Computer Networks COL 334/672

Multi-access rules

Slides adapted from KR

Sem 1, 2025-26

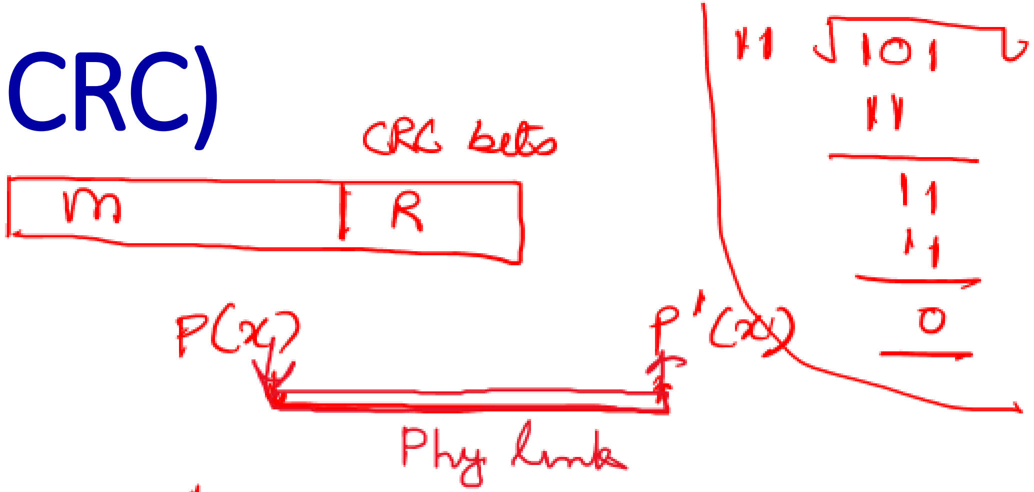
Quiz on Moodlenew

Password: manchester

Link Layer: Services

- Encoding
- Framing
- Error detection
- Addressing
- Link access

Cyclic Redundancy Check (CRC)



- **Algorithm**

1. Multiply $M(x)$ by x^k; that is, add k zeros at the end of the message. Call this zero-extended message $T(x)$.
2. Divide $T(x)$ by G(x) and find the remainder.
3. Subtract the remainder from $T(x)$ → get $P(x)$

$G(x) \mid P'(x)$

$(x+1) \times G'(x) \mid P'(x)$

$(x+1) \mid P(x) + E(x)$

$x+1 \mid E(x)$

- **Claim:** If G(x) has x+1 as one of its factors, all single-bit errors can be detected
- **Ethernet** protocol uses a 32-bit error check

CRC-32 = $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

$x+1 \mid x^k$

Not possible

- Where is CRC implemented?

Hardware (NIC)

End-to-end principle & Error detection

↳ Link layer shouldn't implement ED but done for performance

Link Layer: Services

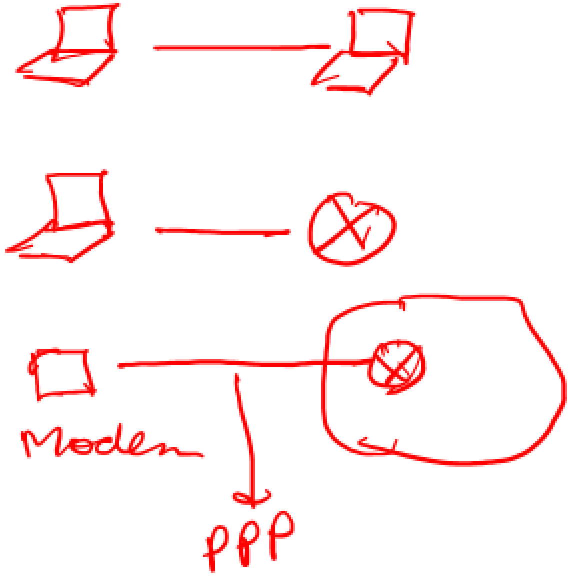
- Encoding
- Framing
- Error detection
- Addressing and link access

multiples

Multiple access links, protocols

two types of “links”:

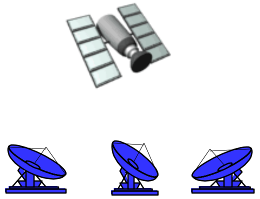
- point-to-point
  - point-to-point link between Ethernet switch, host
  - PPP for dial-up access
- broadcast (shared wire or medium)
  - old-school Ethernet
  - upstream HFC in cable-based access network
  - 802.11 wireless LAN, 4G/4G. satellite



shared wire (e.g., cabled Ethernet)

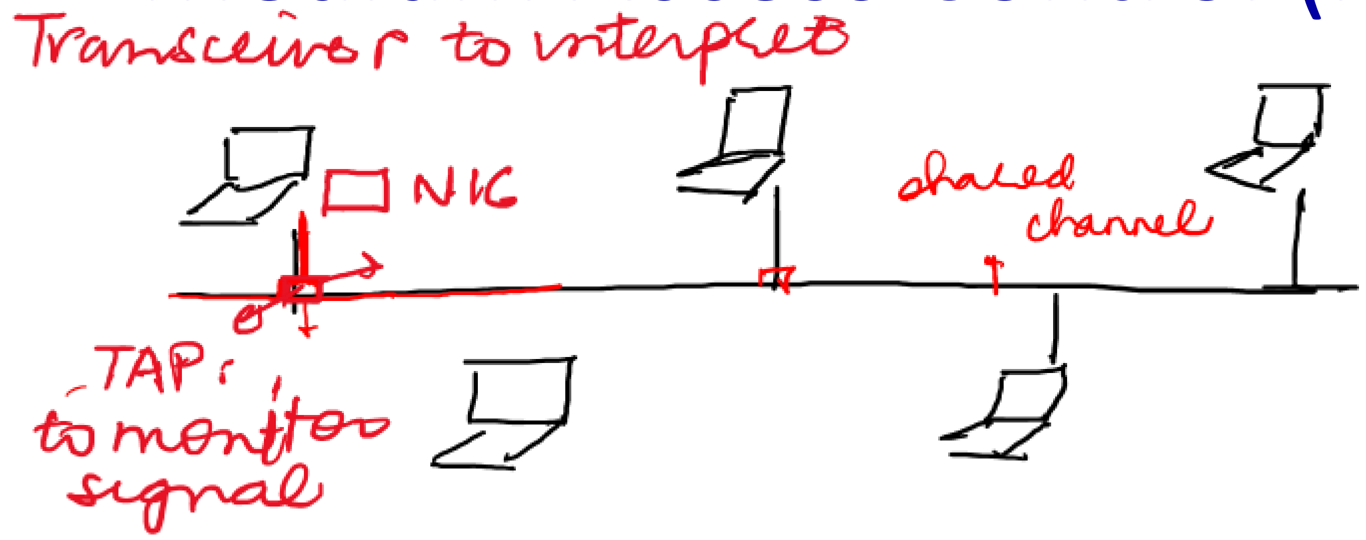
shared radio: 4G/5G

shared radio: WiFi



shared radio: satellite

Medium Access Control (MAC) protocol



Requirement 1: Addressing
- Need a way to identify end nodes
- Ethernet uses a 48-bit MAC address burned in NIC ROM, also sometimes software settable
- E.g., 1A-2F-BB-76-09-AD

Find out your host's MAC using ipconfig / ifconfig

MAC protocol

- 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

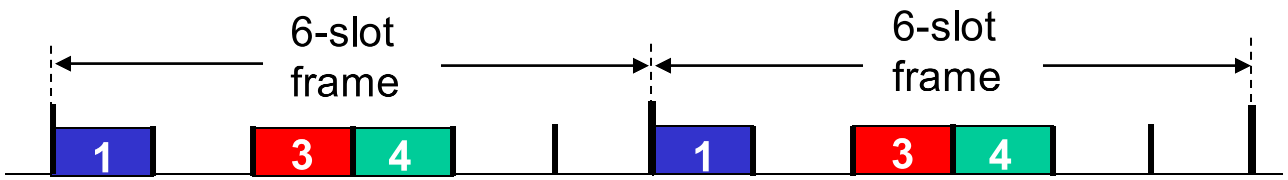
DESIGN GOALS
1. Efficiency
2. Fairness
3. Resilient
4. Simple

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

end host



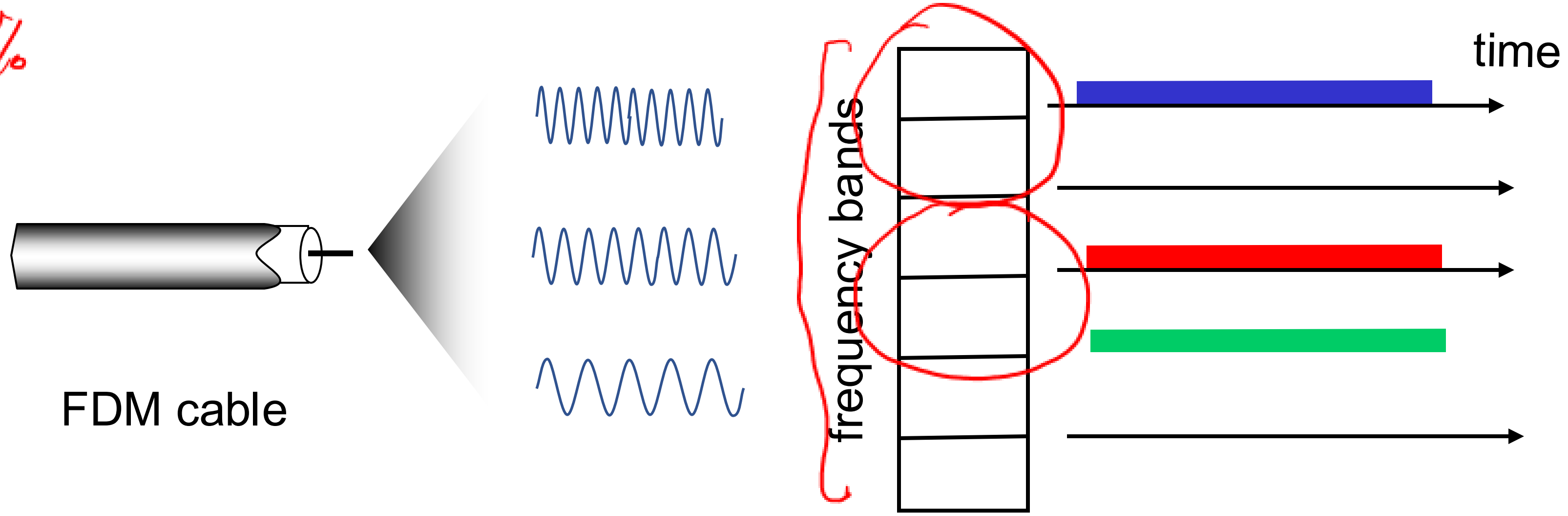
Channel partitioning MAC protocols: FDMA

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

① Fair ✓
② Simple ✓
③ Resilient ✓
④ Efficiency ✗

50%

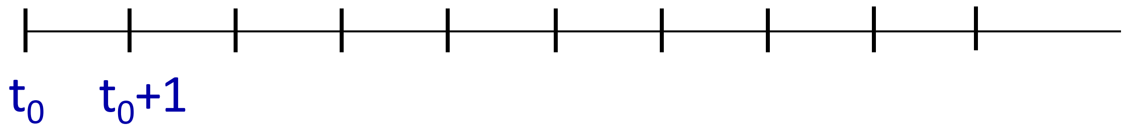


Random access protocols

- 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 protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)

  & how to (try to) avoid collisions
- examples of random access MAC protocols:
  - ALOHA, slotted ALOHA
  - CSMA, CSMA/CD, CSMA/CA

Slotted ALOHA



assumptions:

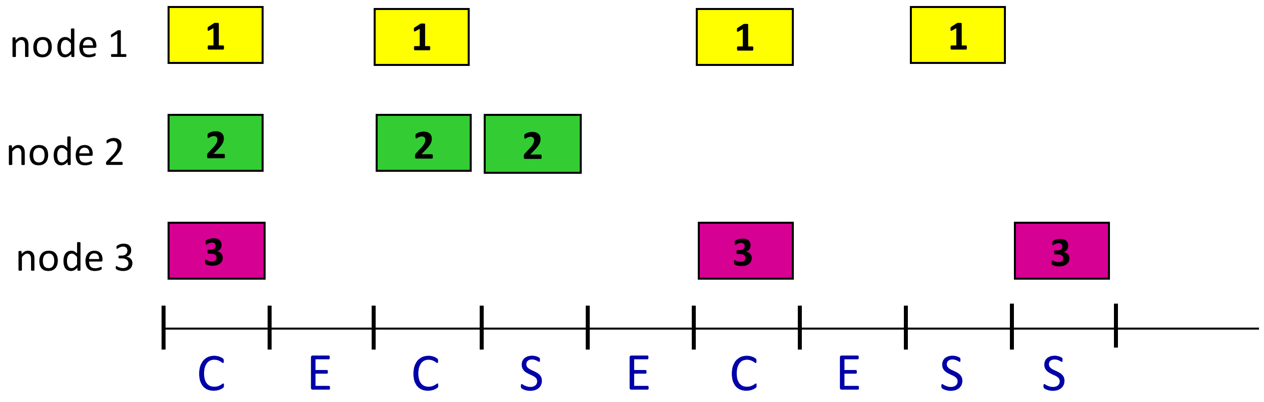
- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

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 probability *p* until success

randomization – *why*?

Slotted ALOHA



C: collision
S: success
E: empty

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