Introduction to Networking

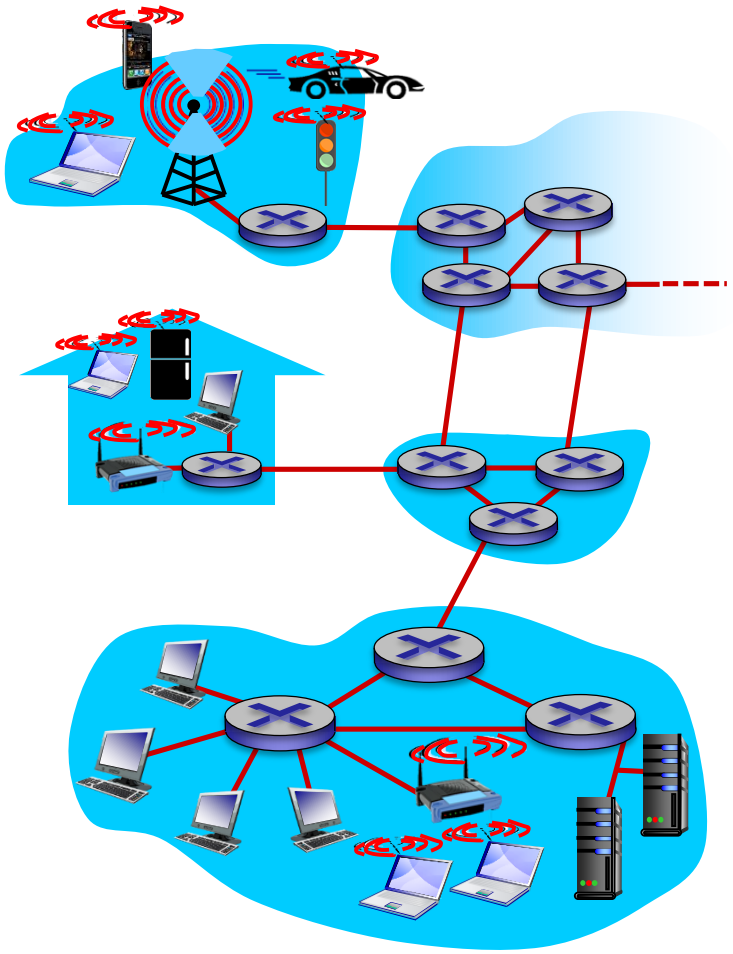
CAN201 – Lecture 9

Lecturer: Dr. Wenjun Fan

Lecture 9 – Link Layer (1)

- **Roadmap**
  1. Services of link layer
  2. Error detection and correction
  3. Multiple access protocols
  4. Addressing and ARP
  5. Ethernet

What's Link Layer



- **Data-Link Layer has responsibility of**
  - transferring datagram
  - from one node to physically adjacent node
  - over a link
- **Node:**
  - Hosts and routers
- **Link:**
  - communication channel
  - connection adjacent nodes
- **Layer-2 packet: frame**
  - Encapsulates datagram

Data Unit

Application Layer

Transport Layer

Network Layer

Link Layer

Physical Layer

?

Datagram

Frame

Message

Segment

Bit (or Signal)

Link layer services

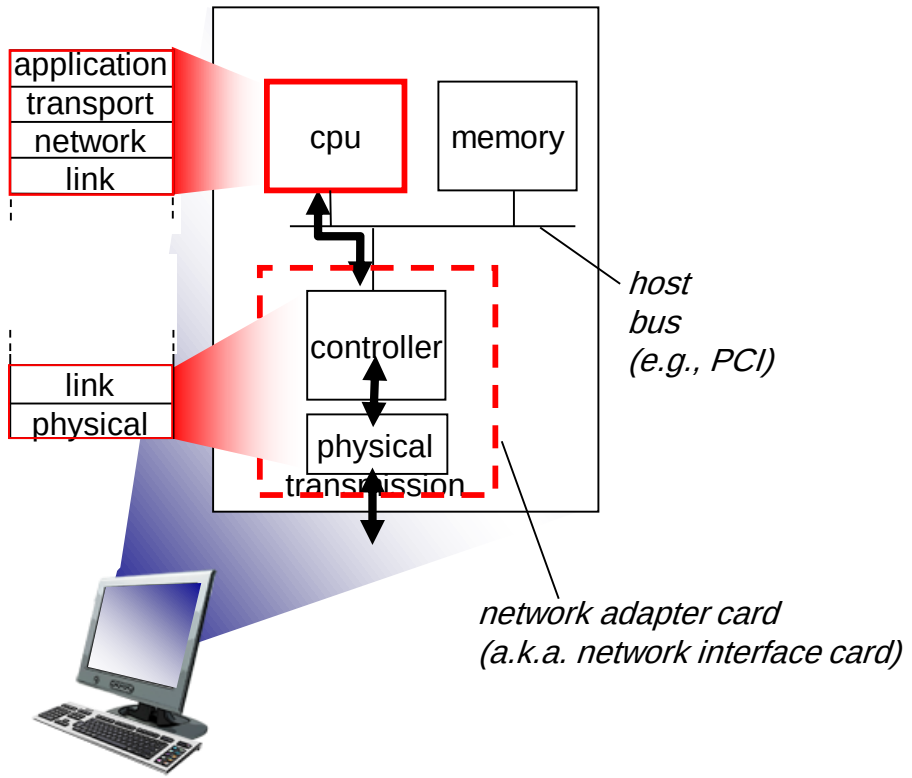
- **Framing**
  - Encapsulate datagram into frame: adding header, trailer
  - Different formats for different protocols
- **Link access**
  - Medium Access Control (MAC) protocol is used to transmit a frame
  - Point to point link or broadcast link
  - MAC addresses used in frame to identify source and destination
    - MAC address vs. IP address

Link layer services

- **Reliable delivery (RD)**
  - Low bit-error link (fiber, coax, some twisted pair): RD seldom used
  - High error rates wireless links: RD often used
    - Recall which protocol on which layer also provides reliability?
- **Error detection and correction**
  - Error caused by signal attenuation, noise…
  - Receiver detects error: retransmission or correction
  - Correction: corrects the bit error without retransmission

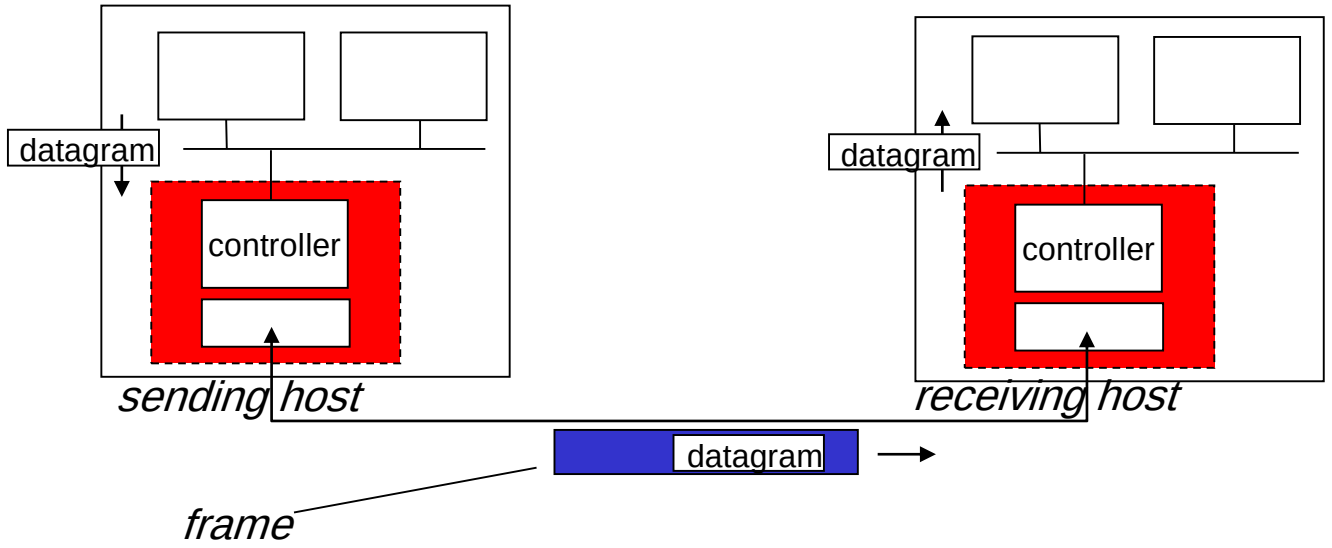
Where is the link layer implemented?

- **In each and every host**
- **Link layer implemented in "Adapter" or on a chip**
  - Ethernet card, Wifi 802.11 card, or chipset
  - For link and physical layers
- **Attached into host's system (motherboard) buses**
  - USB / PCI / Thunderbolt …
- **Hardware / software / firmware**



rdt: reliable data transfer protocol

Adaptors communicating



- Sending side:
  - Encapsulates datagram in frame
  - Adds error checking bits, rdt, flow control, etc.
- Receiving side
  - Looks for errors, rdt, flow control, etc.
  - Extracts datagram, passes to upper layer at receiving side

Lecture 9 – Link Layer (1)

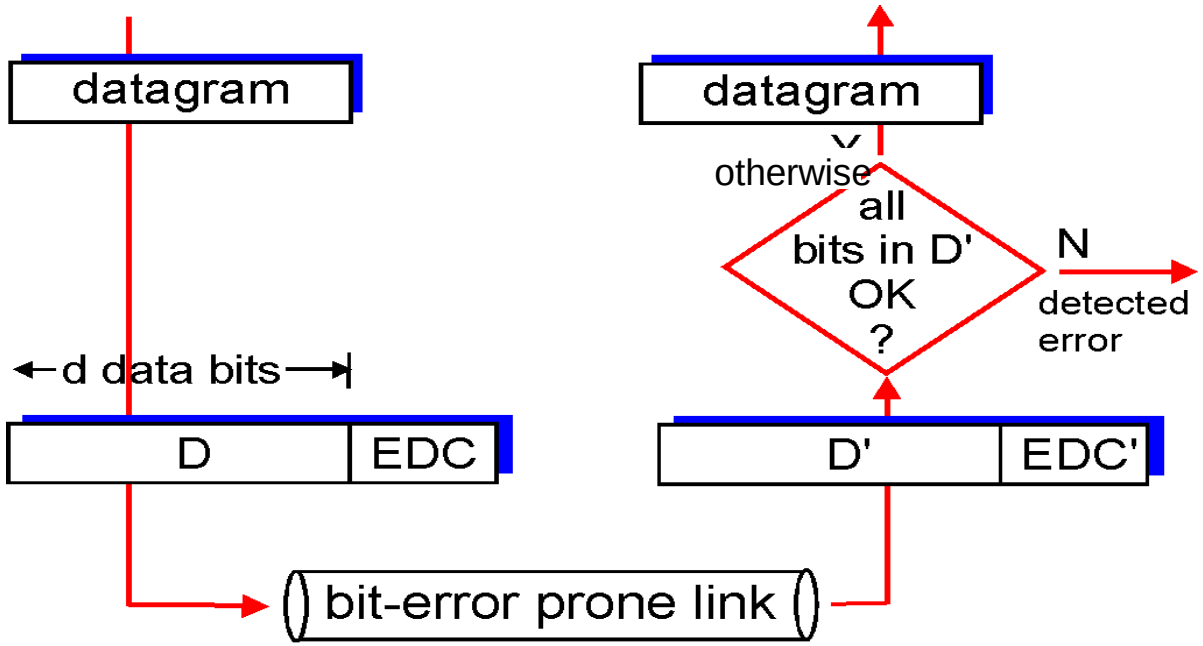
- **Roadmap**
  1. Services of link layer
  2. Error detection and correction
  3. Multiple access protocols
  4. Addressing and ARP
  5. Ethernet

Error detection

D = Data protected by error checking, including header fields

EDC= Error Detection and Correction bits (redundancy)

- Error detection not 100% reliable!
  - protocol may miss some errors, but rarely
  - larger EDC field (more sophisticated) yields better detection and correction (but any problem?)

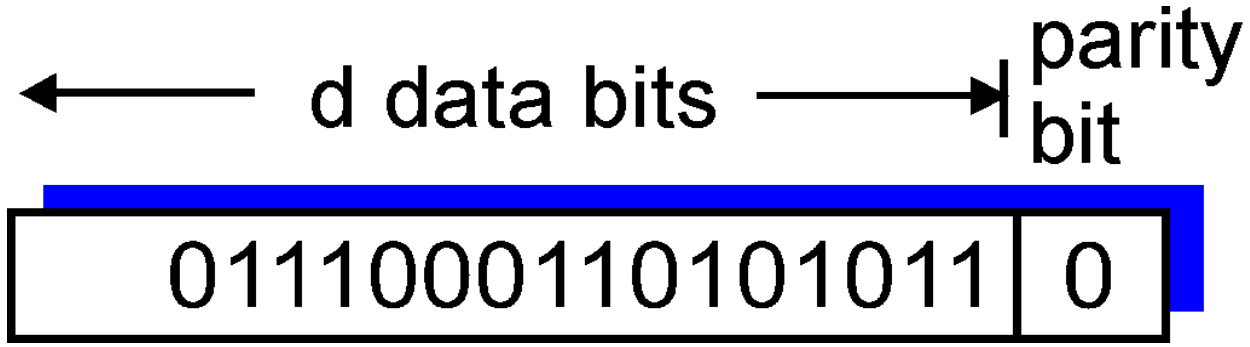


Three techniques for detecting errors

- **Parity Checks**
  - basic ideas
- **Checksum**
  - used in transport layer
- **Cyclic Redundancy Checks (CRC)**
  - used in link layer in an adapter

Parity checking – Single bit parity

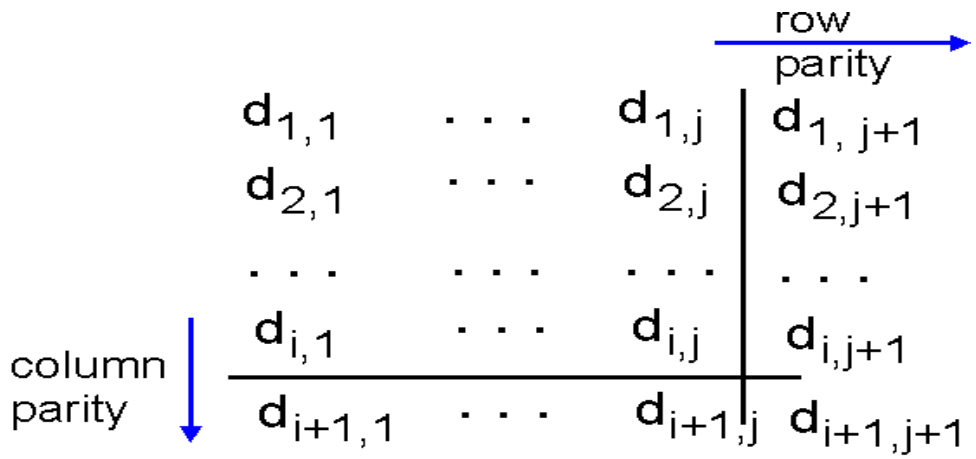
- **Even/Odd parity scheme:**
  - Add an additional bit
  - Total number of 1 (D+1) is even/odd
- **Detect single bit errors**

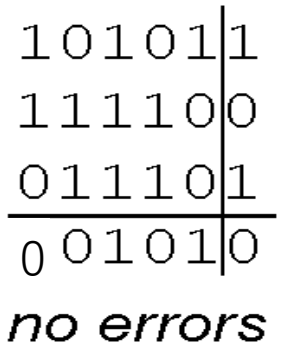


Even scheme
or odd scheme?

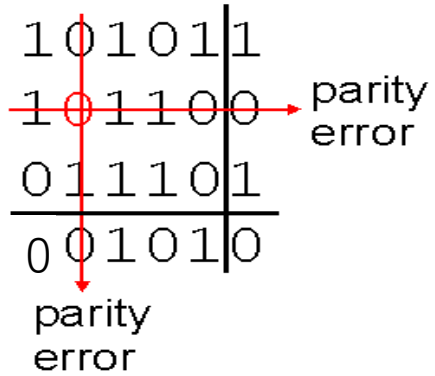
Parity checking – 2D parity

- **Detect and correct single bit errors**
- **Two-dimensional even parity**





no errors



correctable single bit error

Checksum

In the TCP and UDP protocols, the Internet checksum is computed over all fields (header and data fields included).
In IP, the checksum is computed over the IP header (since the UDP or TCP segment has its own checksum).

- **Goal: detect "errors" in transmitted segment**

Sender:

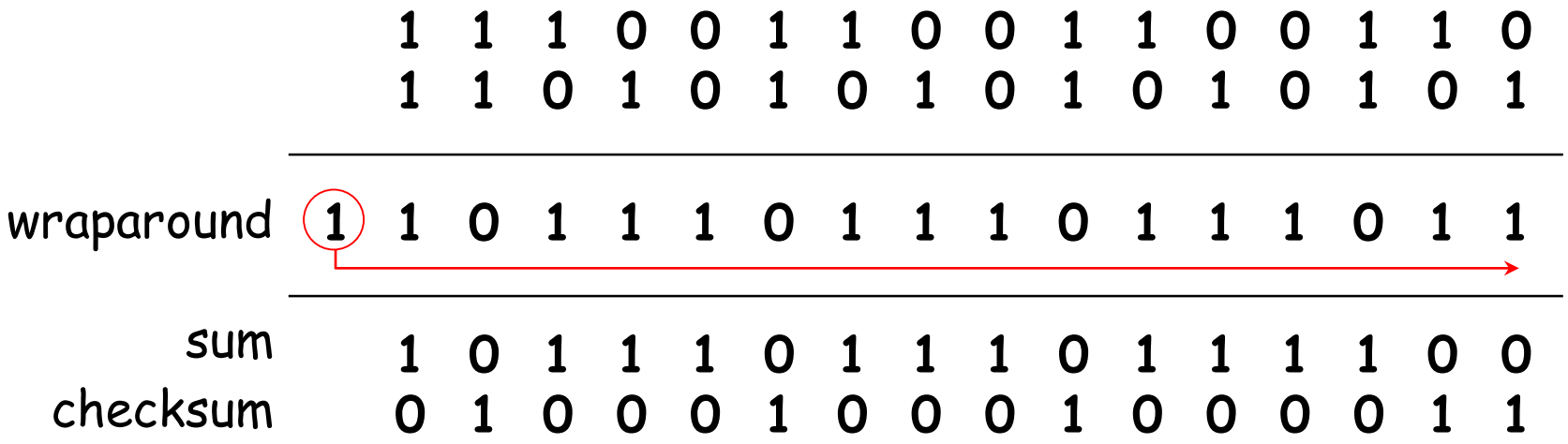
- **Treat segment contents, including header fields, as sequence of 16-bit integers**
- **Checksum: addition (one's complement sum) of segment contents**
- **Sender puts checksum value into (e.g., UDP) checksum field**

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?

Internet checksum: 1s complement

Example: add two 16-bit integers



Note: wraparound - when adding numbers, a carryout from the most significant bit needs to be added to the result

Cyclic redundancy check (CRC)

- **more powerful error-detection coding**
- **widely used in practice (Ethernet, 802.11 WiFi, ATM)**
- **all CRC calculations are done in modulo-2 arithmetic (without carries in addition or borrows in subtraction).**
- **modulo-2 Arithmetic**
  - Addition
  - Subtraction
  - Bitwise exclusive-or (XOR)

1011 + 0101 = ?

1011 - 0101 = ?

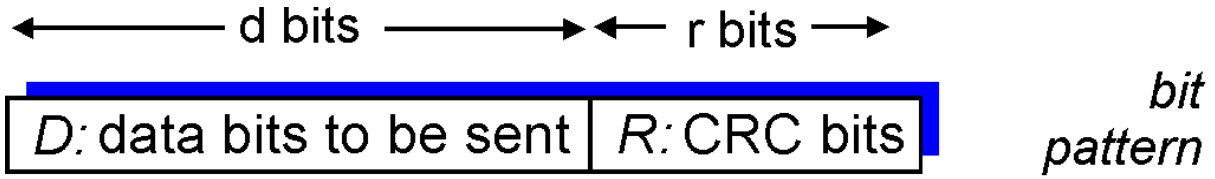
1011 XOR 0101= ?

Cyclic redundancy check (CRC)

- **view Data bits, D, as a binary number**
- **choose an r+1 bit pattern (generator), G,**
  - both sender and receiver agree on that
- **goal: find r CRC bits, (remainder), R, such that**
  - a d+r bit pattern <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

G is given for both sender and receiver. The most significant (leftmost) bit of *G* must be 1

R needs to be computed by the sender.



bit pattern

$D * 2^r$ XOR R

mathematical formula

CRC example

Want:

$D \cdot 2^r$ XOR R = nG

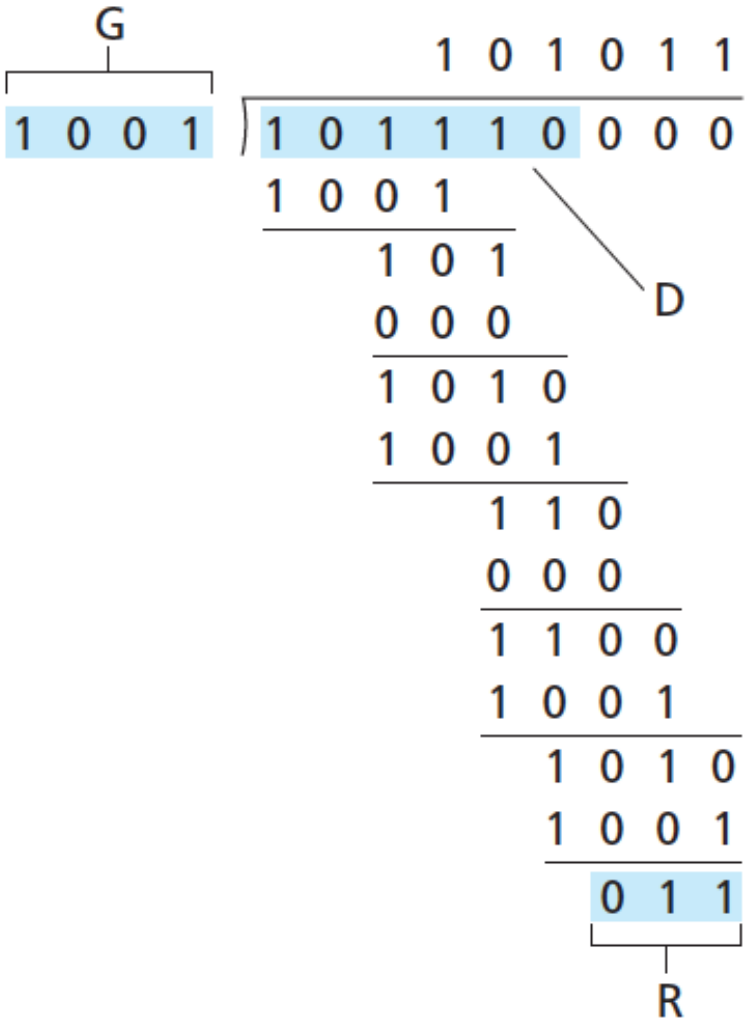
Equivalently:

$D \cdot 2^r$ = nG XOR R

Equivalently:

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

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



Eventual pattern to be sent:
D * 2^r XOR R = ?

CRC Standards for G

CRC-8	X^8+X^2+X+1	0X107	
CRC-12	$X^{12}+X^{11}+X^3+X^2+X+1$	0X180F	telecom systems
CRC-16	$X^{16}+X^{15}+X^2+1$	0X18005	Bisync, Modbus, USB, ANSI X3.28, SIA DC-07, many others; also known as CRC-16 and CRC-16-ANSI
CRC-CCITT	$X^{16}+X^{12}+X^5+1$	0X11021	ISO HDLC, ITU X.25, V.34/V.41/V.42, PPP-FCS
CRC-32	$X^{32}+X^{26}+X^{23}+X^{22}+X^{16}+X^{12}+X^{11}+X^{10}+X^8+X^7+X^5+X^4+X^2+X+1$	0x104C11DB7	ZIP, RAR, IEEE 802 LAN/FDDI, IEEE 1394, PPP-FCS
CRC-32C	$X^{32}+X^{28}+X^{27}+X^{26}+X^{25}+X^{23}+X^{22}+X^{20}+X^{19}+X^{18}+X^{14}+X^{13}+X^{11}+X^{10}+X^9+X^8+X^6+1$	0x11EDC6F41	iSCSI, SCTP,

Lecture 9 – Link Layer (1)

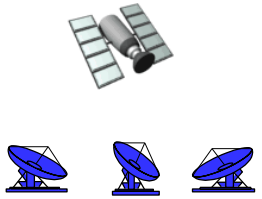
- **Roadmap**
  1. Services of link layer
  2. Error detection and correction
  3. Multiple access protocols
  4. Addressing and ARP
  5. Ethernet

Multiple access links, protocols

- **Two types of links:**
  - Point to point
    - PPP for dial-up access
    - PPPOE: Point to point protocol over Ethernet
  - Broadcast: shared wire or medium
    - Old-fashioned Ethernet
    - Upstream HFC
    - WLAN – 802.11

shared wire (e.g., cabled Ethernet)

shared RF (e.g., 802.11 WiFi)



shared RF (satellite)

humans at a cocktail party (What we share?)

Multiple access protocols

- **One single shared broadcast channel**
- **Two or more simultaneous transmissions by nodes: interference**
  - *collision* if node receives two or more signals at the same time

    in a sense, the signals of the colliding frames become inextricably tangled together.
    all the frames involved in the collision are lost, and the broadcast channel is wasted during the collision interval.

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 (desirable characteristics):

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 (and instantaneous) rate R/M.
3. fully decentralized:
   - no special node to coordinate transmissions
   - no synchronization of clocks, slots
4. Simple, inexpensive to implement.

MAC protocols: taxonomy

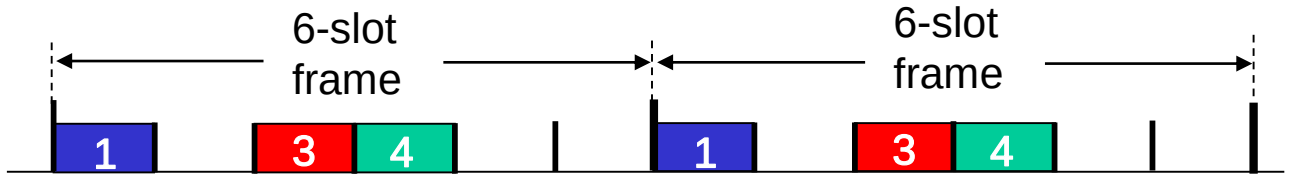
Three broad classes:

- ***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
- ***"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

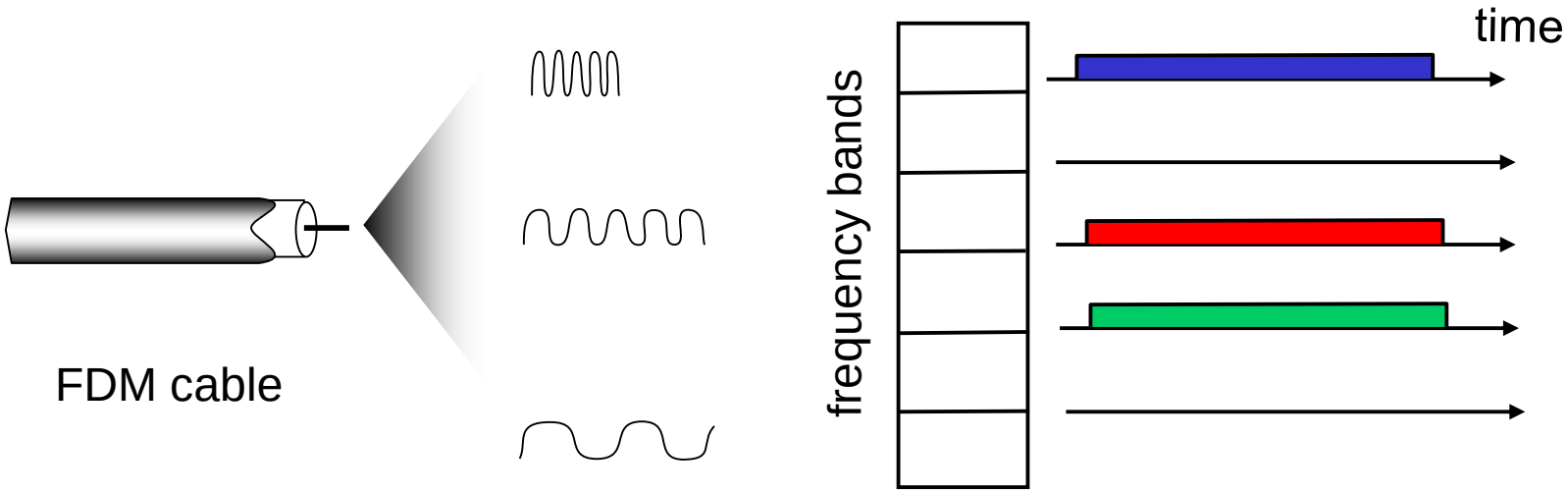
- Access to channel in "rounds"
- Each station/node gets fixed length slot (length = packet transmission time) in each round
- Unused slots go idle
- Example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



Channel partitioning MAC protocols: FDMA

FDMA: frequency division multiple access

- Channel spectrum divided into frequency bands
- Each station/node 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



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

https://en.wikipedia.org/wiki/ALOHAnet

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!

“Taking turns” MAC protocols

Channel partitioning MAC protocols:

- **Pro:** share channel *efficiently* and *fairly* at high load
- **Con:** inefficient at low load, delay in channel access, 1/N bandwidth allocated even if only 1 active node!

Random access MAC protocols

- **Pro:** efficient at low load, single node can fully utilize channel
- **Con:** high load, collision overhead

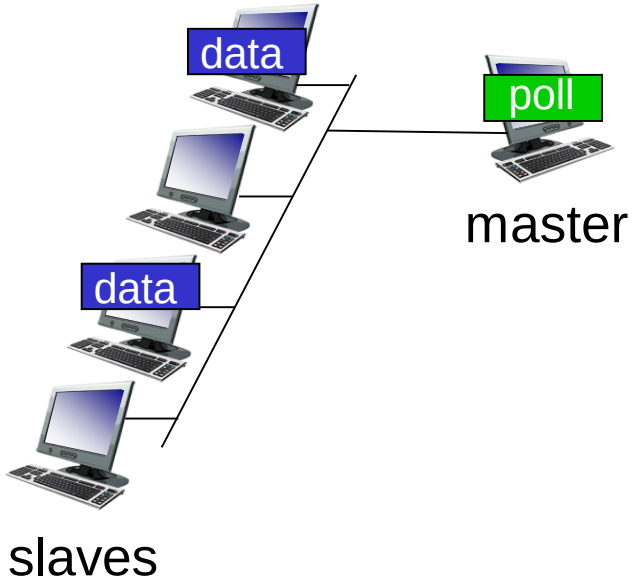
“Taking turns” protocols

look for best of both worlds!

“Taking turns” MAC protocols

Polling:

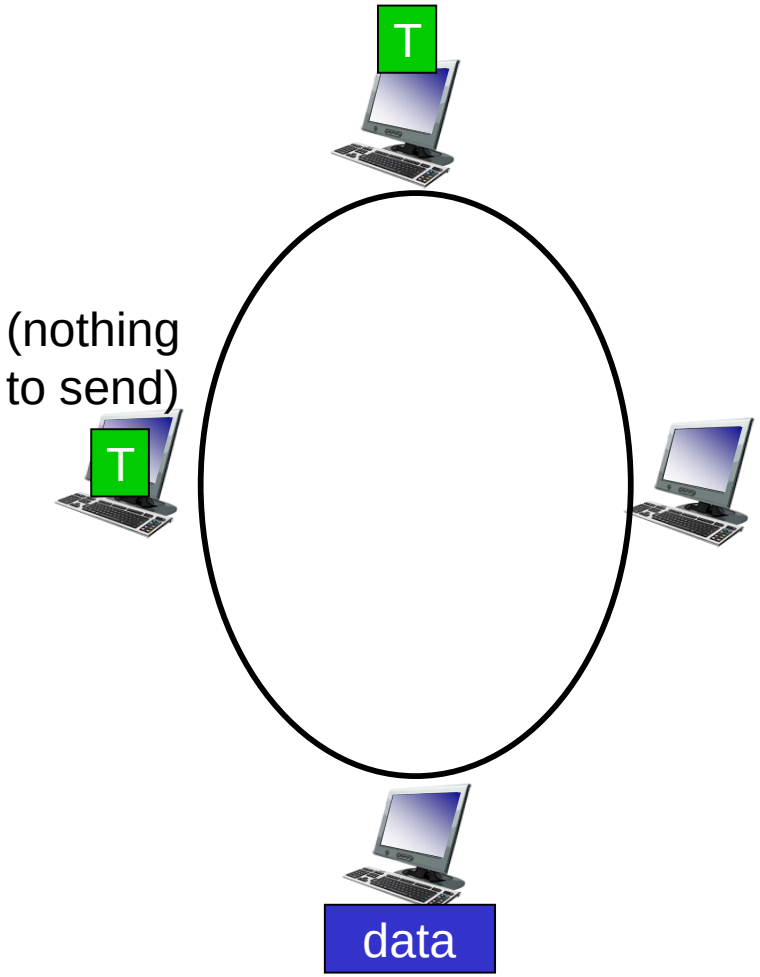
- **Master node “invites” slave nodes to transmit in turn**
- **Typically used with “dumb” slave devices**
- **Drawbacks:**
  - Polling overhead
  - Single point of failure (master)



"Taking turns" MAC protocols

Token passing:

- Control *token* passed from one node to next sequentially.
- Token message
- Drawbacks:
  - Token overhead
  - Node crashes/fails to release token



Summary of MAC protocols

- ***Channel partitioning,* by time, frequency or code**
  - Time Division, Frequency Division
- ***Random access* (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 central site, token passing
  - Bluetooth, FDDI, token ring

Lecture 9 – Link Layer (1)

- **Roadmap**
  1. Services of link layer
  2. Error detection and correction
  3. Multiple access protocols
  4. Addressing and ARP
  5. Ethernet

MAC addresses and ARP

- **IP address**
  - IPv4 (32 bits) and IPv6 (128 bits)
  - Network-layer address
  - Layer-3 forwarding

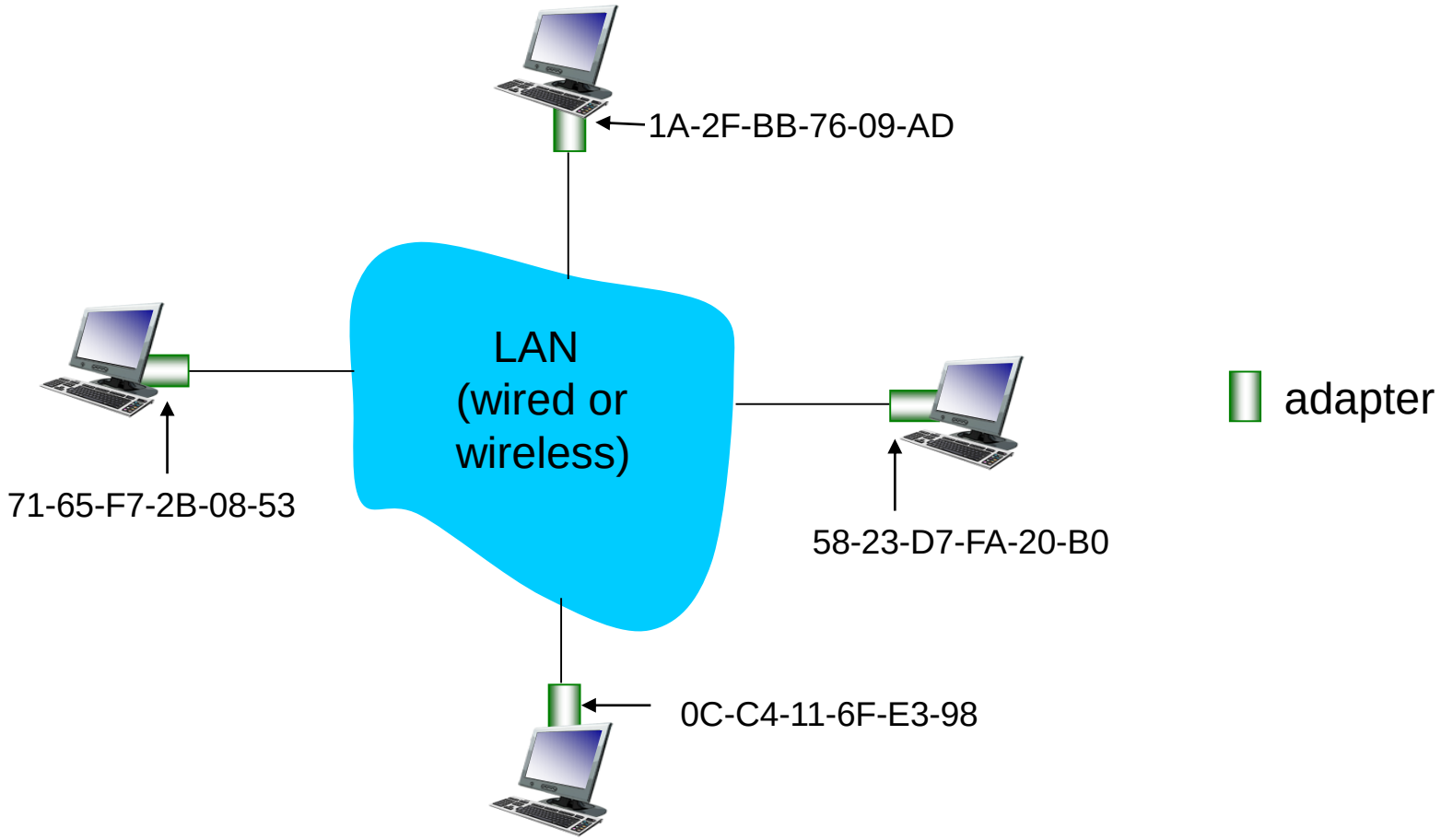
- **MAC (LAN) address**
  - 48 bits (e.g., 12-34-56-78-90-AB)

    hexadecimal (base 16) notation
    (each "numeral" represents 4 bits)

  - Burned in NIC ROM
  - Used "locally" get frame from one interface to another physically-connected interface (same network, in IP-addressing sense)

MAC addresses and ARP

Each adapter on LAN has unique *LAN* address



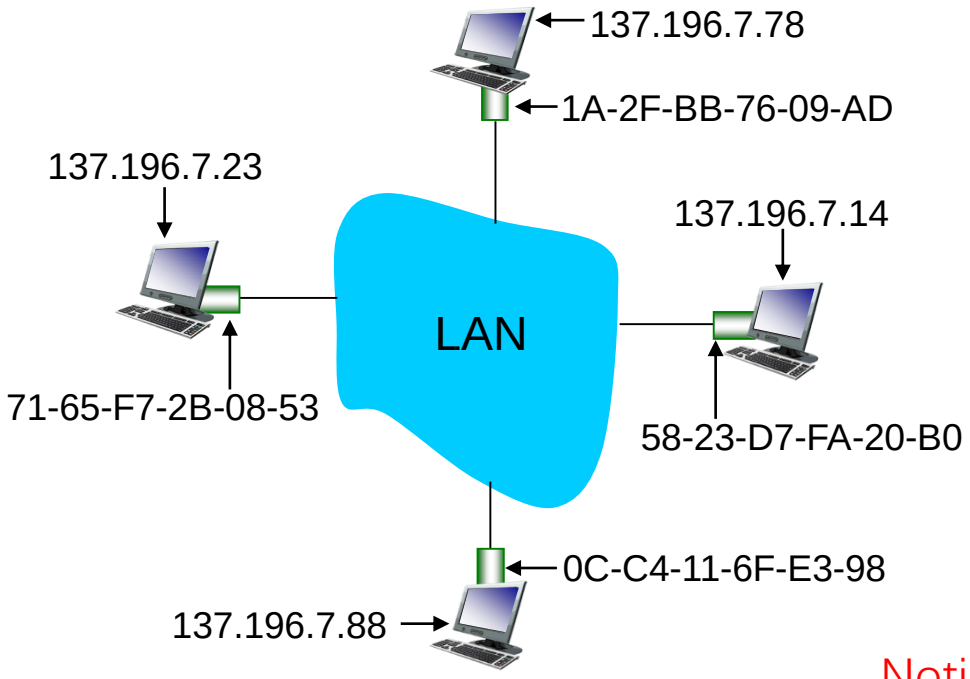
MAC addresses (more)

- **MAC address allocation administered by IEEE**
- **Manufacturer buys portion of MAC address space (to assure uniqueness)**
- **MAC flat address → portability**
  - Can move LAN card from one LAN to another
- **IP hierarchical address *not* portable**
  - IP Address depends on IP subnet to which node is attached
- **Analogy for understanding:**
  - MAC address: like ID card or passport Number
  - IP address: like postal address

IPv6 vs. MAC address ?

ARP: address resolution protocol

- How to determine interface's MAC address, if knowing its IP address?



ARP table: **each IP node (host, router) on LAN has the table**

- An IP-to-MAC address mappings for LAN nodes:

  < IP address; MAC address; TTL>

- A TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

Notice that an ARP table is not necessary to contain an entry for every host and router on the subnet; some may have never been entered into the table, and others may have expired.

ARP protocol: forwarding in the same LAN

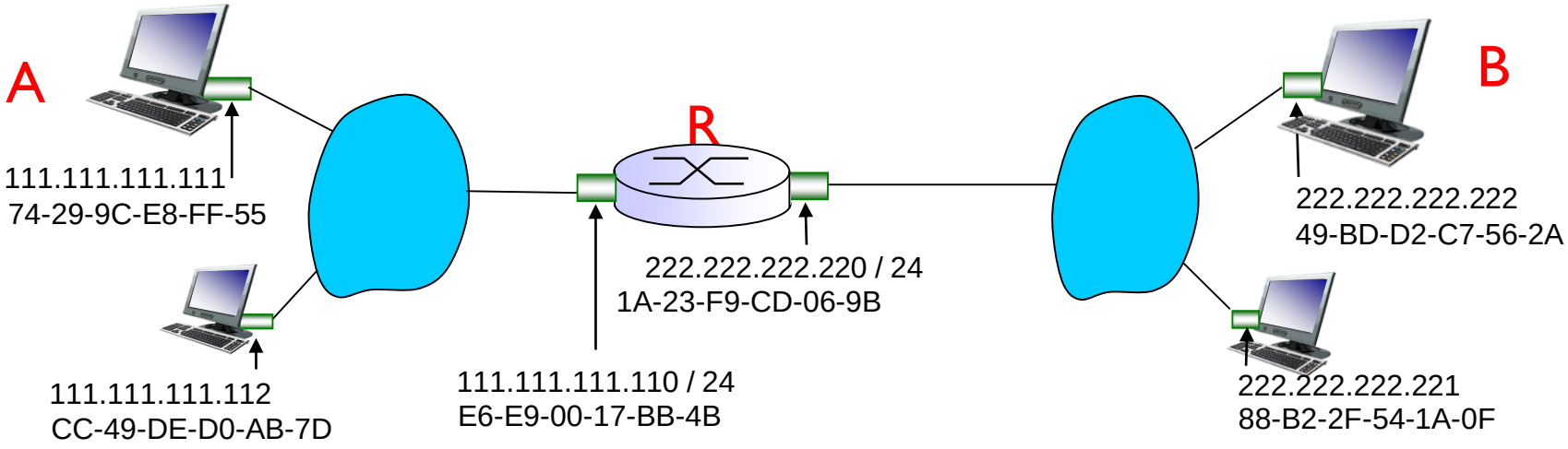
- **A wants to send datagram to B**
  - B's MAC address not in A's ARP table.
- **A broadcasts ARP query packet, containing B's IP address**
  - Destination MAC address = FF-FF-FF-FF-FF-FF
  - All nodes on LAN receive ARP query
- **B receives ARP packet, replies to A with its (B's) MAC address**
  - Frame sent to A's MAC address (unicast)

- **A caches (saves) IP-to-MAC address pair in its ARP table (until information becomes old or times out)**
  - Soft state: information that times out (goes away) unless refreshed
- **ARP is "plug-and-play":**
  - Nodes create their ARP tables automatically *without intervention from net administrator*

Addressing: routing to another LAN

Walkthrough: send datagram from A to B via R

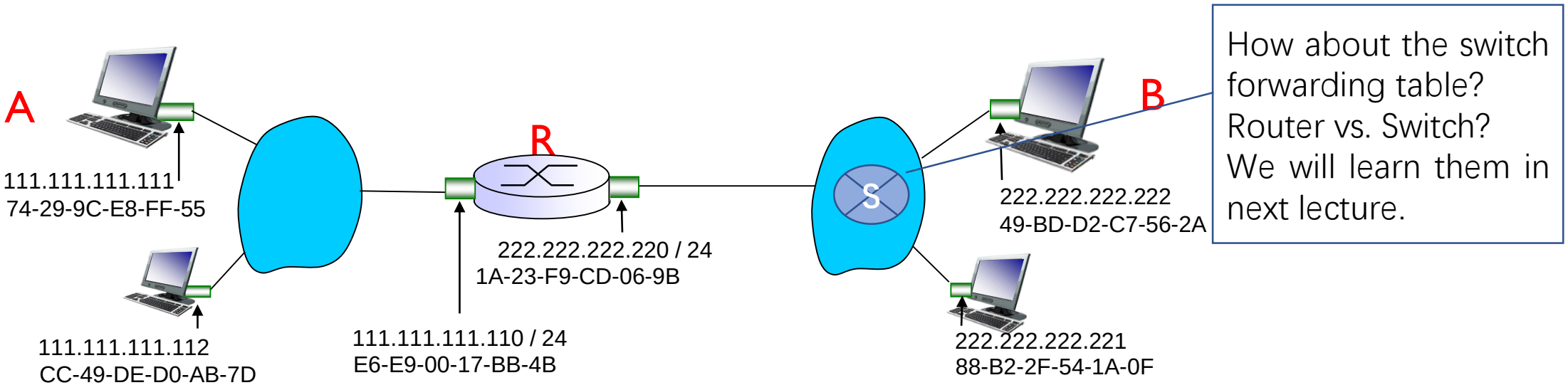
- Focus on addressing – at IP (datagram) and MAC layer (frame)
- Assume A knows B's IP address
- Assume A knows IP address of first hop router, R (how?)
- Assume A knows R's MAC address (how?)
- And how does R obtain B's MAC address?



Addressing: routing to another LAN

Walkthrough: send datagram from A to B via R

- Focus on addressing – at IP (datagram) and MAC layer (frame)
- Assume A knows B's IP address
- Assume A knows IP address of first hop router, R (how?)
- Assume A knows R's MAC address (how?)



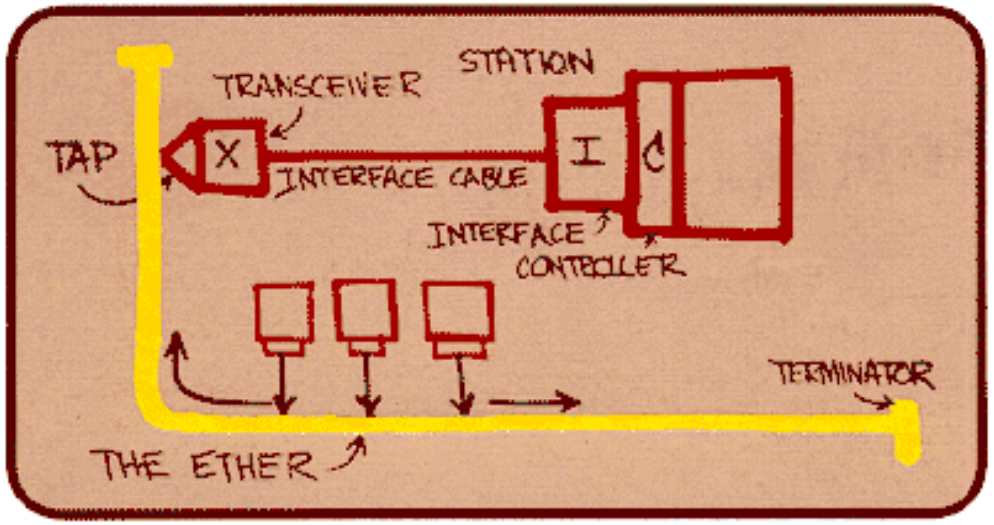
Lecture 9 – Link Layer (1)

- **Roadmap**
  1. Services of link layer
  2. Error detection and correction
  3. Multiple access protocols
  4. Addressing and ARP
  5. Ethernet

Ethernet

"dominant" wired LAN technology:

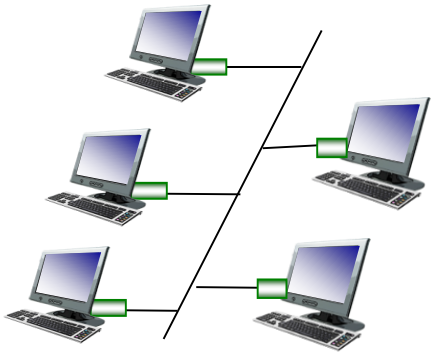
- Single chip, multiple speeds
  - 10Mbps, 100Mbps, 1Gbps, 10Gbps
- First widely used LAN technology
- Simpler, cheap



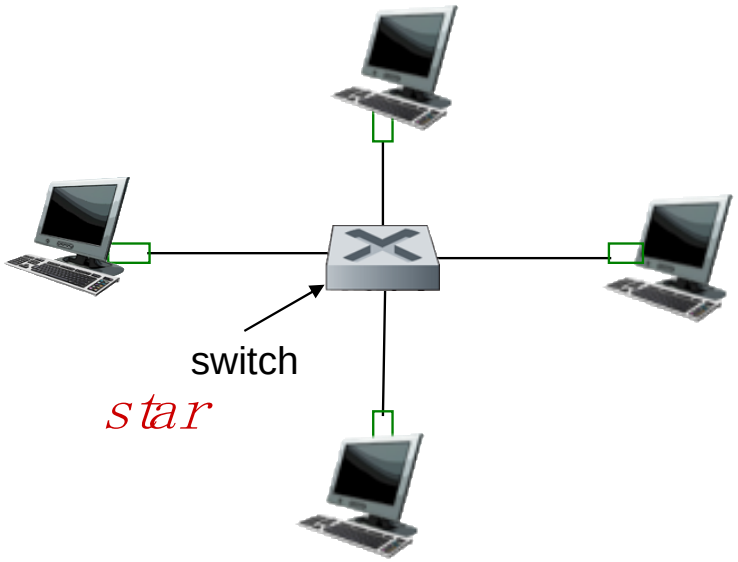
Metcalfe's Ethernet sketch

Ethernet: physical topology

- ***Bus:*** **popular through mid 90s**
  - All nodes in same collision domain (can collide with each other)
- ***Star:*** **prevails today**
  - Active *switch* in center
  - Each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)



bus: coaxial cable



Ethernet frame structure

- **Sending adapter encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame**

preamble: 8-byte

- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

Ethernet frame structure

- ***Addresses:*** **6 byte source, destination MAC addresses**
  - If adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
  - Otherwise, adapter discards frame
- ***Type:*** **indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)**
- ***CRC:*** **cyclic redundancy check at receiver**
  - Error detected: frame is dropped

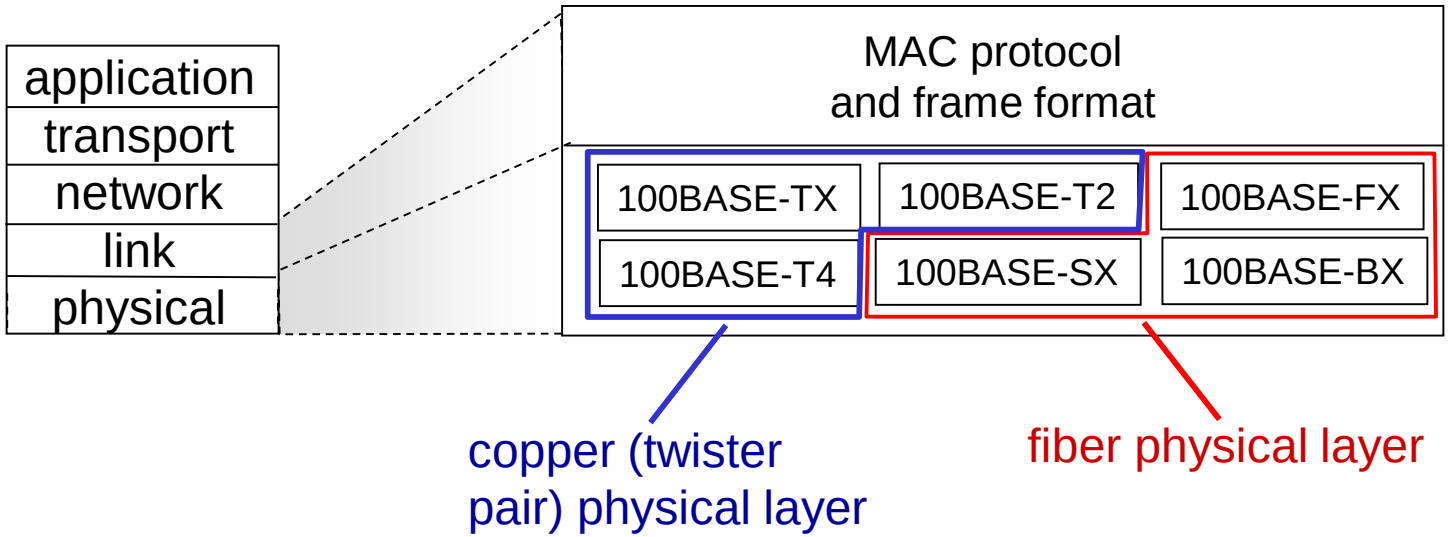
Ethernet: unreliable, connectionless

- ***Connectionless:*** **no handshaking between sending and receiving NICs**
- ***Unreliable:*** **receiving NIC doesn't send acks or nacks to sending NIC**
  - Data in dropped frames recovered only if initial sender uses higher layer rdt (e.g., TCP), otherwise dropped data lost
- **Ethernet's MAC protocol: unslotted *CSMA/CD with binary backoff (using with sensing channel idle)***

rdt: reliable data transfer protocol

802.3 Ethernet standards: link & physical layers

- ***Many*** **different Ethernet standards**
  - Common MAC protocol and frame format
  - Different speeds: 2 Mbps, 10 Mbps, 100 Mbps, 1Gbps, 10 Gbps, 40 Gbps
  - Different physical layer media: fiber, cable



Thanks.

- **Addresses**
  - Email: Wenjun.Fan@xjtlu.edu.cn
  - Office: EE 214
- **Office hours**
  - Monday: 12:00 – 13:00
  - Tuesday: 12:00 – 13:00