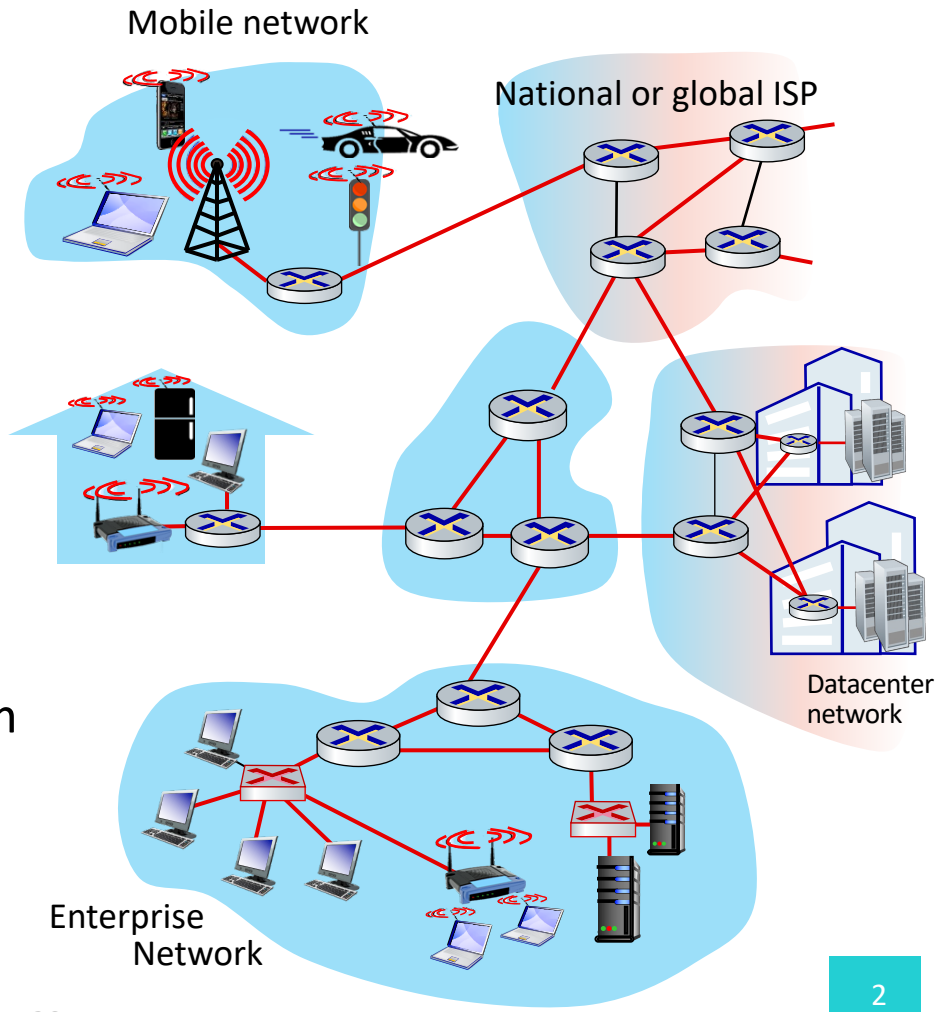
Data Link Layer

Introduction

- Hosts and routers: **Nodes**
- Communication channels that connect adjacent nodes along communication path: **Links**
  - Wired links
  - Wireless links
  - LANs
- Layer-2 packet: **Frame,** encapsulates datagram
- **Data-link layer** has responsibility of transferring datagram from one node to **physically adjacent** node over a link



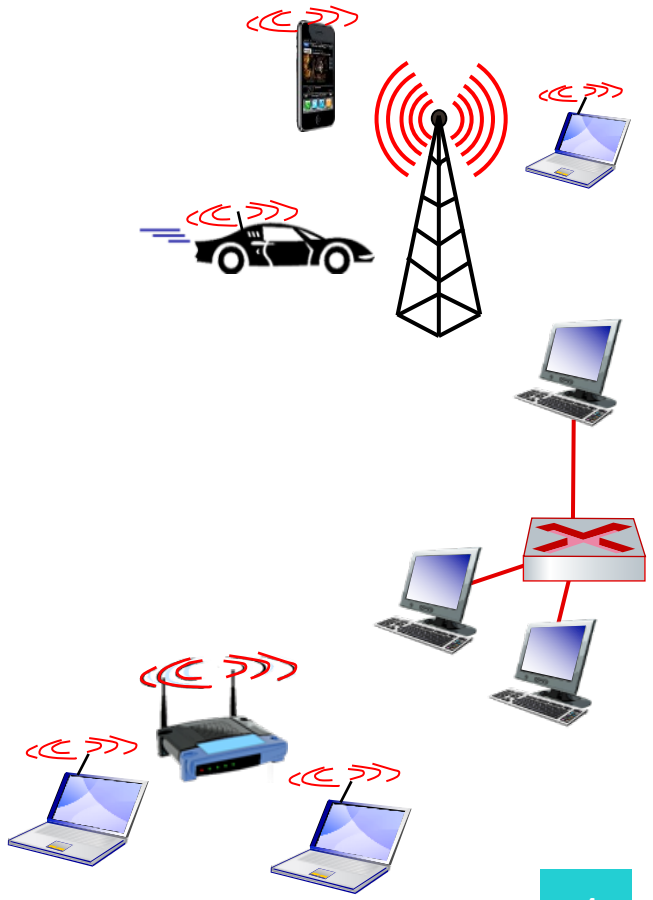
Introduction

- Datagram transferred by different link protocols over different links
  - Example: 802.11 on the first link
    Ethernet on the following links

- Each link protocol provides different services
  - Example: May or may not provide reliable data transfer over link

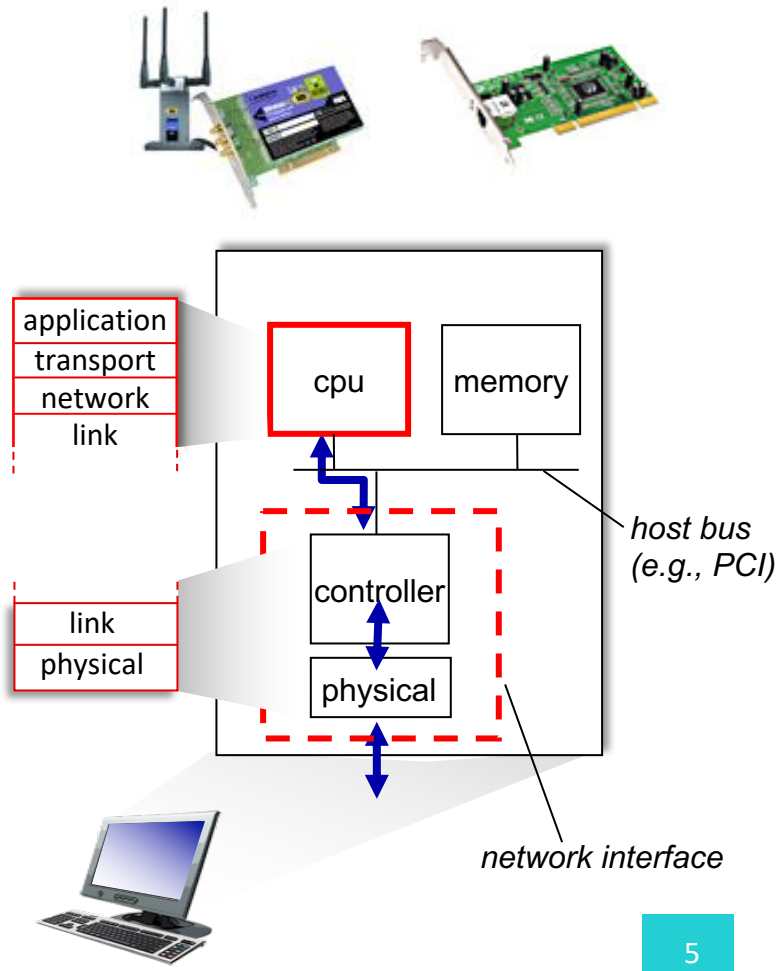
Link Layer Services

- **Framing**
  - Encapsulate datagram into frame (adding header and trailer)
- **Link Access**
  - Channel access if shared medium
  - MAC addresses used in frame headers to identify source, destination
    - Different from IP address!
- **Error Detection and Correction (EDC)**
- **Reliable delivery between adjacent nodes**
  - 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?



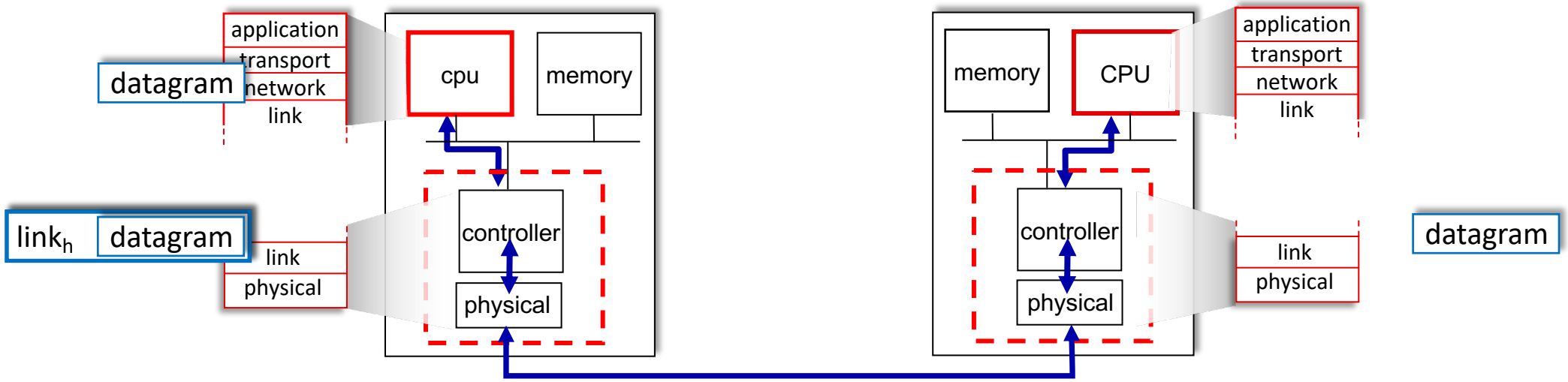
Implementation

- In each and every host
- Link layer implemented in adaptor (**Network Interface Controller:** NIC) or on a chip
  - implements link, physical layer
    - Ethernet card
    - 802.11 card
    - Ethernet chipset
- Attaches into system buses of host
- Combination of hardware, software, firmware



Adaptors Communicating

- Sending side
  - Encapsulates datagram in frame
  - Adds error checking bits, reliable data transfer, flow control, etc.



- Receiving side
  - Looks for errors, reliable data transfer, flow control, etc.
  - Extracts datagram, passes to upper layer at receiving side

Link Layer

- Error detection and error correction
- Multiple access protocols
- LANs
  - Addressing & ARP
  - Ethernet
  - Switches
  - VLANS

Link Layer: EDC

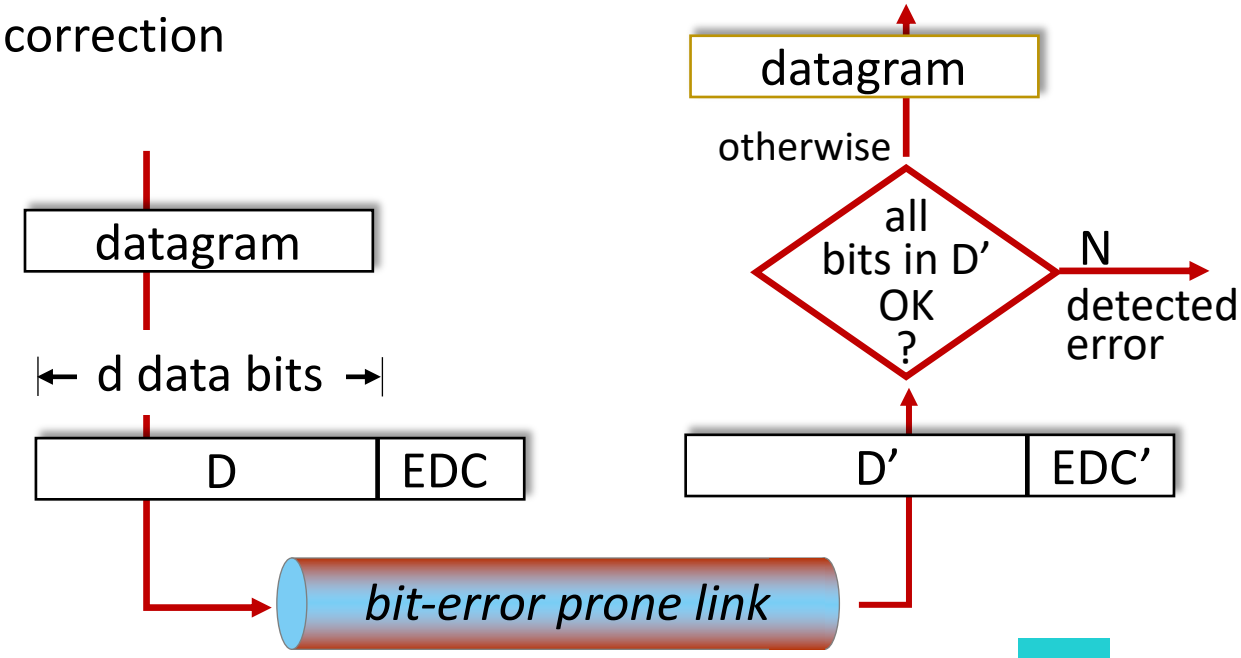
EDC: Error Detection & Correction

- **Error detection**
  - Errors caused by signal attenuation or noise.
  - Receiver detects presence of errors
    - Signals sender for **retransmission** and **drops** frame

- **Error correction**
  - Receiver identifies **and corrects** bit error(s) without resorting to retransmission

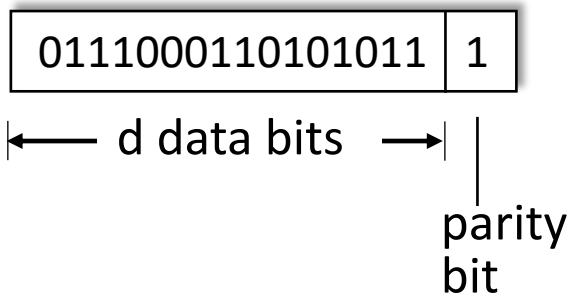
Error Detection

- 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



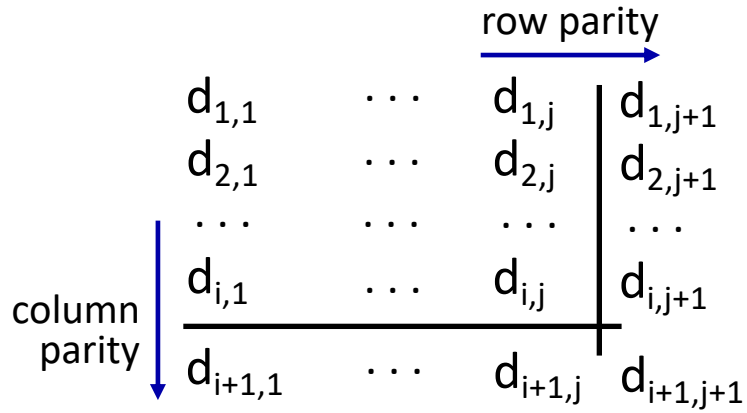
Parity Checking

- Count number of ones
  - Check the total number of bits (data+parity)
    - Even: Expecting even number bits with the value of one
    - Odd: Expecting odd number of bits with the value of one
  - Detect single bit errors
  - What about bursts?



Parity Checking

- Two-dimensional Parity: Single bit in each dimension
- Locating possible
  - Detect and correct single bit errors
- Still not very strong!



no errors:

1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
1	0	1	0	1	0

detected and correctable single-bit error:

1	0	1	0	1	1
1	0	1	1	0	0
0	1	1	1	0	1
1	0	1	0	1	0

parity error

parity error

Checksums

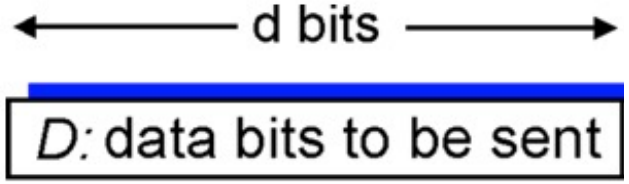
Goal: Detect errors (e.g. flipped bits) in transmitted packet

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

Cyclic Redundancy Check

- More powerful error-detection coding
- View data bits (number of bits: **d** and represented value: **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 (Ethernet, 802.11 WiFi)



Example: CRC

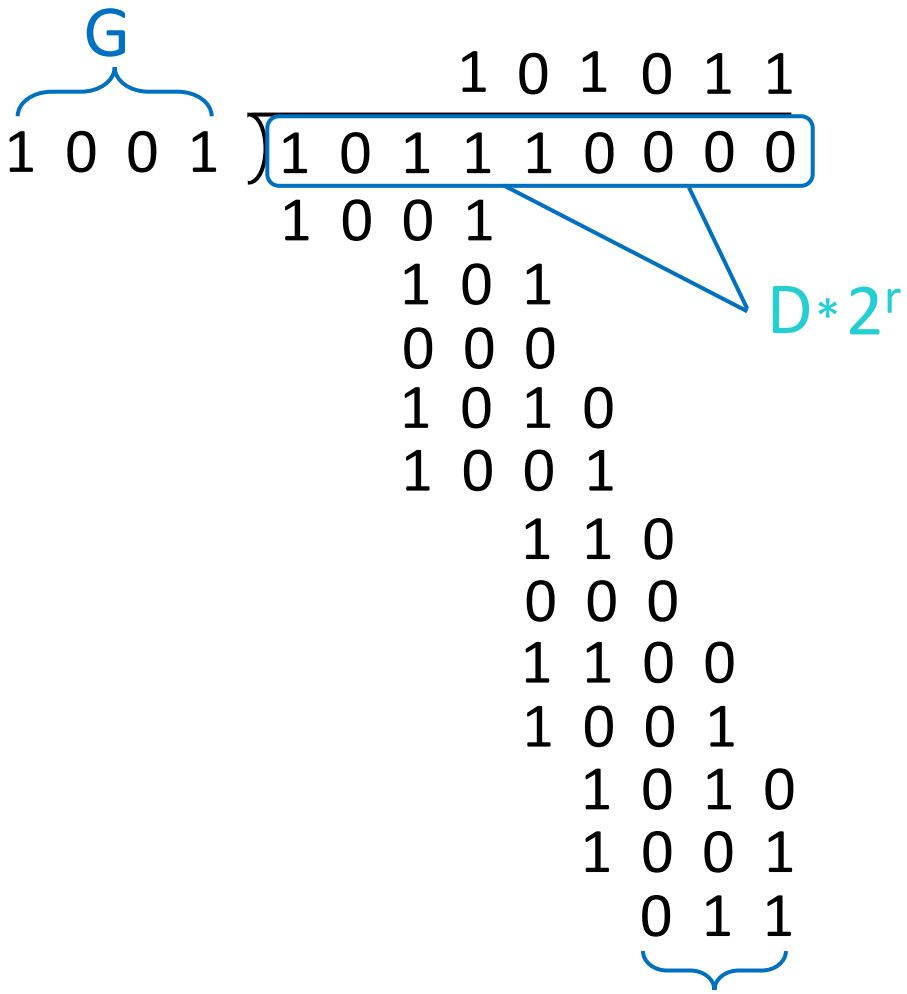
Want: $D.2^r \text{ XOR } R = nG$

Equivalently: $D.2^r = nG \text{ XOR } R$

Equivalently: If we divide $D.2^r$

By **G** want remainder **R** to satisfy:

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



Link Layer: Multiple Access

Multiple Access Links & Protocols

Two types of links

- Point-to-point
  - PPP: Point to Point Protocol
  - HDLC: High Level Data Link Control

- **Broadcast (shared wire or medium)**
  - Old-fashioned Ethernet
  - 802.11 wireless LAN

shared wire (e.g., cabled Ethernet)

shared RF (e.g., 802.11 WiFi)

shared RF (satellite)

humans at a cocktail party (shared air, acoustical)

Multiple Access Protocols

- 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

Ideal Multiple Access Protocol

Given: Broadcast channel of rate ***R bps***

Desiderata

- When one node wants to transmit, it can send at rate ***R***
- When M nodes want to transmit, each can send at average rate ***R/M***
- Fully decentralized
  - No special node to coordinate transmissions
  - No synchronization of clocks, slots
- Simple

MAC Protocols: Taxonomy

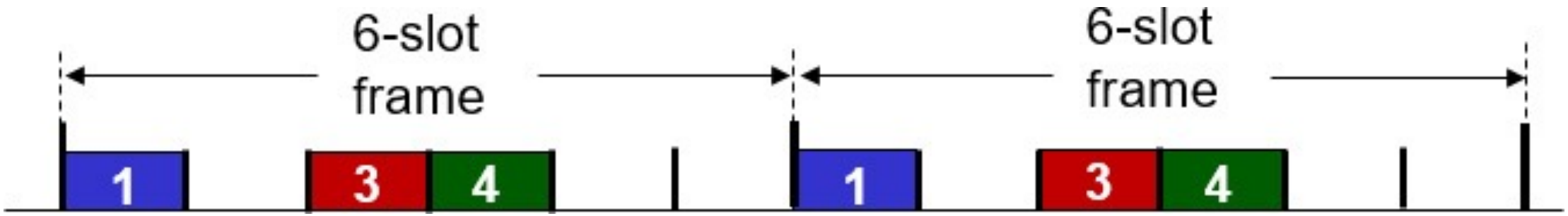
Three broad classes

- **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
- **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

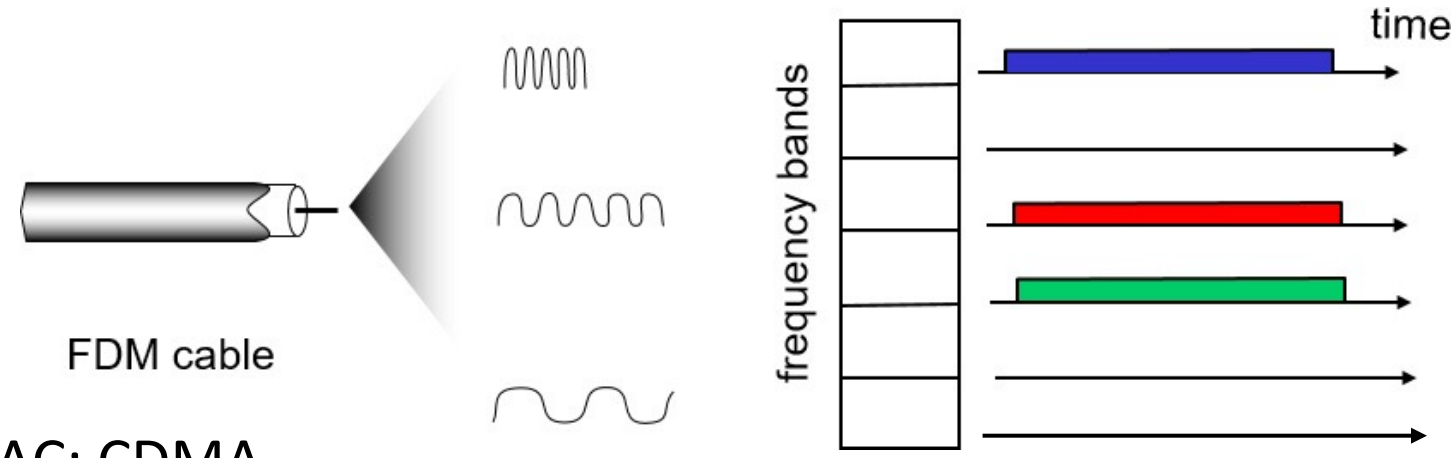
- Access to channel in rounds
- Each station gets fixed length slot (length = packet transmission time) in each round
- Unused slots go idle
- **Example:** 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



Channel Partitioning MAC Protocols: FDMA

FDMA: Frequency Division Multiple Access

- Channel spectrum divided into frequency bands
- Each station 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



Other channel Partitioning MAC: CDMA

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 time slots (time to transmit one frame)
- Nodes start to transmit only at beginnings of slots
- Nodes are synchronized
- If two or more nodes transmit in slot, all nodes detect collision

Slotted ALOHA

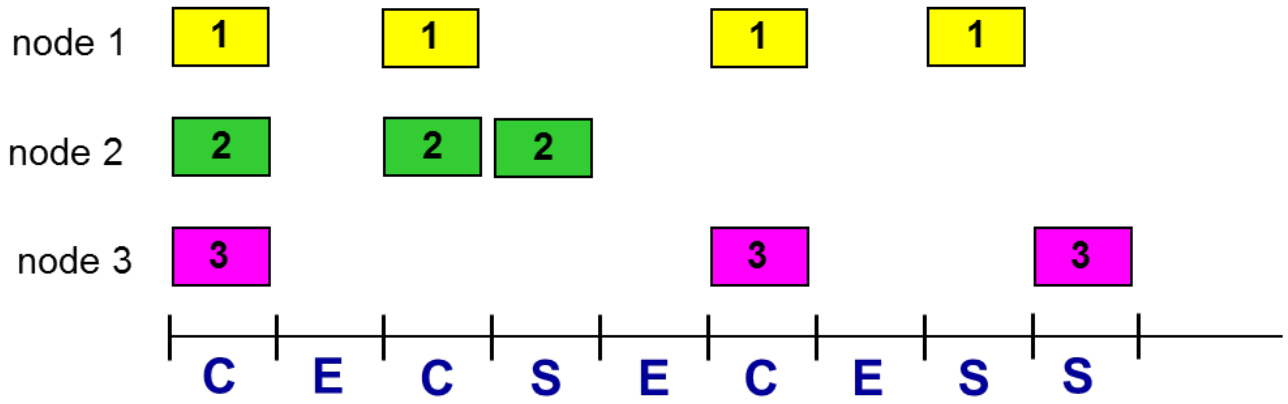
Operation

- 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

Slotted ALOHA

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)

- Suppose: ***N*** nodes with many frames to send. Each node transmits in slot with probability ***P***

- Probability that given node has success in a slot = $P(1-P)^{N-1}$

- Probability that **any** node has a success = $NP(1-P)^{N-1}$

Slotted ALOHA: Efficiency

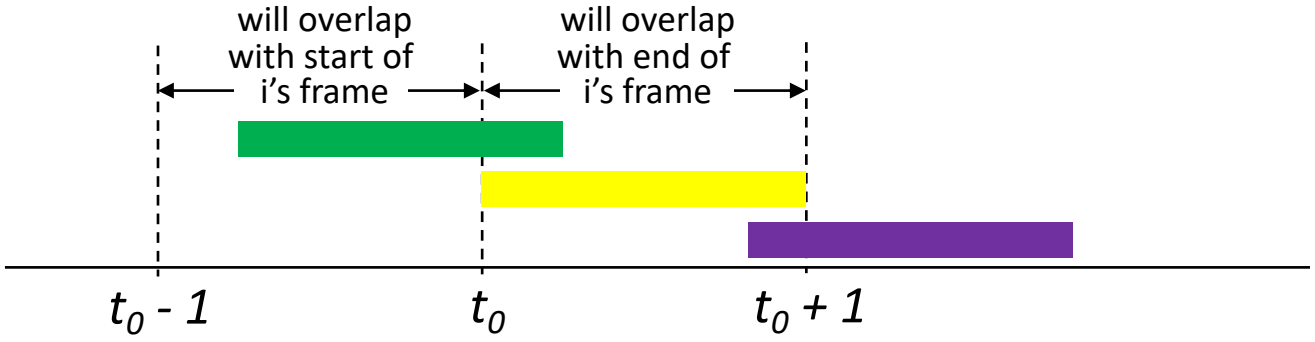
- **Max efficiency:** Find P* that maximizes $NP(1-P)^{N-1}$

- For many nodes, take limit of $NP^*(1-P^*)^{N-1}$
- As *N* goes to infinity, gives: P* $= \frac{1}{e} = .37$

- **At best:** Channel used for useful transmissions 37% of time!

Pure ALOHA

- Unslotted Aloha
  - Simpler
  - No synchronization



- When frame first arrives transmit immediately

- Collision probability increases:
  - Frame sent at t_0 collides with other frames sent in $[t_0-1,t_0+1]$

Pure ALOHA Efficiency

P(success by given node) = P(node transmits)

P(no other node transmits in $[t_0-1,t_0]$)

$= P \cdot (1-P)^{2(N-1)}$

… choosing optimum p and then letting N approach infinity:

$= 1/(2e) = .18$

Even worse than Slotted Aloha!

CSMA (Carrier Sense Multiple Access)

Sensing Protocols

- Human analogy: Do not interrupt others

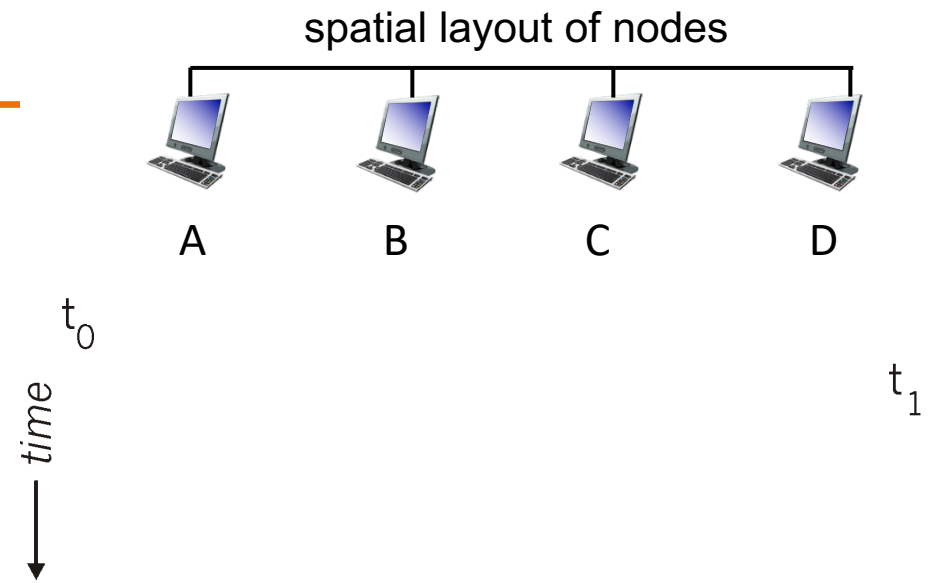
CSMA: Listen before transmit:

- **If channel sensed idle:** Transmit entire frame
- **If channel sensed busy:** Defer transmission

CSMA Collisions

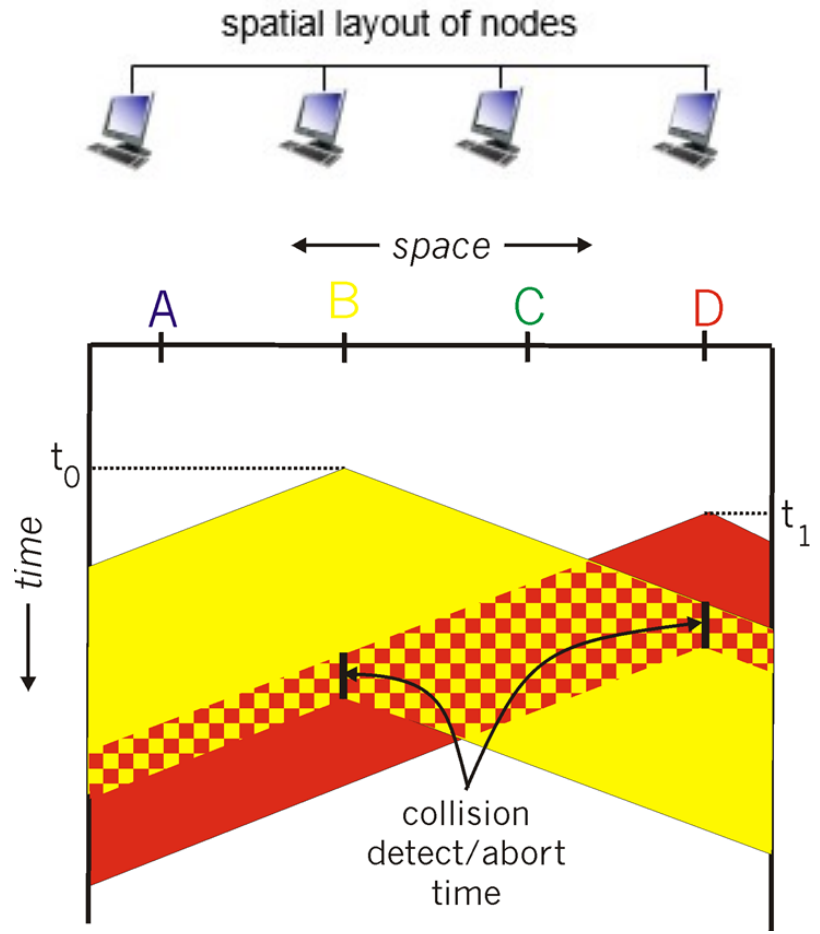
- **Collisions can still occur**
  - Propagation delay means two nodes may not hear each other's transmission

- **Collision**
  - Entire packet transmission time wasted
  - Distance & propagation delay play role in determining collision probability



CSMA Collision Detection

- Human analogy
  - The polite conversationalist



CSMA Collision Detection

CSMA/CD: Carrier sensing, deferral as in CSMA

- Collisions **detected** within short time
- Colliding transmissions aborted: Reduce 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

Ethernet CSMA/CD Algorithm

1. NIC receives datagram from network layer, creates frame

2. If NIC senses channel
   - If idle: Starts frame transmission.
   - If 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 *m*th collision, NIC chooses *K* at random from $\{0,1,2, \ldots, 2^m - 1\}$.
   - NIC waits Kx512 bit times, returns to Step 2
   - Longer backoff interval with more collisions

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!

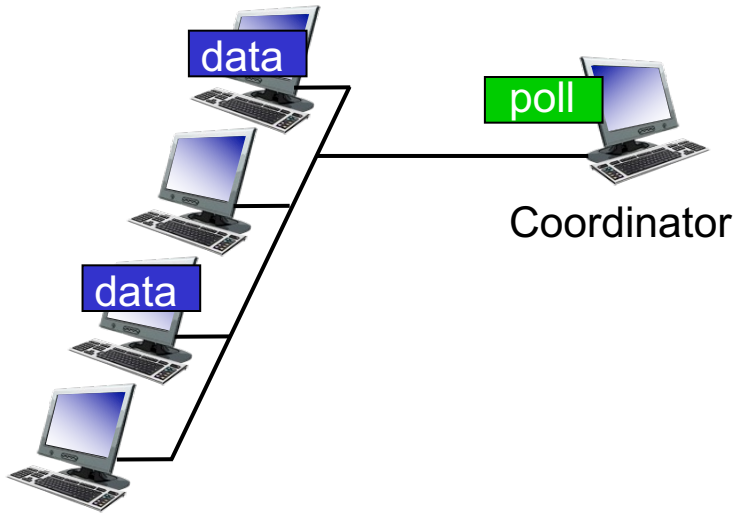
- **Random access MAC protocols**
  - Efficient at low load: Single node can fully utilize channel
  - High load: Collision overhead

- They look for the best of both worlds!

Taking-Turns MAC Protocols

Polling

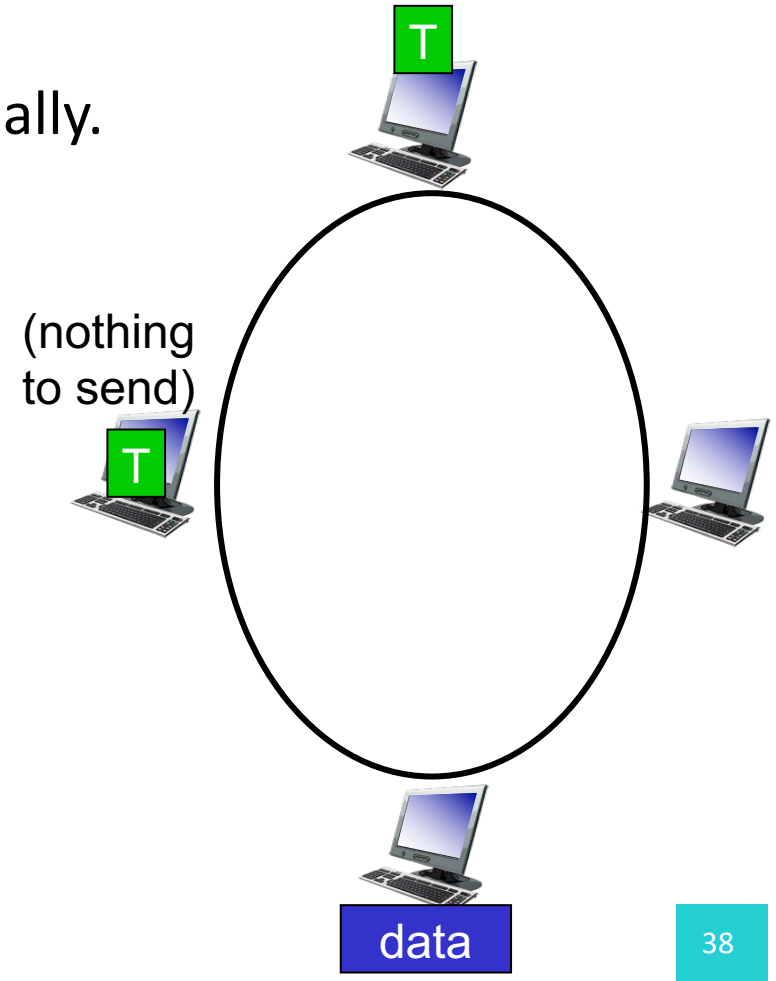
- Coordinator node **invites** sender nodes to transmit in turn
- Concerns
  - Polling overhead
  - Latency
  - Single point of failure (Coordinator)



Taking-Turns MAC Protocols

Token passing

- Control **token** passed from one node to next sequentially.
- Token message
- Concerns
  - Token overhead
  - Latency
  - Single point of failure (token)



Link Layer: LAN

MAC Addresses & ARP

- 32-bit IP address
  - **Network-layer** address for interface
  - Used for layer 3 (network layer) forwarding

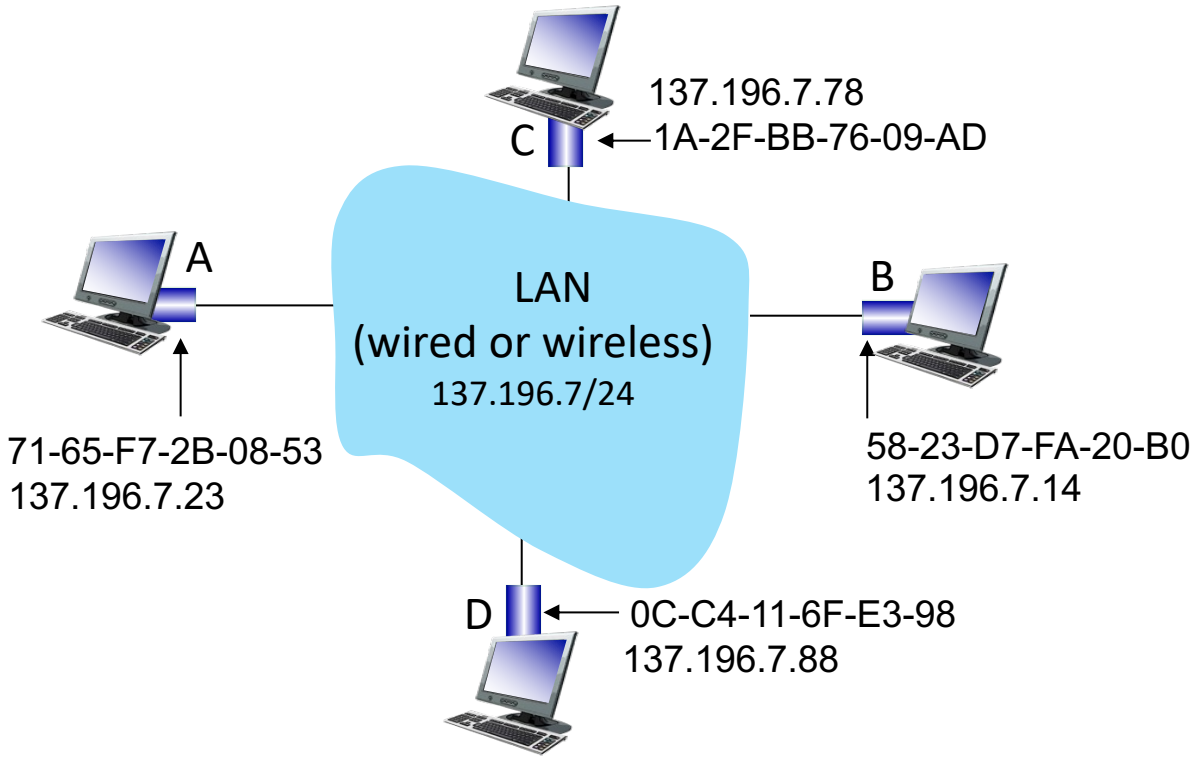
- MAC (or LAN or physical or Ethernet) address
  - Function: **Used locally to get frame from one interface to another physically-connected interface (same network, in IP-addressing sense)**
  - 48-bit MAC address (for most LANs) burned in NIC ROM, also sometimes software settable
  - Example: 1A-2F-BB-76-09-AD

LAN Address

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

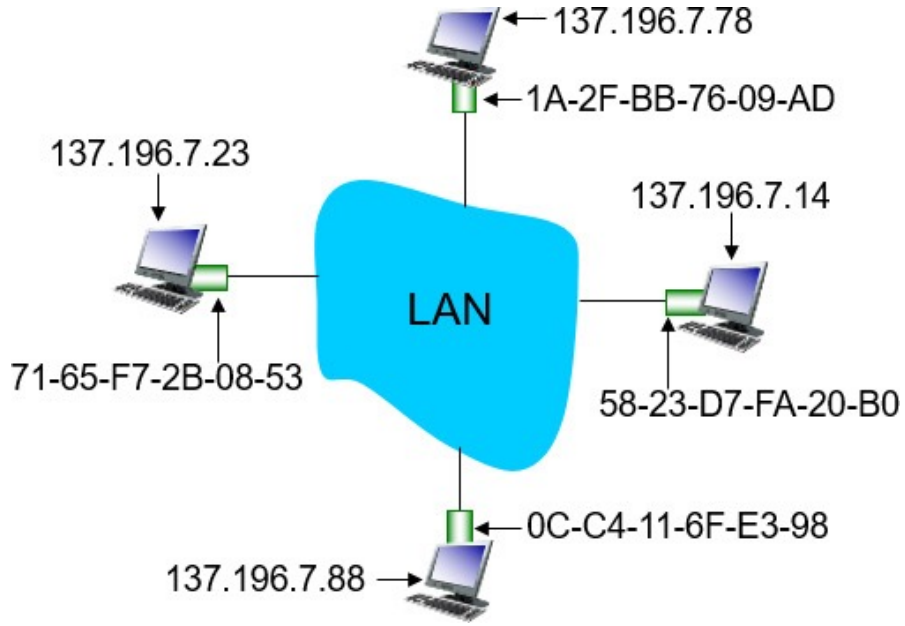
MAC Addresses & ARP

- Each adapter on LAN has unique **LAN** address



ARP: Address Resolution Protocol

Question: How to determine interface MAC address knowing its IP address?



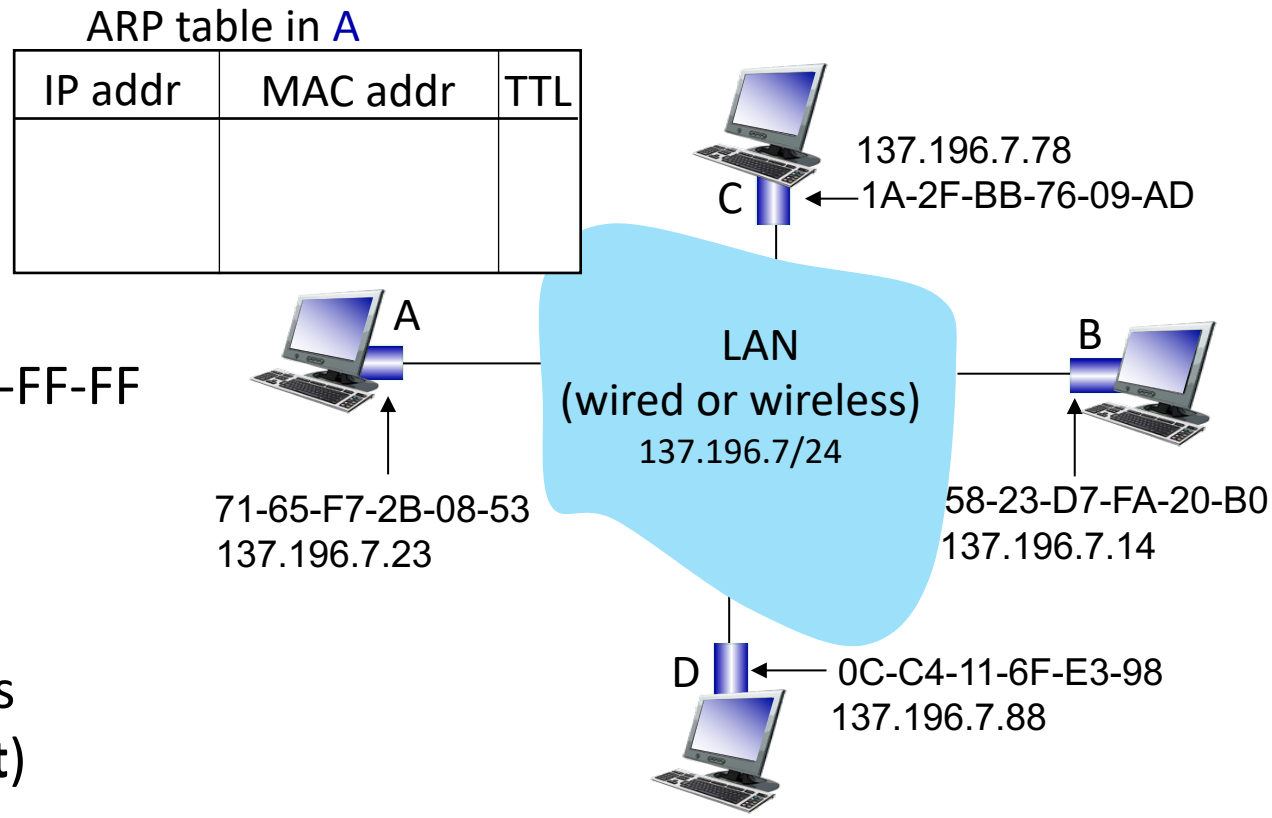
ARP table: Each IP node (host, router) on LAN has table

- IP/MAC address mappings for some LAN nodes
  < IP address; MAC address; TTL>

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

ARP Protocol: 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 IP address for B
  - Destination MAC address = FF-FF-FF-FF-FF-FF
  - All nodes on LAN receive ARP query
- B receives ARP packet
  - B replies to A with its (B's) MAC address
  - Frame sent to A's MAC address (**unicast**)



ARP Protocol: Same LAN

- A caches (saves) IP-to-MAC address pair in its ARP table

ARP table in A

IP addr	MAC addr	TTL
137.196.7.14	58-23-D7-FA-20-B0	500

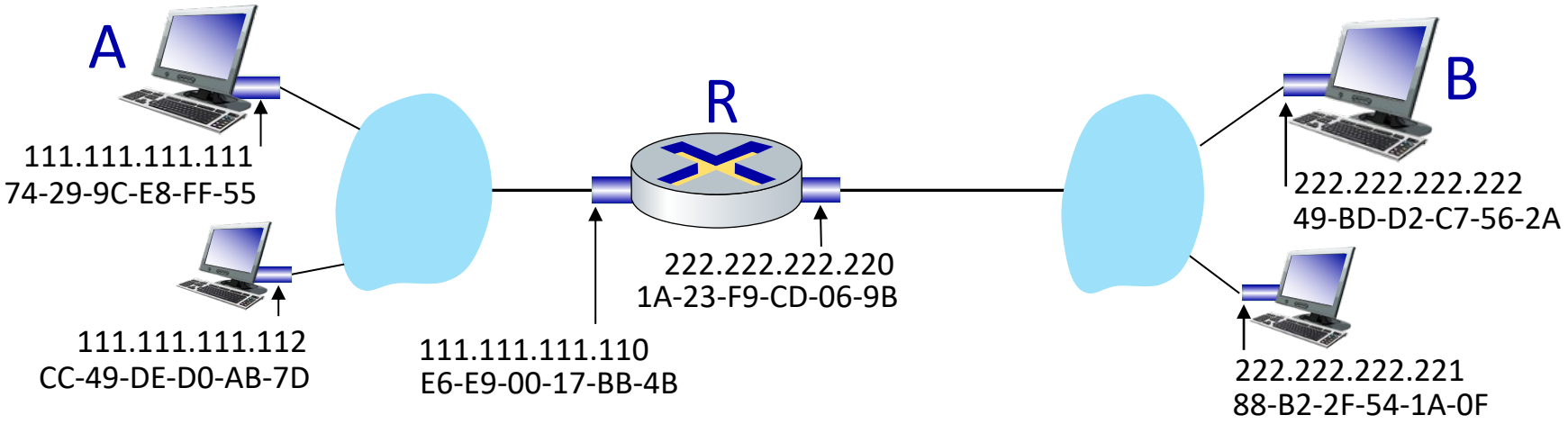
  - Until information becomes old (times out)
  - Soft state: Information that times out (goes away) unless refreshed

- ARP is **plug-and-play**:
  - Nodes create their ARP tables **without intervention from net administrator**

Addressing: Routing to Another LAN

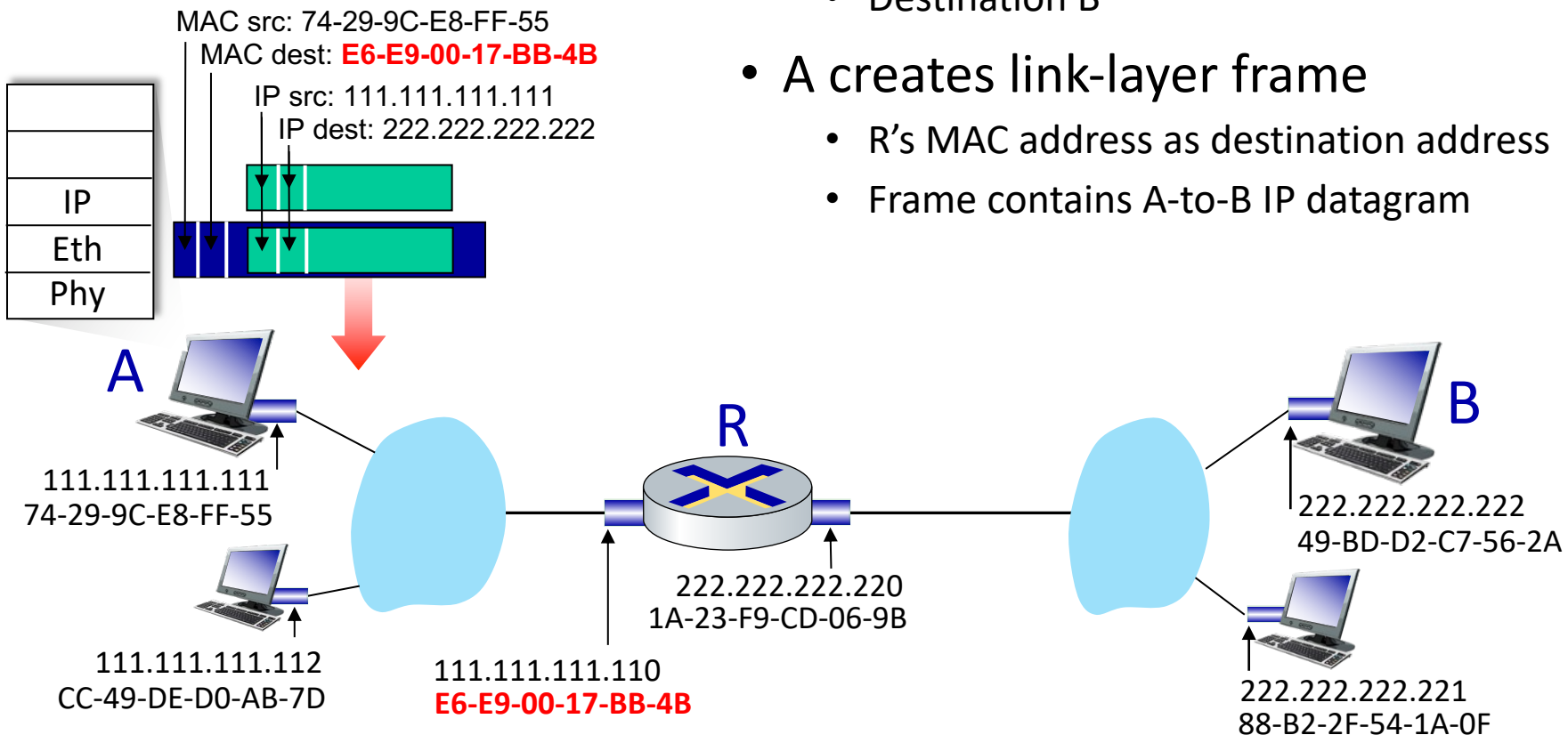
Walkthrough**: Send datagram from A to B via R**

- 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?)



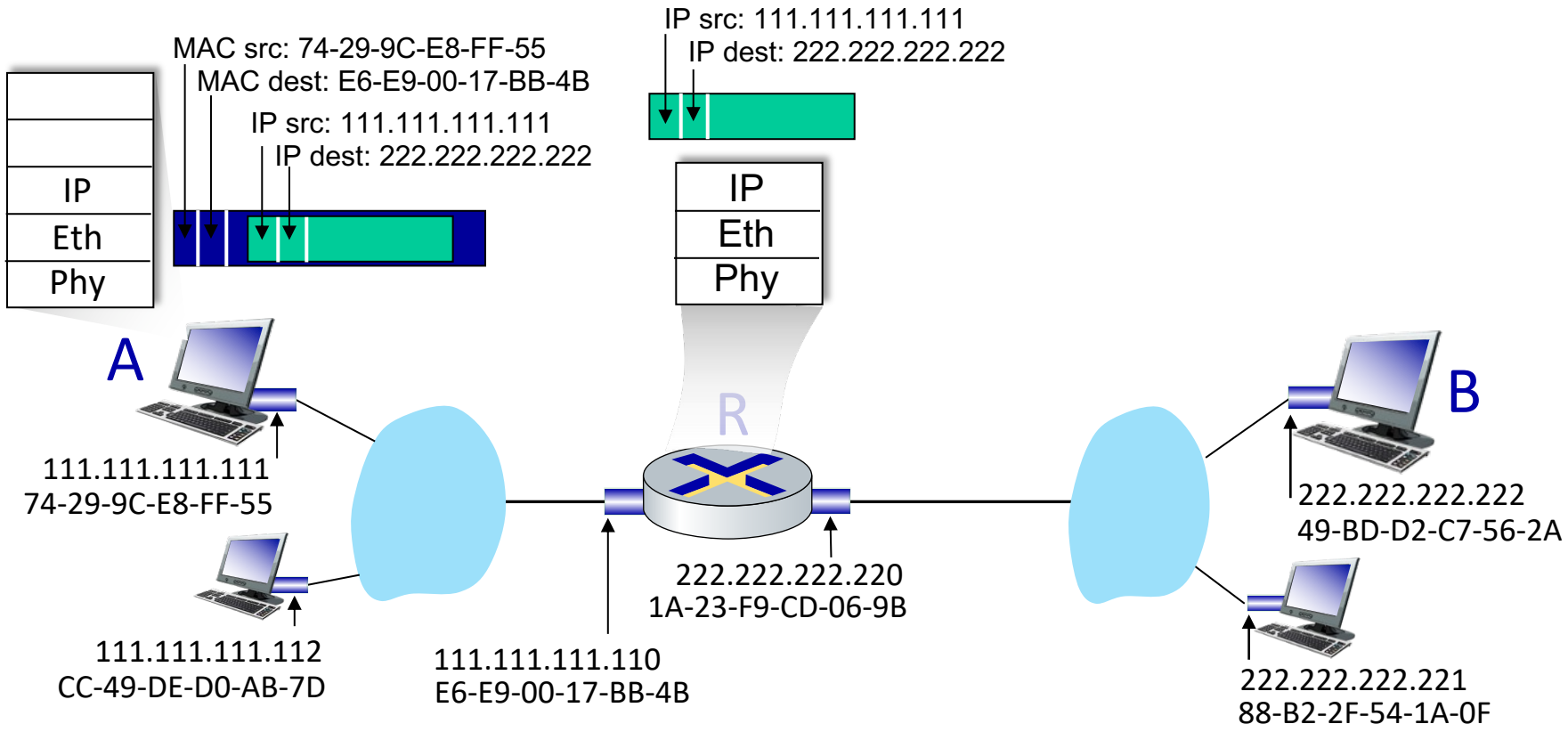
Addressing: Routing to Another LAN

- A creates IP datagram
  - IP source A
  - Destination B
- A creates link-layer frame
  - R's MAC address as destination address
  - Frame contains A-to-B IP datagram



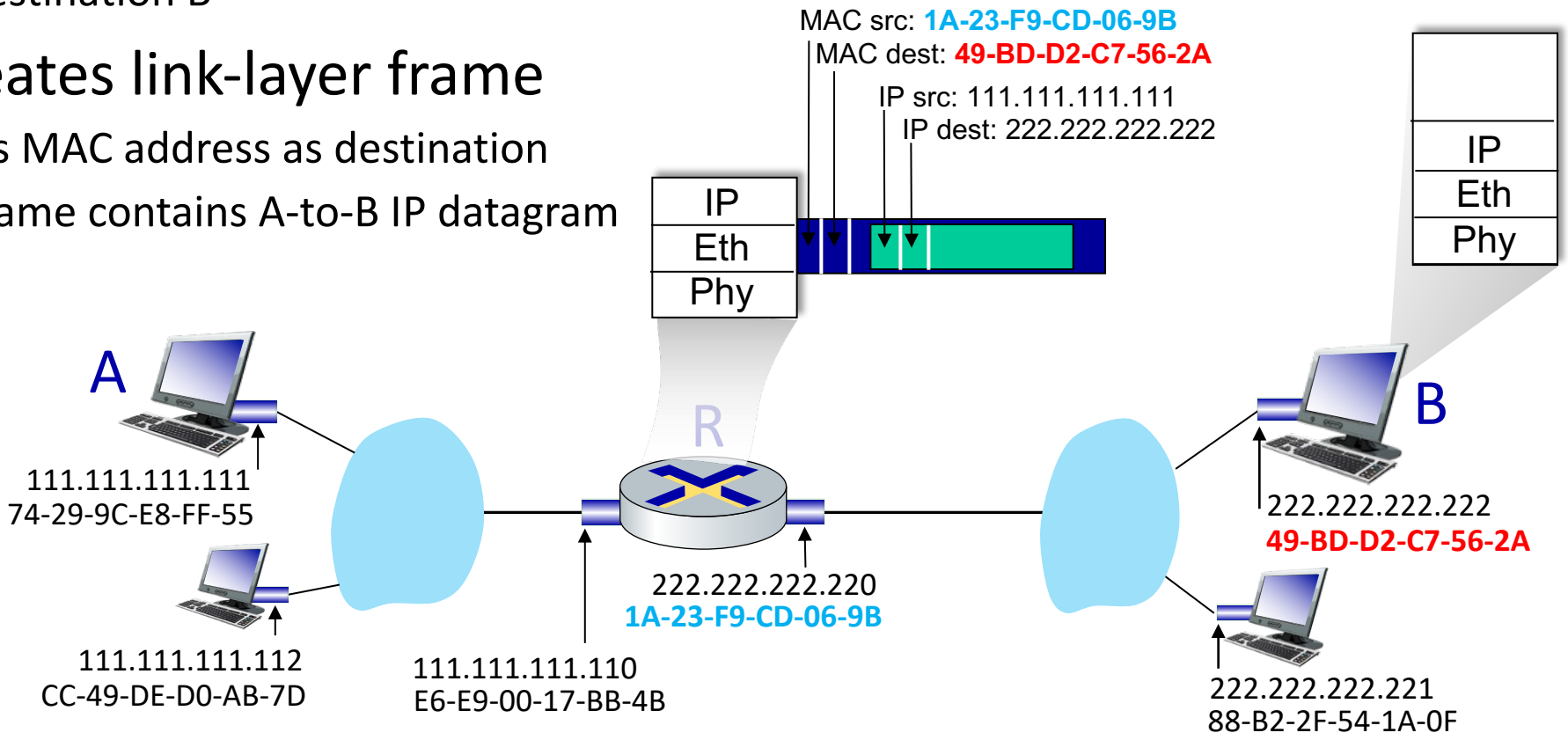
Addressing: Routing to Another LAN

- Frame sent from A to R
- Frame received at R → Datagram removed → Passed up to IP



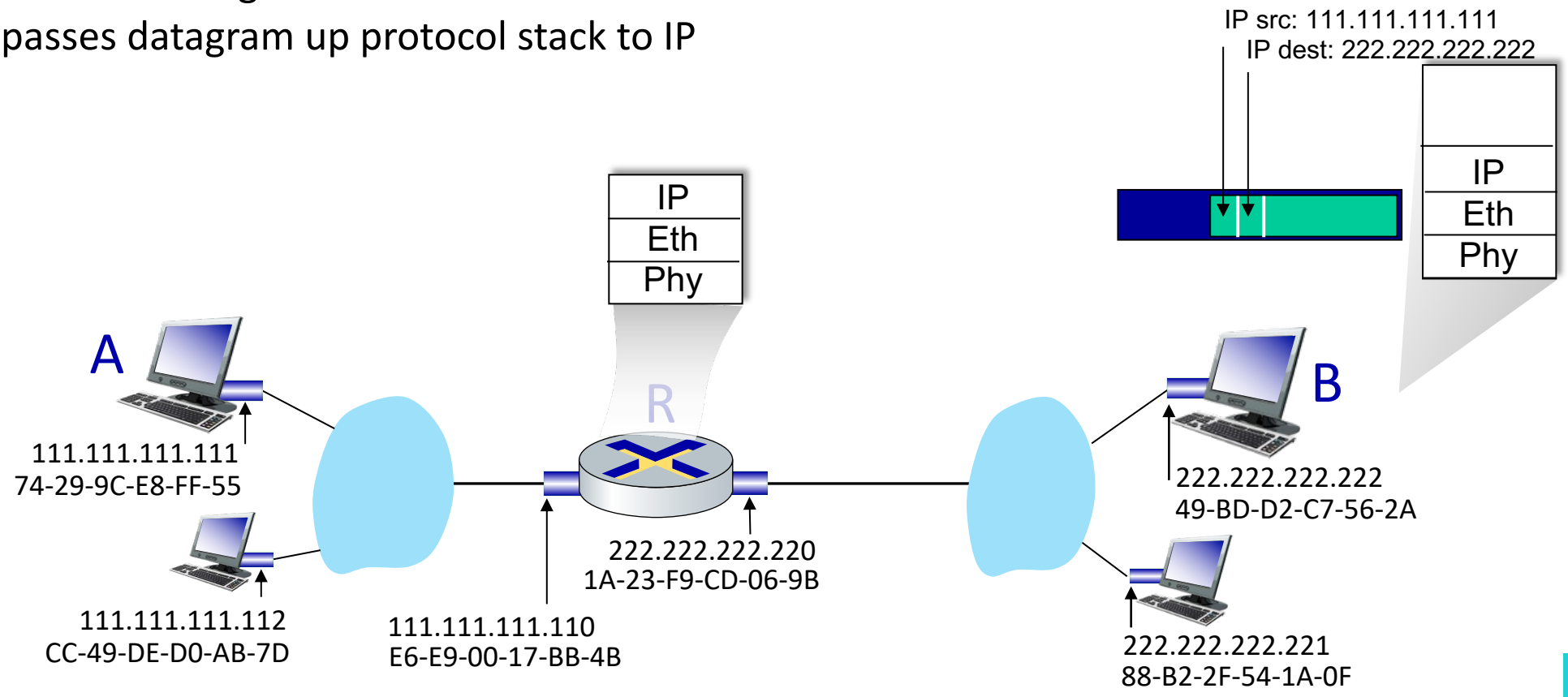
Addressing: Routing to Another LAN

- R forwards datagram
  - IP source A
  - Destination B
- R creates link-layer frame
  - B's MAC address as destination
  - Frame contains A-to-B IP datagram



Addressing: Routing to Another LAN

- B receives frame
  - Extracts IP datagram destination B
  - B passes datagram up protocol stack to IP



Link Layer

- Error detection, correction
- Multiple access protocols

✓**LANs**

- Addressing, ARP

✓**Ethernet**

- Switches
- VLANS

Ethernet

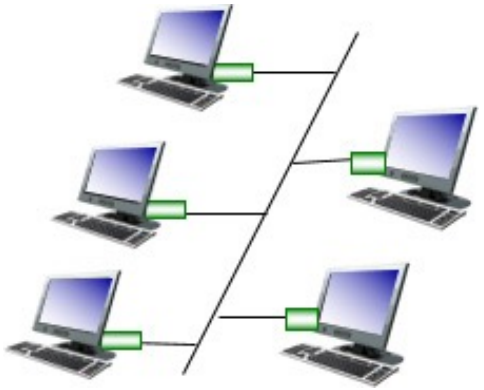
- Dominant wired LAN technology
  - Single chip, multiple speeds (e.g., Broadcom BCM5761)
  - First widely used LAN technology
  - Kept up with speed race: 10 Mbps – 10 Gbps

- **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)

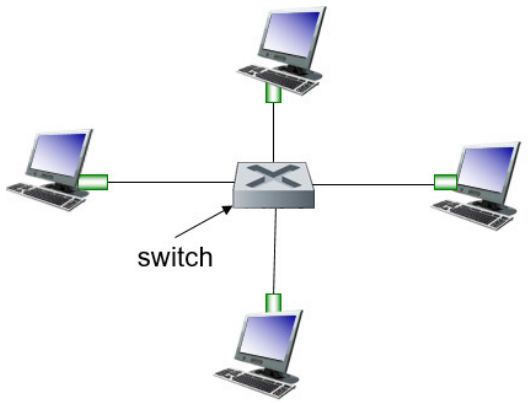
Ethernet: Physical Topology

- **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



Star



Ethernet Frame Structure

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

Preamble:

- 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

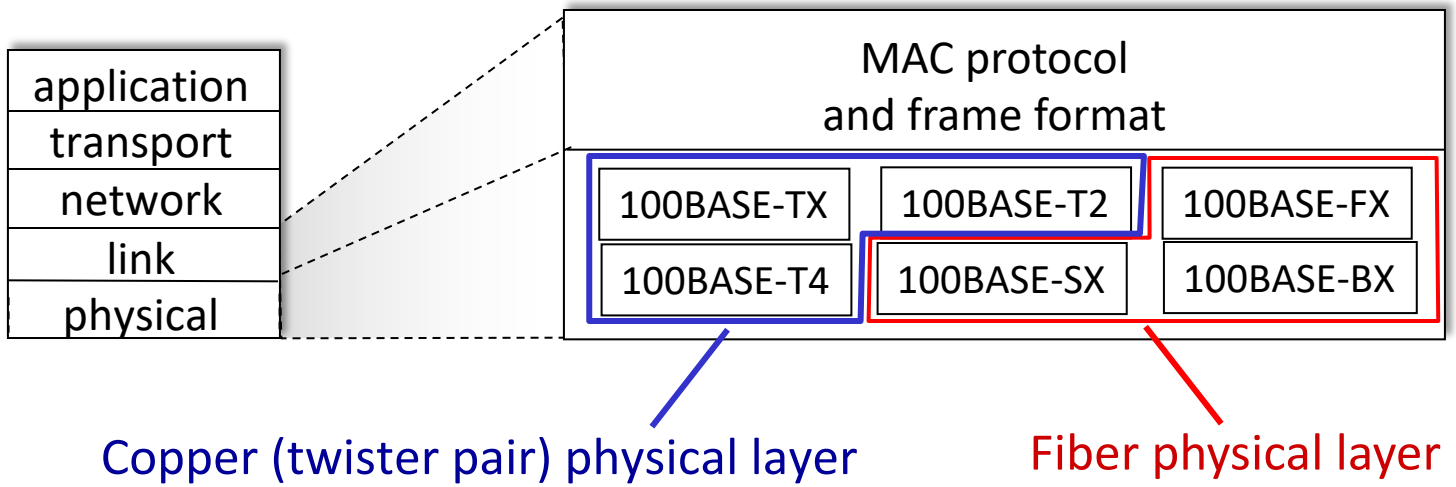
- **Type:** Indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)
- **CRC:** Cyclic redundancy check at receiver
  - Error detected: Frame is dropped

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**

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



Link Layer

- Error detection, correction
- Multiple access protocols

✓**LANs**

  - Addressing, ARP
  - Ethernet

  ✓**Switches**

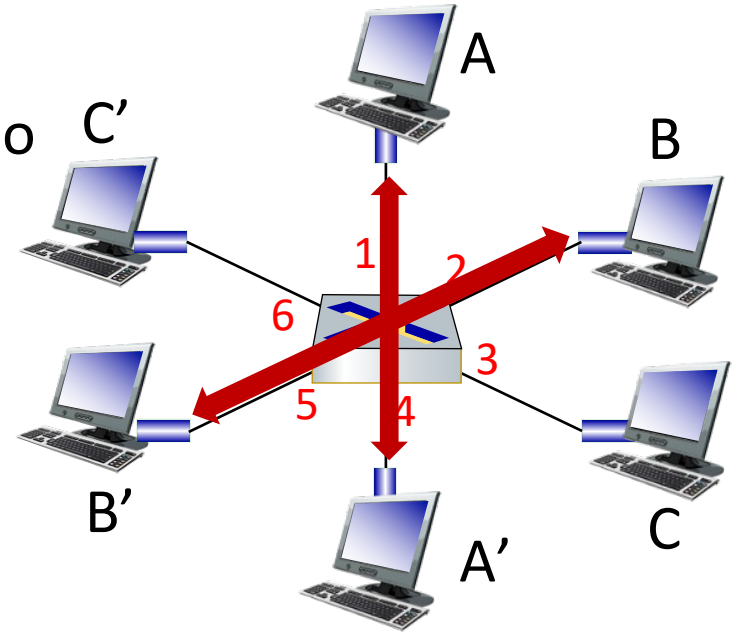
  - VLANS

Link Layer Switch

- **Link-layer device: Takes an active role**
  - Store and forward Ethernet frames
  - Examine incoming frame's MAC address
  - **Selectively** forward frame to one-or-more outgoing links when frame is to be forwarded on segment, uses CSMA/CD to access segment
- **Transparent**
  - Hosts are unaware of presence of switches
- **Plug-and-play & self-learning**
  - Switches do not need to be configured

Switch: Multiple Simultaneous Transmissions

- Hosts have dedicated, direct connection to switch
- Ethernet protocol used on **each** incoming link, but no collisions: Full duplex
  - Each link is its own collision domain
- **Switching:** A-to-A' and B-to-B' can transmit simultaneously, without collisions



switch with six interfaces (1,2,3,4,5,6)

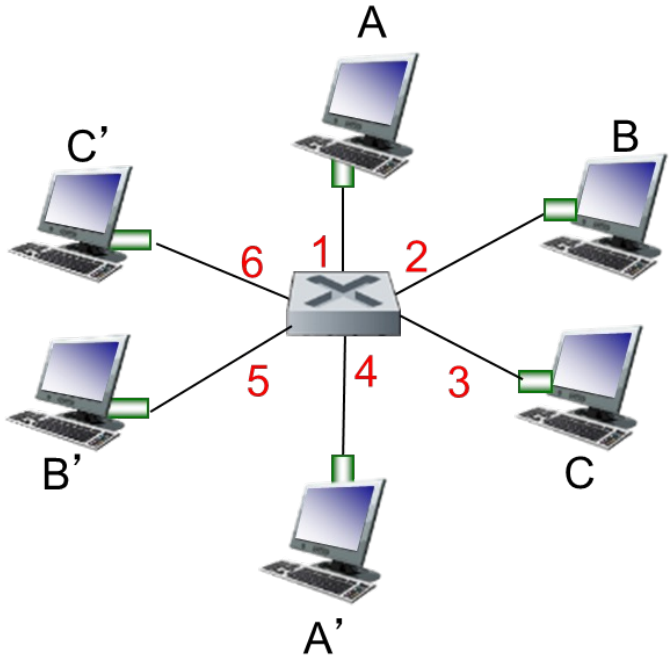
Switch Forwarding Table

Q: How does switch know A' reachable via interface 4, B' reachable via interface 5?

A: Each switch has a **switch table,** each entry:

- (MAC address of host, interface to reach host, time stamp)
- Looks like a routing table!

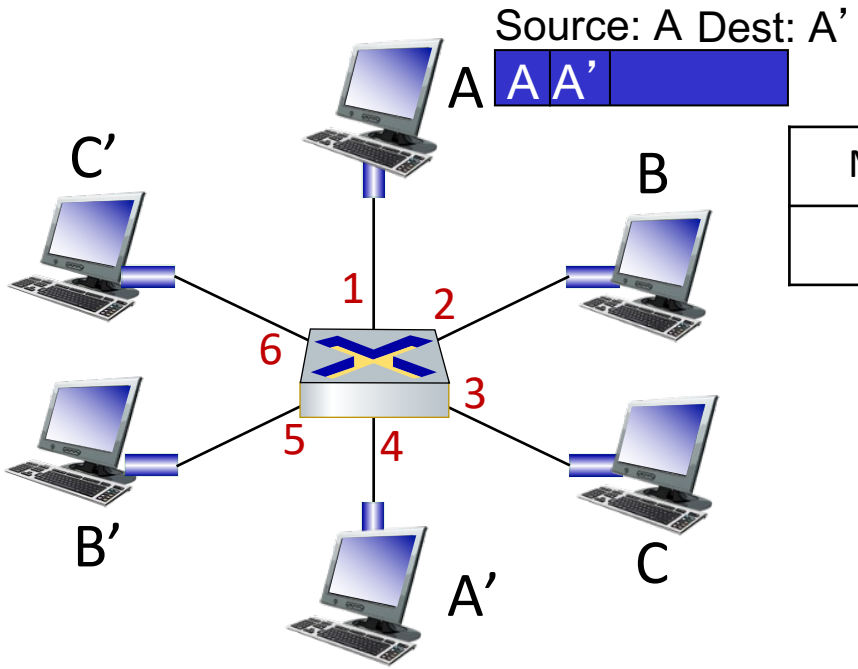
Q: How are entries created, maintained in switch table?
Something like a routing protocol?



switch with six interfaces (1,2,3,4,5,6)

Switch: Self-Learning

- Switch **learns** which hosts can be reached through which interfaces
  - When frame received, switch "learns" location of sender: Incoming LAN segment
  - Records sender/location pair in switch table



MAC addr	interface	TTL

Switch table (initially empty)

Switch: Frame Filtering/Forwarding

When frame received at switch

- Record incoming link, MAC address of sending host
- Index switch table using MAC destination address

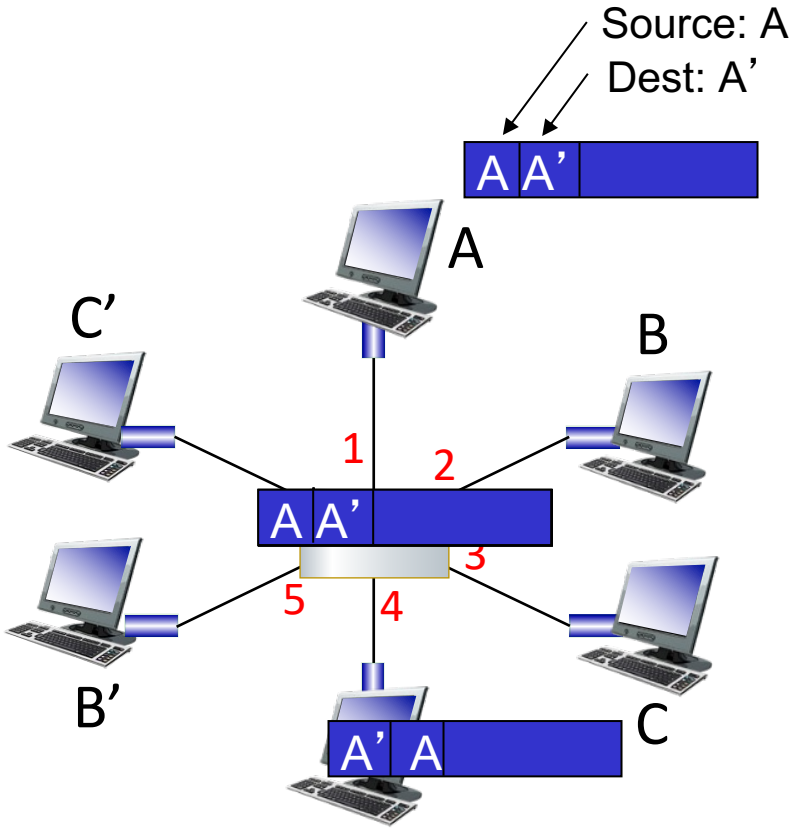
```
If entry found for destination
then {
if destination on segment from which frame arrived
   then drop frame
   else forward frame on interface indicated by entry
 }
 else flood /* forward on all interfaces except arriving interface */
```

Example: Self-Learning & Forwarding

- Frame destination, A’, location unknown: **Flood**
- Destination A location known: **Selectively send on just one link**

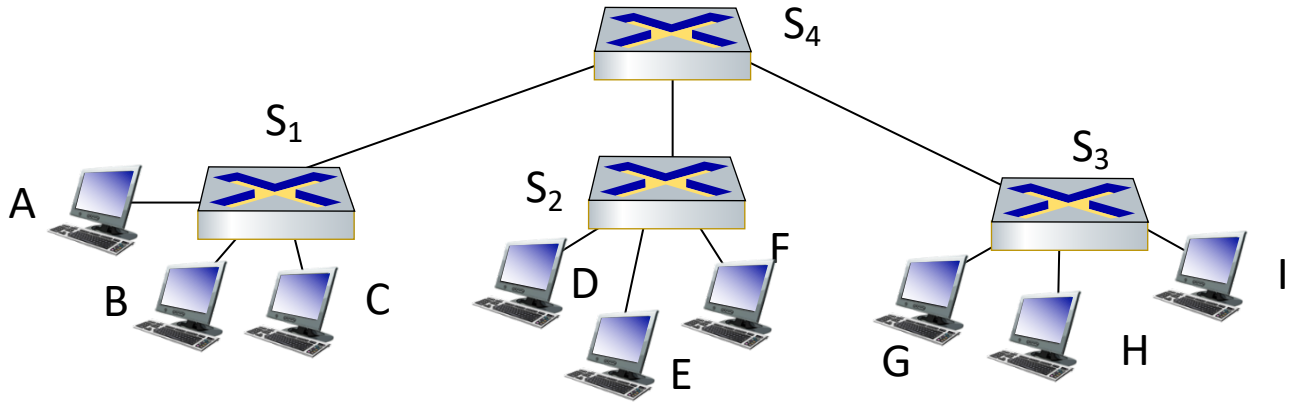
MAC addr	interface	TTL

Switch table (initially empty)



Interconnecting Switches

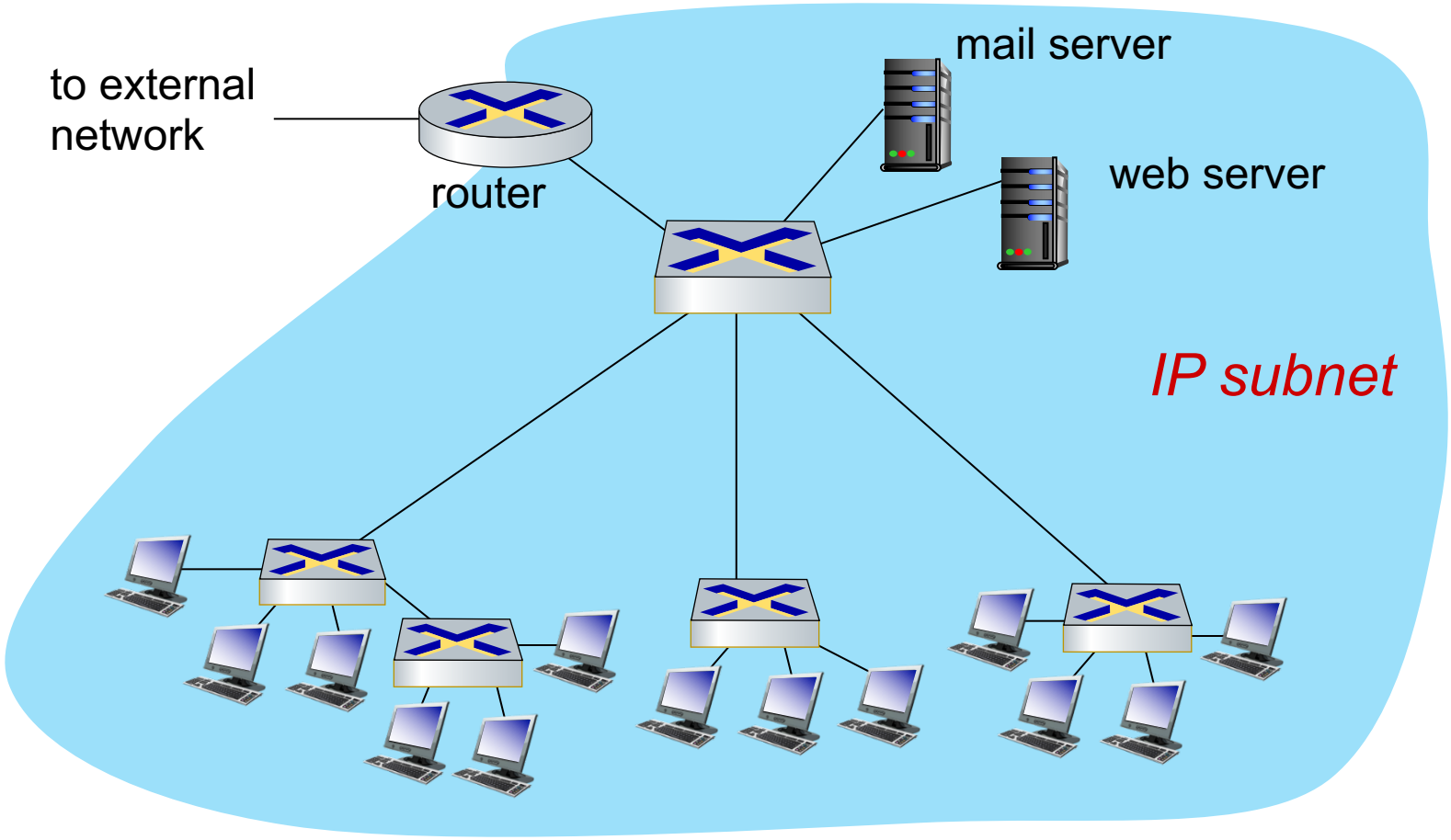
- Self-learning switches can be connected together:



Q: Sending from A to G: How does S_1 know to forward frame destined to G via S_4 and S_3?

A: Self learning! Works exactly the same as in single-switch case!

Institutional Network



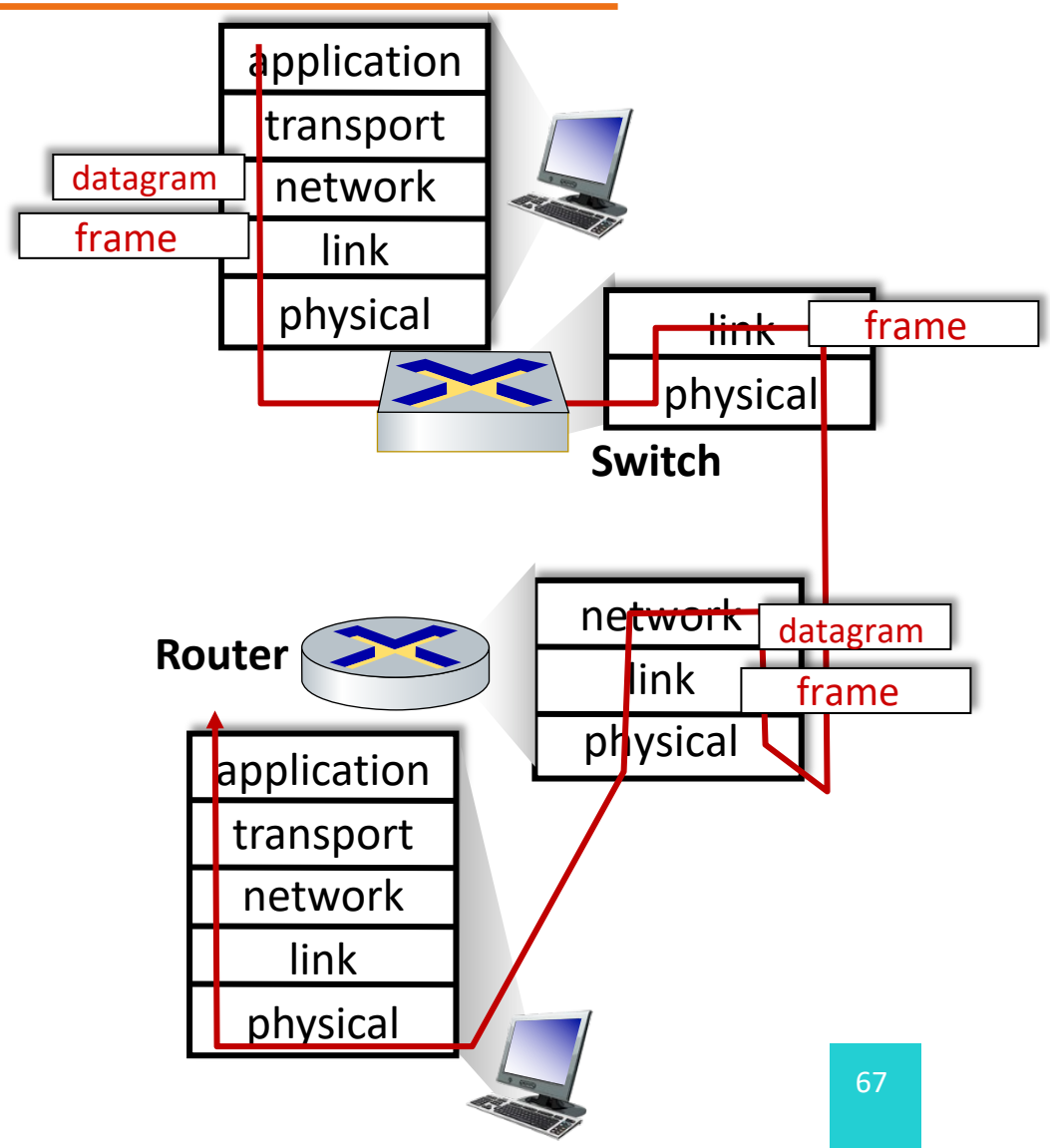
Switches vs. Routers

Both are store-and-forward:

- **Routers:** Network-layer devices (examine network-layer headers)
- **Switches:** Link-layer devices (examine link-layer headers)

Both have forwarding tables:

- **Routers:** Compute tables using routing algorithms, IP addresses
- **Switches:** Learn forwarding table using flooding, learning, MAC addresses



Link Layer

- Error detection, correction
- Multiple access protocols

✓**LANs**

  - Addressing, ARP
  - Ethernet
  - Switches

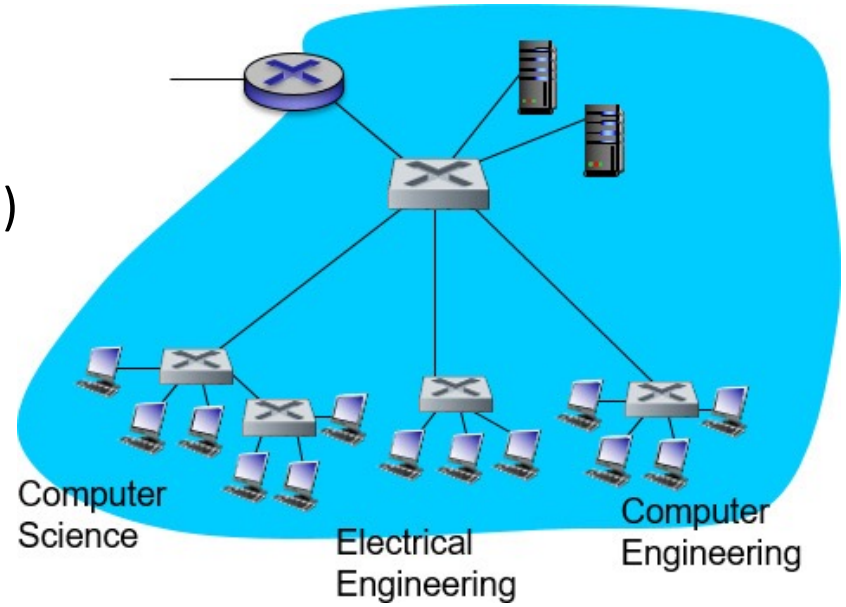
  ✓**VLANS**

VLANs: Motivation

Consider

- CS user moves office to EE, but wants connect to CS switch?

- Single broadcast domain:
  - All layer-2 broadcast traffic must cross entire LAN
    (ARP, DHCP, unknown location of destination MAC address)
  - Security/privacy, efficiency issues



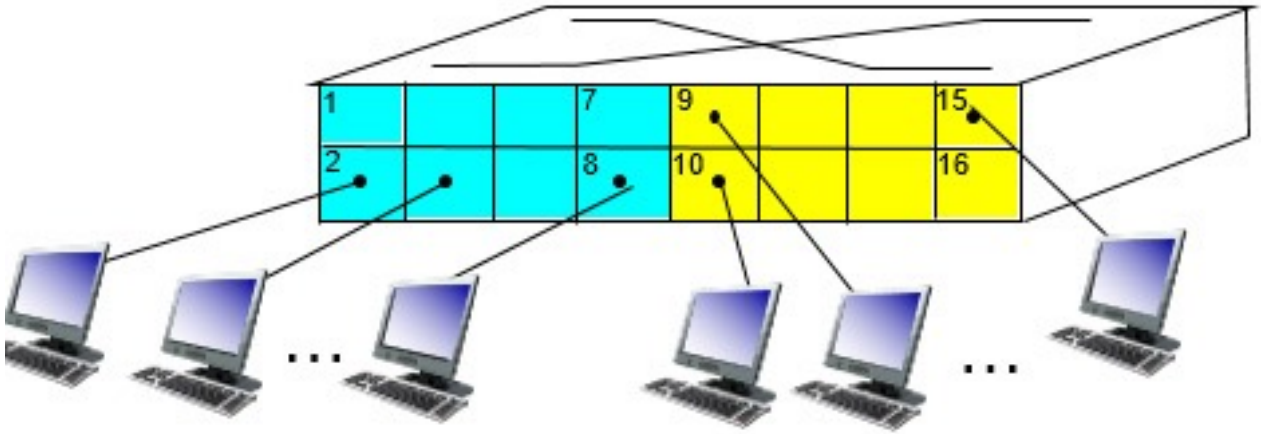
VLANs

Virtual Local Area Network

- Switch(es) supporting VLAN capabilities can be configured to define multiple **virtual** LANS over single physical LAN infrastructure.

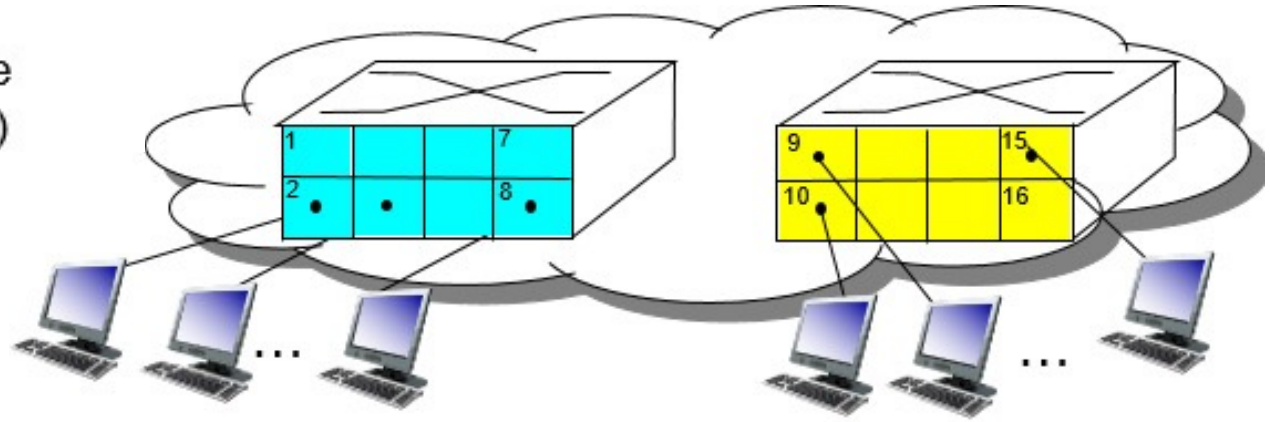
- **Port-based VLAN:** Switch ports grouped (by switch management software) → **single** physical switch operates as **multiple** virtual switches

VLANs



Electrical Engineering
(VLAN ports 1-8)

Computer Science
(VLAN ports 9-15)



Electrical Engineering
(VLAN ports 1-8)

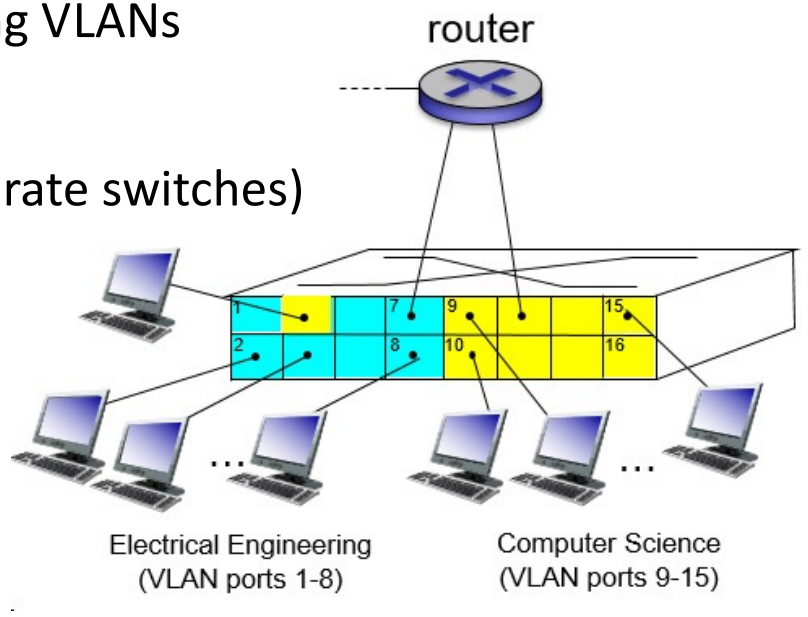
Computer Science
(VLAN ports 9-16)

Port-Based VLAN

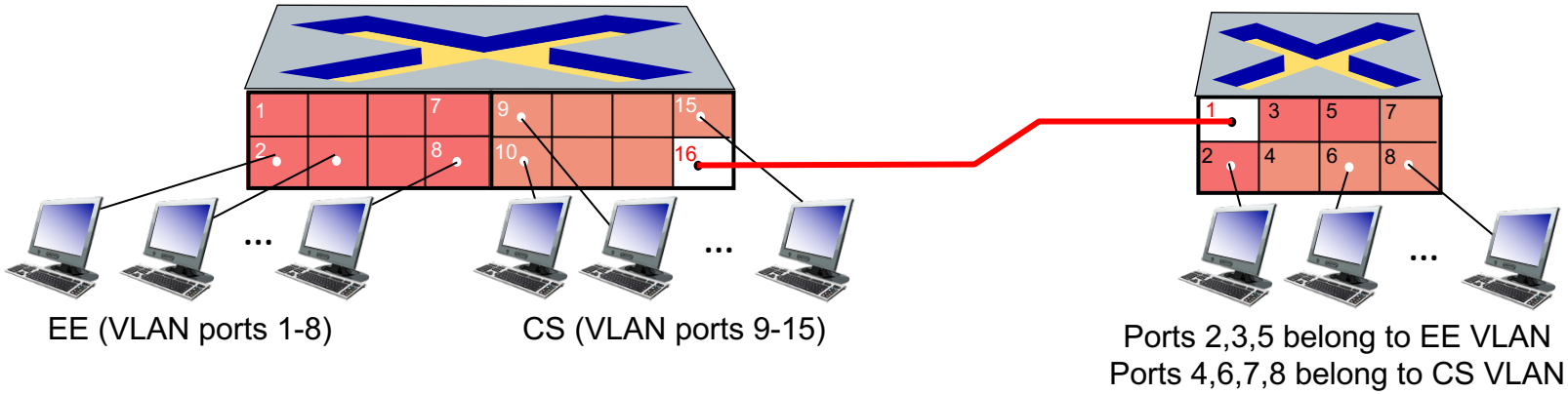
- **Traffic isolation:** Frames to/from ports 1-8 can **only** reach ports 1-8
  - Can also define VLAN based on MAC addresses of endpoints, rather than switch port

- **Dynamic membership:** Ports can be dynamically assigned among VLANs

- **Forwarding between VLANS:** Done via routing (just as with separate switches)
  - In practice vendors sell combined switches plus routers

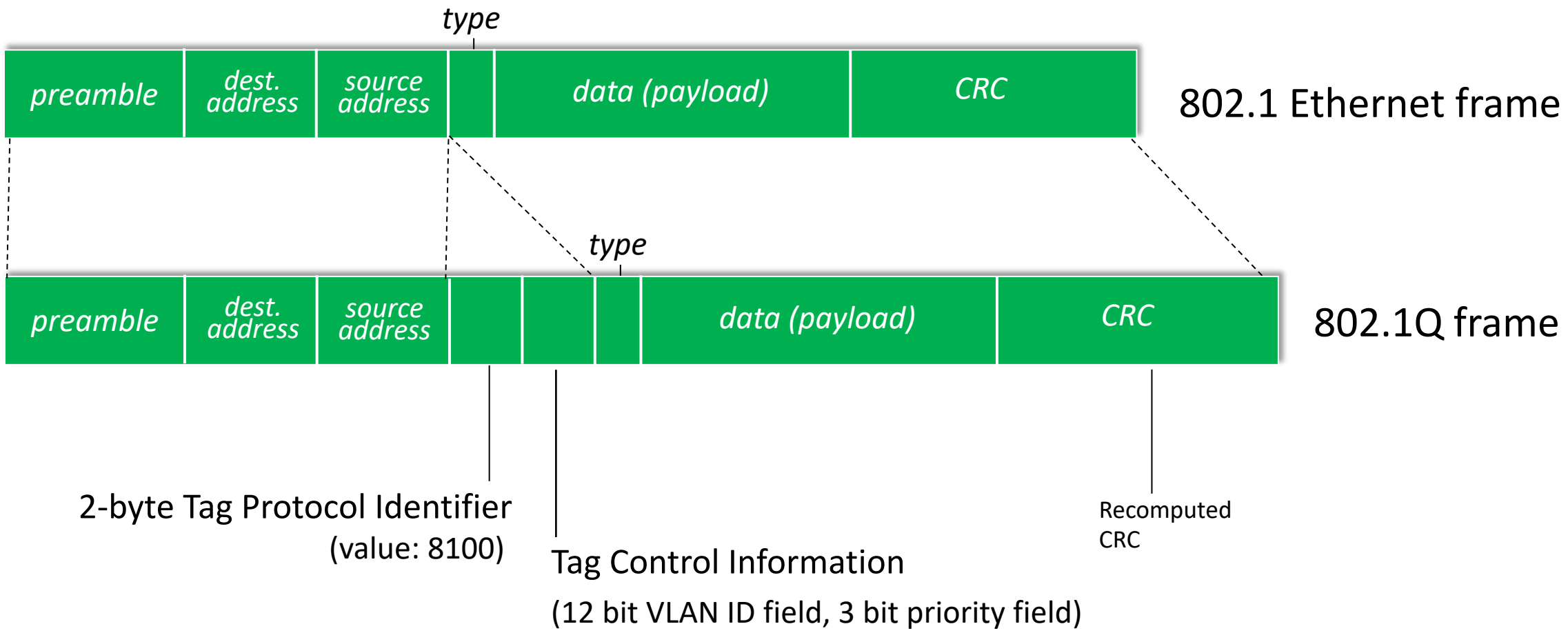


VLANS Spanning Multiple Switches



- **Trunk port:** Carries frames between VLANs defined over multiple physical switches
  - Frames forwarded within VLAN between switches can't be vanilla 802.1 frames (must carry VLAN ID info)
  - 802.1q protocol adds/removed additional header fields for frames forwarded between trunk ports

802.1Q VLAN Frame Format



Summary

- Principles behind data link layer services
  - Error Detection and Correction
  - Sharing a broadcast channel: Multiple Access
  - Link layer addressing

- Instantiation and implementation of various link layer technologies
  - Ethernet
  - Switched LANS, VLANs
  - Virtualized networks as a link layer: MPLS

Acknowledgements

- The following materials have been used in preparation of this presentation:

  [1] Textbook and (edited) Slides: Computer Networking: A Top-Down Approach
  James Kurose, Keith Ross
  7th and 8th Edition, Pearson
  http://gaia.cs.umass.edu/kurose_ross/

  [2] Reference: Computer Networks: A Systems Approach
  https://www.systemsapproach.org/book.html

- Recommended Additional Resources:

  [1] Interactive Exercises (Chapter six)
  http://gaia.cs.umass.edu/kurose_ross/interactive/