

# CS 352

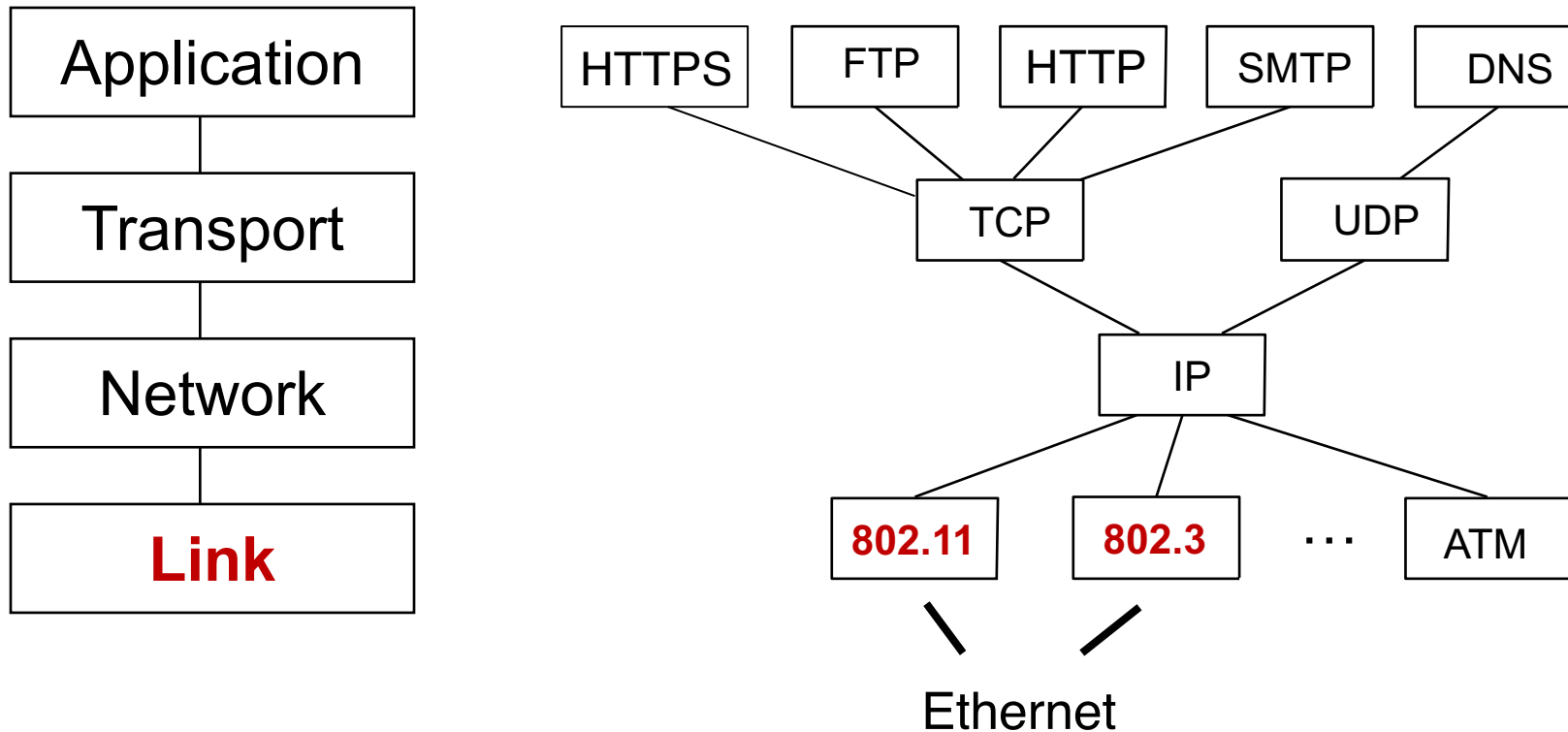
## Link Layer: Introduction

CS 352, Lecture 22.1

<http://www.cs.rutgers.edu/~sn624/352>

Srinivas Narayana

# Link layer



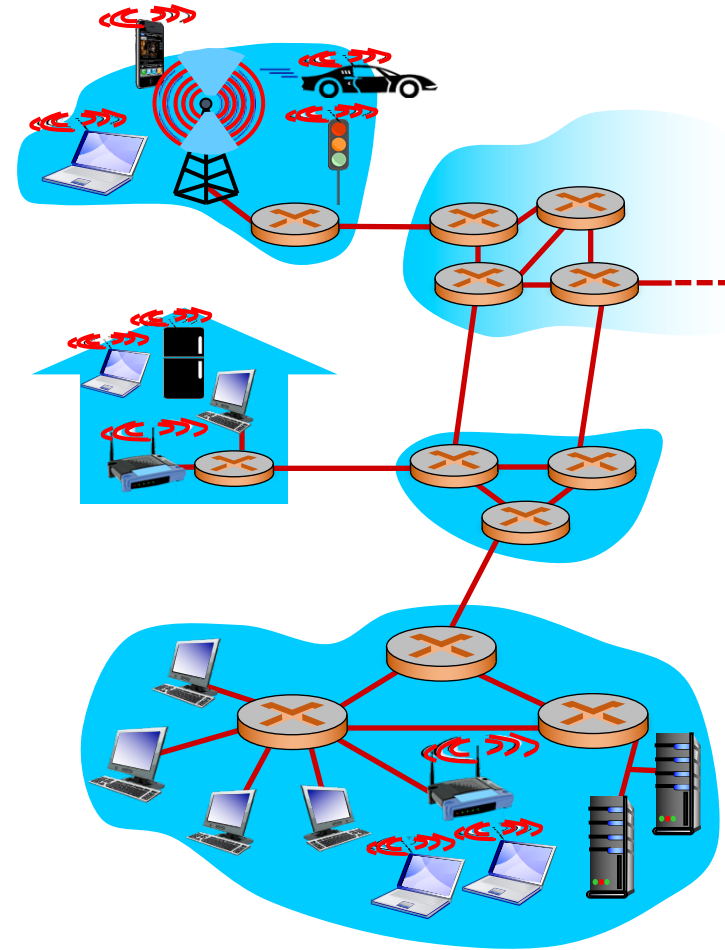
The main function of the link layer is **link-local delivery**: getting packets from one side of the link to the other.

# Link layer: introduction

## Terminology:

- Endpoints and routers: **nodes**
- communication channels that connect adjacent nodes along communication path: **links**
  - wired links, wireless links, local area networks (LANs)
- layer-2 packet: **frame**, encapsulates (IP) datagram

**Link layer** has the responsibility of transferring datagram from one node to **physically adjacent** node over a link.



Slides in this presentation were heavily adapted from those of Kurose and Ross, who own the copyright on the material.

# Link layer: context

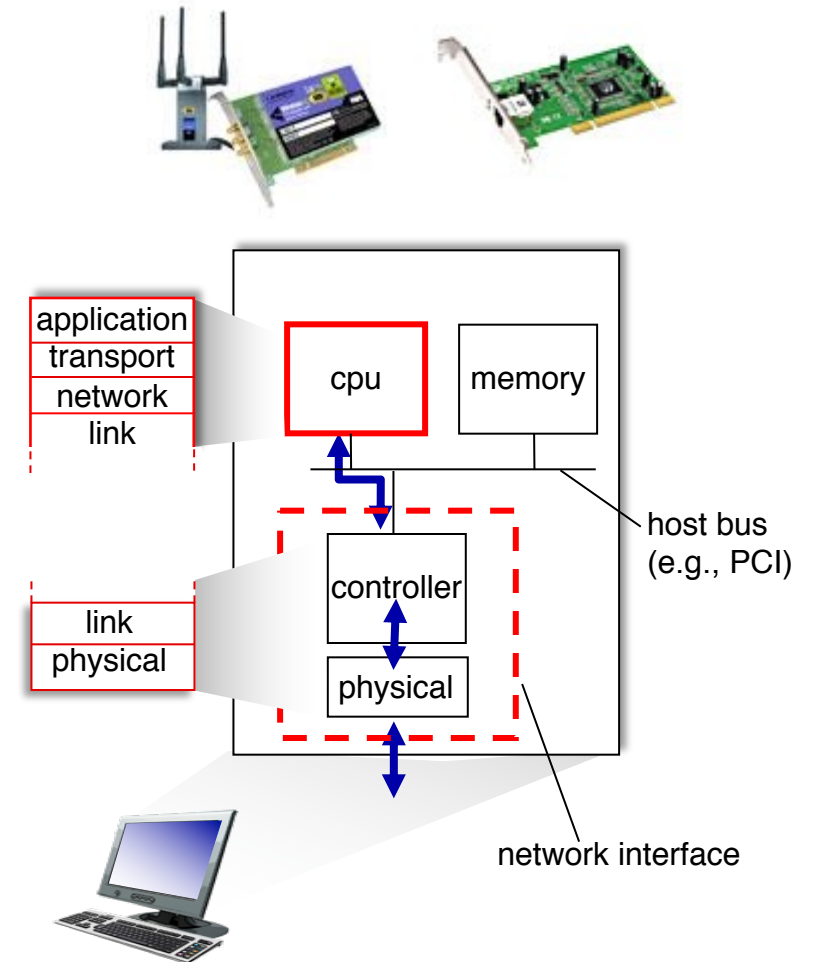
- Datagram transferred by different link layer protocols over different links
  - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- Each link layer protocol may provide different services
  - e.g., some links may provide additional reliability mechanisms

## Analogy:

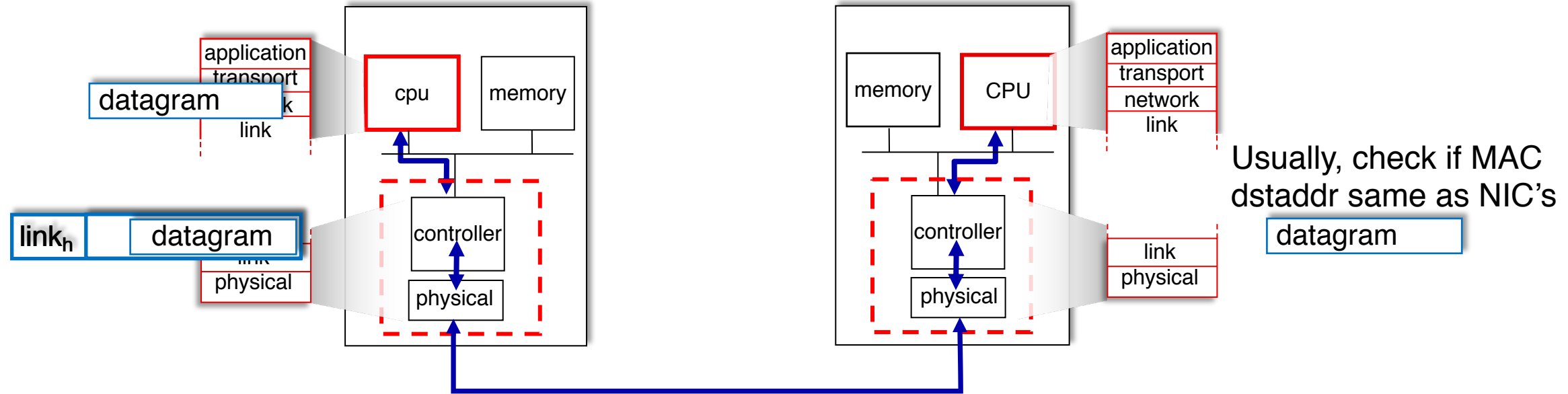
- trip from Piscataway, NJ to Palo Alto, CA
  - limo: Piscataway to JFK
  - plane: JFK to San Francisco
  - train: San Francisco to Palo Alto
- tourist = **datagram**
- transport segment (road/flight/rail) = **communication link**
- transportation mode (car/plane/train) = **link layer protocol**

# Where is the link layer implemented?

- in every endpoint & router
- link layer implemented in **network interface card (NIC)**
  - Ethernet, WiFi card or chip
  - Router input and output ports
- At endpoint, attaches into its system buses (e.g., PCIe)
- combination of hardware, software, firmware



# Interfaces 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 (MAC) addresses**

# Link layer addresses

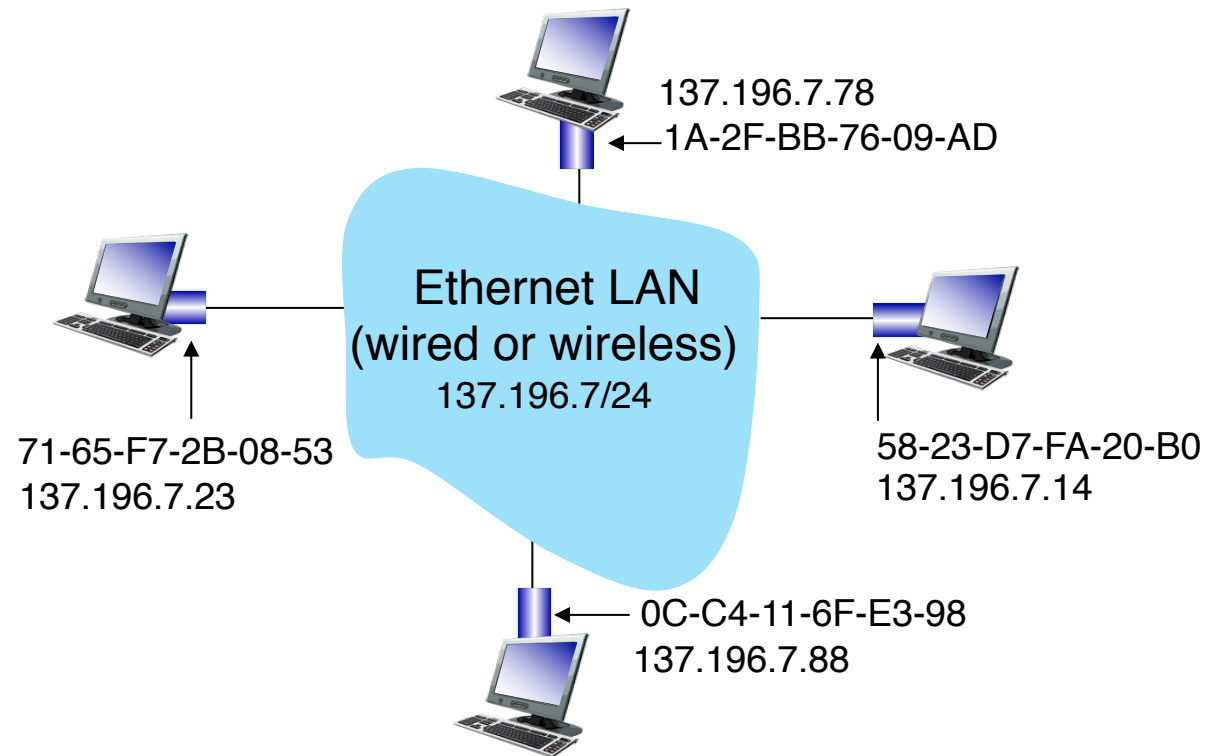
- Review: we've looked at 32-bit IPv4 addresses
  - network-layer address for interface
  - used for layer 3 (network layer) forwarding
  - e.g.: 128.119.40.136
- **MAC or physical or link-layer address**
  - Used “locally” to get frame from one interface to another physically-connected interface (same subnet, in IP-addressing sense)
  - This course: **Ethernet family** of link layer protocols
  - **48-bit** Ethernet MAC address burned in NIC ROM
    - Sometimes, the address can be set in software
  - e.g.: 1A-2F-BB-76-09-AD \_\_\_\_\_ hexadecimal (base 16) notation  
(each “numeral” represents 4 bits)



# MAC addresses

each interface on LAN

- has unique 48-bit MAC address
- has a locally unique 32-bit IP address (as we've seen)



# MAC addresses

- 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 is a **flat address**: portability
  - e.g., can move interface from one Ethernet LAN to another
  - Recall: IP address is not portable: depends on IP network to which node is attached

**Link layer services**

# Link layer: services

- **Line termination (serialize/deserialize)**
  - Physical signaling (into the wire) & extracting digital signal (out of the wire)
  - Encapsulate datagram into frame (**framing**). Add header, trailer
- **Error detection:**
  - errors caused by signal attenuation, noise.
  - receiver detects errors, signals retransmission, or drops frame
- **Error correction:**
  - receiver identifies **and corrects** bit error(s) without retransmission
- **Medium access control**
  - channel **medium access** in a shared medium:
  - “Who should talk?”

# Link layer: services

- **Line termination (serialize/deserialize)**
  - Physical signaling (into the wire) & extracting digital signal (out of the wire)
  - Encapsulate datagram into frame (**framing**). Add header, trailer
- **Error detection:** **This lecture**
  - errors caused by signal attenuation, noise
  - receiver detects errors, signals retransmission, or drops frame
- **Error correction:**
  - receiver identifies **and corrects** bit error(s) without retransmission

- **Medium access control** **Next 2 lectures**
  - channel **medium access** in a shared medium
  - “Who should talk?”

# Link layer: services: there's more!

- **Reliable delivery between adjacent nodes**
  - Seldom used on **low bit-error** links
  - Not strictly needed for functionality
  - **An optimization that significantly improves performance** over pure end-to-end reliable delivery over high-error-rate links (e.g., wireless)
- **Flow control:**
  - pacing between adjacent sending and receiving nodes
  - Used in **lossless** link layer technologies (e.g., Infiniband, lossless Ethernet)

Not covered in this course

# Link layer: services

- Line termination (serialize/deserialize)
- Physical signaling (into the wire) & extracting digital signal (out of the wire)
- Encapsulate datagram into frame (framing), Add header, trailer
- **Encoding, error detection, and error correction**
  - Error detection:
    - errors caused by signal attenuation, noise
    - receiver detects errors, signals retransmission, or drops frame
  - Error correction
    - receiver identifies and corrects bit error(s) without retransmission
- Medium access control
  - channel medium access in a shared medium:
  - “Who should talk?”





# CS 352

## Encoding, Error Detection, and Correction

CS 352, Lecture 22.2

<http://www.cs.rutgers.edu/~sn624/352>

Srinivas Narayana

# Recall: Link layer services

- **Line termination (serialize/deserialize)**
  - Physical signaling (into the wire) & extracting digital signal (out of the wire)
  - Encapsulate datagram into frame (**framing**). Add header, trailer
- **Error detection:**
  - errors caused by signal attenuation, noise.
  - receiver detects errors, signals retransmission, or drops frame
- **Error correction:**
  - receiver identifies **and corrects** bit error(s) without retransmission
- **Medium access control**
  - channel **medium access** in a shared medium:
  - “Who should talk?”

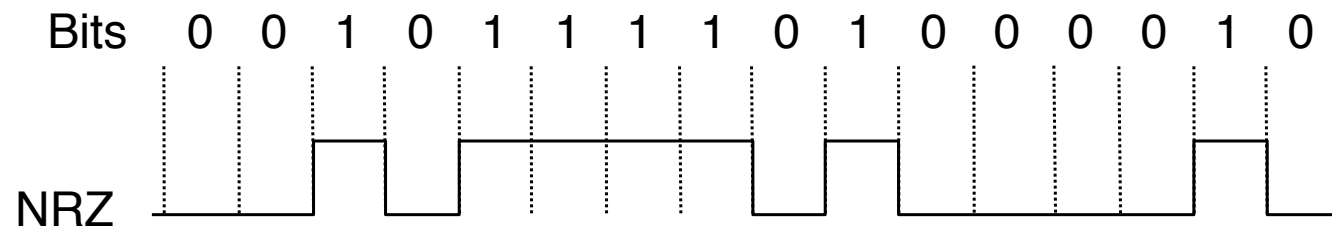
# Link layer: services

- Line termination (serialize/deserialize)
  - Physical signaling (into the wire) & extracting digital signal (out of the wire)
  - Encapsulate datagram into frame (adding) header, trailer
- **Encoding, error detection, and error correction**
  - **Error detection:**
    - errors caused by signal attenuation, noise
    - receiver detects errors, signals retransmission, or drops frame
  - **Error correction:**
    - receiver identifies and corrects bit error(s) without retransmission
- Medium access control
  - channel medium access in a shared medium:
  - “Who should talk?”

# Handling digital and physical Information

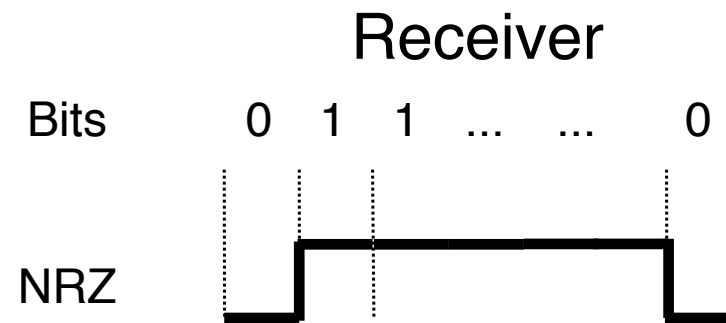
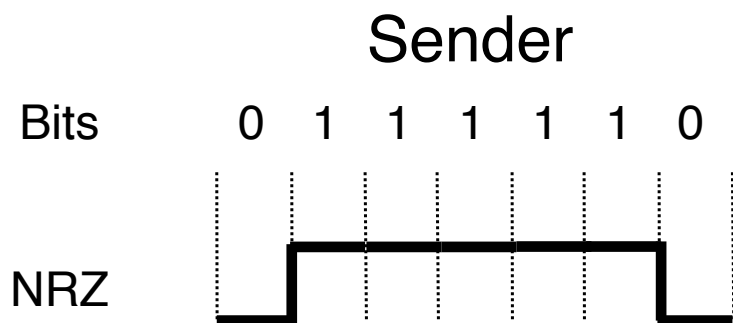
# Encoding and Decoding

- Signals propagate over a physical medium
  - modulate electromagnetic waves
  - e.g., vary voltage
- Encode binary data onto signals
  - e.g., 0 as low signal and 1 as high signal
  - known as **Non-Return to zero** (NRZ)



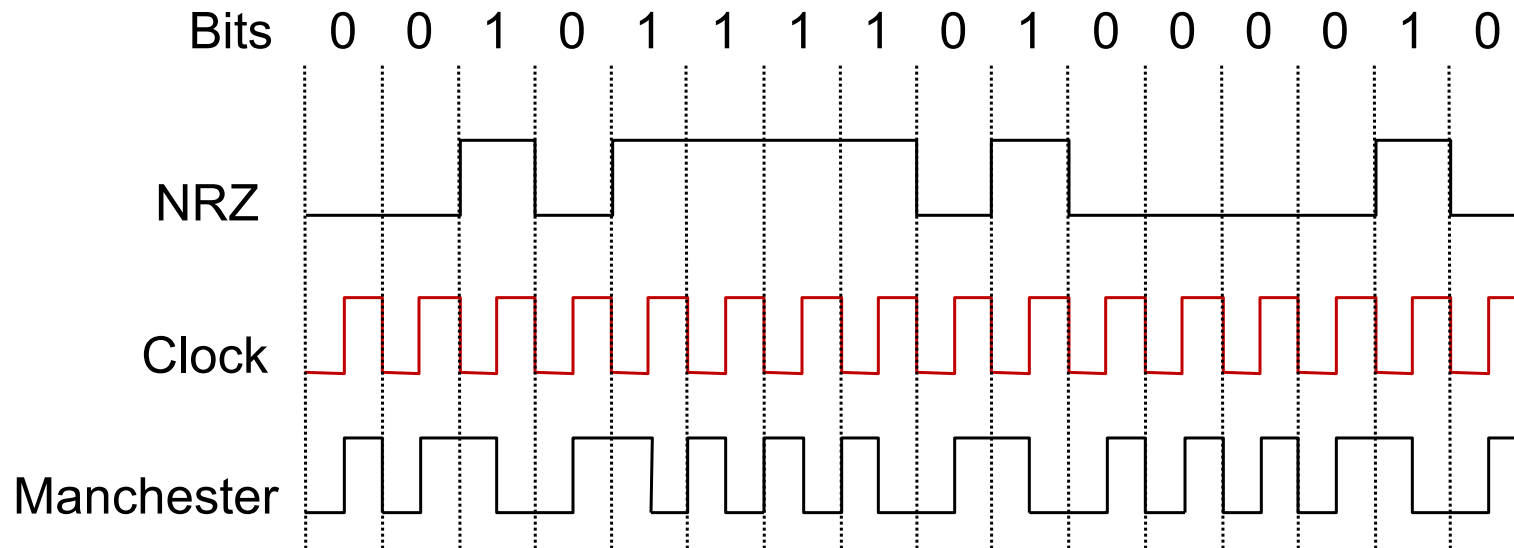
# Clock sync and recovery

- Receiver needs to know when a symbol begins and ends
- One approach: send a clock signal together with data signal
  - Lowers data rate by 2x!
- Another approach: look for **transitions** in the data signal to re-synchronize the clock
- **Long strings of 0s and 1s** make synchronization challenging
- It's like trying to dance in sync without a beat



How many 1s?

# Self-clocking encoding



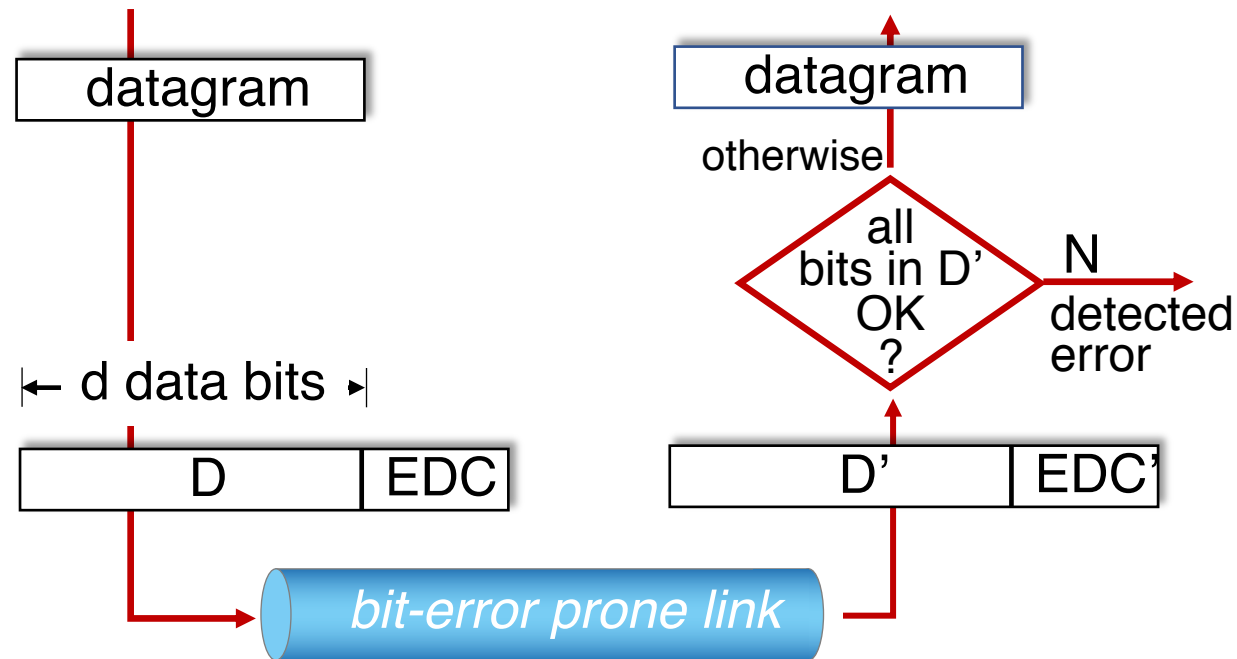
- **Manchester encoding**
- 0 encoded by a positive transition, 1 by a negative transition
- Construct using  $XOR(\text{bit}, \text{clock})$
- Used in early Ethernet standards (up to 10 Mbit/s)

# Error Detection & Correction



# Error detection

EDC: error detection and correction bits (e.g., redundancy)  
D: data protected by error checking, may include header fields



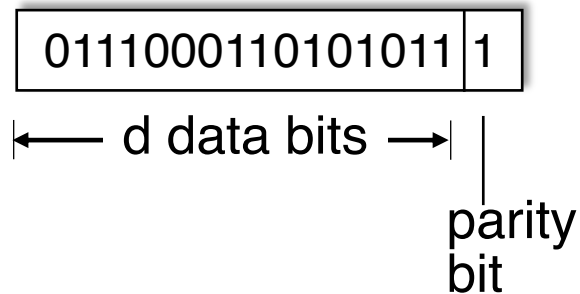
Error detection not 100% reliable!

- protocol may miss some errors, but rarely
- larger EDC field yields better detection and correction

# Parity checking

## single bit parity:

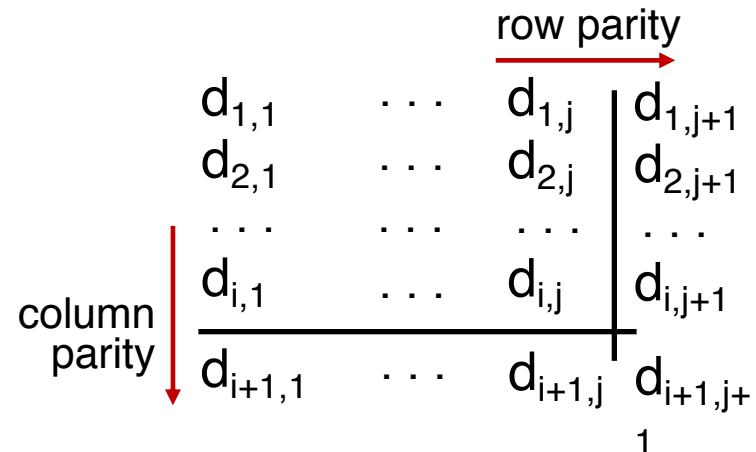
- detect single bit errors



**Even parity:** set parity bit so there is an even number of 1's

## two-dimensional bit parity:

- detect **and correct** single bit errors



no errors:

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

**detected and correctable single-bit error:**

1	0	1	0	1	1
<del>1</del>	<del>0</del>	<del>1</del>	<del>1</del>	<del>0</del>	<del>0</del>
0	1	1	1	0	1
0	0	1	0	1	0

# Review: Transport (UDP/TCP) checksum

*Goal:* detect errors (*i.e.*, flipped bits) in transmitted segment

## sender:

- treat contents of transport segment as sequence of 16-bit integers
- **checksum:** addition (roughly) of segment content
- checksum value put into UDP checksum field

## receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - Yes: *assume* no error
  - No: declare error

# Cyclic Redundancy Check (CRC)

- more powerful error-detection coding than checksums
- D: data bits (given, think of these as a binary number)
- G: bit pattern (**generator**), of  $r+1$  bits (given)



Goal: choose  $r$  CRC bits,  $R$ , such that  $\langle D, R \rangle$  exactly divisible by  $G$

**Do all arithmetic mod 2: all additions, subtractions replaced by XOR**

- receiver knows  $G$ , divides  $\langle D, R \rangle$  by  $G$ . If non-zero remainder: error detected!
- can detect all **burst errors (continuous bit errors)** less than  $r+1$  bits
- widely used in practice (Ethernet, 802.11 WiFi)



# Summary

- Self-clocking encoding useful to synchronize sender & receiver
- Error detection and correction mechanisms:
  - Parity bits: single or a few bits of error
  - CRCs: bursty errors up to a certain size
- Error detection and correction codes widely used across many computer systems



# CS 352

## Connecting Multiple Endpoints into a Single Network

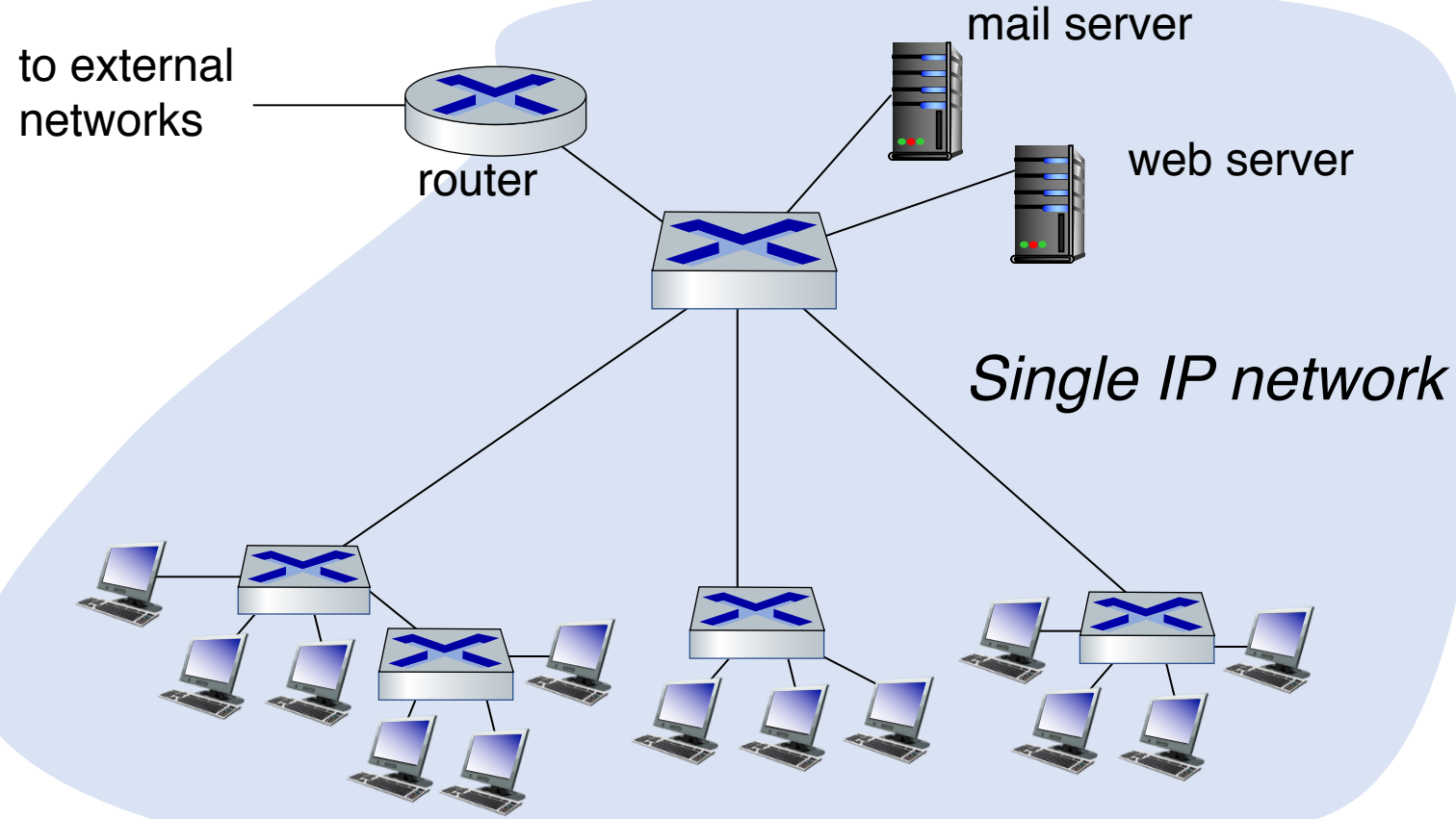
CS 352, Lecture 22.3

<http://www.cs.rutgers.edu/~sn624/352>

Srinivas Narayana



# A small organizational network today



## Switches

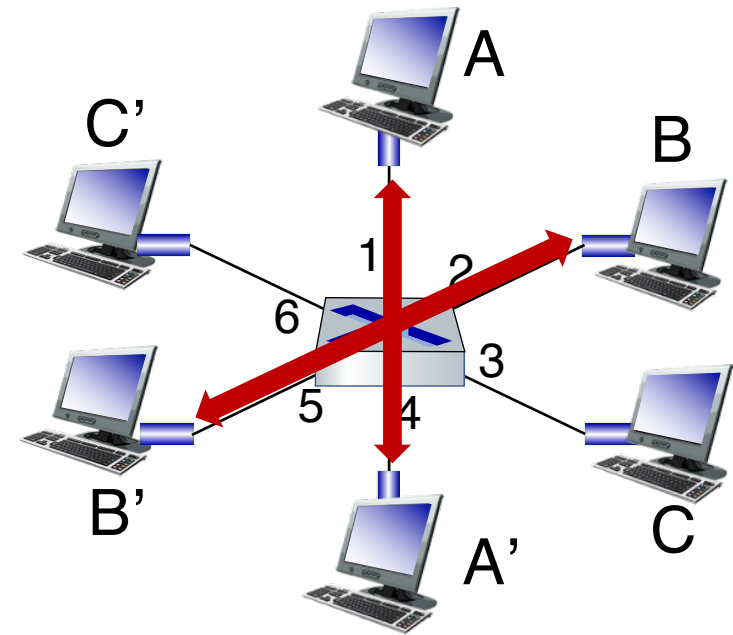
Located in switching closets, connect directly to endpoints.

# Switching

- A **switch** is a link-layer device
  - Examine incoming frame's destination MAC address
  - Selectively forward frame to one-or-more outgoing links when frame is to be forwarded
  - Can store link layer frames in switch buffers
- **Transparent**: hosts unaware of presence of switches
- Plug-and-play, **self-learning**
  - Switches do not need to be configured

# Switch: multiple simultaneous transmissions

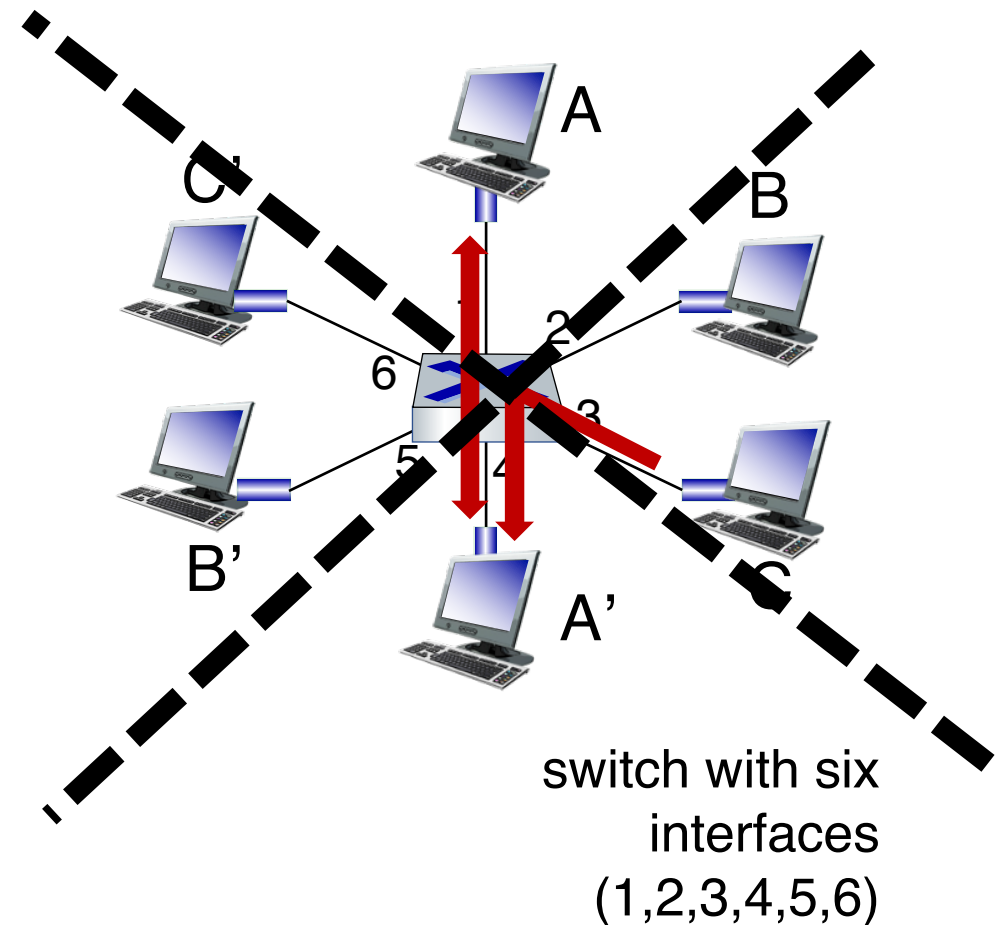
- hosts have dedicated, direct connection to switch
- switches **buffer** packets
- Same link layer protocol used on each incoming link
  - full duplex links
  - No medium access control needed (more next lecture)
- switching: A-to-A' and B-to-B' can transmit **simultaneously**



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

# Switch: multiple simultaneous transmissions

- hosts have dedicated, direct connection to switch
- switches **buffer** packets
- Same link layer protocol used on each incoming link
  - full duplex links
  - No medium access control needed (more next lecture)
- switching: A-to-A' and B-to-B' can transmit **simultaneously**
- However, A → A' and C → A' can't happen simultaneously



# Switched LAN

- If switches don't need to be configured, how can switches route to the correct endpoints?
- Process known as **MAC learning** or **layer-2 bridging**
  - a configuration-free, **learning-based** routing protocol

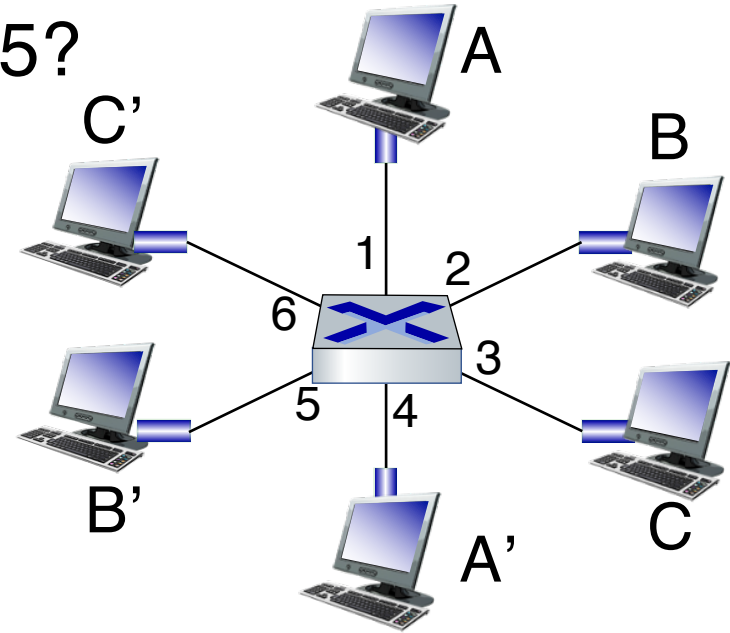
# Switch forwarding table

Ex: how does switch know A' reachable via interface 4, B' reachable via interface 5?

**Each switch has a MAC table.**

Each entry:

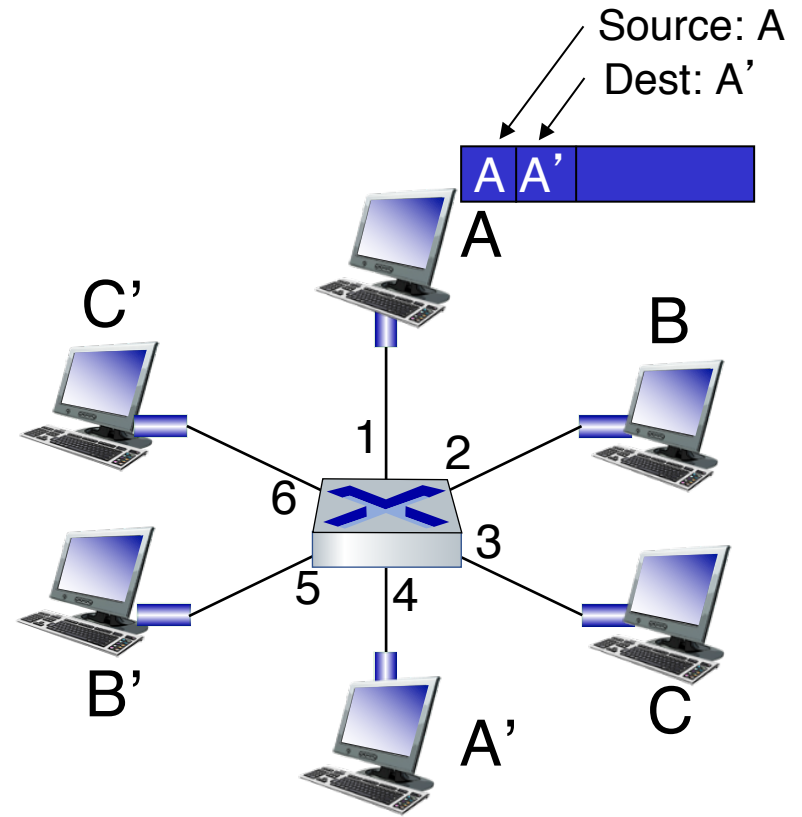
- (MAC address of host, interface to reach host, timestamp)
- **looks like a forwarding table!**



How are entries created and maintained in the MAC table?

# MAC 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
A	1	60

*Switch table  
(initially empty)*

# MAC learning: frame forwarding

when a frame received at switch:

1. record incoming link, **source MAC** address
2. index switch table using **destination MAC** address
3. if **entry found** for destination  
then {  
    if destination on link from which frame arrived  
    then **drop frame**  
    else **forward frame** on interface indicated by entry  
}  
else **flood** /\* forward on all ports except arriving interface \*/

MAC	port	TTL
A	1	60

Flooding is only acceptable because all endpoints are in the same org/IP network.



# Forwarding: Example

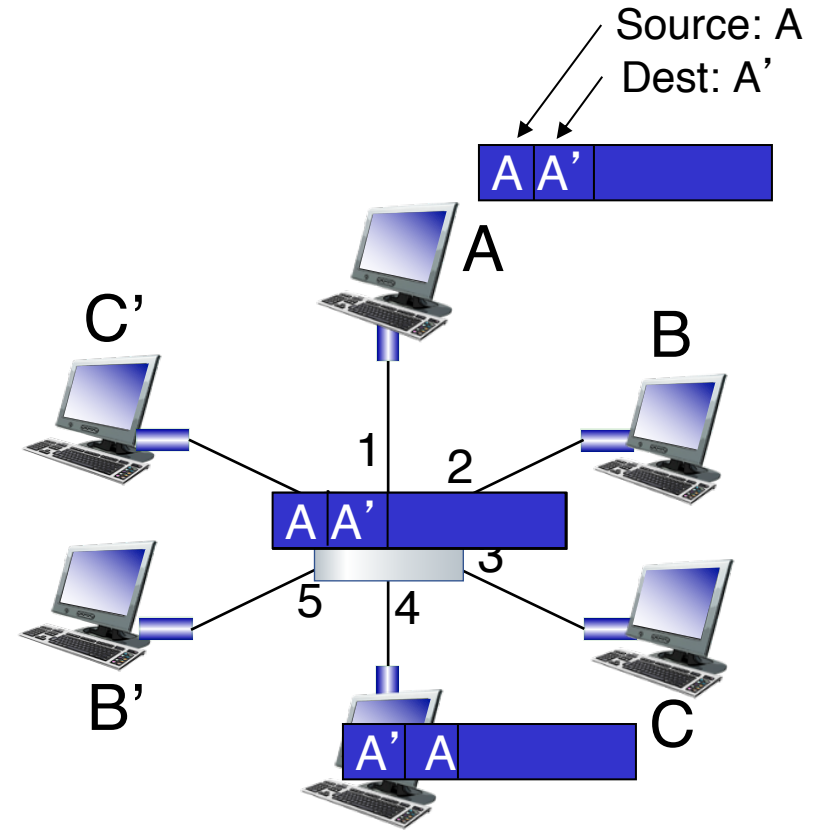
- frame destination A'.  
Interface unknown  
(not in table)

**flood**

- destination A location  
known:

**Selectively send on one link**

- Subsequent  $A \leftrightarrow A'$   
packets not flooded

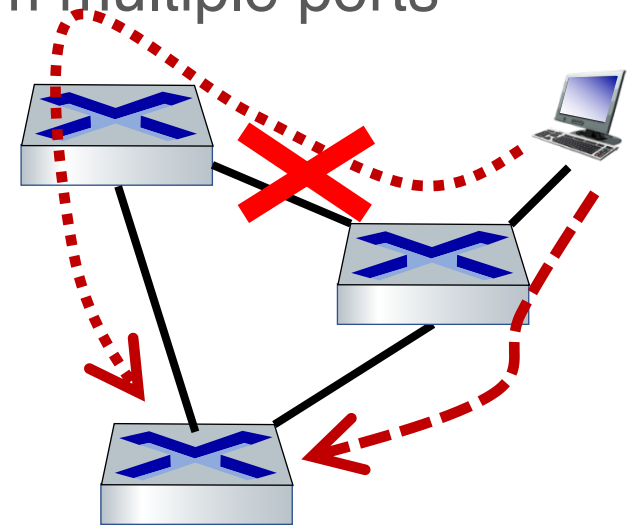


MAC addr	interface	TTL
A	1	60
A'	4	60

*switch table  
(initially empty)*

# Interconnecting switches

- MAC learning switches can be connected together
  - The algorithm works the same way!
- **Complication:** what if there are **loops** in the switch topology?
  - Flooding may result in the same packet arriving from multiple ports
- **Ethernet: spanning tree protocol**
  - Switches discover the switch-level graph
  - Process akin to link state advertisements flooding
  - Then, switches use a **loop-free** subset of links
    - A spanning tree of the network graph



# Switches vs. routers

Both can store, buffer, and forward.

- **routers**: network-layer devices (examine network-layer headers)
- **switches**: link-layer devices (examine link-layer headers)

Both have forwarding tables.

- **routers**: compute forwarding tables using routing algorithms, link configurations, and announced IP addresses
- **switches**: learn forwarding table using flooding and learning MAC addresses

# Summary

- Enterprises often use switches for their ease of configuration and plug-and-play nature
  - Switched Ethernet: popular in dorms and office buildings
- Switches can discover endpoints
- **Flooding** facilitates reachability across endpoints. Only possible as all endpoints part of the same IP network
- **MAC learning** records where endpoints send from, enabling the discovery of endpoint-port associations without prior knowledge