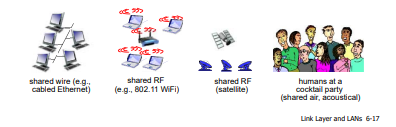
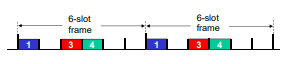
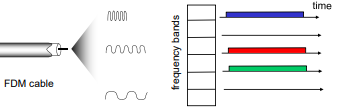
* Multiple Access Links, protocols
  + two types of “links”:
    - point-to-point
      * • PPP for dial-up access
      * • point-to-point link between Ethernet switch, host
    - broadcast (shared wire or medium)
      * • old-fashioned Ethernet
      * • upstream HFC
      * • 802.11 wireless LAN
  + 
  + 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
  + desiderata:
    - 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
    - Example: Internet
  + “taking turns”
    - • nodes take turns, but nodes with more to send can take longer turns
  + Xfinity Modem/Router uses all three protocols.
* Channel Partitioning MAC Protocols: TDMA

TDMA: time division multiple access

* + access to channel in "rounds"
  + each station gets fixed length slot (length = packet transmission time) in each round
  + unused slots go idle
  + example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle
  + High traffic load will fully utilize this.
  + 
* Channel Partitioning MAC Protocols: FDMA

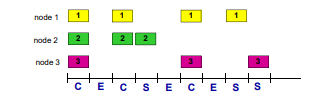
FDMA: frequency division multiple access

* + 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 packet to send, frequency bands 2,5,6 idle
  + 
* CDMA: Code division multiple access
  + Well designed : everyone gets a code
  + Together, the receiver, after receiving, with the code. They can decode it correctly.
  + Code is not random. It is intelligently distributed.
* 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 divided into equal size slots (time to transmit 1 frame)
  + nodes start to transmit only slot beginning
  + nodes are synchronized
  + if 2 or more nodes transmit in slot, all nodes detect collision

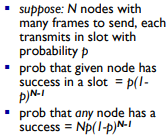
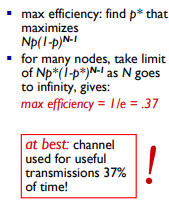
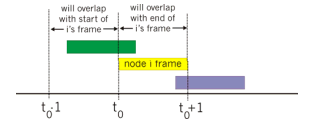
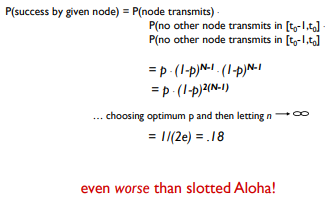
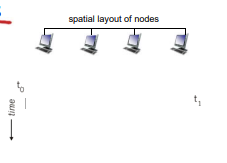
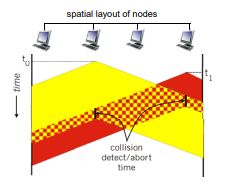
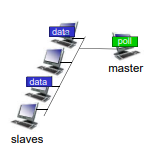
operation:

* + when node obtains fresh frame, transmits in next slot
    - if *no collision*: node can send new frame in next slot
    - if *collision*: node retransmits frame in each subsequent slot with prob. p until success
* 

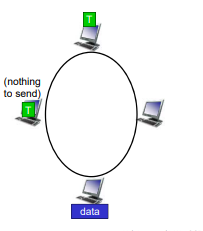
Pros:

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

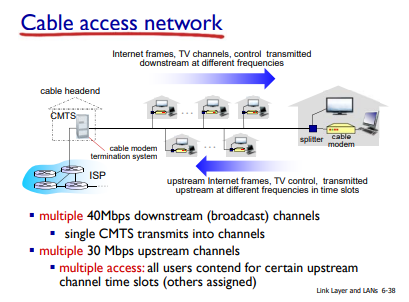
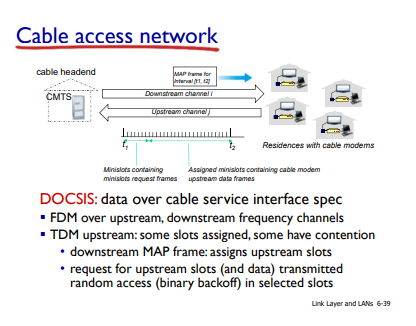
Cons:

* + 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: long-run fraction of successful slots (many nodes, all with many frames to send)
* Pure (unslotted) ALOHA
  + unslotted Aloha: simpler, no synchronization
  + when frame first arrives
    - • transmit immediately
  + collision probability increases:
  + • frame sent at t0 collides with other frames sent in [t0- 1, t0+1]
  + 
* Pure ALOHA efficiency
  + 
* CMSA (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!
* CMSA Collisions
  + collisions can still occur: propagation delay means two nodes may not hear each other’s transmission
  + collision: entire packet transmission time wasted
    - distance & propagation delay play role in 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: received signal strength overwhelmed by local transmission strength
  + human analogy: the polite conversationalist
* 
* Ethernet CSMA/CD Algorithm
  + 1. NIC receives datagram from network layer, creates frame
  + 2. If NIC senses channel idle, starts frame transmission. If NIC senses channel busy, waits until channel idle, then transmits.
  + 3. If NIC transmits entire frame without detecting another transmission, NIC is done with frame !
  + 4. If NIC detects another transmission while transmitting, aborts and sends jam signal
  + 5. After aborting, NIC enters binary (exponential) backoff:
    - • after mth collision, NIC chooses K at random from {0,1,2, …, 2m-1}. NIC waits K·512 bit times, returns to Step 2
    - • longer backoff interval with more collisions
  + CSMA/CD Efficiency
    - Tprop = max prop delay between 2 nodes in LAN
    - ttrans = time to transmit max-size frame
      * 
    - efficiency goes to 1
      * • as tprop goes to 0
      * • as ttrans goes to infinity
    - better performance than ALOHA: and simple, cheap, decentralized!
* “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!

polling:

* + master node “invites” slave nodes to transmit in turn
  + typically used with “dumb” slave devices
  + 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
    - single point of failure (token)
* 
* 
* 