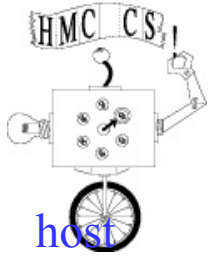
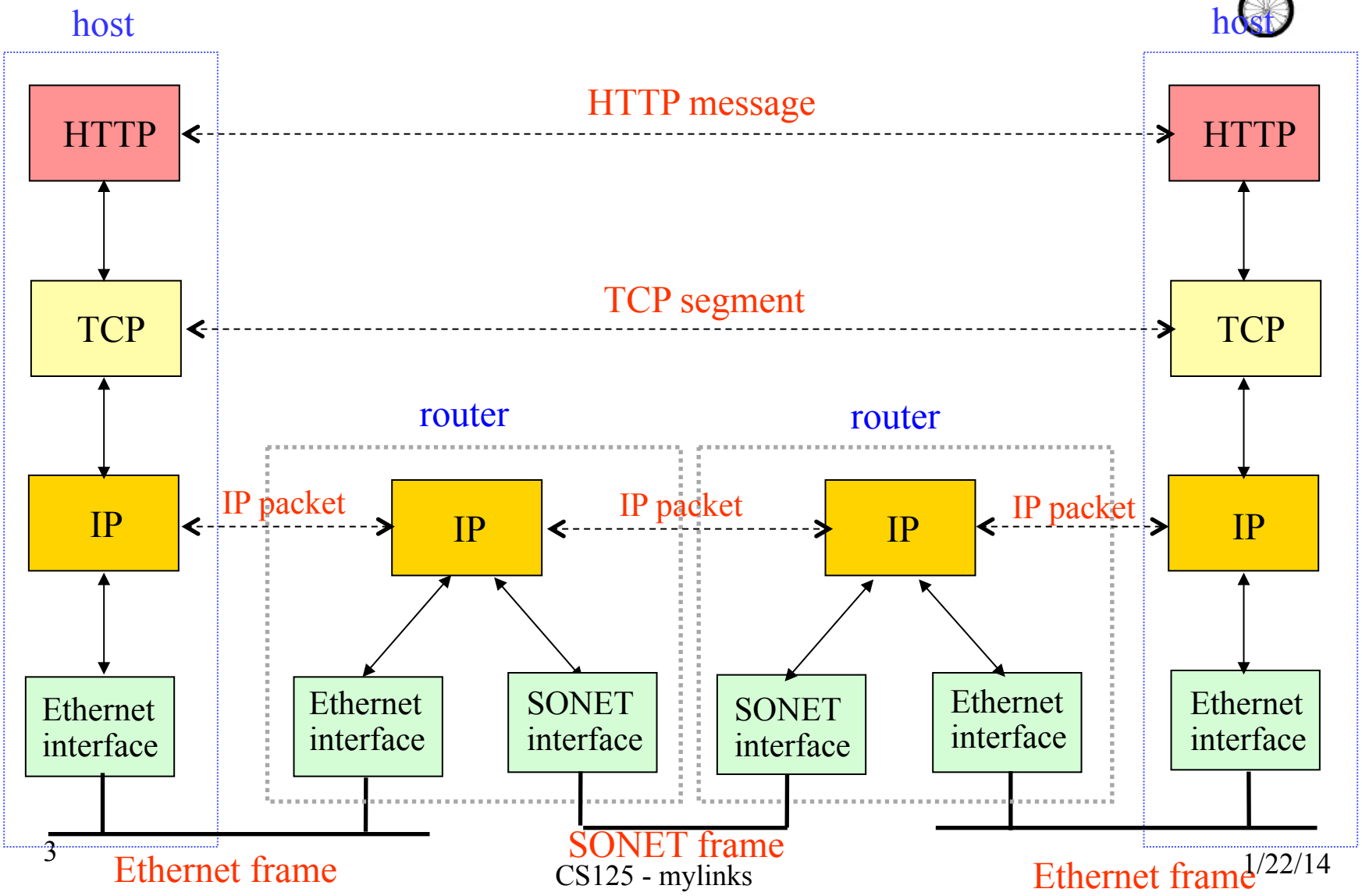
Links

Goals of Today's Lecture

- Link-layer services
  - Encoding, framing, and error detection
  - Error correction and flow control
- Sharing a shared media
  - Channel partitioning
  - Taking turns
  - Random access
- Shared Networks
  - Ethernet
  - Token ring
  - Wireless

Message, Segment, Packet, and Frame

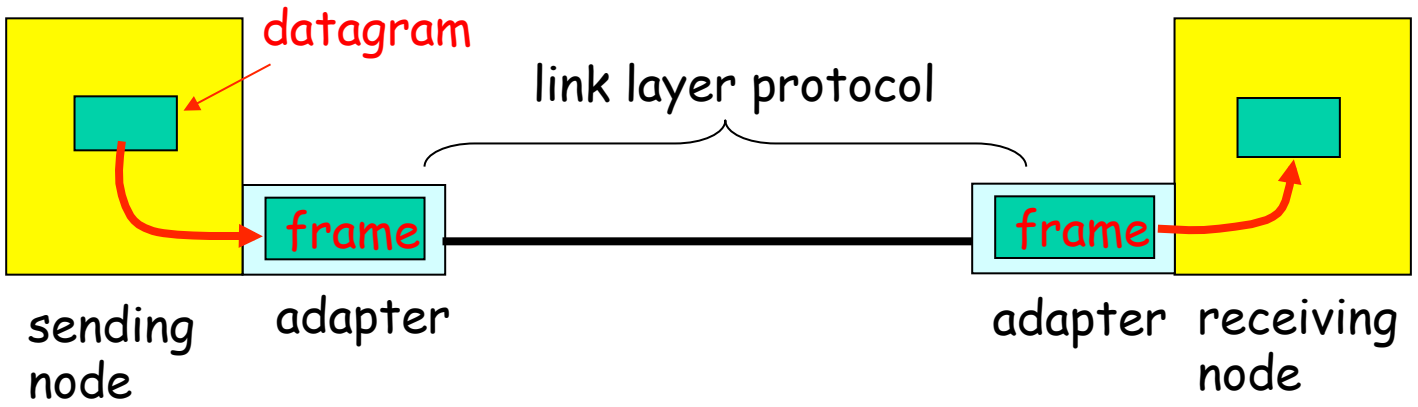




Link Layer Protocol for Each Hop

- IP packet transferred over multiple hops
  - Each hop has a link layer protocol
  - May be different on different hops
- Analogy: trip from HMÇ to Auckland
  - Limo: HMC to LAX
  - Plane: LAX to Auckland AP
  - Car: Auckland AP to hotel
- Refining the analogy
  - Tourist == packet
  - Transport segment == communication link
  - Transportation mode == link-layer protocol
  - Travel agent == routing algorithm

Adaptors Communicating



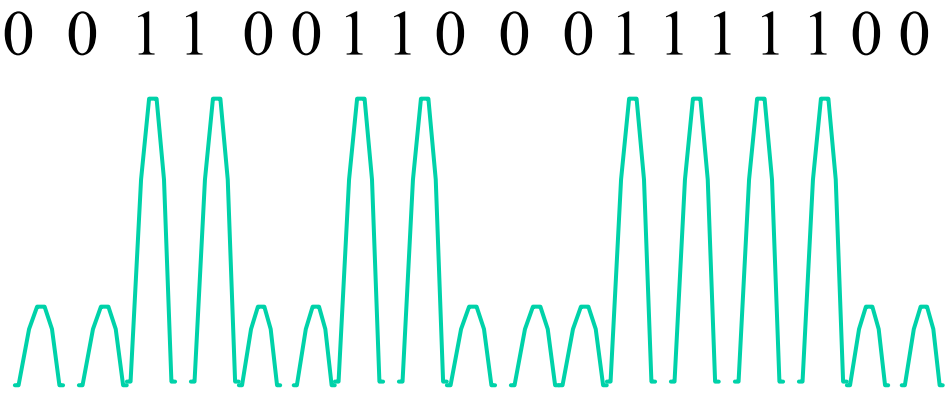
- Link layer implemented in adaptor (network interface card)
  - Ethernet card, PCMCI card, 802.11 card
- Sending side:
  - Encapsulates datagram (packet) in a frame
  - Adds error checking bits, flow control, etc.??
- Receiving side
  - Looks for errors, flow control, etc.
  - Extracts datagram (how is datagram found) and passes to receiving node

Link-Layer Services

- Encoding
  - Representing the 0s and 1s on the link
- Framing
  - Encapsulating packet into frame, adding header, trailer
  - Using MAC addresses, rather than IP addresses
- Error detection
  - Errors caused by signal attenuation, noise.
  - Receiver detecting presence of errors
- Error correction
  - Receiver correcting errors without retransmission
- Flow control
  - Pacing between adjacent sending and receiving nodes

Encoding

- Signals propagate over physical links
  - Source node encodes the bits into a signal
  - Receiving node decodes the signal back into bits
- Simplify some electrical engineering details
  - Assume two discrete signals, high and low
  - E.g., could correspond to two different voltages
- Simple approach
  - High for a 1,
  - low for a 0
  - NRZ:
  - NonReturn to Zero



Problem With Simple Approach

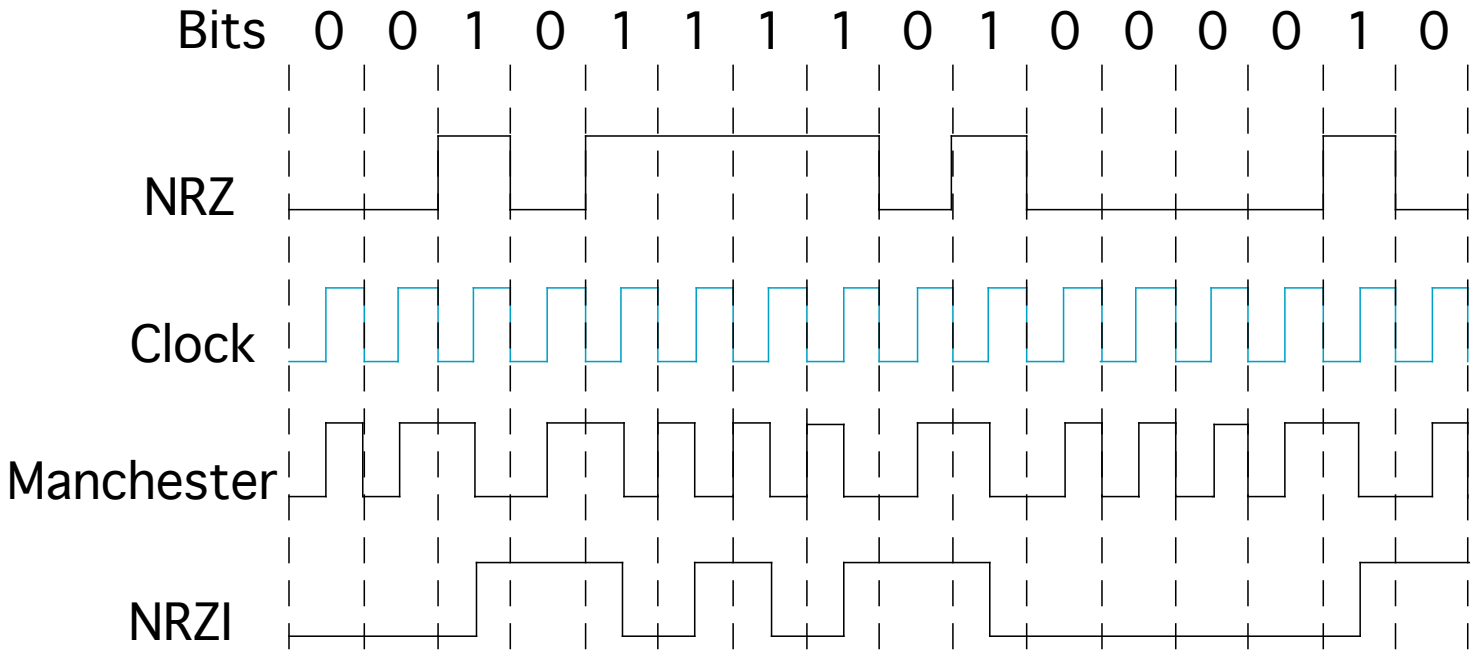
- Long strings of 0s or 1s introduce problems
  - No transitions from low-to-high, or high-to-low..Why?
- Receiver keeps average of signal it has received
  - Uses the average to distinguish between high and low
  - Long flat strings make receiver sensitive to small change
- Transitions also necessary for clock recovery
  - Receiver uses transitions to drive its own clock
  - Long flat strings do not produce any transitions
  - Can lead to clock drift at the receiver
- Alternatives (see Section 2.2)
  - Non-return to zero inverted, and Manchester encoding

Alternative Encodings

- Non-return to Zero Inverted (NRZI)
  - make a transition from current signal to encode a one; stay at current signal to encode a zero
  - solves the problem of consecutive ones

- Manchester
  - transmit XOR of the NRZ encoded data and the clock
  - only 50% efficient (bit rate = 1/2 baud rate)

Encodings (cont)

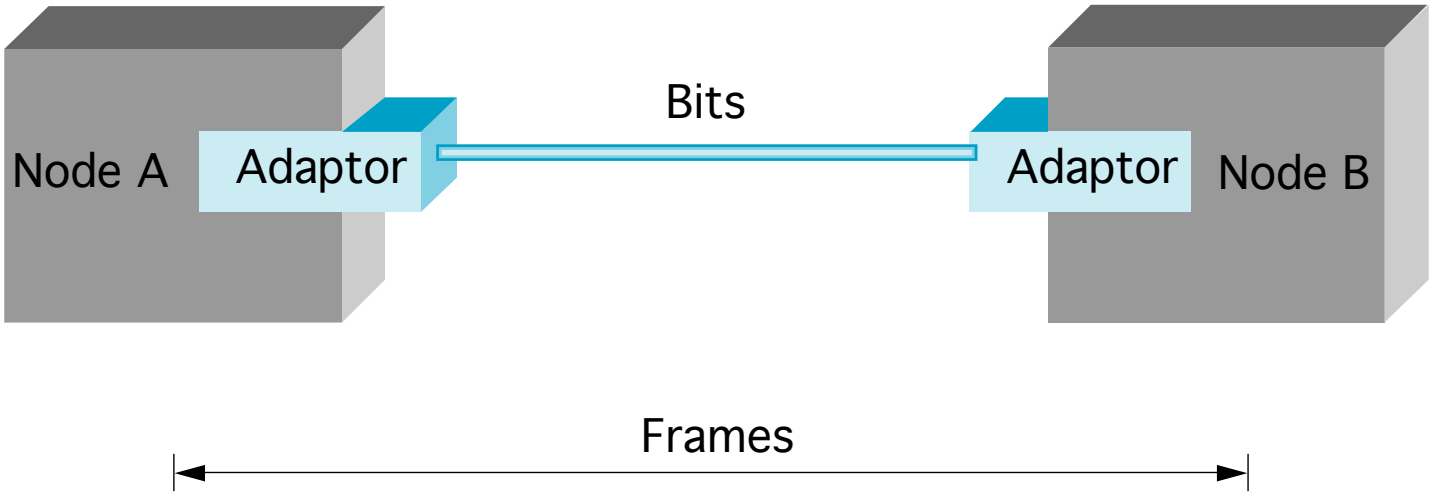


Encodings (cont)

- 4B/5B
  - every 4 bits of data encoded in a 5-bit code
  - 5-bit codes selected to have no more than one leading 0 and no more than two trailing 0s
  - thus, never get more than three consecutive 0s
  - resulting 5-bit codes are transmitted using NRZI
  - achieves 80% efficiency

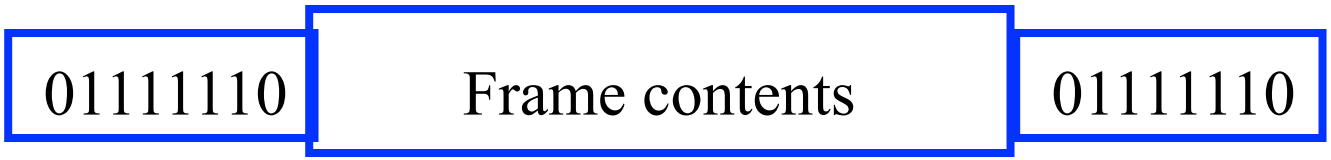
Framing

- Break sequence of bits into a frame
- Typically implemented by network adaptor



Framing

- Sentinel-based
  - Delineate frame with special pattern (e.g., 01111110)



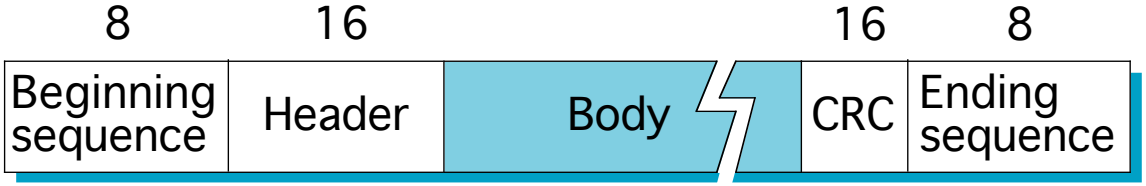
  - Problem: what if special patterns occurs within frame?
  - Solution: escaping the special characters
    - E.g., sender always inserts a 0 after five 1s
    - … and receiver always removes a 0 appearing after five 1s
  - Similar to escaping special characters in C programs

Framing (Continued)

- Counter-based
  - Include the payload length in the header
  - … instead of putting a sentinel at the end
  - Problem: what if the count field gets corrupted?
    - Causes receiver to think the frame ends at a different place
  - Solution: catch later when doing error detection
    - And wait for the next sentinel for the start of a new frame
- Clock-based
  - Make each frame a fixed size
  - No ambiguity about start and end of frame
  - But, may be wasteful of bandwidth

Framing Approaches

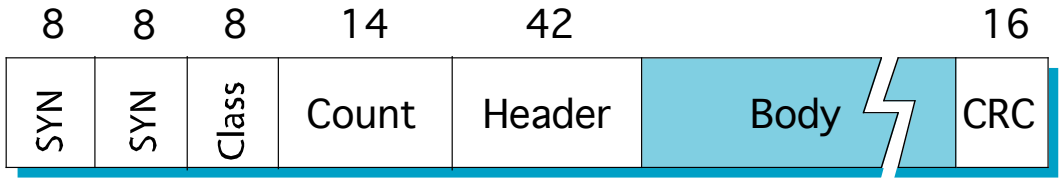
- Sentinel-based
  - delineate frame with special pattern: 01111110
  - e.g., HDLC, SDLC, PPP



  - problem: special pattern appears in the payload
  - solution: *bit stuffing*
    - sender: insert 0 after five consecutive 1s
    - receiver: delete 0 that follows five consecutive 1s

Framing Approaches (cont)

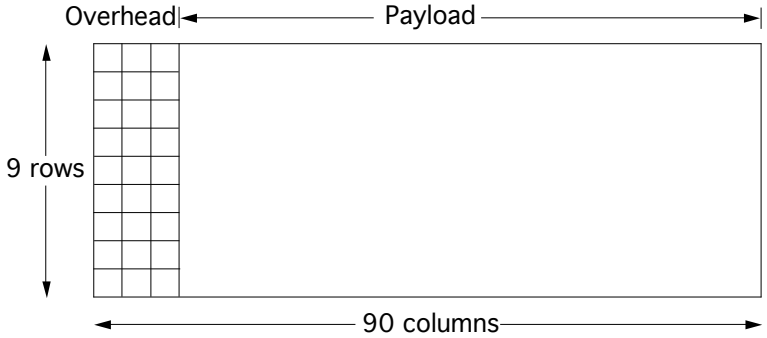
- Counter-based
  - include payload length in header
  - e.g., DDCMP

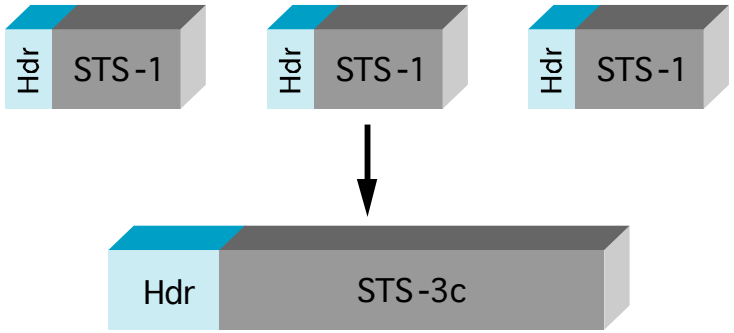


  - problem: count field corrupted
  - solution: catch when CRC fails

Framing Approaches (cont)

- Clock-based
  - each frame is 125us long
  - e.g., SONET: Synchronous Optical Network
  - STS-*n* (STS-1 = 51.84 Mbps)





Error Detection

- Errors are unavoidable
  - Electrical interference, thermal noise, etc.
- Error detection
  - Transmit extra (redundant) information
  - Use redundant information to detect errors
  - Extreme case: send two copies of the data
  - Trade-off: accuracy vs. overhead
- Techniques for detecting errors
  - Parity checking
  - Checksum
  - Cyclic Redundancy Check (CRC)

Error Detection Techniques

- Parity check
  - Add an extra bit to a 7-bit code
  - Odd parity: ensure an odd number of 1s
    - E.g., 0101011 becomes 01010111
  - Even parity: ensure an even number of 1s
    - E.g., 0101011 becomes 01010110
- Checksum
  - Treat data as a sequence of 16-bit words
  - Compute a sum of all the 16-bit words, with no carries
  - Transmit the sum along with the packet
- Cyclic Redundancy Check (CRC)
  - See book for details

Errors
Cyclic Redundancy Check

- Add k bits of redundant data to an n-bit message
  - want $k << n$
  - e.g., $k = 32$ and $n = 12{,}000$ (1500 bytes)
- Represent n-bit message as n-1 degree polynomial
  - e.g., MSG=10011010 as $M(x) = x^7 + x^4 + x^3 + x^1$
- Let there be some divisor polynomial
  - e.g., $C(x) = x^3 + x^2 + 1$
- k bits will be used to hold remainder of M/C

Errors cont: Selecting $C(x)$

- All single-bit errors, as long as the x^k and x^0 terms have non-zero coefficients.
- All double-bit errors, as long as $C(x)$ contains a factor with at least three terms
- Any odd number of errors, as long as $C(x)$ contains the factor $(x + 1)$
- Any 'burst' error (i.e., sequence of consecutive error bits) for which the length of the burst is less than k bits.
- Most burst errors of larger than k bits can also be detected
- See book for common *C(x)*

Errors cont: Internet Checksum Algorithm

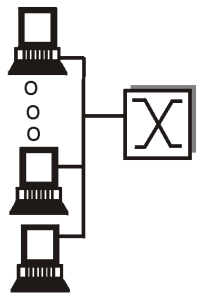
- View message as a sequence of 16-bit integers; sum using 16-bit ones-complement arithmetic; take ones-complement of the result.

```
u_short
cksum(u_short *buf, int count)
{
    register u_long sum = 0;
    while (count--)
    {
        sum += *buf++;
        if (sum & 0xFFFF0000)
        {
            /* carry occurred, so wrap around */
            sum &= 0xFFFF;
            sum++;
        }
    }
    return ~(sum & 0xFFFF);
}
```

Links

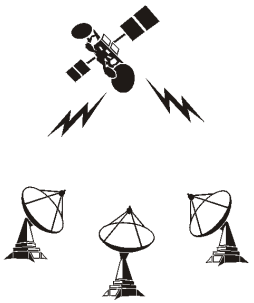
Point-to-Point vs. Broadcast Media

- Point-to-point
  - PPP for dial-up access
  - Point-to-point link between Ethernet switch and host
- Broadcast (shared wire or medium)
  - Traditional Ethernet
  - 802.11 wireless LAN



shared wire
(e.g. Ethernet)

shared wireless
(e.g. Wavelan)



satellite

cocktail party

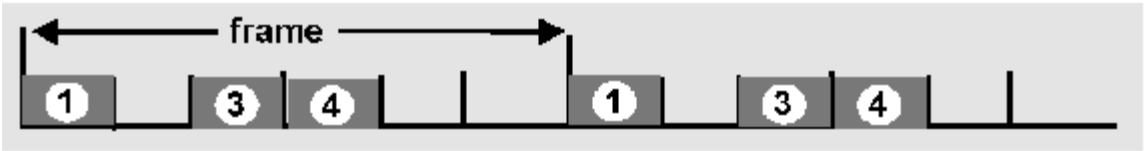
Multiple Access Protocol

- Single shared broadcast channel
  - Avoid having multiple nodes speaking at once
  - Otherwise, collisions lead to garbled data
- Multiple access protocol
  - Distributed algorithm for sharing the channel
  - Algorithm determines which node can transmit
- Classes of techniques
  - Channel partitioning: divide channel into pieces
  - Taking turns: passing a token for the right to transmit
  - Random access: allow collisions, and then recover

Channel Partitioning: TDMA

TDMA: time division multiple access

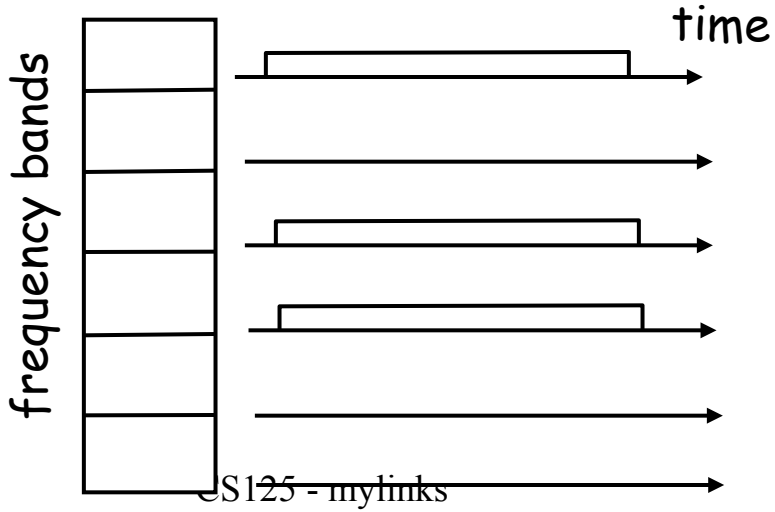
- Access to channel in "rounds"
  - Each station gets fixed length slot in each round
- Time-slot length is packet transmission time
  - Unused slots go idle
- Example: 6-station LAN with slots 1, 3, and 4



Channel Partitioning: FDMA

FDMA: frequency division multiple access

- Channel spectrum divided into frequency bands
  - Each station assigned fixed frequency band
- Unused transmission time in bands go idle
- Example: 6-station LAN with bands 1, 3, and 4



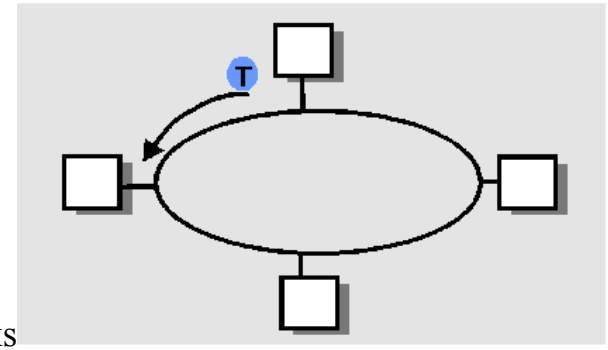
"Taking Turns" MAC protocols

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
  - Single point of failure (token)



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
- Examples
  - ALOHA and Slotted ALOHA
  - CSMA, CSMA/CD, CSMA/CA

Key Ideas of Random Access

- Carrier sense
  - *Listen before speaking, and don't interrupt*
  - Checking if someone else is already sending data
  - … and waiting till the other node is done
- Collision detection
  - *If someone else starts talking at the same time, stop*
  - Realizing when two nodes are transmitting at once
  - …by detecting that the data on the wire is garbled
- Randomness
  - *Don't start talking again right away*
  - Waiting for a random time before trying again

Slotted ALOHA – 70's

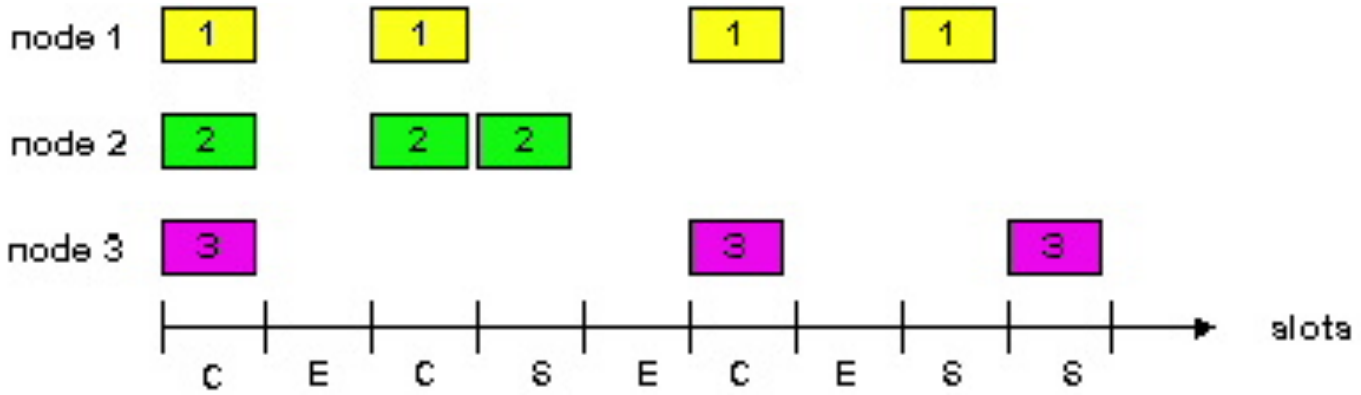
Assumptions

- All frames same size
- Time divided into equal slots (time to transmit a frame)
- Nodes start to transmit frames only at start of slots
- Nodes are synchronized
- If two or more nodes transmit, all nodes detect collision

Operation

- When node obtains fresh frame, transmits in next slot
- No collision: node can send new frame in next slot
- Collision: node retransmits frame in each subsequent slot with probability p until success

Slotted ALOHA



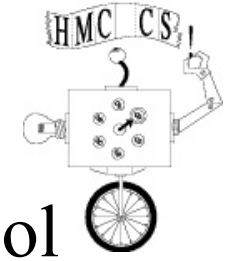
Pros

- Single active node can continuously transmit at full rate of channel
- Highly decentralized: only nodes in slots 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

CSMA (Carrier Sense Multiple Access)



- Collisions hurt the efficiency of ALOHA protocol
  - At best, channel is useful 37% of the time

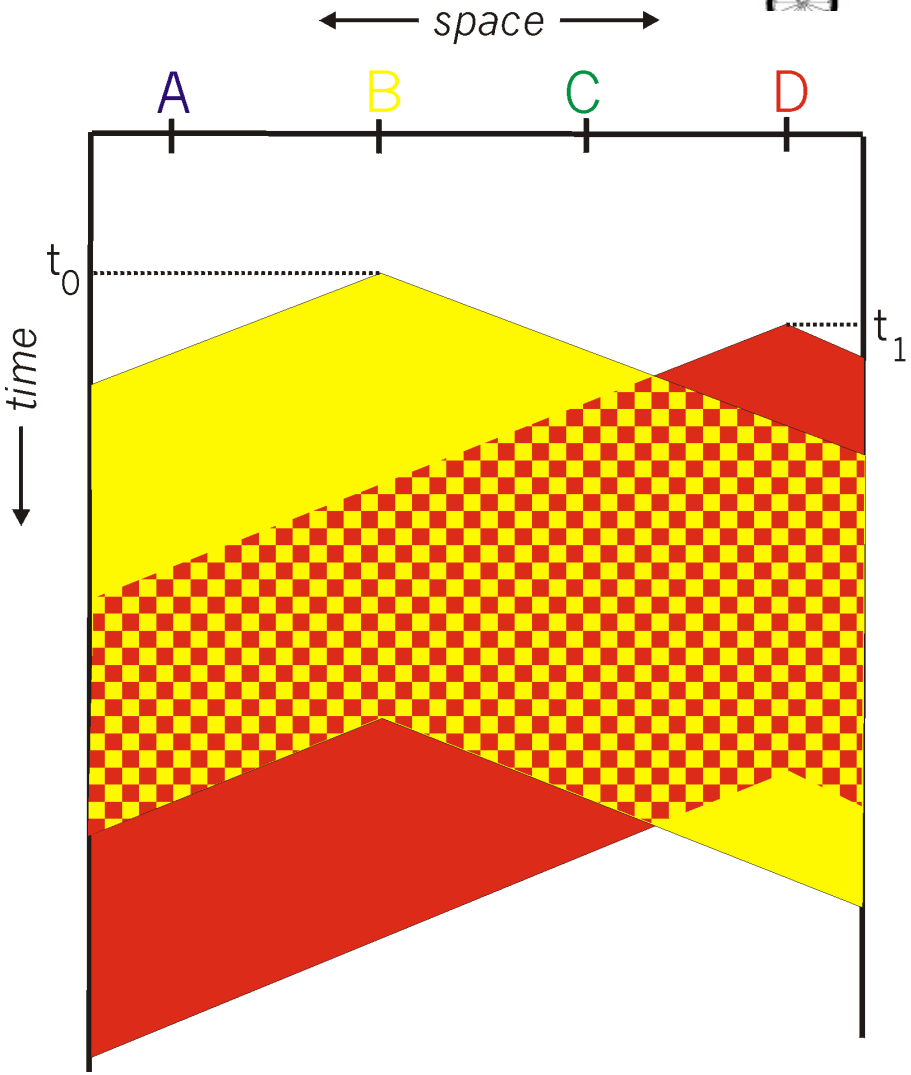
- CSMA: listen before transmit
  - If channel sensed idle: transmit entire frame
  - If channel sensed busy, defer transmission

- Human analogy: don't interrupt others!

CSMA Collisions

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

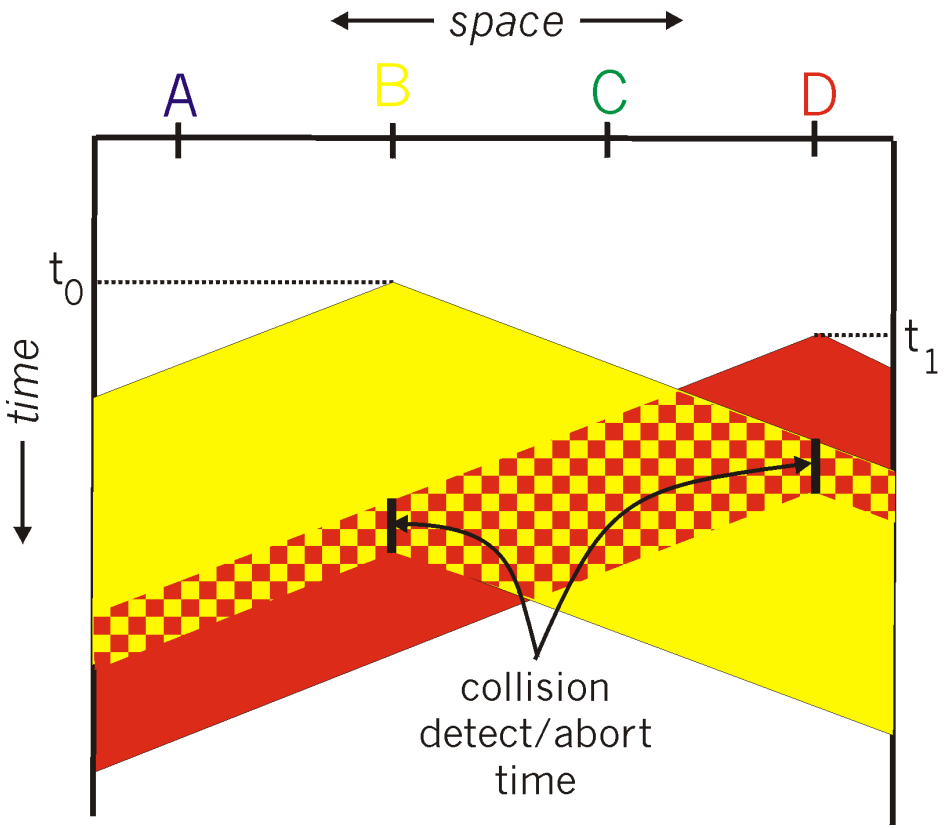
Collision:
entire packet transmission time wasted



CSMA/CD (Collision Detection)

- CSMA/CD: carrier sensing, deferral as in CSMA
  - Collisions detected within short time
  - Colliding transmissions aborted, reducing 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

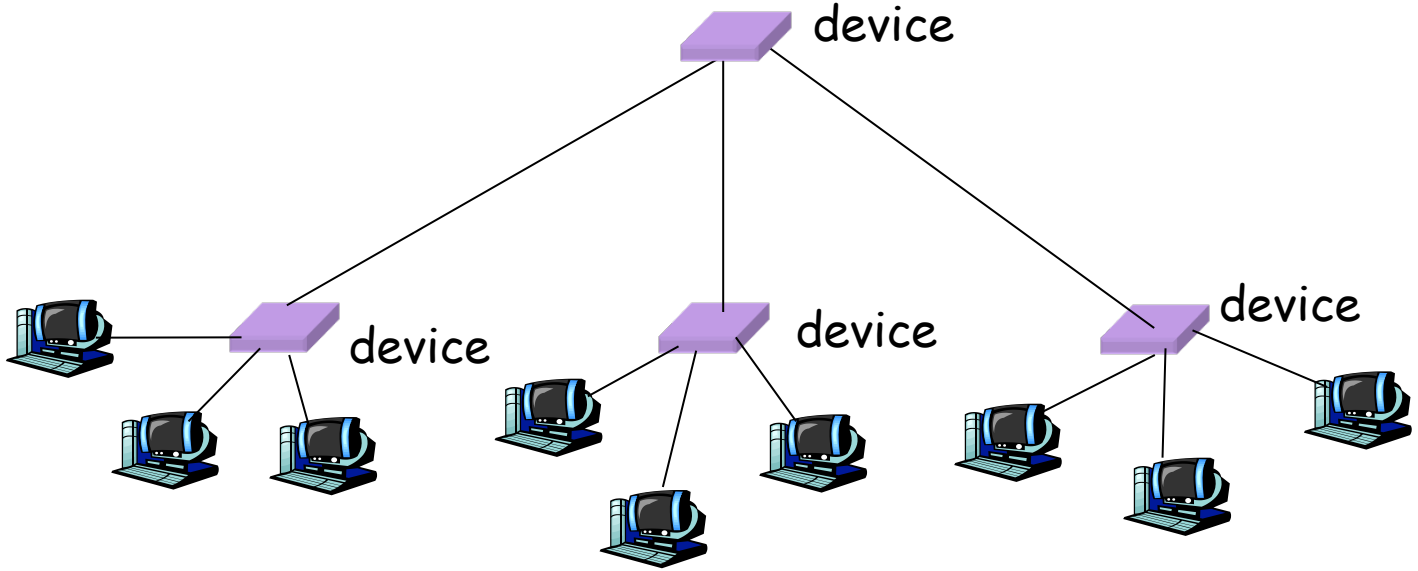
CSMA/CD Collision Detection



Three Ways to Share the Media

- 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!
- "Taking turns" protocols
  - Eliminates empty slots without causing collisions
  - Vulnerable to failures (e.g., failed node or lost token)
- Random access MAC protocols
  - Efficient at low load: single node can fully utilize channel
  - High load: collision overhead

Next Issue: Interconnecting LANS



Conclusions

- IP runs on a variety of link layer technologies
  - Point-to-point links vs. shared media
  - Wide varieties within each class
- Link layer performs key services
  - Encoding, framing, and error detection
  - Optionally error correction and flow control
- Shared media introduce interesting challenges
  - Decentralized control over resource sharing
  - Partitioned channel, taking turns, and random access
  - Ethernet as a wildly popular example