

CS 457 - Lecture 24

Multiple Access Protocols

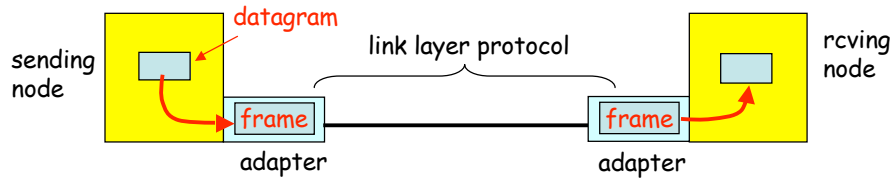
Link Layer Errors and Addressing

Spring 2008

Review

- Chapter 1
 - Throughput, Loss, and Delay
- Chapter 2
 - Application Protocol Concepts, HTTP, FTP, Email, DNS, P2P, Socket Programming
- Chapter 3
 - Multiplexing/Demultiplexing
 - Unreliable Transport (UDP)
 - Reliable Transport (stop and wait, GoBackN, Selective Repeat)
 - TCP: Reliable Delivery, Flow Control, Fairness, Throughput, Delay
 - TCP Congestion Control (AIMD, Slow Start, Fast Recovery, Thresholds)
- Chapter 4
 - Network layer, **virtual circuits, forwarding vs routing, longest match**
 - **Router Architectures** and IP basics (**fragmentation, addressing**)
 - **IP addressing, NAT, ICMP, IPv6**
 - ***Routing algorithms (link state and distance vector)***
 - ***Internet Routing: RIP and OSPF and BGP Routing Protocols***
 - Broadcast and **Multicast**
- Chapter 5
 - Link layer and **Multiple Access Protocols**
 - Link layer error detection and addressing

Adaptors Communicating

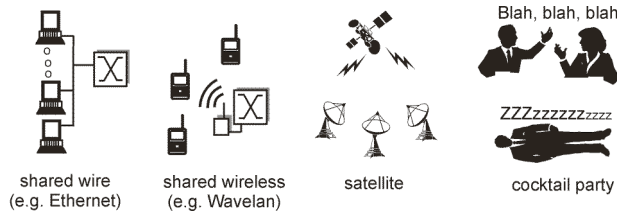


- link layer implemented in “adaptor” (aka NIC)
 - Ethernet card, PCMCIA card, 802.11 card
- sending side:
 - encapsulates datagram in a frame
 - adds error checking bits, rdt, flow control, etc.
- receiving side
 - looks for errors, rdt, flow control, etc
 - extracts datagram, passes to rcving node
- adapter is semi-autonomous
- link & physical layers

Multiple Access Links and Protocols

Two types of “links”:

- point-to-point
 - PPP for dial-up access
 - point-to-point link between Ethernet switch and host
- **broadcast** (shared wire or medium)
 - traditional Ethernet
 - upstream HFC
 - 802.11 wireless LAN



Multiple Access protocols

- 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

Ideal Multiple Access Protocol

Broadcast channel of rate R bps

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: a taxonomy

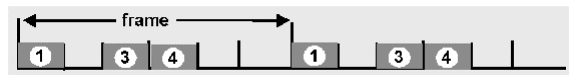
Three broad classes:

- **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

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

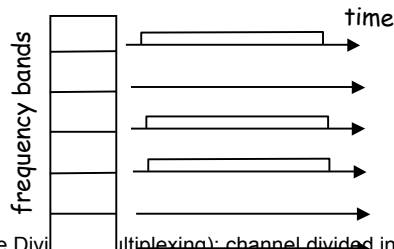


- TDM (Time Division Multiplexing): channel divided into N time slots, one per user; inefficient with low duty cycle users and at light load.
- FDM (Frequency Division Multiplexing): frequency subdivided.

Channel Partitioning MAC protocols: FDMA

FDMA: frequency division multiple access

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



- TDM (Time Division Multiplexing): channel divided into N time slots, one per user; inefficient with low duty cycle users and at light load.
- FDM (Frequency Division Multiplexing): frequency subdivided.

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 (e.g., via delayed retransmissions)
- Examples of random access MAC protocols:
 - slotted ALOHA
 - ALOHA
 - CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA

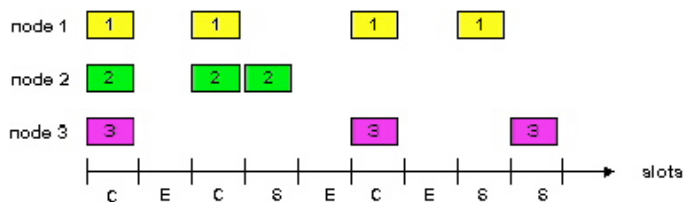
Assumptions

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

Operation

- when node obtains fresh frame, it transmits in next slot
- no collision, node can send new frame in next slot
- 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
- nodes may be able to detect collision in less than time to transmit packet
- clock synchronization

Slotted Aloha efficiency

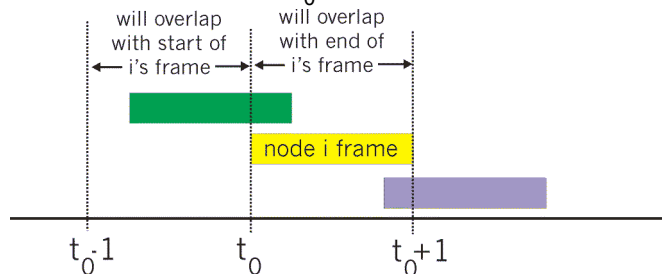
Efficiency is the long-run fraction of successful slots when there are many nodes, each with many frames to send

- Suppose N nodes with many frames to send, each transmits in slot with probability p
- prob that node 1 has success in a slot = $p(1-p)^{N-1}$
- prob that any node has a success = $Np(1-p)^{N-1}$
- For max efficiency with N nodes, find p^* that maximizes $Np(1-p)^{N-1}$
- For many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives $1/e = .37$

At best: channel used for useful transmissions 37% of time!

Pure (unslotted) ALOHA

- unslotted Aloha: simpler, no synchronization
- when frame first arrives
 - transmit immediately
- collision probability increases:
 - frame sent at t_0 collides with other frames sent in $[t_0 -$



Pure Aloha efficiency

P(success by given node) =

P(node transmits) ·

P(no other node transmits in $[p_0-1, p_0]$ ·

P(no other node transmits in $[p_0-1, p_0]$

$$= p \cdot (1-p)^{N-1} \cdot (1-p)^{N-1}$$

$$= p \cdot (1-p)^{2(N-1)}$$

... choosing optimum p and then letting n → ∞ ...

$$= 1/(2e) = .18$$

Even worse !

CSMA (Carrier Sense Multiple Access)

CSMA: listen before transmit:

If channel sensed idle: transmit entire frame

- If channel sensed busy, defer transmission

- Human analogy: don't interrupt others!

CSMA collisions

collisions *can* still occur:

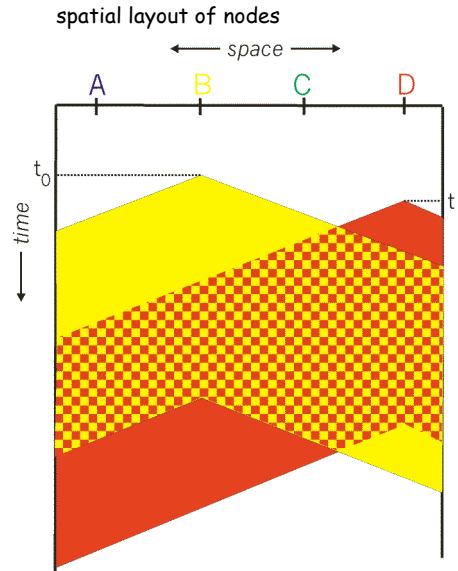
propagation delay means
two nodes may not hear
each other's transmission

collision:

entire packet transmission
time wasted

note:

role of distance & propagation
delay in determining collision
probability

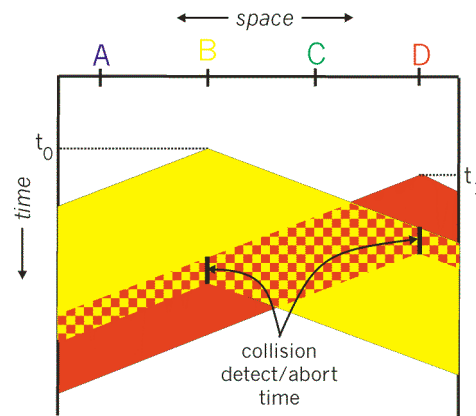


CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, deferral as in CSMA

- collisions *detected* within short time
- colliding transmissions aborted, reducing channel wastage
- 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

CSMA/CD collision detection



“Taking Turns” MAC protocols

channel partitioning MAC protocols:

- share channel efficiently and fairly at high load
- inefficient at low load: delay in channel access, $1/N$ 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!

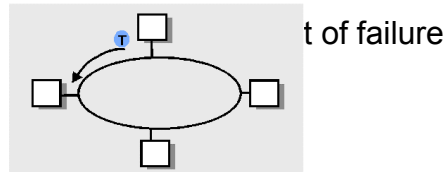
“Taking Turns” MAC protocols

Polling:

- master node “invites” slave nodes to transmit in turn
- concerns:
 - polling overhead
 - latency
 - single point of failure (master)

Token passing:

- control **token** passed from one node to next sequentially.
- token message
- concerns:
 - token overhead
 - latency



Summary of MAC protocols

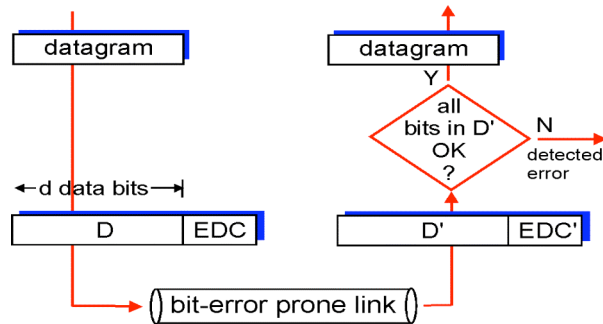
- What do you do with a shared media?
 - Channel Partitioning, by time, frequency or code
 - Time Division, Frequency Division
 - Random partitioning (dynamic),
 - ALOHA, S-ALOHA, CSMA, CSMA/CD
 - carrier sensing: easy in some technologies (wire), hard in others (wireless)
 - CSMA/CD used in Ethernet
 - CSMA/CA used in 802.11
 - Taking Turns
 - polling from a central site, token passing

Error Detection

EDC= Error Detection and Correction bits (redundancy)

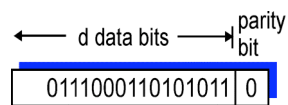
D = Data protected by error checking, may include header fields

- Error detection not 100% reliable!
 - protocol may miss some errors, but rarely
 - larger EDC field yields better detection and correction

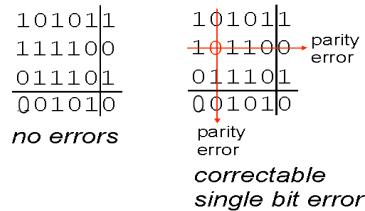
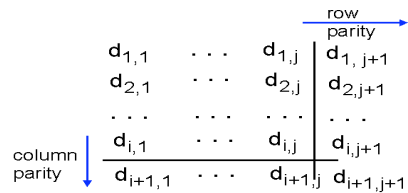


Parity Checking

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:

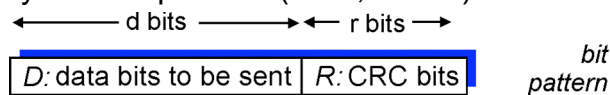
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1’s complement sum) of segment contents
- sender puts checksum value into UDP checksum field

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

Checksumming: Cyclic Redundancy Check

- view data bits, **D**, as a binary number
- choose r+1 bit pattern (generator), **G**
- goal: choose r CRC bits, **R**, such that
 - $\langle D, R \rangle$ exactly divisible by G (modulo 2)
 - receiver knows G, divides $\langle D, R \rangle$ by G. If non-zero remainder: error detected!
 - can detect all burst errors less than r+1 bits
- widely used in practice (ATM, HDCL)



$$D * 2^r \text{ XOR } R$$

mathematical formula

CRC Example

Want:

$$D \cdot 2^r \text{ XOR } R = nG$$

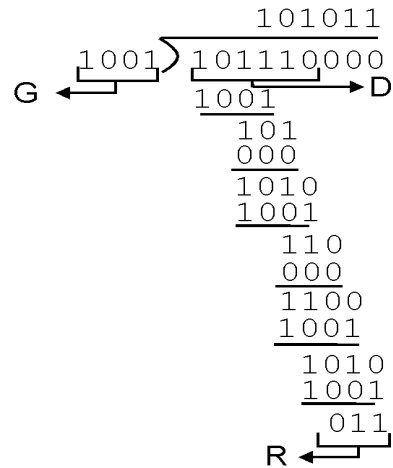
equivalently:

$$D \cdot 2^r = nG \text{ XOR } R$$

equivalently:

if we divide $D \cdot 2^r$ by G ,
want remainder R

$$R = \text{remainder} \left[\frac{D \cdot 2^r}{G} \right]$$

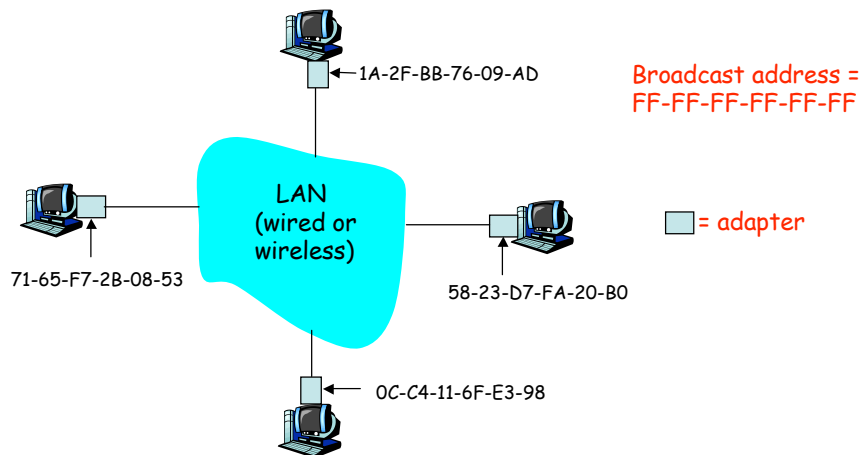


MAC Addresses and ARP

- 32-bit IP address:
 - *network-layer* address
 - used to get datagram to destination IP subnet
- MAC (or LAN or physical or Ethernet) address:
 - used to get datagram from one interface to another physically-connected interface (same network)
 - 48 bit MAC address (for most LANs)
burned in the adapter ROM

LAN Addresses and ARP

Each adapter on LAN has unique LAN address

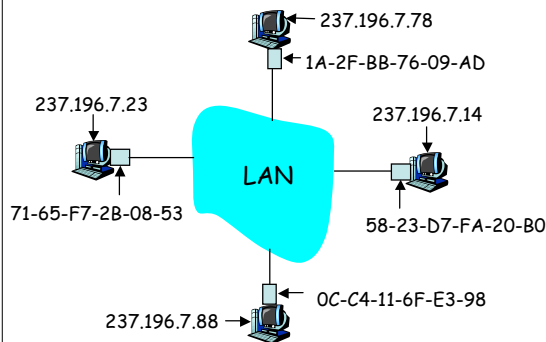


LAN Address (more)

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- Analogy:
 - (a) MAC address: like Social Security Number
 - (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 IP subnet to which node is attached

ARP: Address Resolution Protocol

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



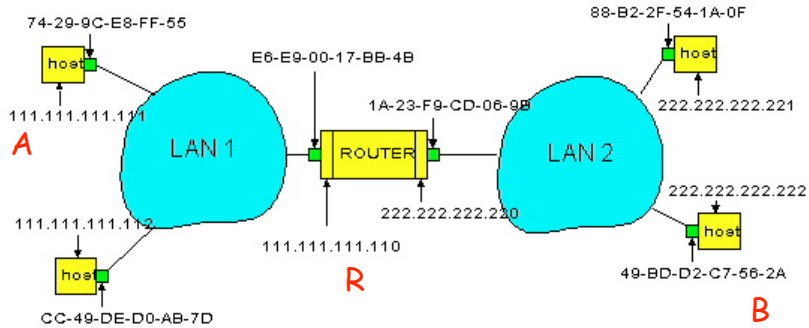
- Each IP node (Host, Router) on LAN has **ARP** table
- ARP Table: IP/MAC address mappings for some LAN nodes
< IP address; MAC address; TTL >
 - TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

ARP protocol: Same LAN (network)

- A wants to send datagram to B, and B's MAC address not in A's ARP table.
- A **broadcasts** ARP query packet, containing B's IP address
 - Dest MAC address = FF-FF-FF-FF-FF-FF
 - all machines on LAN receive ARP query
- B receives ARP packet, replies to A with its (B's) MAC address
 - frame sent to A's MAC address (unicast)
- A caches (saves) IP-to-MAC address pair in its ARP table until information becomes old (times out)
 - soft state: information that times out (goes away) unless refreshed
- ARP is "plug-and-play":
 - nodes create their ARP tables without intervention from net administrator

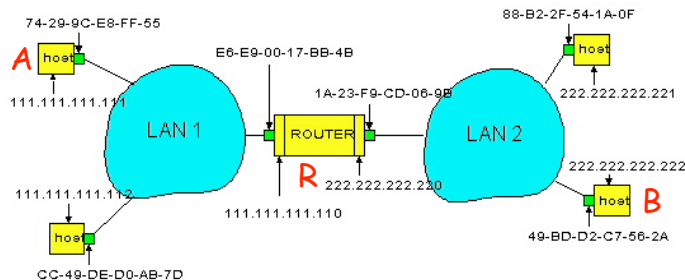
Routing to Another LAN

Objective: send datagram from A to B via R
 assume A know's B IP address



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

- A creates datagram with source A, destination B
- 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 adapter sends frame
- R's adapter 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



What's Next

- You Should Read all of Chapter 1, 2, 3, 4, and 5.1-5.5
- Homework
 - New Homework assigned today, due Thursday
 - Project 3 is **due last day of class**
- Next Lecture Topics
 - **Ethernet**