEE 802.5 Ring: Maintenance
(not in must-study material)

Centralised: a "monitor" station oversees the ring:

- generates token when lost
- cleans the ring when garbled/orphan frames appear

If the monitor goes away, a convention protocol ensures that another station is *elected* as a monitor (e.g. the one with highest identity)

If the monitor gets "mad", though…..

IEEE 802.5 Ring: Priority Algorithm
(not in must-study material)

Station S

upon arrival of frame f:

set prior(f) := max{prior(f), prior(S)}

forward(f)

upon arrival of T

if prior(T)>prior(S) then forward(T)

else send own frame f with prior(f):=0

wait until f comes back

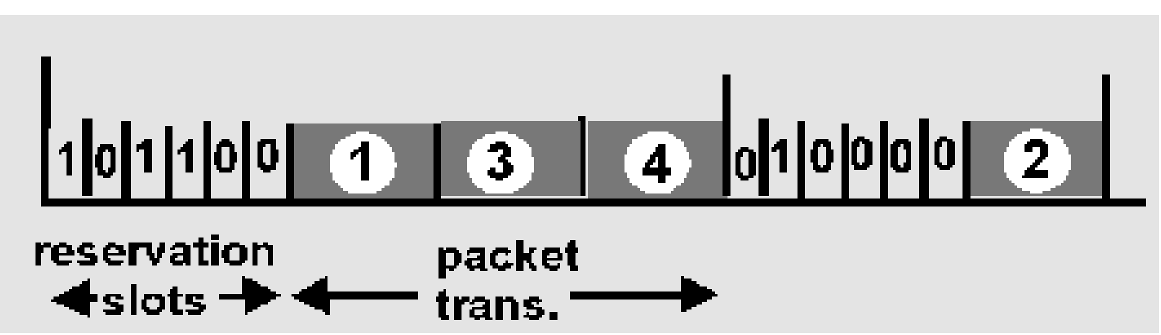
prior(T):=prior(f)

forward(T)

Reservation-based protocols

Distributed Polling – Bit-map protocol:

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



Switches (bridges): cont.

- Link Layer devices: operate on frames, examining header and selectively forwarding frame based on its destination
  - filtering: same-LAN-segment frames not forwarded to other seg's
- Advantages:
  - Isolates collision domains:
    - higher total max throughput
    - no limit on number of nodes nor distances
  - Can connect different net-types (translational, …)
  - Transparent: no need for any change to hosts LAN adapters

forwarding: how to know LAN segment on which to forward frame?
  - looks like a routing problem…