The Data Link Layer

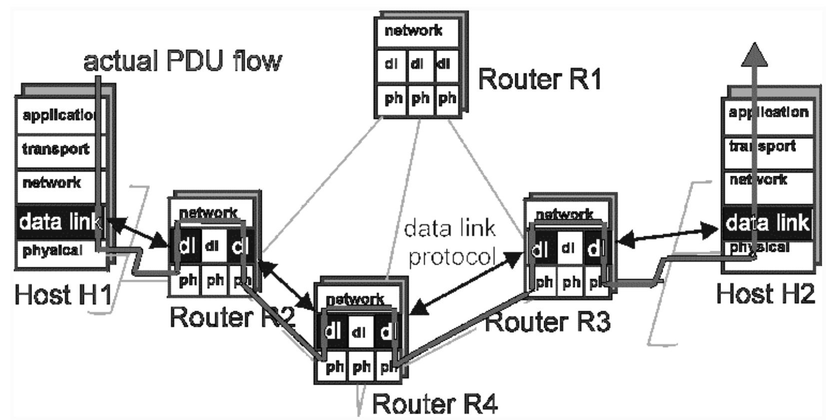
Based on slides by Shiv. Kalyanaraman and B. Sidkar RPI
Kurose and Ross from their book
Modified by Michalis Faloutsos UCR

Understand principles behind data link layer services:
- error detection, correction
- sharing a broadcast channel: multiple access
- link layer addressing
- reliable data transfer, flow control

- Instantiation and implementation of various link layer technologies

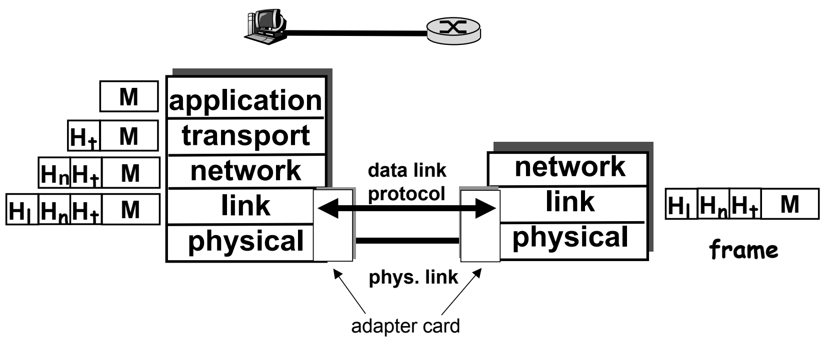
- link layer services
- error detection, correction
- multiple access protocols and LANs
- link layer addressing, ARP
- specific link layer technologies:
  - Ethernet
  - IEEE 802.11 LANs
  - PPP
  - ATM
  - hubs, bridges, switches

Link Layer: setting the context - 1



Link Layer: setting the context - 2

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



Link Layer Services - 1

- **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, dest
    - different from IP address!

Link Layer Services - 2

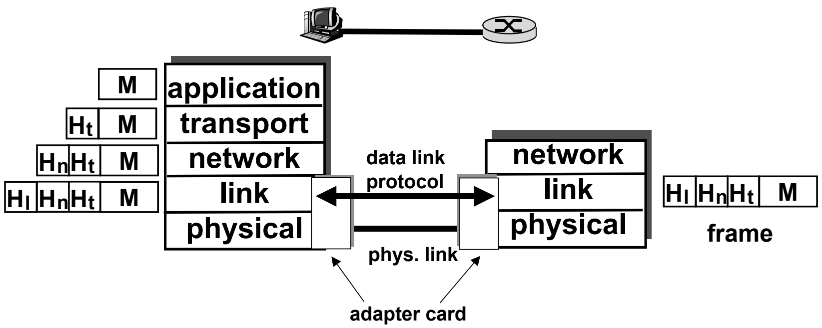
- Reliable delivery between two physically connected devices:
  - seldom used on low bit error link (fiber, some twisted pair)
  - wireless links: high error rates
    - Q: why both link-level and end-end reliability?

Link Layer Services - 3

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

Link Layer: Implementation

- Implemented in "adapter"
  - e.g., PCMCIA card, Ethernet card
  - typically includes: RAM, DSP chips, host bus interface, and link interface



Some basic concepts

- Bandwidth and throughput
- Link capacity limits
- Bandwidth delay product

Bandwidth and Throughput

- See PetDav p.40
- Bandwidth: the range of frequencies a channel can use
  - Tel. Line: 300 - 3300Hz -> 3000Hz
- Bandwidth commonly used as number of bits per sec that can be transmitted
- Throughput: the number of bits that can be delivered successfully defined per layer
  - For Link level, same as bandwidth
  - For application level, information that can go through
- Goodput: useful throughput
  - throughput - overhead

Link Capacity

- How much capacity (data rate) can a link support?
- Shannon's theorem - classic theorem:
  - C = B log2 (1 + S/N)
- Where
  - C is link capacity
  - B is the bandwidth of the line
  - S is average signal power
  - N is average noise power
- S/N is the signal to noise ratio
  - Expressed in decibels: db = 10 log10 (S/N)
  - Signal strength is reported relative to noise
- For db = 30, and B = 3300 - 300Hz -> C = 30Kbps

Bandwidth delay product

- B, Bandwidth is how fast I can push bits on the link
- D, Delay is the time it takes for information to arrive at the destination
- B x D is the amount of bits that can be in flight on a link before the receiver is aware of anything
- Often this is used for D = round trip time
  - Amount of information that is on the link before sender is informed that something arrived at the receiver

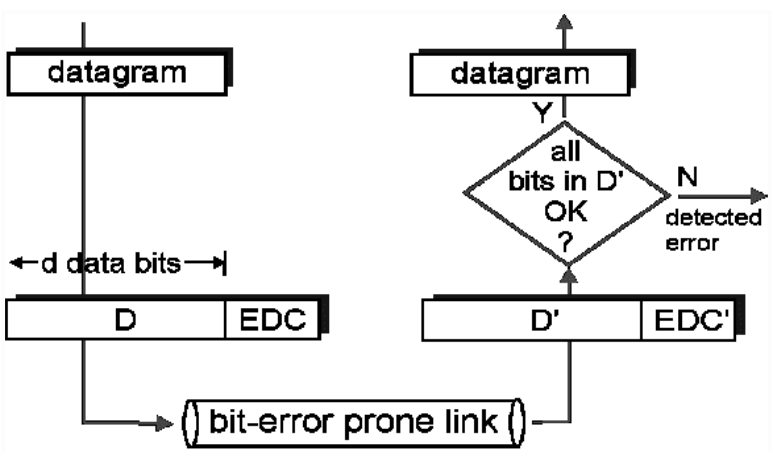
Error Detection - 1

EDC= Error Detection and Correction bits (redundancy)

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

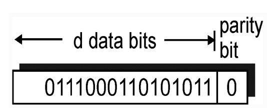
Error detection not 100% reliable!
- protocol may miss some errors, but rarely
- larger EDC field yields better detection and correction

Error Detection - 2

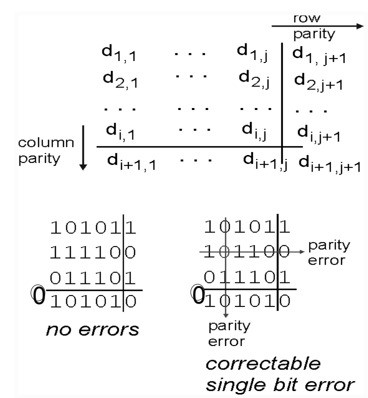


Parity Checking

Single Bit Parity:
Detect single bit errors



Two Dimensional Bit Parity:
Detect and correct single bit errors



Internet checksum

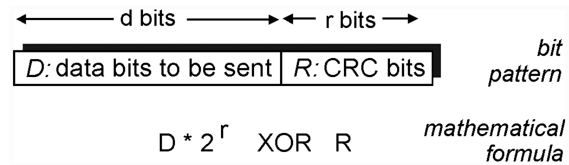
Goal: detect "errors" (e.g., flipped bits) in transmitted segment (note: used at transport layer *only*)

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
  - <D,R> exactly divisible by G (modulo 2)
  - receiver knows G, divides <D,R> by G. If non-zero remainder: error detected!
  - can detect all burst errors less than r+1 bits
- Widely used in practice (ATM, HDCL)



CRC Example

Want:

D·2r XOR R = nG

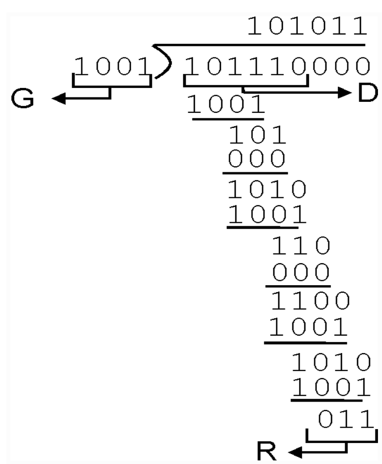
equivalently:

D·2r = nG XOR R

equivalently:

if we divide D·2r by G, want reminder R

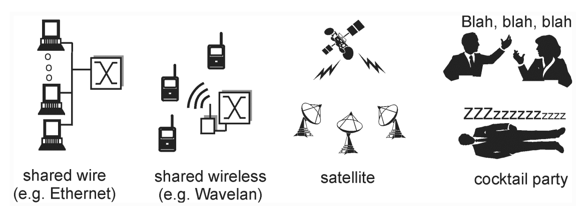
$$R = \text{remainder}\left[\frac{D \cdot 2r}{G}\right]$$



See picture from book (this is weird)

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, Wavelan, etc.)



- Switched (e.g., switched Ethernet, ATM etc)

Multiple Access Protocols - 1

- single shared communication channel
- two or more simultaneous transmissions by nodes: interference
  - only one node can send successfully at a time
- *multiple access protocol:*
  - distributed algorithm that determines how stations share channel, i.e., determine when station can transmit

Multiple Access protocols - 2

- *multiple access protocol (cont.):*
  - 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

Multiple Access protocols - 3

- claim: humans use multiple access protocols all the time
- How do we share?

MAC Protocols: a taxonomy

Three broad classes:

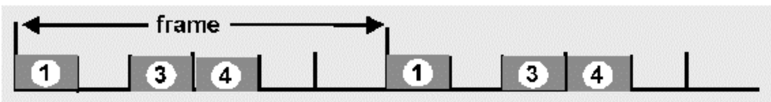
- Channel Partitioning
  - divide channel into smaller “pieces” (time slots, frequency)
  - allocate piece to node for exclusive use
- Random Access
  - allow collisions
  - “recover” from collisions
- “Taking turns”
  - tightly coordinate shared access to avoid collisions

Goal: efficient, fair, simple, decentralized

Channel Partitioning MAC protocols: TDMA - 1

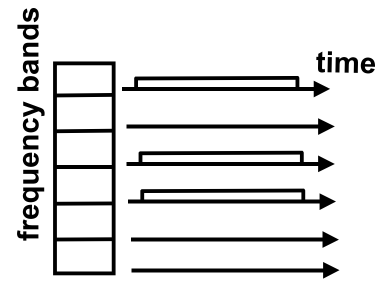
TDMA: time division multiple access

- 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



Channel Partitioning MAC protocols: FDMA - 2

- Example: 6-station LAN, 1,3,4 have pkt, frequency bands 2,5,6 idle



Channel Partitioning (CDMA) - 1

CDMA (Code Division Multiple Access)

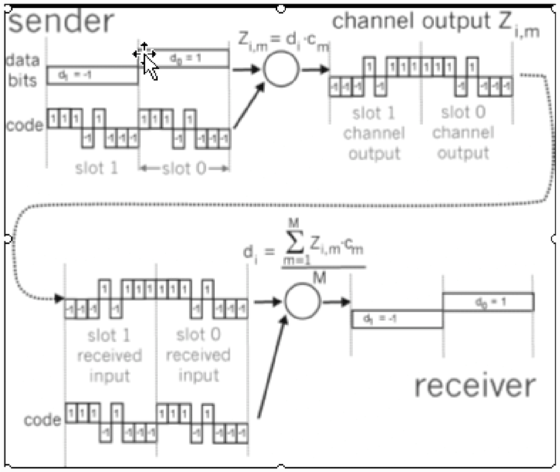
- unique “code” assigned to each user; ie, code set partitioning
- used mostly in wireless broadcast channels (cellular, satellite, etc)
- all users share same frequency, but each user has own “chipping” sequence (ie, code) to encode data

Channel Partitioning (CDMA) - 2

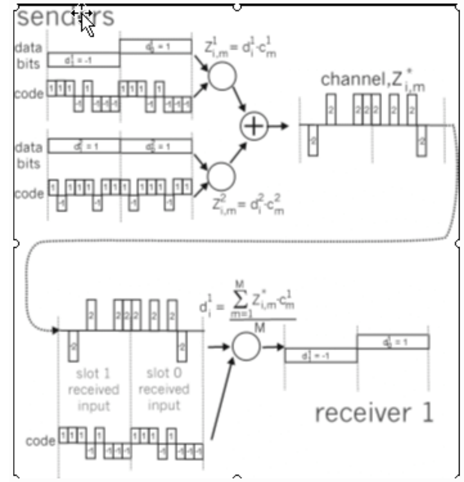
- Encoded signal = (original data) X (chipping sequence)
- Decoding: inner-product of encoded signal and chipping sequence
- allows multiple users to “coexist” and transmit simultaneously with minimal interference (if codes are “orthogonal”)

Not in Peterson Davie!

CDMA Encode/Decode



CDMA: two-sender interference



Performance of Fixed Assignment Protocols - 1

- **Fixed assignment protocols are ideal for continuous streams such as video or audio**
- **What about for packet switched data?**
- **A "perfect" multiple access scheme would always use the channel when there are packets waiting (statistical multiplexing)**
- **The mean delay for statistical multiplexing is just like for the M / M / 1 queue:** $E(T) = \frac{1}{\mu - \lambda}$, **where λ is the arrival rate and μ is the service rate**

Performance of Fixed Assignment Protocols - 2

- **OTOH fixed assignment protocols divide the channel into N separate independent, μ/N identical subchannels**
- **If each user has arrival rate λ/N, each user/subchannel pair can be modeled as a separate M / M / 1 queue**
- **And the mean delay for a packet is**

$$E(T) = \frac{1}{\mu/N - \lambda/N} = \frac{N}{\mu - \lambda}$$

- **So, if we use fixed assignment protocols for packet switched data, mean delay goes up by a factor of N!!**

Random Access Protocols - 1

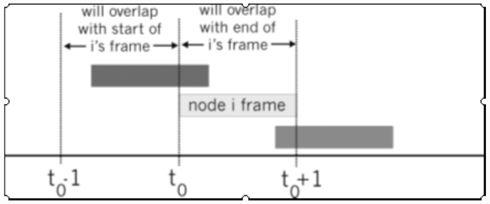
- When node has packet to send
  - transmit at full channel data rate R.
  - no *a priori* coordination among nodes
- Two or more transmitting nodes -> "collision",
- Random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)

Random Access Protocols - 2

- Examples of random access MAC protocols:
  - ALOHA (not in PetDavie)
  - slotted ALOHA (not in PetDavie)
  - CSMA and CSMA/CD

Pure (unslotted) ALOHA - 1

- Unslotted Aloha: simpler, no synchronization
- pkt needs transmission:
  - send when you have a packet
- Collision probability increases:
  - pkt sent at t_0 collide with other pkts sent in $[t_0-1, t_0+1]$



Pure (unslotted) ALOHA - 2

Success if none starts right before or right after:

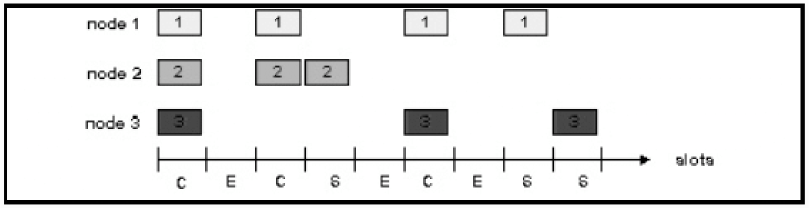
P(success by given node) = P(node transmits) $\cdot$
P(no other node transmits in $[t_0-1,t_0]$ $\cdot$
P(no other node transmits in $[t_0,t_0+1]$
$= p \cdot (1-p)^{(N-1)} \cdot (1-p)^{(N-1)}$

P(success by any of N nodes) = $N\ p \cdot (1-p)^{(N-1)} \cdot (1-p)^{(N-1)}$

… choosing optimum p as N -> infty ...
= 1/(2e) = .18

Slotted Aloha

- time is divided into equal size slots (= pkt trans. time)
- node with new arriving pkt: transmit at beginning of next slot
- if collision: retransmit pkt in future slots with probability p, until successful.



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

Slotted Aloha Efficiency

Q: What is max fraction slots successful?

A: Suppose N stations have packets to send

- each transmits in slot with probability *p*
- prob. successful transmission *S* is:

by single node: $S = p (1-p)^{(N-1)}$

by any of N nodes

S = Prob (only one transmits)

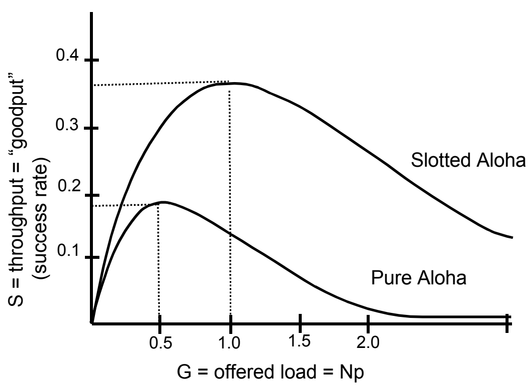
$= N p (1-p)^{(N-1)}$

... choosing optimum p as n -> infty ...

= 1/e = .37 as N -> infty

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

Performance Comparison



***protocol* constrains effective channel throughput!**

The Rules of Sharing

- CSMA: Collision Sensing Multiple Access
  - Sense the channel before sending
- CSMA/CD: CSMA with Collision Detection
  - Sense the channel before sending
  - Sense the channel WHILE sending too

Carrier Sensing Multiple Access (CSMA) - 1

- **In some shorter distance networks, it is possible to listen to the channel before transmitting**
- **In radio networks, this is called " sensing the carrier"**
- **The CSMA protocol works just like Aloha except: If the channel is sensed busy, then the user waits to transmit its packet, and a collision is avoided**
- **This really improves the performance in short distance networks!**

Carrier Sensing Multiple Access (CSMA) - 2

- How long does a blocked user wait before trying again to transmit its packet? Three basic variants:
- 1-persistent: Blocked user continuously senses channel until its idle, then transmits
- 0-persistent: Blocked user waits a randomly chosen amount of time before sensing channel again

Carrier Sensing Multiple Access (CSMA) - 3

- P-persistent: Let τ = end-to-end propagation delay
  - **If channel is idle then transmit packet**
  - **If channel busy then toss coin *[with P(heads) = P]***
  - **Heads: Transmit at first idle**
  - **Tails: wait until first idle plus *T*, sense, repeat**
- **Human analogy: Don't interrupt others**

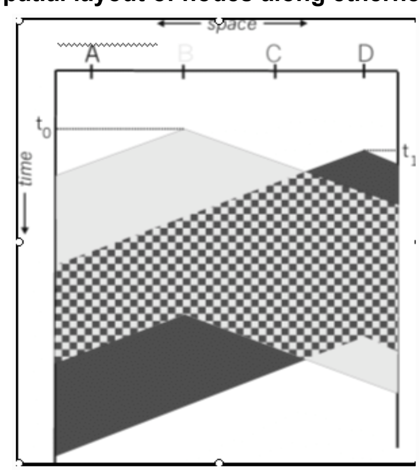
CSMA collisions

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

collision: entire packet transmission time wasted

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

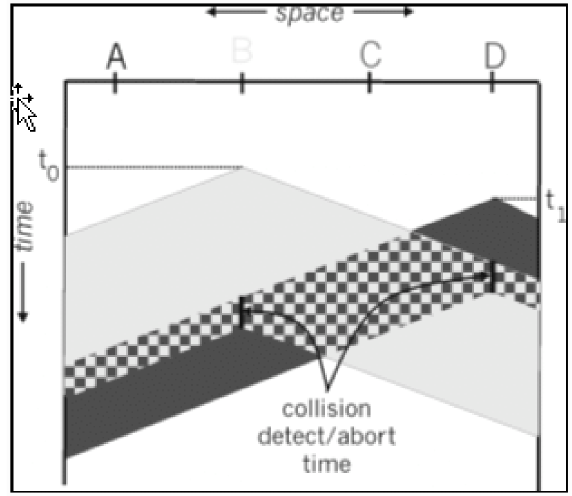
spatial layout of nodes along ethernet



CSMA/CD (Collision Detection)

- CSMA improves performance, but still it wastes the channel during collisions
- In some very short distance networks (e.g. coax LANs), it is possible to listen while transmitting (in addition to listening before transmitting)
- If we detect a collision while transmitting, we can abort the transmission and free up the channel sooner
- This idea was proposed by R. Metcalfe and Boggs at Xerox PARC in the mid 1970s under the name Ethernet.
- Human analogy: the polite conversationalist

CSMA/CD collision detection



"Taking Turns" MAC protocols - 1

Channel partitioning MAC protocols:

- share channel efficiently 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!

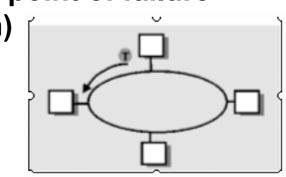
"Taking Turns" MAC protocols - 2

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:

- Control token passed from one node to next sequentially.
- Token message
- Concerns:
  - token overhead
  - latency
  - single point of failure (token)



Reservation-based protocols - 1

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**

Reservation-based protocols - 2

- **After reservation slots, message transmissions ordered by known priority**