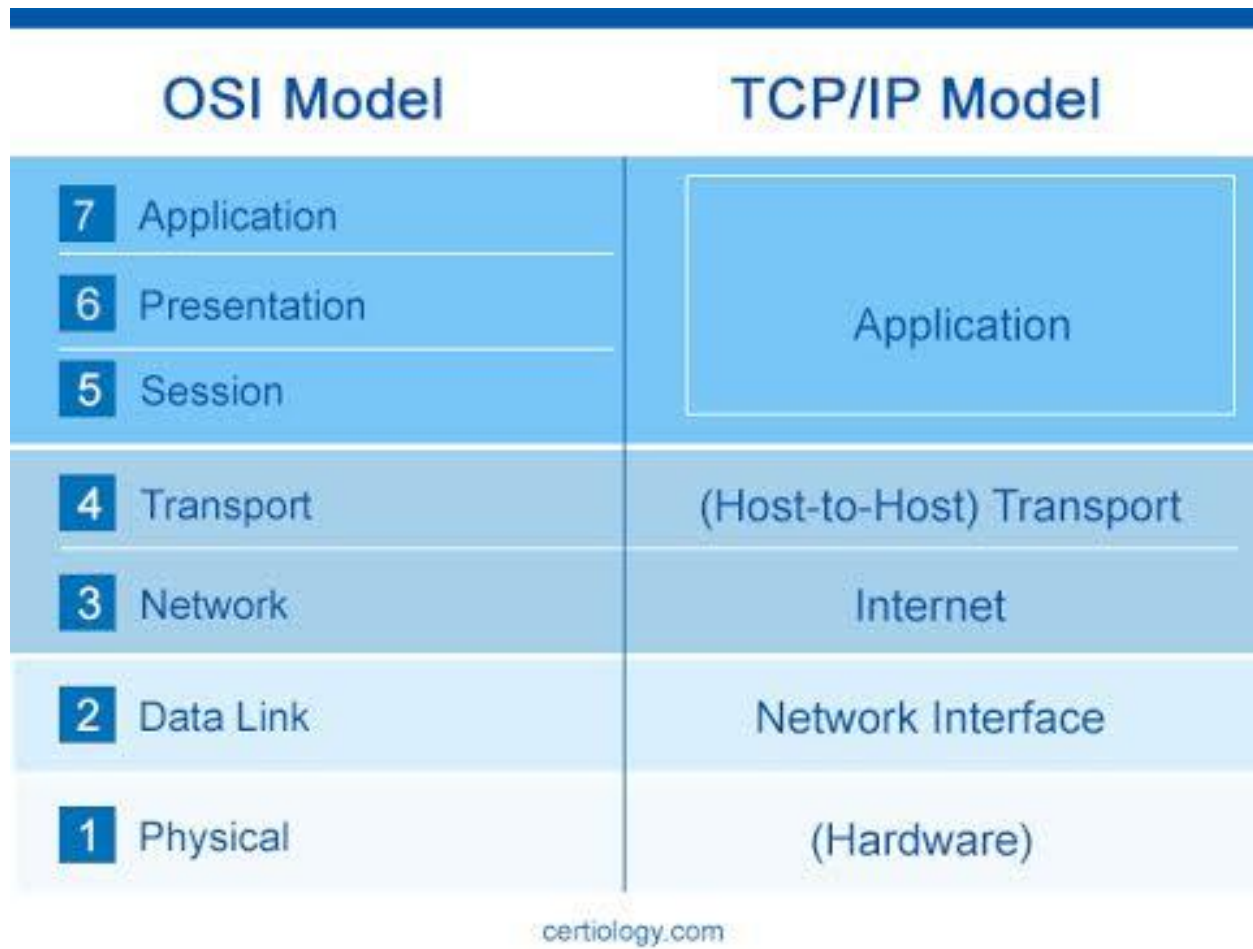
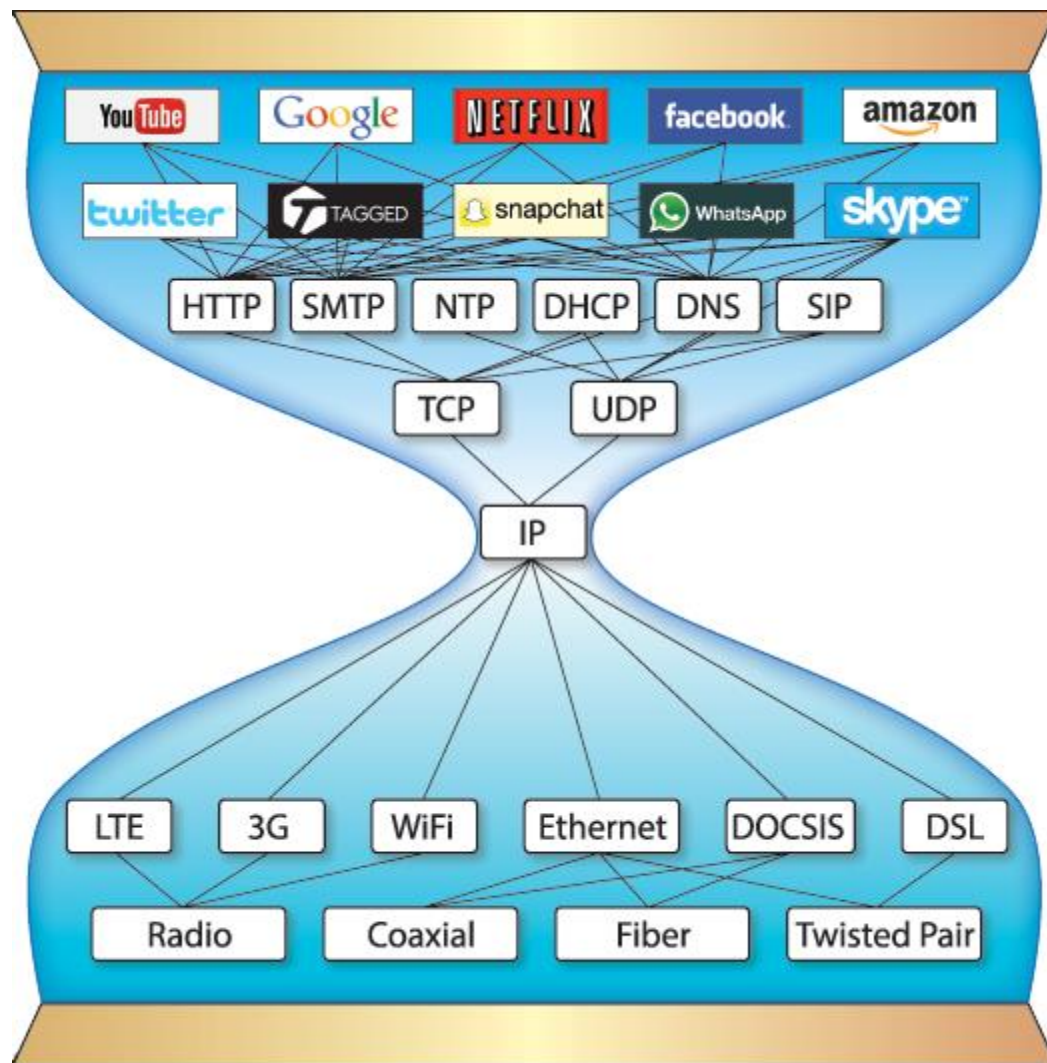


回顾：传输层与网络层



回顾：传输层与网络层



回顾：传输层与网络层

□ 从原理到实现

- 传输层的可靠数据传输：RDT到TCP
- 网络层的数据传输机制：
 - 数据报到IP
- 网络层路由算法
 - 链路状态算法到OSPF
 - 距离向量算法到RIP

回顾：传输层与网络层

▣ 实现细节体现设计思想

○ 传输层TCP

- 连接管理：三次握手、四次挥手
- 流控制
- 拥塞控制：AIMD、慢启动

○ 网络层IP

- CIDR
- 路由地址聚合
- IPv4 and IPv6
- 层次化路由

回顾：传输层与网络层

□ 经验同样重要

○ 传输层TCP

- RTT的估计：RTT的均值与方差、历史值与当前值

$$\text{TimeoutInterval} = \text{EstimatedRTT} + 4 * \text{DevRTT}$$

$$\text{EstimatedRTT} = (1-\alpha) * \text{EstimatedRTT} + \alpha * \text{SampleRTT}$$

(typically, $\alpha = 0.125$)

$$\text{DevRTT} = (1-\beta) * \text{DevRTT} + \beta * |\text{SampleRTT} - \text{EstimatedRTT}|$$

(typically, $\beta = 0.25$)

- 超时与三次重复ACK
- AIMD、慢启动、阈值

回顾：传输层与网络层

▣ 经验同样重要

○ 网络层

- 路由器缓冲器大小
- IP地址分配：ABC类
- RIP、OSPF协议中的参数
- TTL

Chapter 5: The Data Link Layer

Our goals:

- ❑ understand principles behind data link layer services:
 - error detection, correction
 - sharing a broadcast channel: multiple access
 - link layer addressing
 - reliable data transfer, flow control: *done!*
- ❑ instantiation and implementation of various link layer technologies

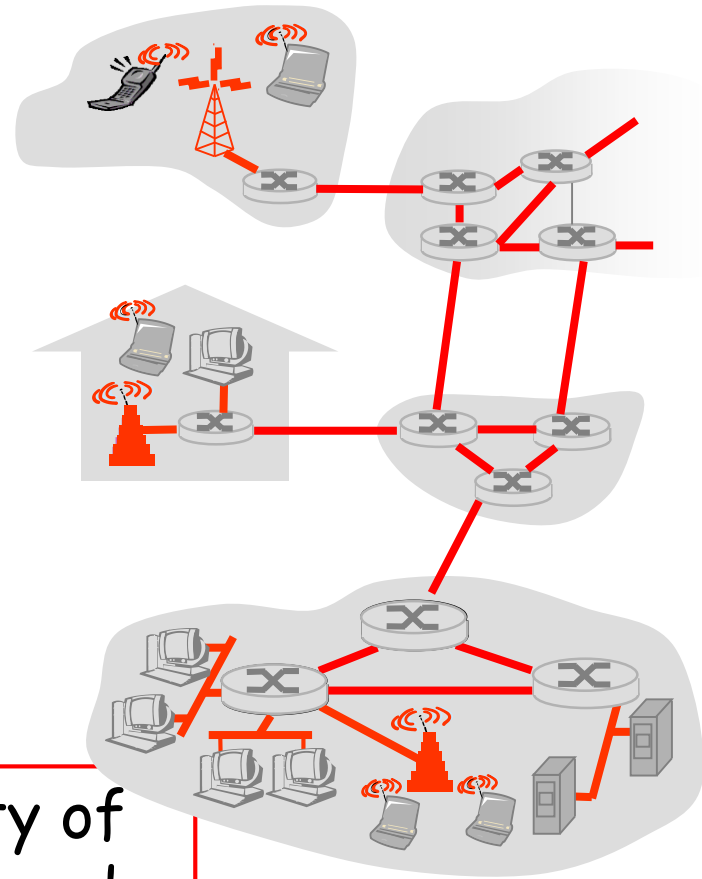
Link Layer

- ❑ 5.1 Introduction and services
- ❑ 5.2 Error detection and correction
- ❑ 5.3 Multiple access protocols
- ❑ 5.4 Link-layer Addressing
- ❑ 5.5 Ethernet
- ❑ 5.6 Link-layer switches
- ❑ 5.7 PPP
- ❑ 5.8 Link virtualization: ATM, MPLS

Link Layer: Introduction

Some terminology:

- ❑ hosts and routers are **nodes**
- ❑ communication channels that connect adjacent nodes along communication path are **links**
 - wired links
 - wireless links
 - LANs
- ❑ layer-2 packet is a **frame**, encapsulates datagram



data-link layer has responsibility of transferring datagram from one node to adjacent node over a link

Link layer: context

- ❑ datagram transferred by different link protocols over different links:
 - e.g., Ethernet on first link, frame relay on intermediate links, 802.11 on last link
- ❑ each link protocol provides different services
 - e.g., may or may not provide rdt over link

transportation analogy

- ❑ trip from Princeton to Lausanne
 - limo: Princeton to JFK
 - plane: JFK to Geneva
 - train: Geneva to Lausanne
- ❑ tourist = **datagram**
- ❑ transport segment = **communication link**
- ❑ transportation mode = **link layer protocol**
- ❑ travel agent = **routing algorithm**

Link Layer Services

❑ *framing, link access:*

- encapsulate datagram into frame, adding header, trailer
- channel access if shared medium
- "MAC" addresses used in frame headers to identify source, dest
 - different from IP address!

❑ *reliable delivery between adjacent nodes*

- we learned how to do this already (chapter 3)!
- 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?

Link Layer Services (more)

❑ *flow control:*

- pacing between adjacent sending and receiving nodes

❑ *error detection:*

- errors caused by signal attenuation, noise.
- receiver detects presence of errors:
 - signals sender for retransmission or drops frame

❑ *error correction:*

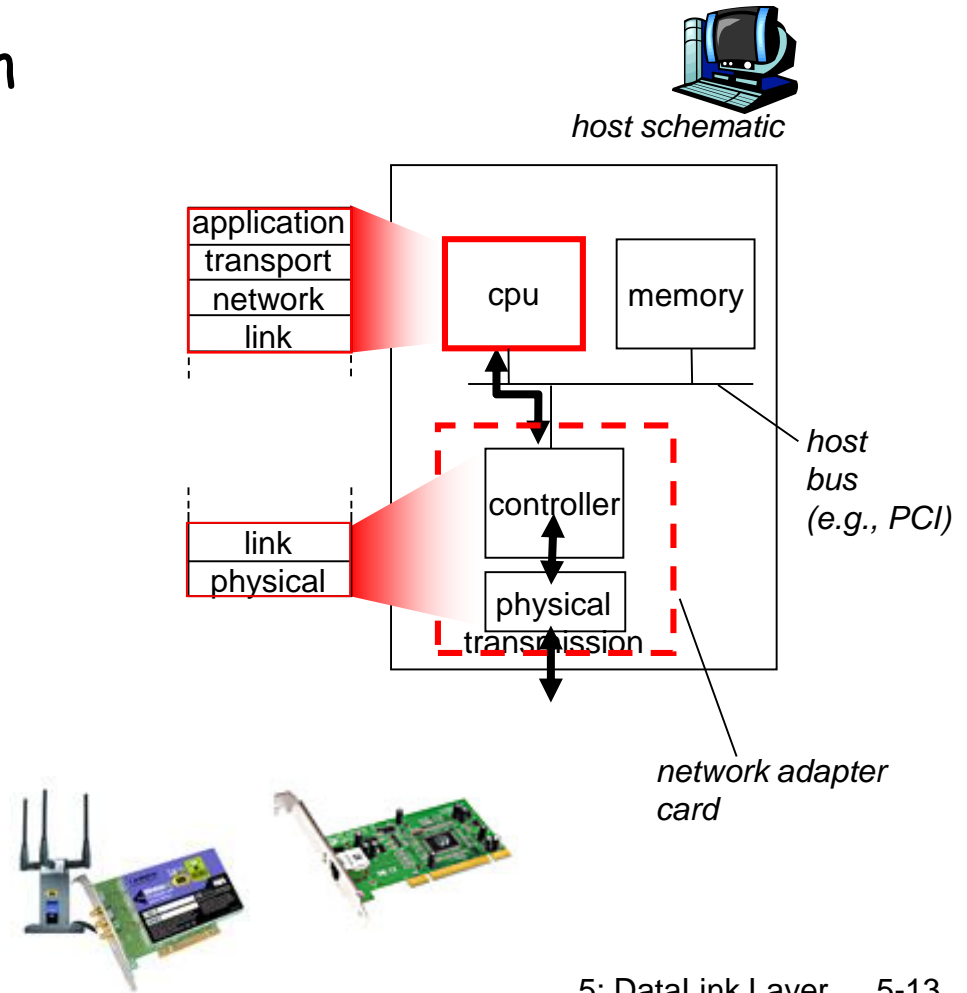
- receiver identifies *and corrects* bit error(s) without resorting to retransmission

❑ *half-duplex and full-duplex*

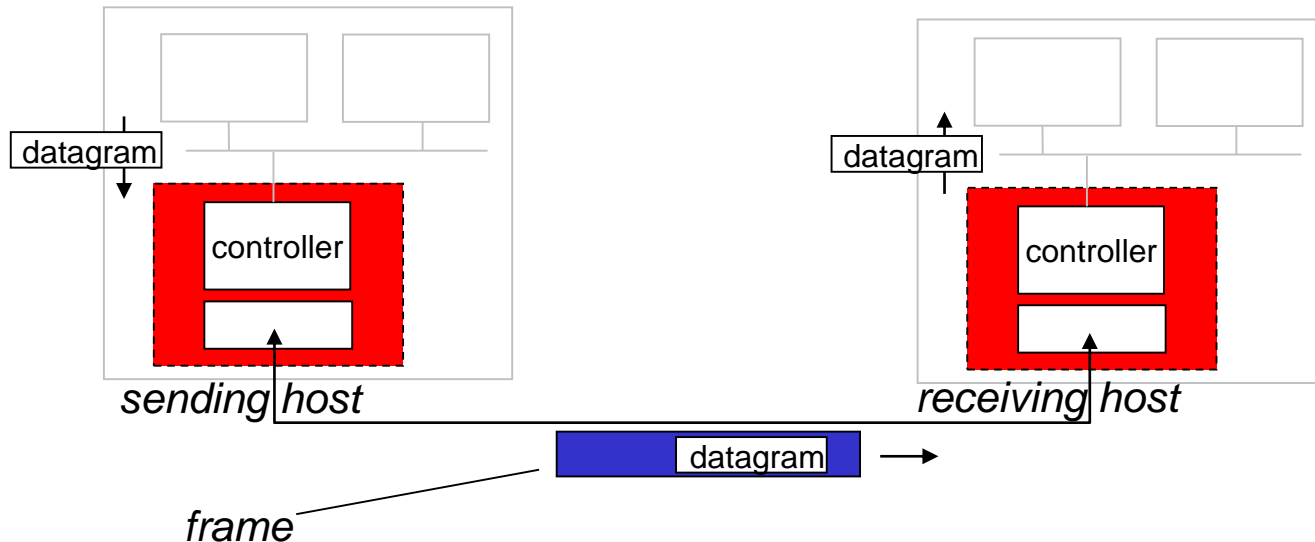
- with half duplex, nodes at both ends of link can transmit, but not at same time

Where is the link layer implemented?

- ❑ in each and every host
- ❑ link layer implemented in "adaptor" (aka **network interface card NIC**)
 - Ethernet card, PCMCIA card, 802.11 card
 - implements link, physical layer
- ❑ attaches into host's system buses
- ❑ combination of hardware, software, firmware



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

Link Layer

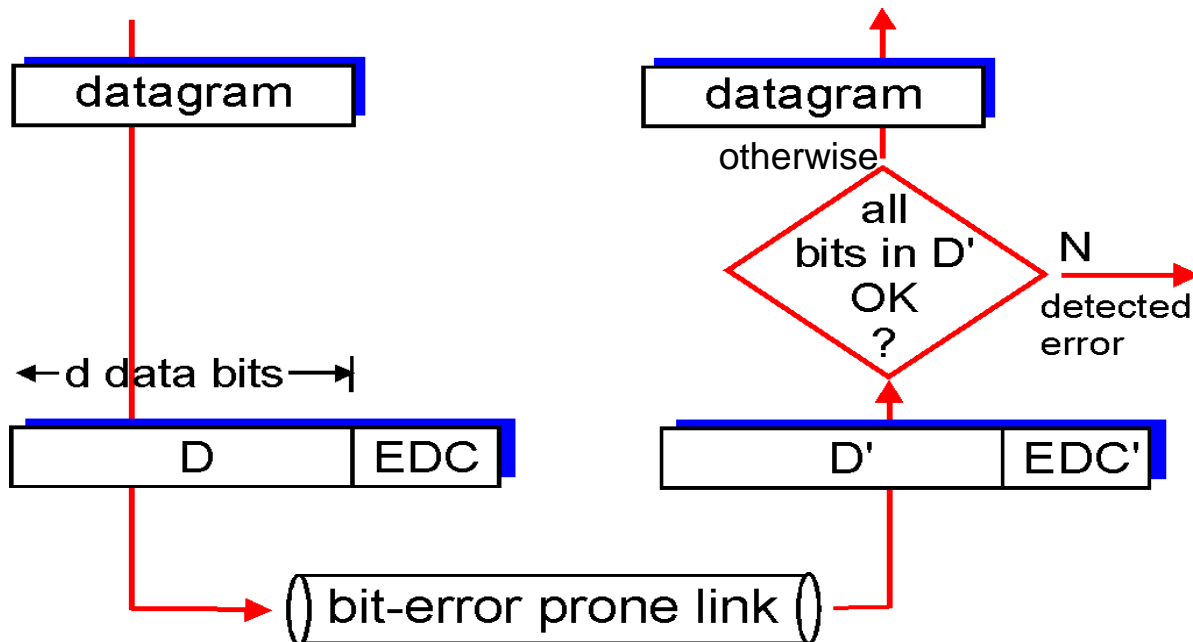
- ❑ 5.1 Introduction and services
- ❑ 5.2 Error detection and correction
- ❑ 5.3 Multiple access protocols
- ❑ 5.4 Link-layer Addressing
- ❑ 5.5 Ethernet
- ❑ 5.6 Link-layer switches
- ❑ 5.7 PPP
- ❑ 5.8 Link Virtualization: ATM. MPLS

Error Detection

EDC= Error Detection and Correction bits (redundancy)

D = Data protected by error checking, may include header fields

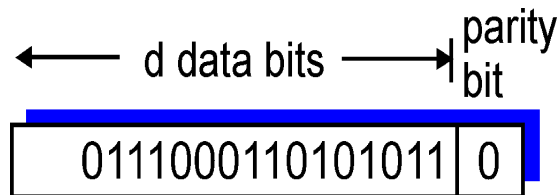
- Error detection not 100% reliable!
 - protocol may miss some errors, but rarely
 - larger EDC field yields better detection and correction



Parity Checking

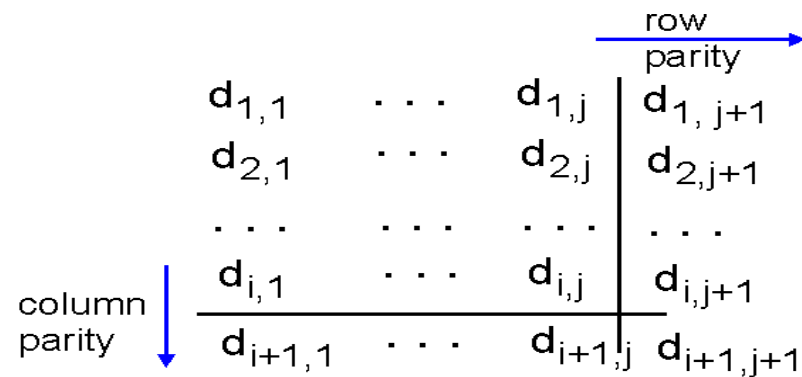
Single Bit Parity:

Detect single bit errors



Two Dimensional Bit Parity:

Detect and correct single bit errors



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

no errors

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

parity
error

*correctable
single bit error*

Internet checksum (review)

Goal: detect "errors" (e.g., flipped bits) in transmitted packet (note: used at transport and network layers)

Sender:

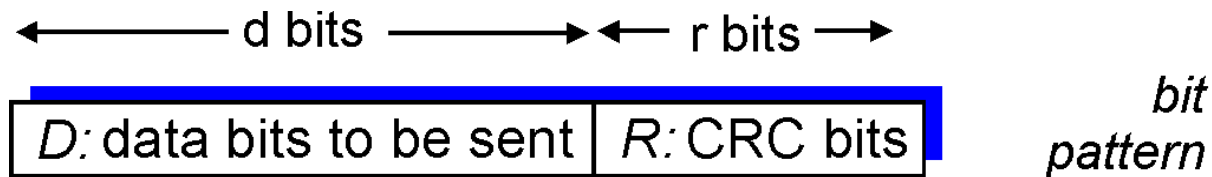
- ❑ treat segment contents as sequence of 16-bit integers
- ❑ checksum: addition (1's complement sum) of segment contents
- ❑ sender puts checksum value into UDP checksum field

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?

Checksumming: Cyclic Redundancy Check

- ❑ view data bits, **D**, as a binary number
- ❑ choose $r+1$ bit pattern (generator), **G**
- ❑ goal: choose r CRC bits, **R**, such that
 - $\langle D, R \rangle$ exactly divisible by G (modulo 2)
 - receiver knows G , divides $\langle D, R \rangle$ 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, ATM)



$$D * 2^r \text{ XOR } R$$

mathematical formula

CRC Example

Want:

$$D \cdot 2^r \text{ XOR } R = nG$$

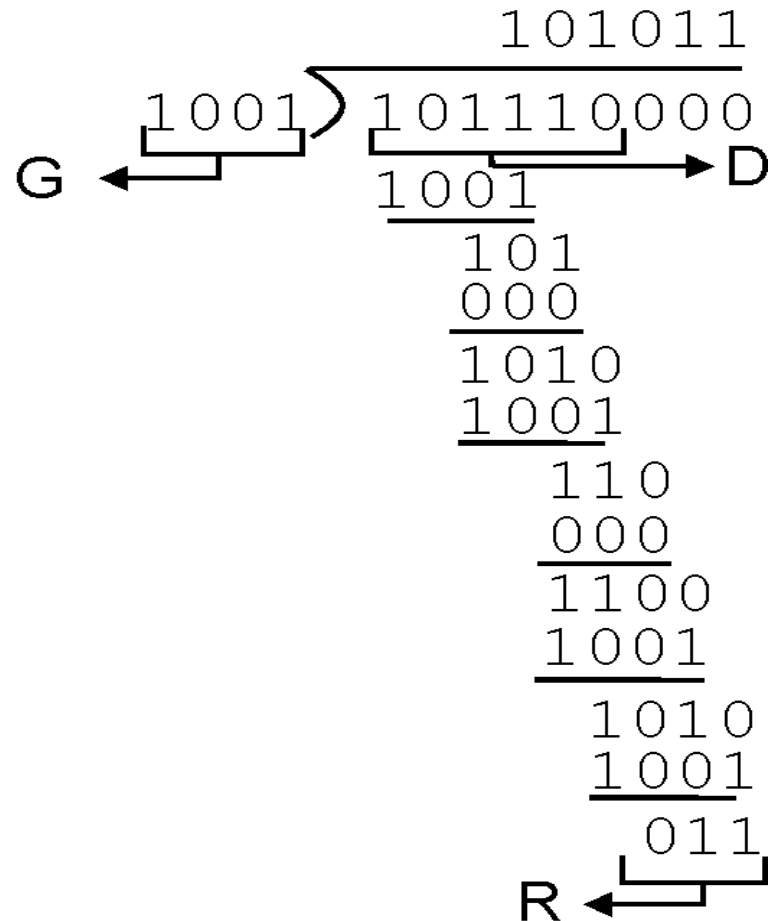
equivalently:

$$D \cdot 2^r = nG \text{ XOR } R$$

equivalently:

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

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



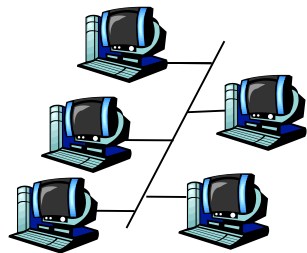
Link Layer

- ❑ 5.1 Introduction and services
- ❑ 5.2 Error detection and correction
- ❑ 5.3 Multiple access protocols
- ❑ 5.4 Link-layer Addressing
- ❑ 5.5 Ethernet
- ❑ 5.6 Link-layer switches
- ❑ 5.7 PPP
- ❑ 5.8 Link Virtualization: ATM, MPLS

Multiple Access Links and Protocols

Two types of "links":

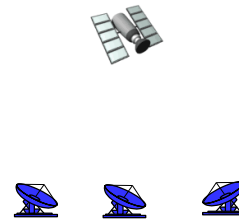
- ❑ point-to-point
 - PPP for dial-up access
 - point-to-point link between Ethernet switch and host
- ❑ **broadcast** (shared wire or medium)
 - old-fashioned Ethernet
 - upstream HFC
 - 802.11 wireless LAN



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

Broadcast channel of rate R bps

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 rate R/M
3. fully decentralized:
 - no special node to coordinate transmissions
 - no synchronization of clocks, slots
4. simple

MAC Protocols: a 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

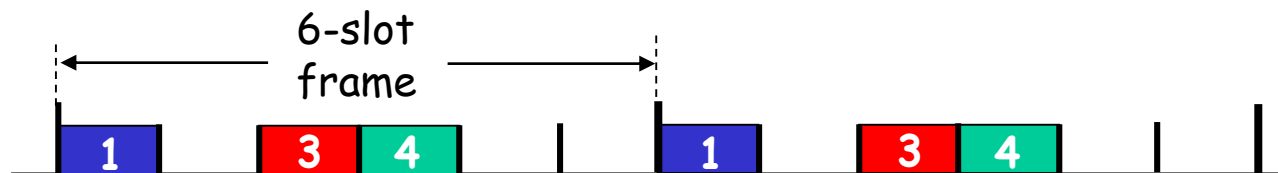
❑ "Taking turns"

- nodes take turns, but nodes with more to send can take longer turns

Channel Partitioning MAC protocols: TDMA

TDMA: time division multiple access

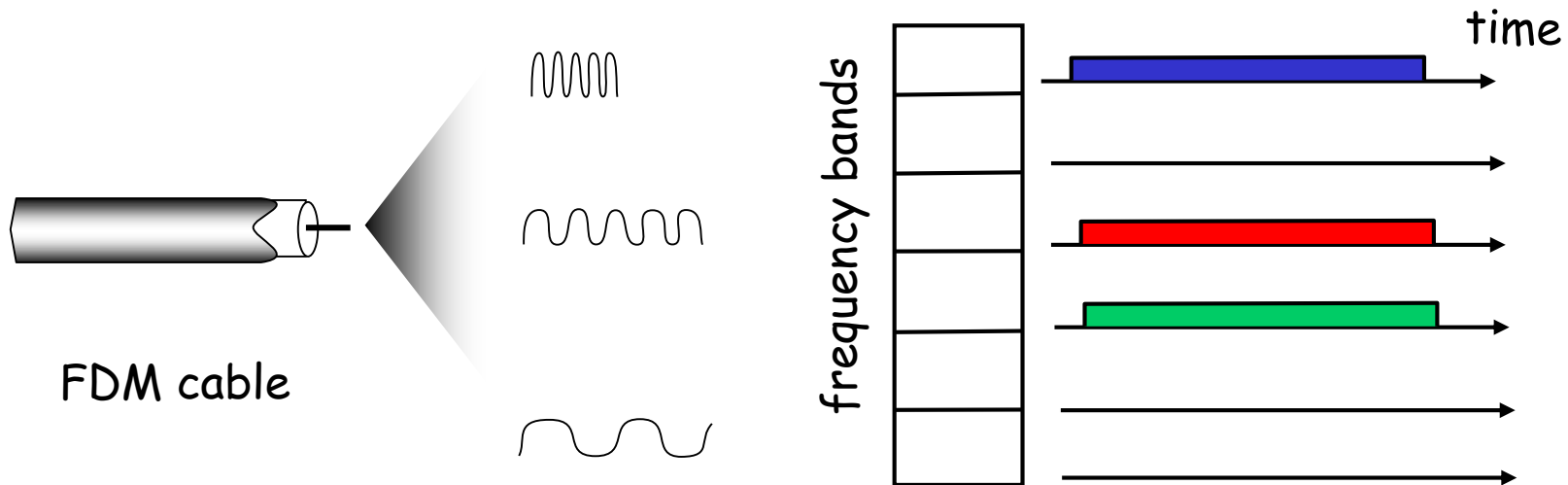
- ❑ access to channel in "rounds"
- ❑ each station gets fixed length slot (length = pkt trans time) in each round
- ❑ unused slots go idle
- ❑ example: 6-station LAN, 1,3,4 have pkt, slots 2,5,6 idle




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 pkt, 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

Slotted ALOHA

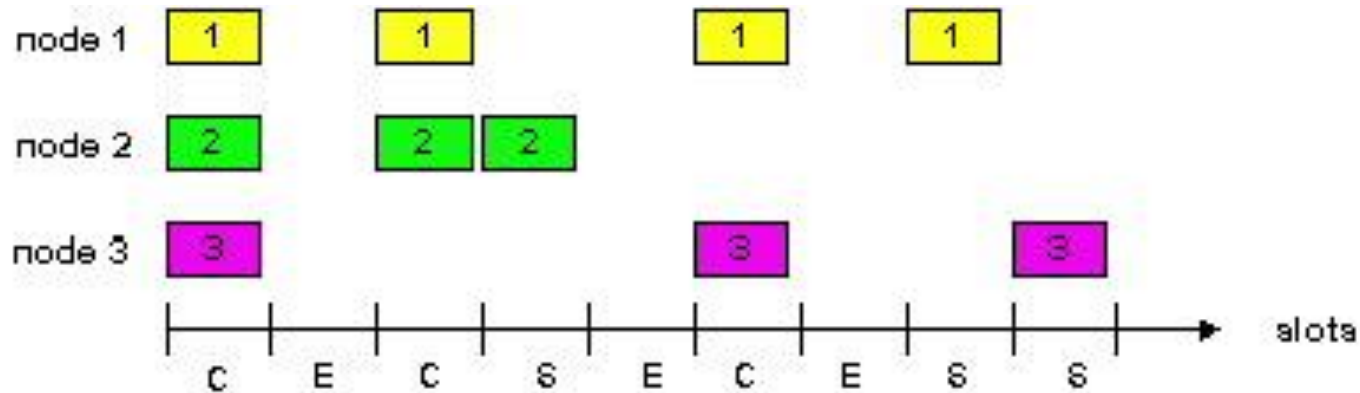
Assumptions:

- ❑ 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 prob. 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 transmits in slot with probability p
- prob that given node has success in a slot = $p(1-p)^{N-1}$
- prob that any node has a success = $Np(1-p)^{N-1}$

- 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:

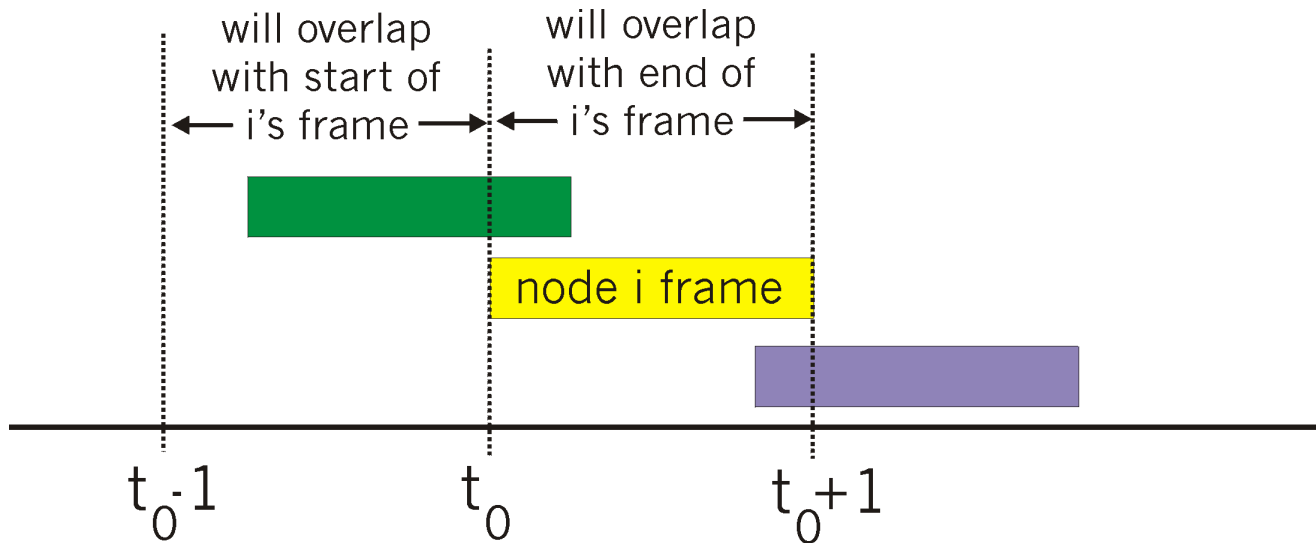
$$\text{Max efficiency} = 1/e = .37$$

At best: channel used for useful transmissions 37% of time!



Pure (unslotted) ALOHA

- ❑ unslotted Aloha: simpler, no synchronization
- ❑ when frame first arrives
 - transmit immediately
- ❑ collision probability **increases:** *Why?*
 - frame sent at t_0 collides with other frames sent in $[t_0-1, t_0+1]$



Pure Aloha efficiency

$P(\text{success by given node}) = P(\text{node transmits}) \cdot$

$P(\text{no other node transmits in } [t_0-1, t_0] \cdot$

$P(\text{no other node transmits in } [t_0, t_0+1])$

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

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

... choosing optimum p and then letting $n \rightarrow \infty$...

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

even worse than slotted Aloha!

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!

CSMA collisions

collisions can still occur:

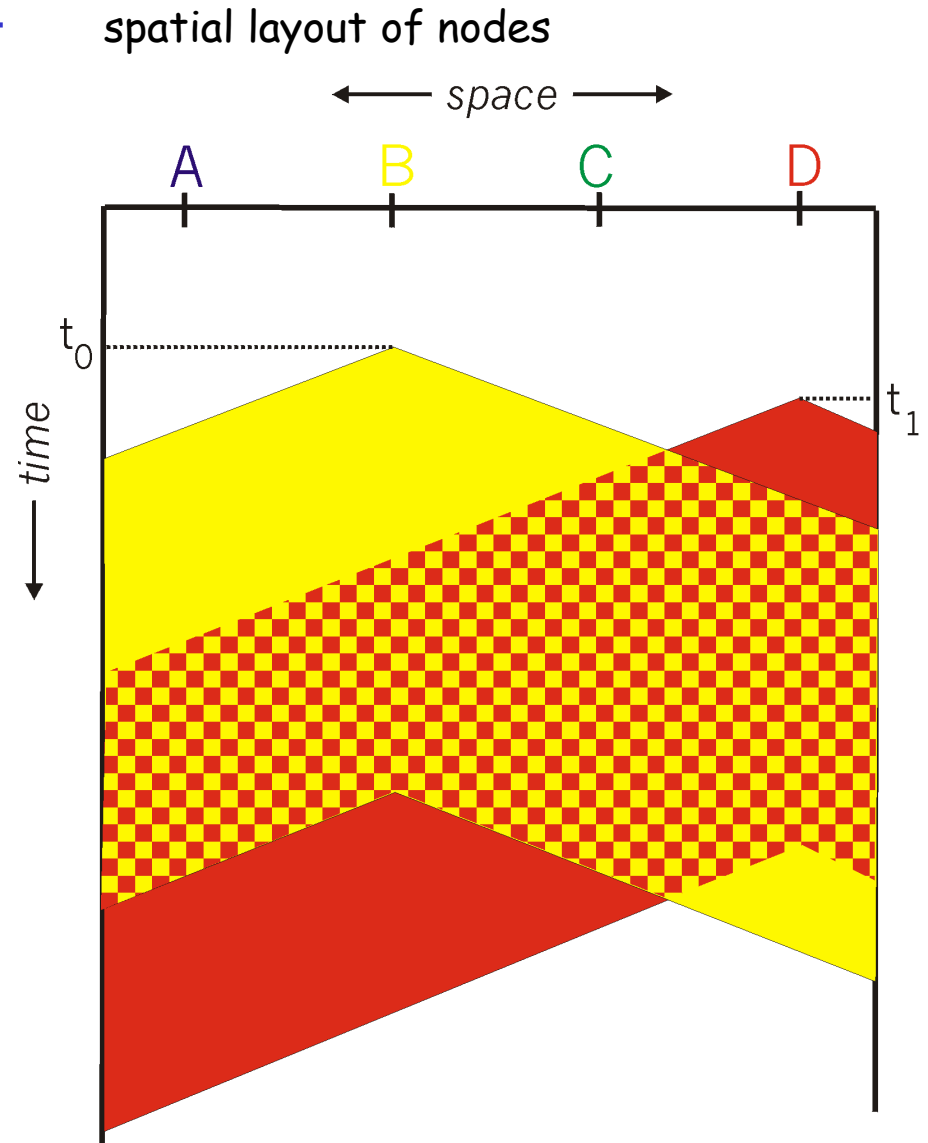
propagation delay means
two nodes may not hear
each other's transmission

collision:

entire packet transmission
time wasted

note:

role of distance & propagation
delay in determining collision
probability



CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, deferral as in CSMA

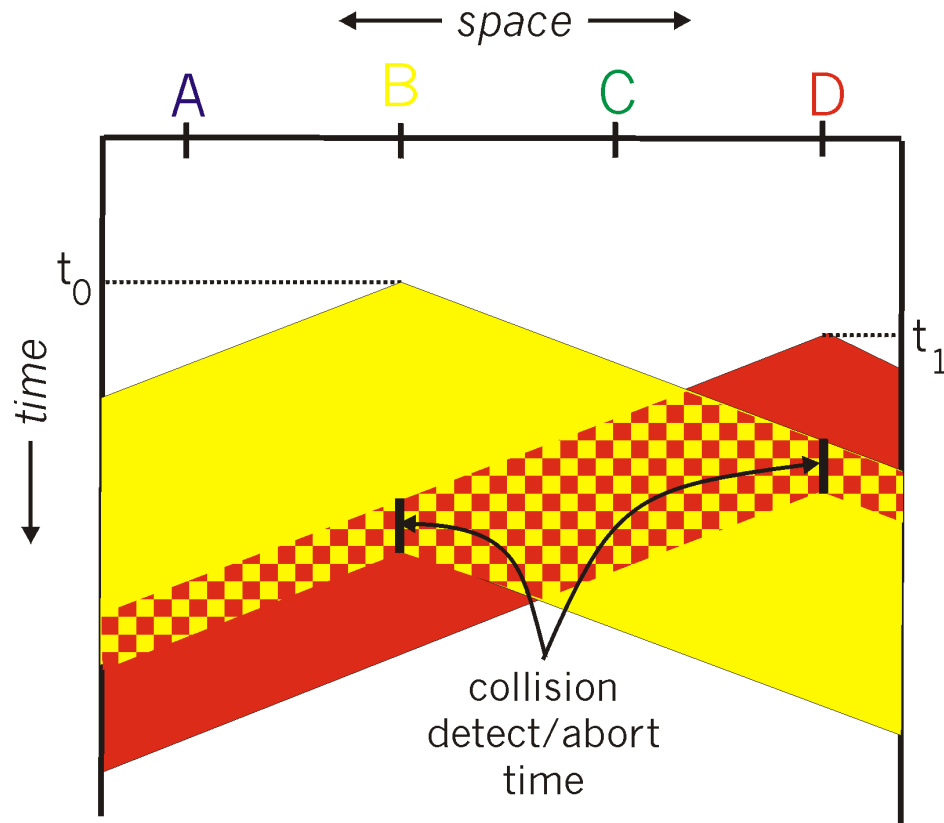
- collisions detected within short time
- colliding transmissions aborted, reducing channel wastage

□ 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

□ human analogy: the polite conversationalist

CSMA/CD collision detection



"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

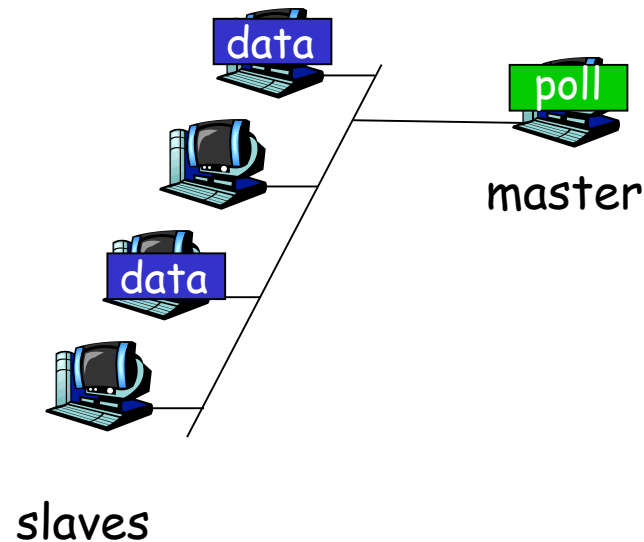
"taking turns" protocols

look for best of both worlds!

"Taking Turns" MAC protocols

Polling:

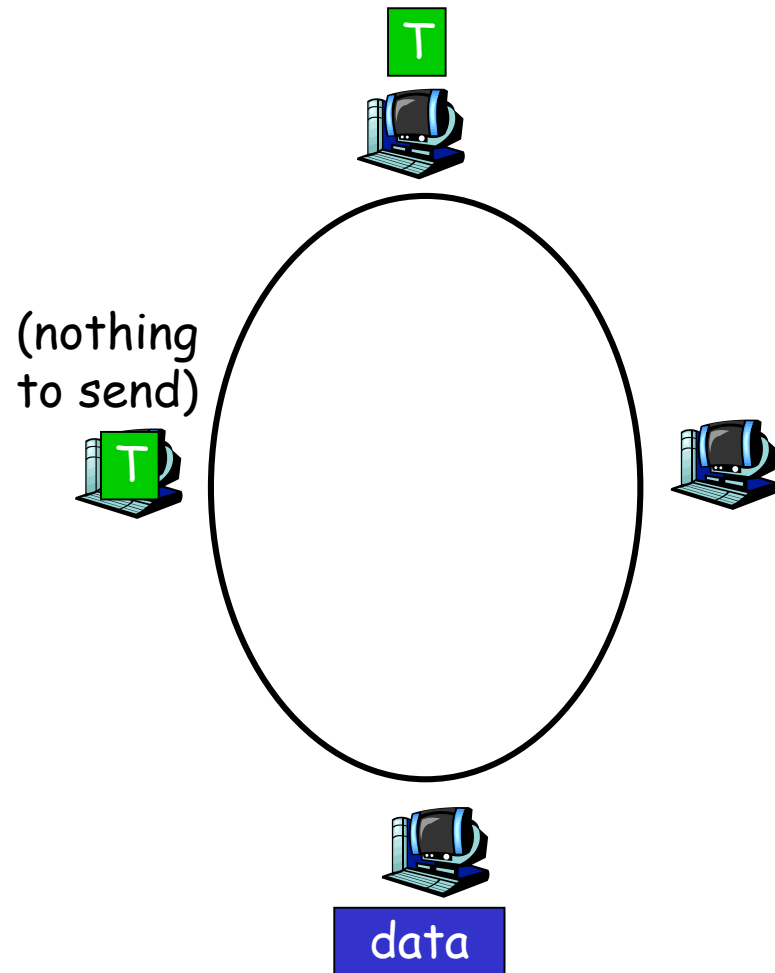
- ❑ master node
 - "invites" slave nodes to transmit in turn
- ❑ typically used with "dumb" slave devices
- ❑ concerns:
 - polling overhead
 - latency
 - single point of failure (master)



"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)



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, IBM Token Ring

Link Layer

- ❑ 5.1 Introduction and services
- ❑ 5.2 Error detection and correction
- ❑ 5.3 Multiple access protocols
- ❑ 5.4 Link-Layer Addressing
- ❑ 5.5 Ethernet
- ❑ 5.6 Link-layer switches
- ❑ 5.7 PPP
- ❑ 5.8 Link Virtualization: ATM, MPLS

MAC Addresses and ARP

□ 32-bit IP address:

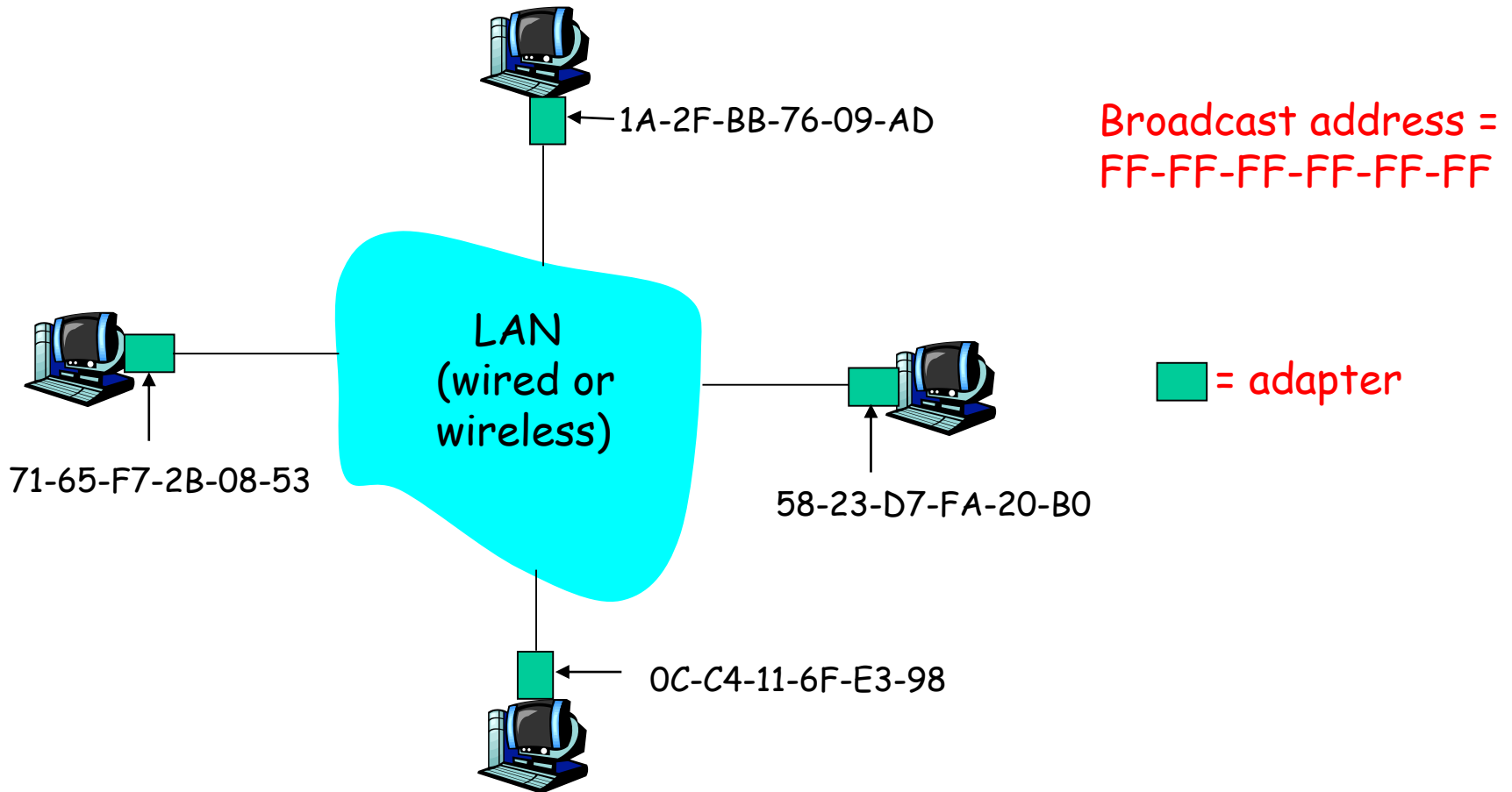
- *network-layer* address
- used to get datagram to destination IP subnet

□ MAC (or LAN or physical or Ethernet) address:

- function: *get frame from one interface to another physically-connected interface (same network)*
- 48 bit MAC address (for most LANs)
 - burned in NIC ROM, also sometimes software settable

LAN Addresses and ARP

Each adapter on LAN has unique LAN address



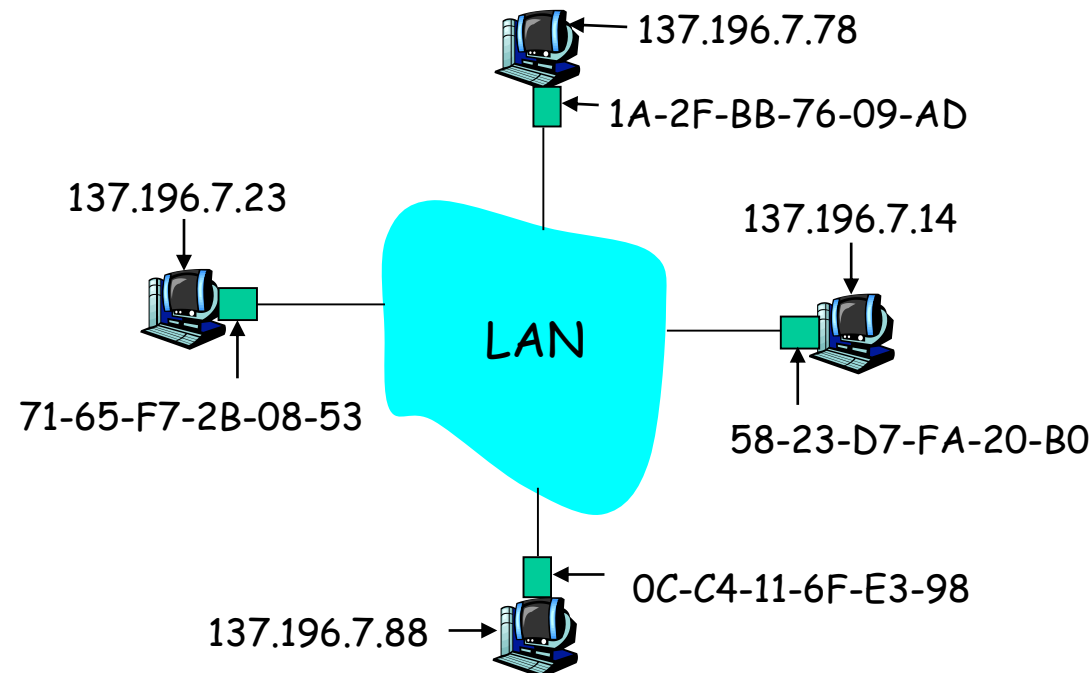
LAN Address (more)

- ❑ MAC address allocation administered by IEEE
- ❑ manufacturer buys portion of MAC address space (to assure uniqueness)
- ❑ analogy:
 - (a) MAC address: like Social Security Number
 - (b) 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

ARP: Address Resolution Protocol

Question: how to determine MAC address of B knowing B's IP address?

- ❑ Each IP node (host, router) on LAN has **ARP** table
- ❑ ARP 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)



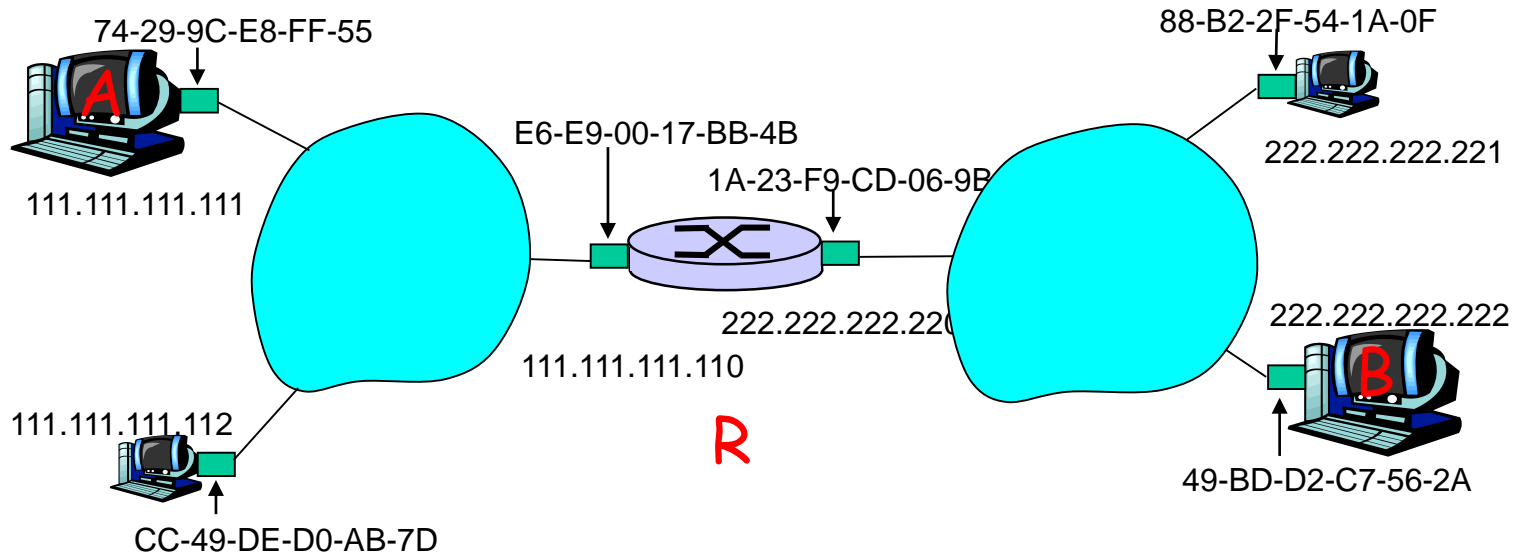
ARP protocol: Same LAN (network)

- ❑ A wants to send datagram to B, and B's MAC address not in A's ARP table.
- ❑ A **broadcasts** ARP query packet, containing B's IP address
 - dest MAC address = FF-FF-FF-FF-FF-FF
 - all machines 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 (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**

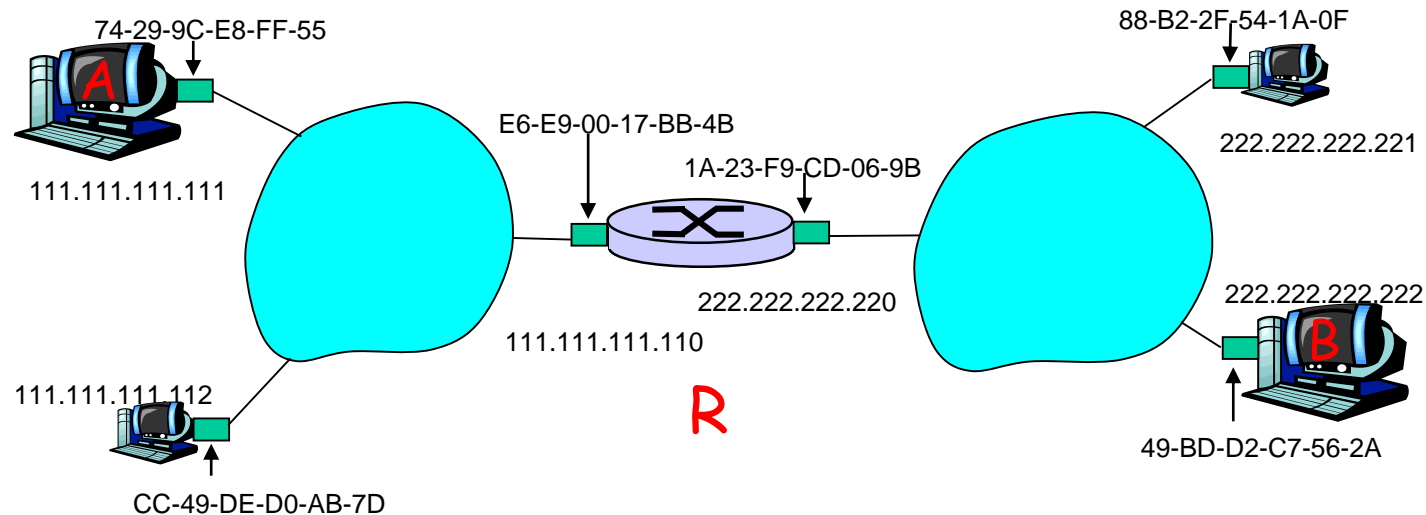
assume A knows B's IP address



- two ARP tables in router R, one for each IP network (LAN)

- ❑ A creates IP datagram with source A, destination B
- ❑ A uses ARP to get R's MAC address for 111.111.111.110
- ❑ A creates link-layer frame with R's MAC address as dest, frame contains A-to-B IP datagram
- ❑ A's NIC sends frame
- ❑ R's NIC receives frame
- ❑ R extracts IP datagram from Ethernet frame, sees its destined to B
- ❑ R uses ARP to get B's MAC address
- ❑ R creates frame containing A-to-B IP datagram sends to B

This is a **really** important example - make sure you understand!



ARP攻击

□ 原理

- **ARP攻击**就是通过伪造**IP**地址和**MAC**地址实现**ARP**欺骗，能够在网络中产生大量的**ARP**通信量使网络阻塞，攻击者只要持续不断的发出伪造的**ARP**响应包就能更改目标主机**ARP**缓存中的**IP-MAC**条目，造成网络中断或中间人攻击。

□ 举例

- 黑客经过收到的**ARP Request**广播包，能够偷听到其它节点的 **(IP, MAC)** 地址，黑客就伪装为**A**，告诉**B** (受害者) 一个假地址，使得**B**在发送给**A** 的数据包都被黑客截取，而**A, B** 浑然不知
- 紫荆的“毒王”

ARP攻击

□ 原因

- **ARP** 是个早期的网络协议，**RFC826**在 **1980**就出版了。早期的互联网采取的是信任模式，在科研、大学内部使用，追求功能、速度，没考虑网络安全。
- 尤其以太网的泛洪特点，能够很方便的用来查询。但这也为日后的黑客开了方便之门。

□ 防止

- 不能。但伤害已经很小。

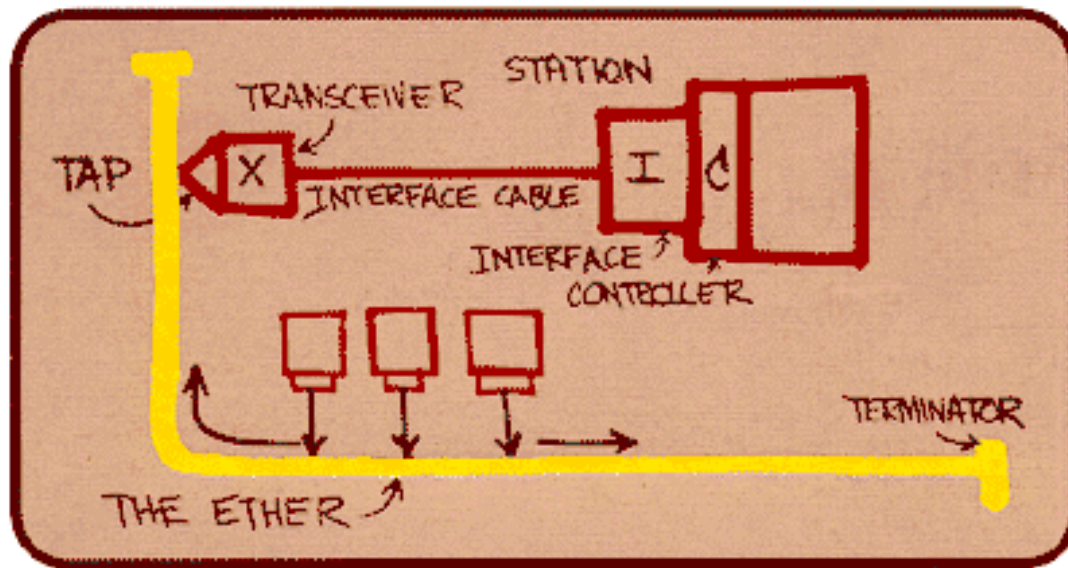
Link Layer

- ❑ 5.1 Introduction and services
- ❑ 5.2 Error detection and correction
- ❑ 5.3 Multiple access protocols
- ❑ 5.4 Link-Layer Addressing
- ❑ 5.5 Ethernet
- ❑ 5.6 Link-layer switches
- ❑ 5.7 PPP
- ❑ 5.8 Link Virtualization: ATM and MPLS

Ethernet

"dominant" wired LAN technology:

- ❑ cheap \$20 for NIC
- ❑ first widely used LAN technology
- ❑ simpler, cheaper than token LANs and ATM
- ❑ kept up with speed race: 10 Mbps - 1000 Gbps



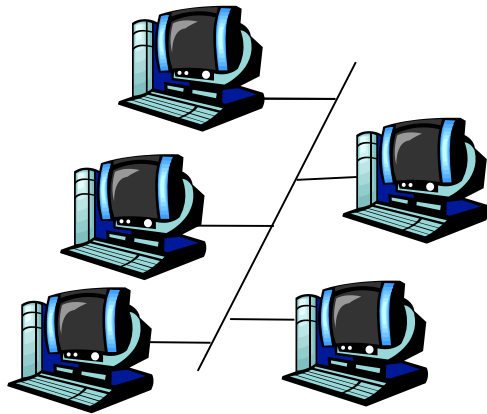
Metcalfe's Ethernet sketch

Ethernet

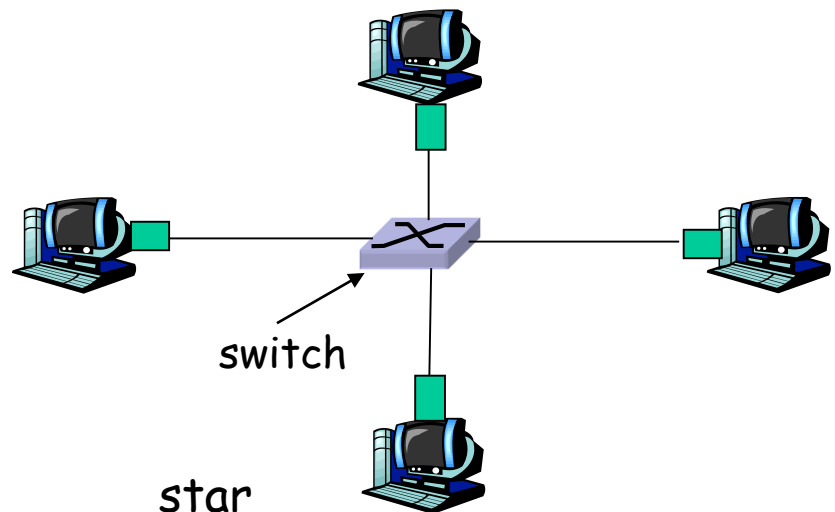
- ❑ 1972年底，**Metcalfe**和**David Boggs**设计了一套网络，将不同的**ALTO**计算机连接起来，接着又把**NOVA**计算机连接到**EARS**激光打印机。
- ❑ 这天，**Metcalfe**写了一段备忘录，称他已将该网络改名为以太网(**Ethernet**)，其灵感来自于"电磁辐射是可以通过发光的以太来传播的这一想法"。
- ❑ 1976年6月，**Metcalfe**和**Boggs**发表了题为："以太网：局域网的分布型信息包交换"的著名论文，1977年底，**Metcalfe**和他的三位合作者获得了"具有冲突检测的多点数据通信系统"的专利，多点传输系统被称为**CSMA/CD**(载波监听多路存取和冲突检测)。从此，以太网就正式诞生了。

Star topology

- ❑ bus topology popular through mid 90s
 - all nodes in same collision domain (can collide with each other)
- ❑ today: star topology prevails
 - active *switch* in center
 - each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)

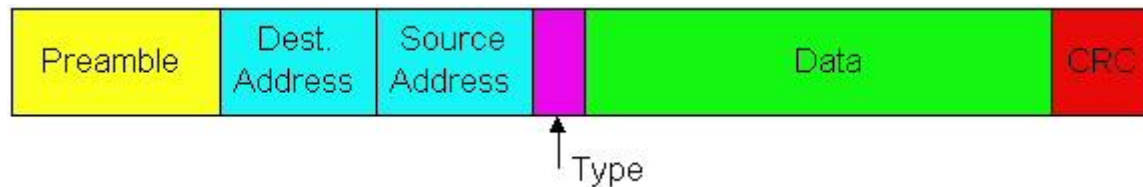


bus: coaxial cable



Ethernet Frame Structure

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

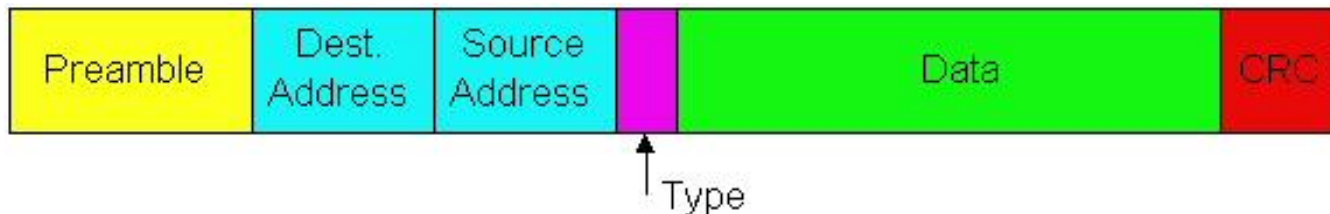


Preamble:

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

Ethernet Frame Structure (more)

- ❑ **Addresses:** 6 bytes
 - if adapter receives frame with matching destination address, or with broadcast address (eg 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:** checked at receiver, if error is 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
 - stream of datagrams passed to network layer can have gaps (missing datagrams)
 - gaps will be filled if app is using TCP
 - otherwise, app will see gaps
- ❑ Ethernet's MAC protocol: unslotted **CSMA/CD**

Ethernet CSMA/CD algorithm

1. NIC receives datagram from network layer, creates frame
2. If NIC senses channel idle, starts frame transmission
If NIC senses channel 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 **exponential backoff**: after m^{th} collision, NIC chooses K at random from $\{0, 1, 2, \dots, 2^m - 1\}$. NIC waits $K \cdot 512$ bit times, returns to Step 2

Ethernet's CSMA/CD (more)

Jam Signal: make sure all other transmitters are aware of collision; 48 bits

Bit time: 0.1 microsec for 10 Mbps Ethernet ;
for $K=1023$, wait time is about 50 msec

See/interact with Java applet on AWL Web site: highly recommended !

Exponential Backoff:

- **Goal:** adapt retransmission attempts to estimated current load
 - heavy load: random wait will be longer
- first collision: choose K from $\{0,1\}$; delay is $K \cdot 512$ bit transmission times
- after second collision: choose K from $\{0,1,2,3\}$...
- after ten collisions, choose K from $\{0,1,2,3,4,...,1023\}$

CSMA/CD efficiency

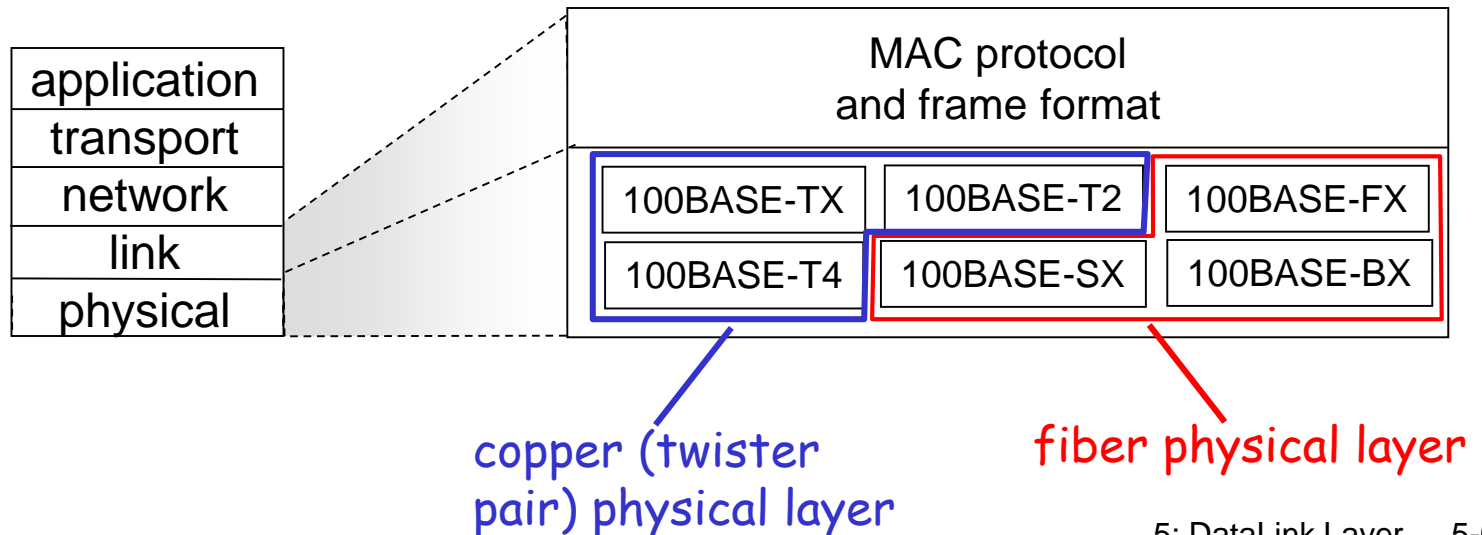
- T_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

$$\text{efficiency} = \frac{1}{1 + 5t_{\text{prop}}/t_{\text{trans}}}$$

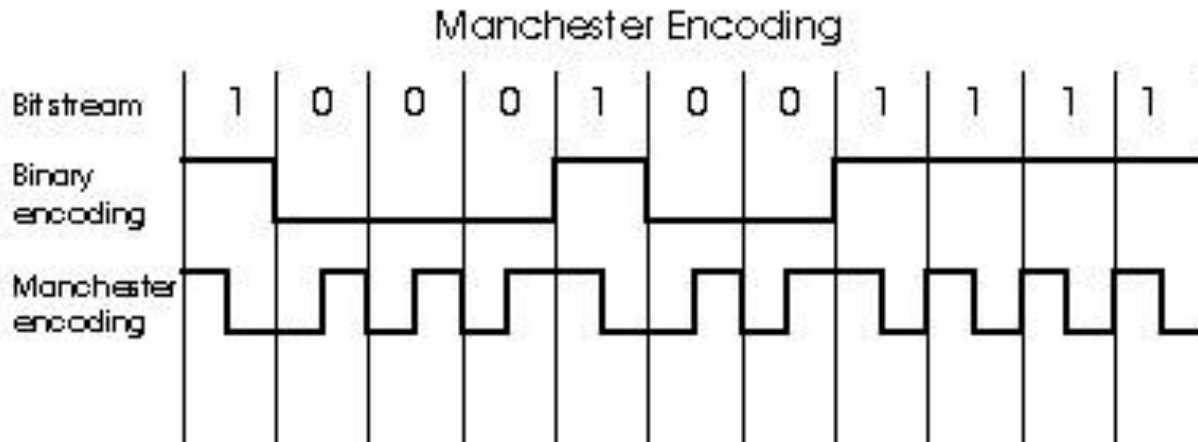
- efficiency goes to 1
 - as t_{prop} goes to 0
 - as t_{trans} goes to infinity
- better performance than ALOHA: and simple, cheap, decentralized!

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, 10G, 100G, 1000G, ...
 - different physical layer media: fiber, cable



Manchester encoding



- ❑ used in 10BaseT
- ❑ each bit has a transition
- ❑ allows clocks in sending and receiving nodes to synchronize to each other
 - no need for a centralized, global clock among nodes!
- ❑ Hey, this is physical-layer stuff!

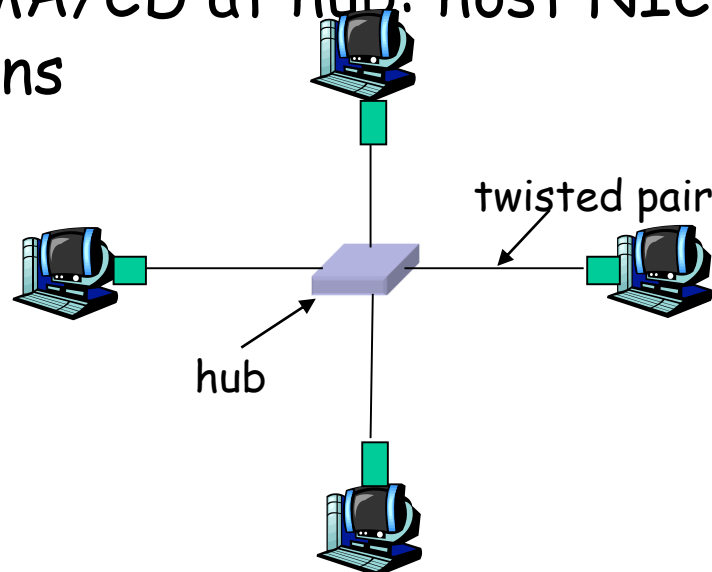
Link Layer

- ❑ 5.1 Introduction and services
- ❑ 5.2 Error detection and correction
- ❑ 5.3 Multiple access protocols
- ❑ 5.4 Link-layer Addressing
- ❑ 5.5 Ethernet
- ❑ 5.6 Link-layer switches
- ❑ 5.7 PPP
- ❑ 5.8 Link Virtualization: ATM, MPLS

Hubs

... physical-layer ("dumb") repeaters:

- bits coming in one link go out *all* other links at same rate
- all nodes connected to hub can collide with one another
- no frame buffering
- no CSMA/CD at hub: host NICs detect collisions

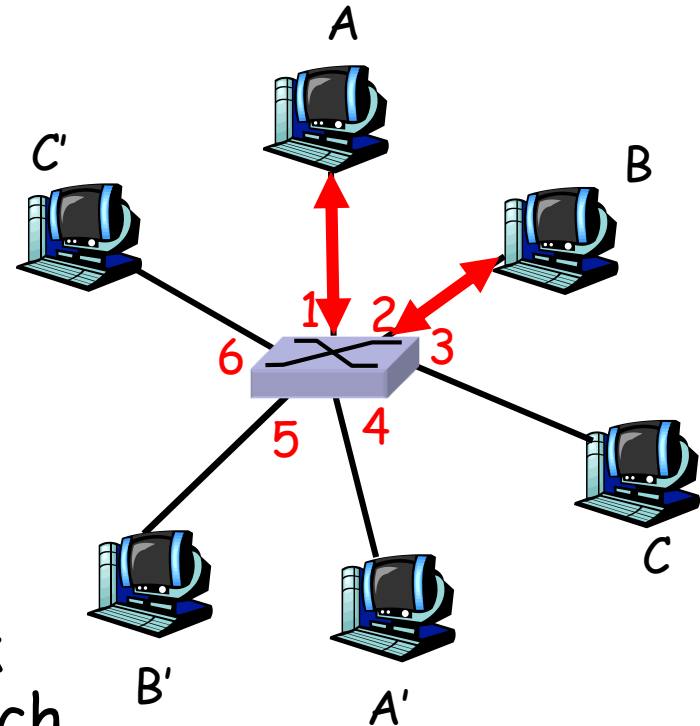


Switch

- ❑ link-layer device: smarter than hubs, take *active role*
 - store, 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: allows multiple simultaneous transmissions

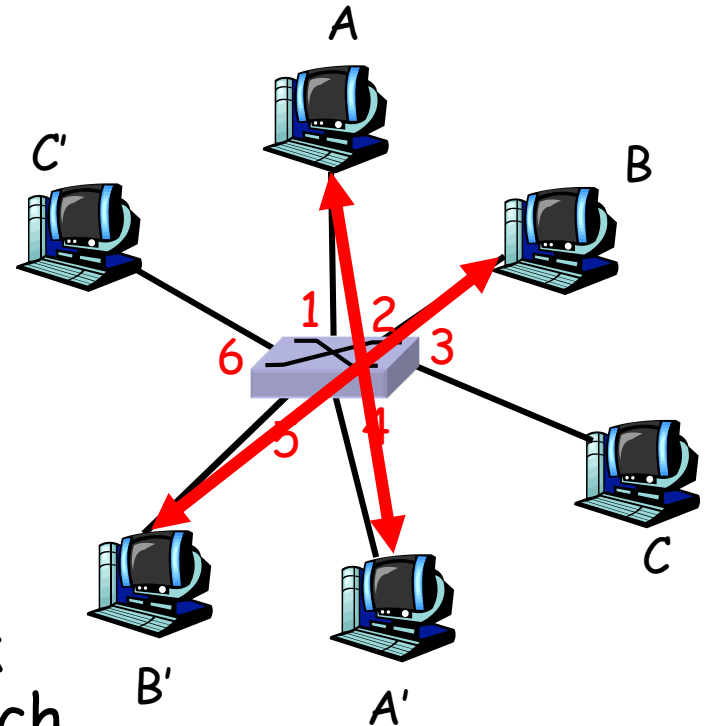
- ❑ hosts have dedicated, direct connection to switch
- ❑ switches buffer packets
- ❑ Ethernet protocol used on each incoming link:
 - each link is its own collision domain
- ❑ **switching:** point to point link between each node and switch



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

Switch: allows multiple simultaneous transmissions

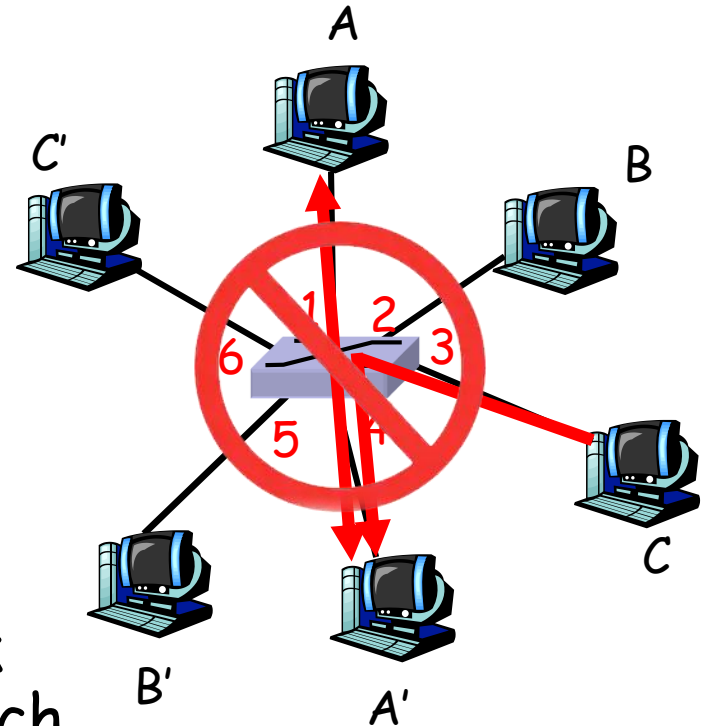
- ❑ hosts have dedicated, direct connection to switch
- ❑ switches buffer packets
- ❑ Ethernet protocol used on each incoming link:
 - each link is its own collision domain
- ❑ **switching:** point to point link between each node and switch
 - A-to-A', B-to-B' simultaneously without collisions
 - NOT possible with dumb hub



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

Switch: allows multiple simultaneous transmissions

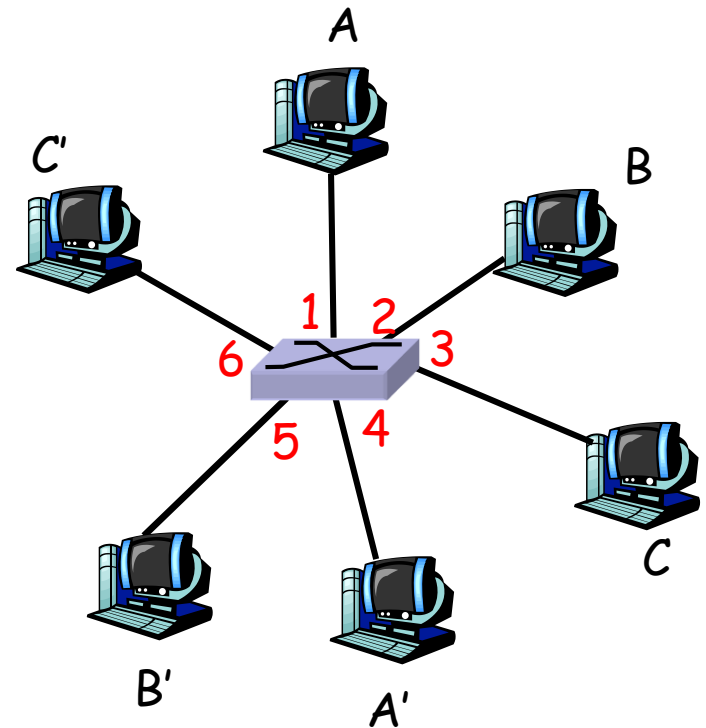
- ❑ hosts have dedicated, direct connection to switch
- ❑ switches buffer packets
- ❑ Ethernet protocol used on each incoming link:
 - each link is its own collision domain
- ❑ **switching:** point to point link between each node and switch
 - A-to-A', B-to-B' simultaneously without collisions
 - NOT possible with dumb hub
 - but A-to-A' and C-to-A' can not happen simultaneously



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

Switch Table

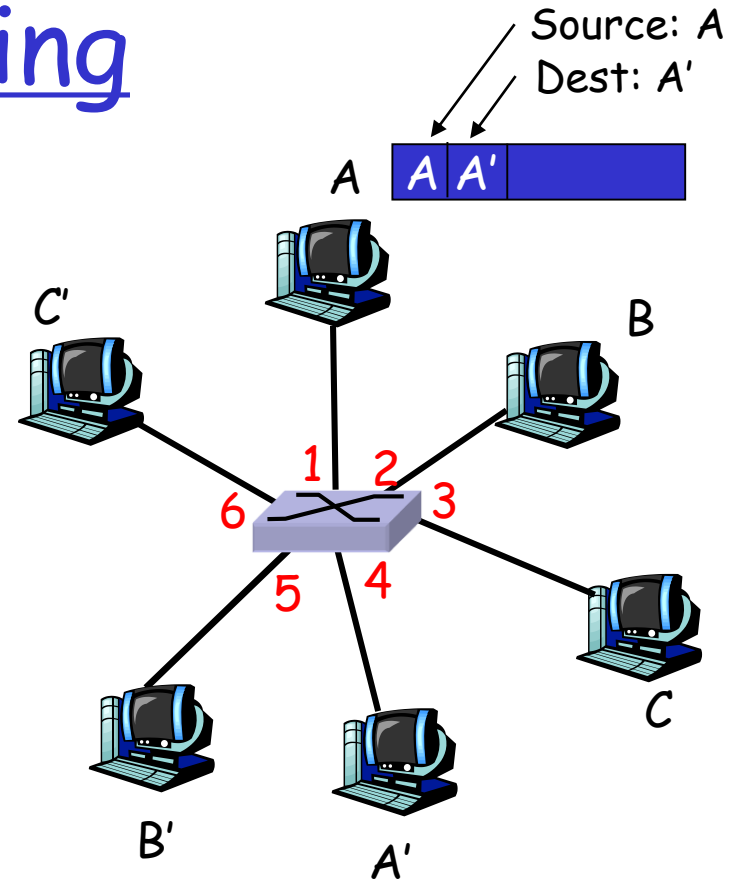
- ❑ Q: how does switch know that A' reachable via interface 4, B' reachable via interface 5?
- ❑ A: each switch has a **switch table**, each entry:
 - (MAC address of host, interface to reach host, time stamp)
- ❑ looks like a routing table!
- ❑ Q: how are entries created, maintained in switch table?
 - 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
A	1	60

*Switch table
(initially empty)*

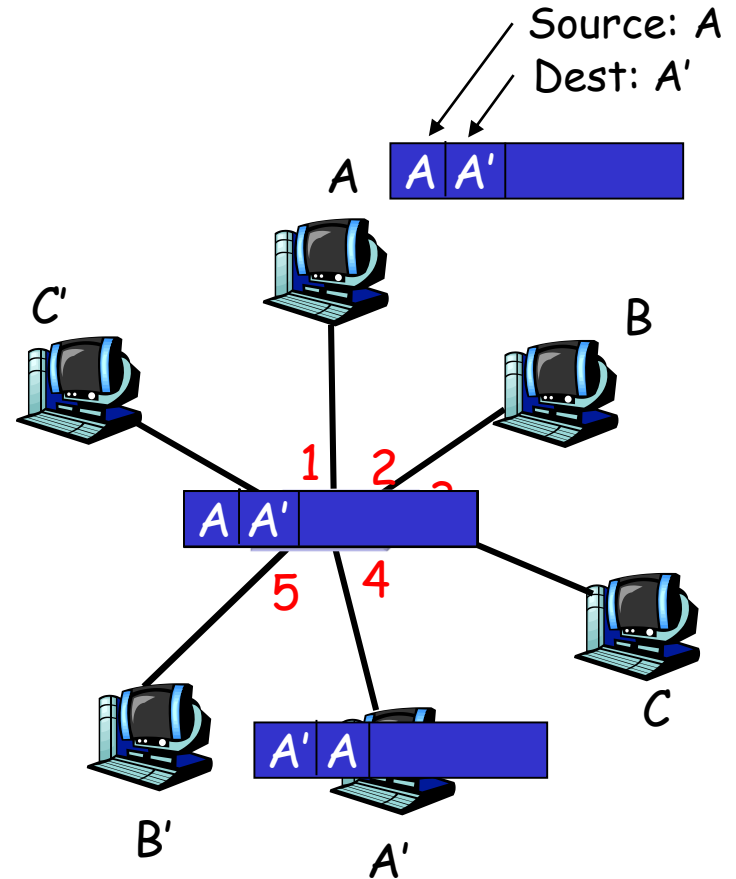
Switch: frame filtering/forwarding

When frame received:

1. record link associated with sending host
 2. index switch table using MAC dest address
 3. **if** entry found for destination
 then {
 if dest on segment from which frame arrived
 then drop the frame
 else forward the frame on interface indicated
 }
 else flood
- forward on all but the interface
on which the frame arrived*
- 

Self-learning, forwarding: example

- ❑ frame destination unknown: *flood*
- ❑ destination A location known: *selective send*

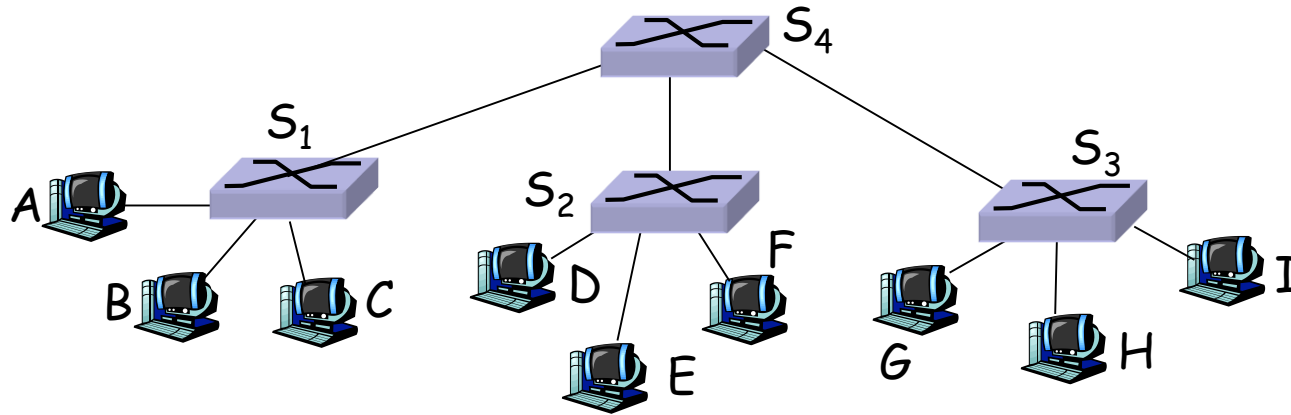


MAC addr	interface	TTL
A	1	60
A'	4	60

Switch table
(initially empty)

Interconnecting switches

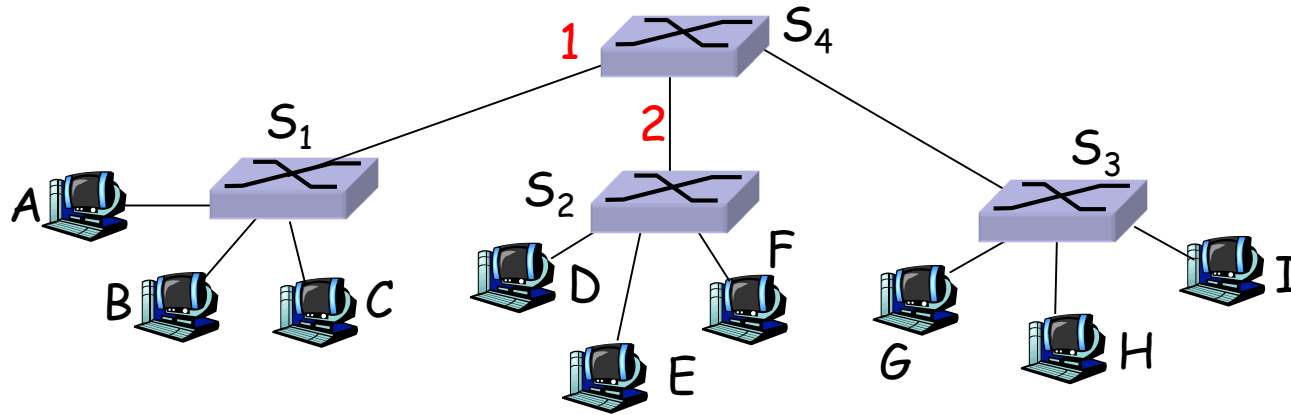
- switches can be connected together



- **Q:** sending from A to G - how does S₁ know to forward frame destined to G via S₄ and S₃?
- **A:** self learning! (works exactly the same as in single-switch case!)

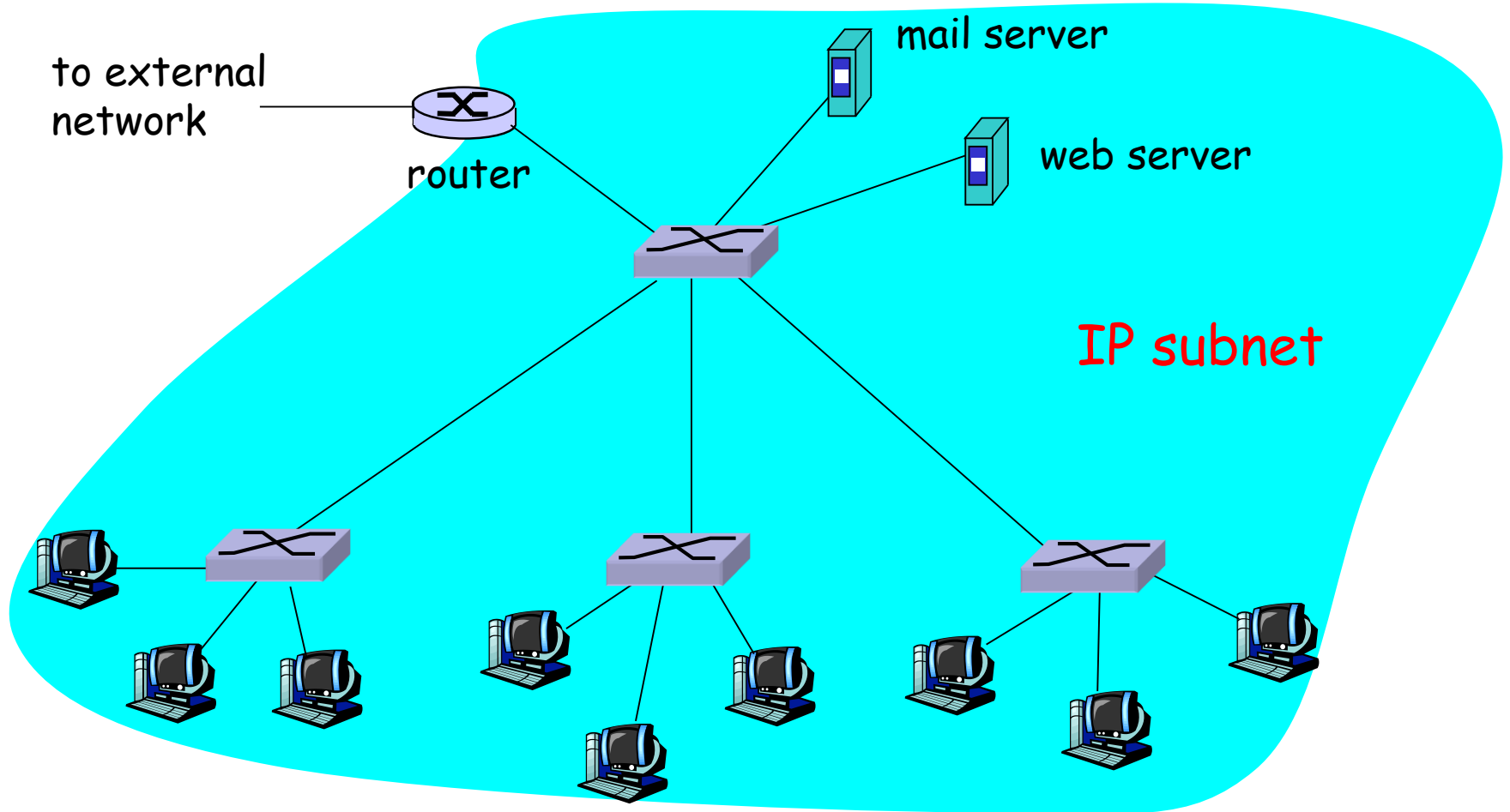
Self-learning multi-switch example

Suppose C sends frame to I, I responds to C



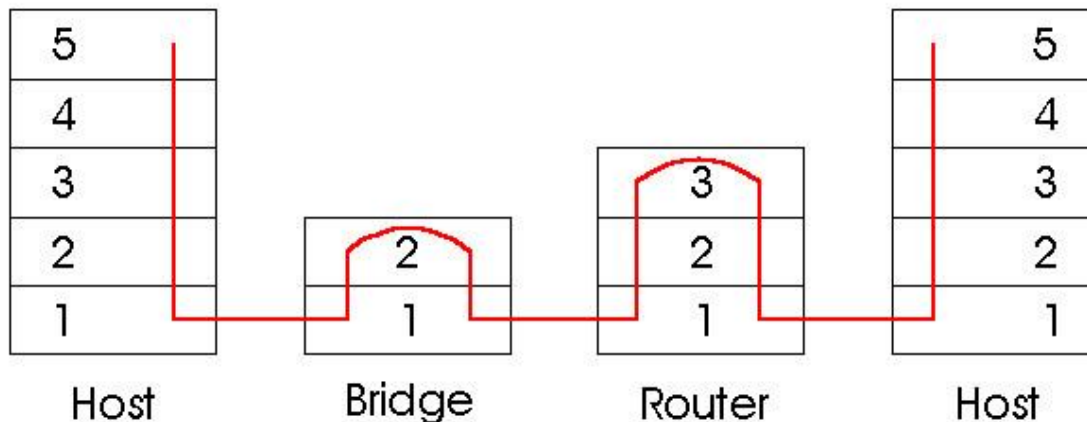
- Q: show switch tables and packet forwarding in S₁, S₂, S₃, S₄

Institutional network



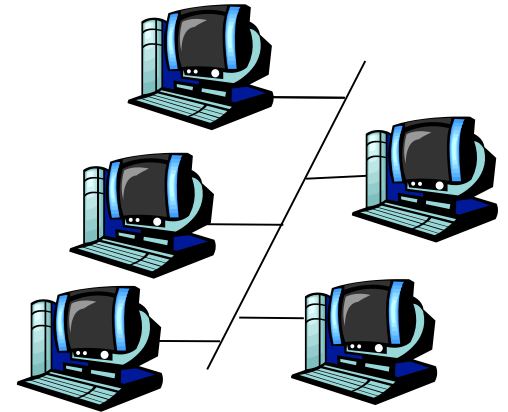
Switches vs. Routers

- ❑ both store-and-forward devices
 - routers: network layer devices (examine network layer headers)
 - switches are link layer devices
- ❑ routers maintain routing tables, implement routing algorithms
- ❑ switches maintain switch tables, implement filtering, learning algorithms

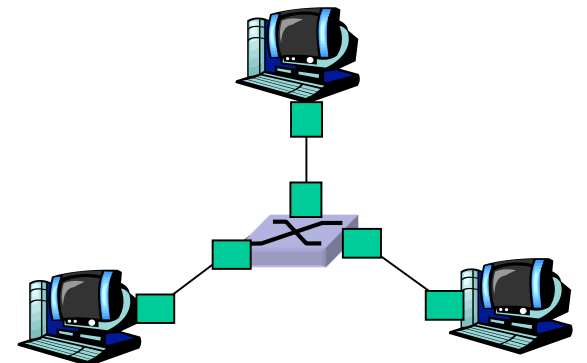


Recall CSMA/CD

- ❑ CSMA: Listen before transmit
- ❑ CD: collisions detection
- ❑ Multiple access protocol
 - shared broadcast channel
 - collision if node receives two or more signals at the same time
- ❑ Switch: point to point link between each node and switch



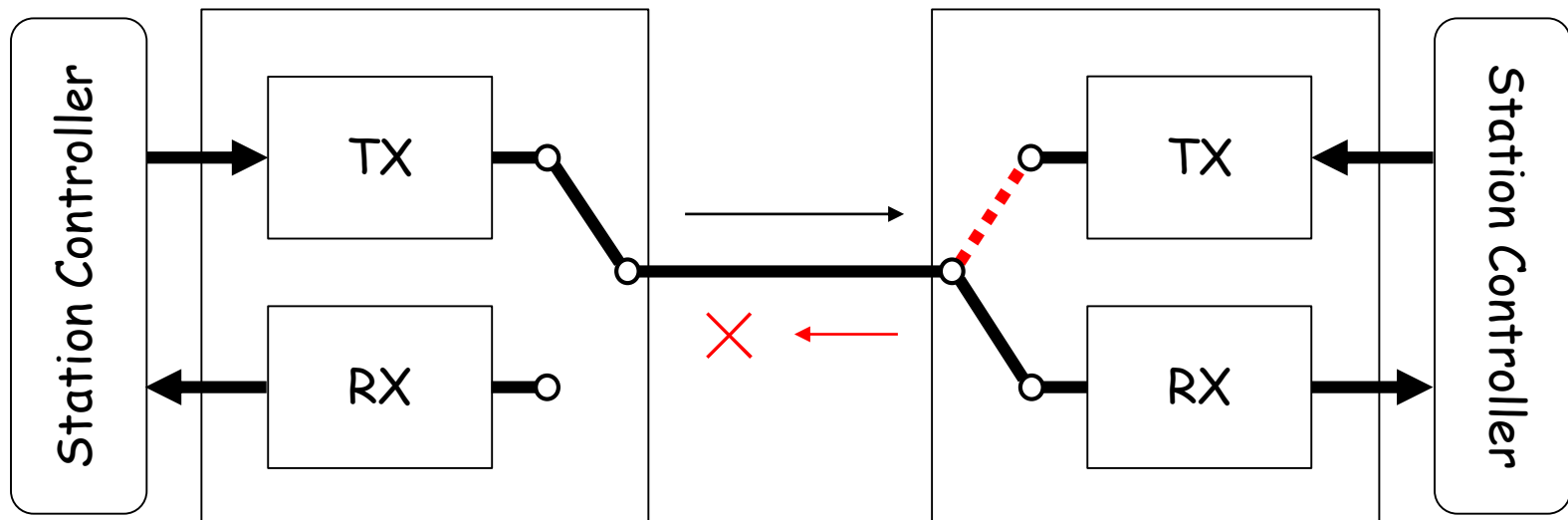
bus (broadcast channel)



star (point to point link)

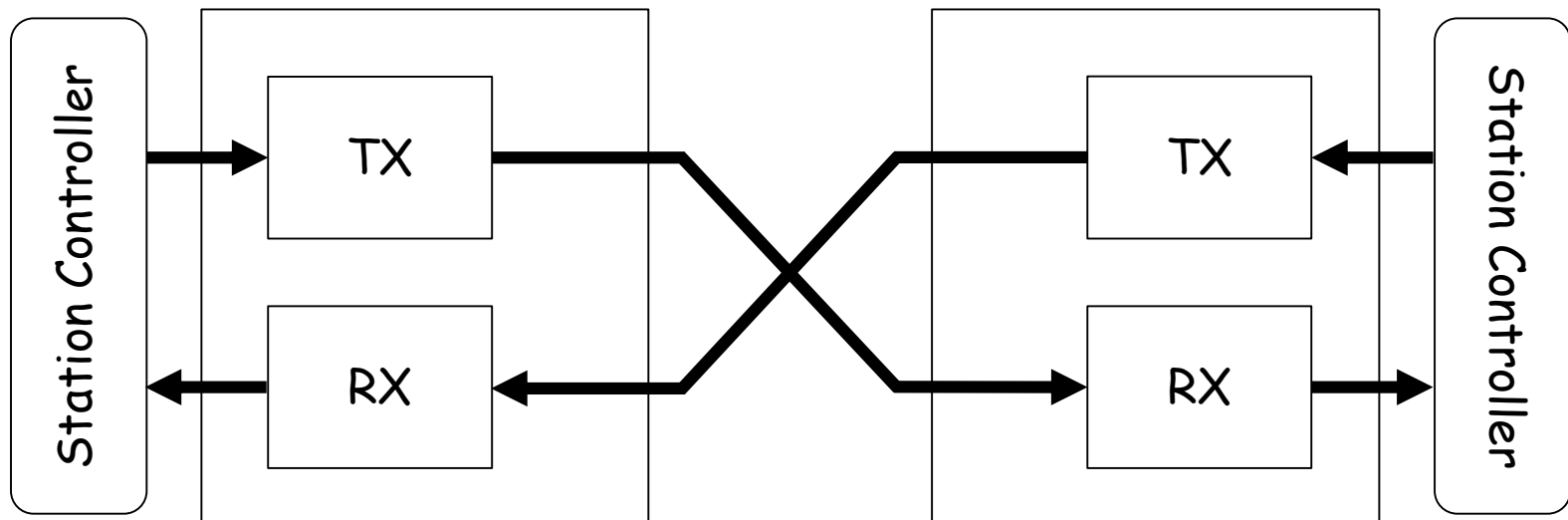
Original switch: half duplex

- ❑ Both stations can send and receive, but not at the same time
 - 10BASE5, 10BASE2, 10BASE-FB, 100BASE-T4, ...
- ❑ Collision can still happen !



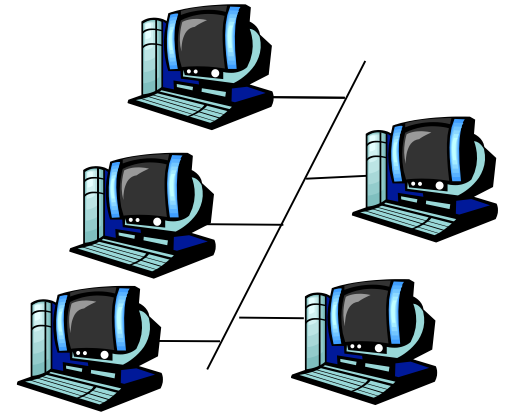
Modern switch: full duplex

- ❑ Both stations can transmit & receive simultaneously
 - 100BASE-TX, 100BASE-FX, 1000BASE-T, ...
- ❑ No collision -> **No need for CSMA/CD !**

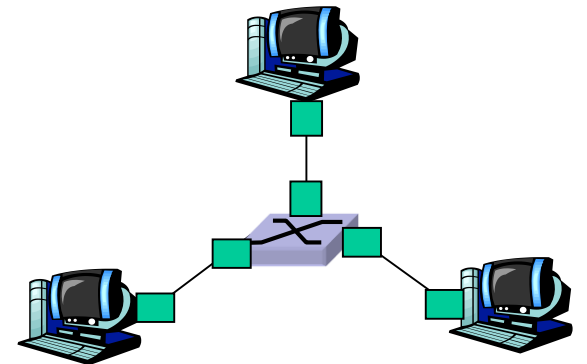


回顾CSMA

- ❑ Bus型以太网需要全局**CSMA**
- ❑ 早期Star型以太网只在局部（直接相连网卡之间）需要**CSMA**
- ❑ 现代Star型以太网（全双工交换机）在局部（直接相连网卡之间）都不需要**CSMA**了
- ❑ ...
- ❑ 讲了个寂寞
- ❑ **CSMA**是多余的？



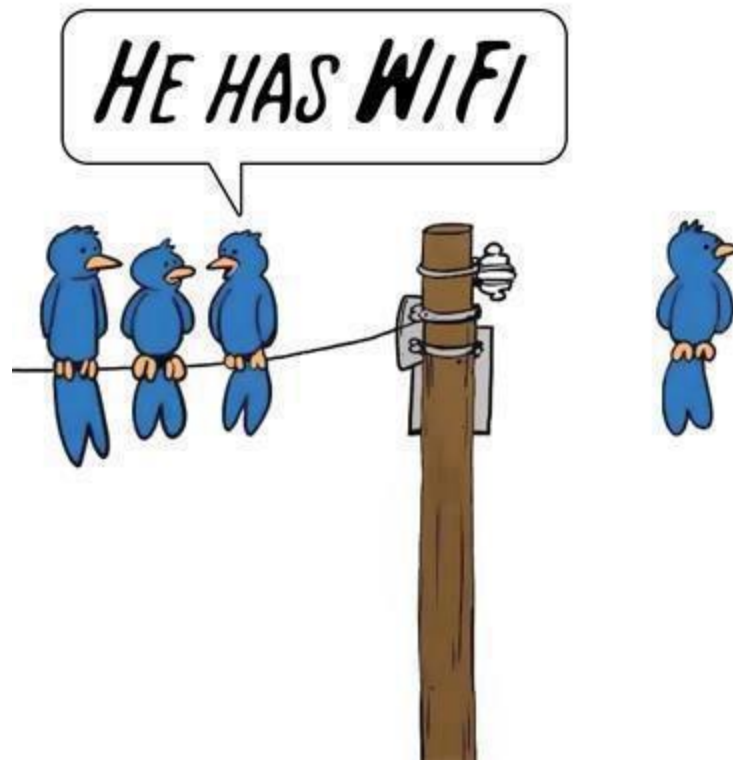
bus (broadcast channel)



star (point to point link)

链路层协议的发展趋势

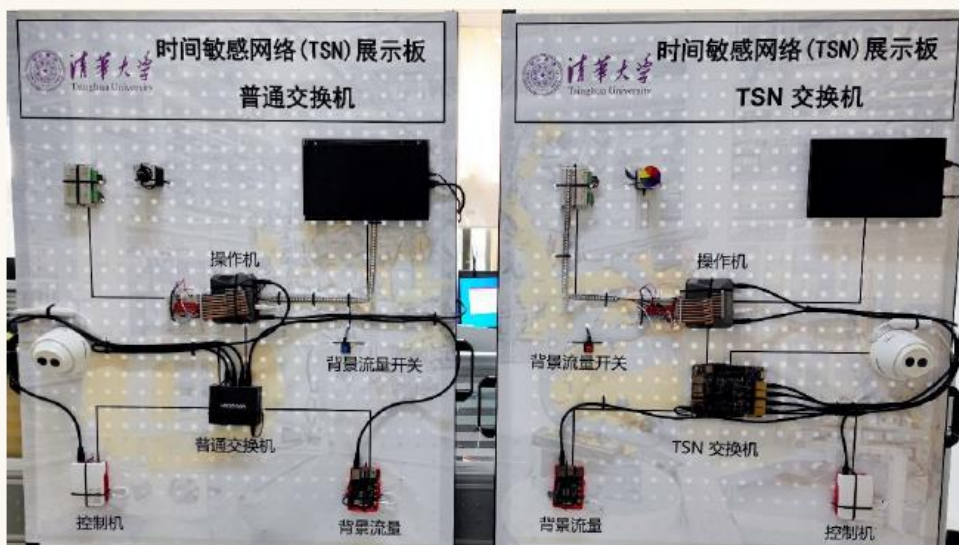
□ 无线 Wireless



链路层协议的发展趋势

□ 实时性、确定性

时间敏感网络 (Time Sensitive Networking, TSN) 是工业互联网的核心网络技术，利用时间同步、流量整形、管理控制等机制，保证网络的**实时可靠与灵活高效**，实现**OT网络与IT网络的融合**。



确定性转发

超低时延传输

网络控制管理

SMT流量调度算法

开发实验平台

链路层协议的发展趋势

□ 实时性、确定性

知更Ziggo系列TSN交换机：

- 全面支持802.1AS、Qav、Qbv、Qcc
- 支持IT流量与OT流量的共网传输
- 实现关键数据的确定性低时延
- 同时支持时间触发和事件触发关键流量
- 基于SDN、OPC UA的控制管理接口

纳秒

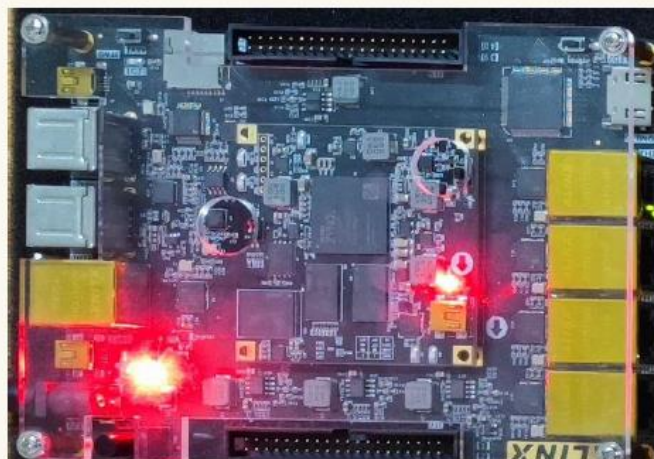
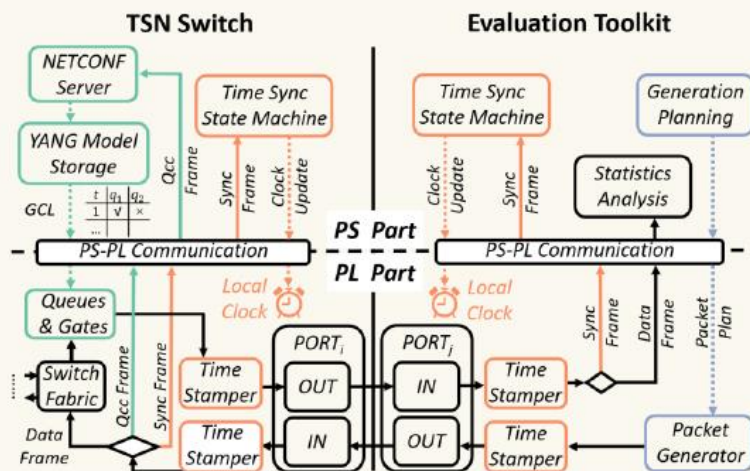
时间同步精度达到纳秒级

微秒

每跳时延抖动小于一微秒

万兆

带宽最大支持万兆以太网



时间敏感网络 TSN

Time Sensitive Network TSN



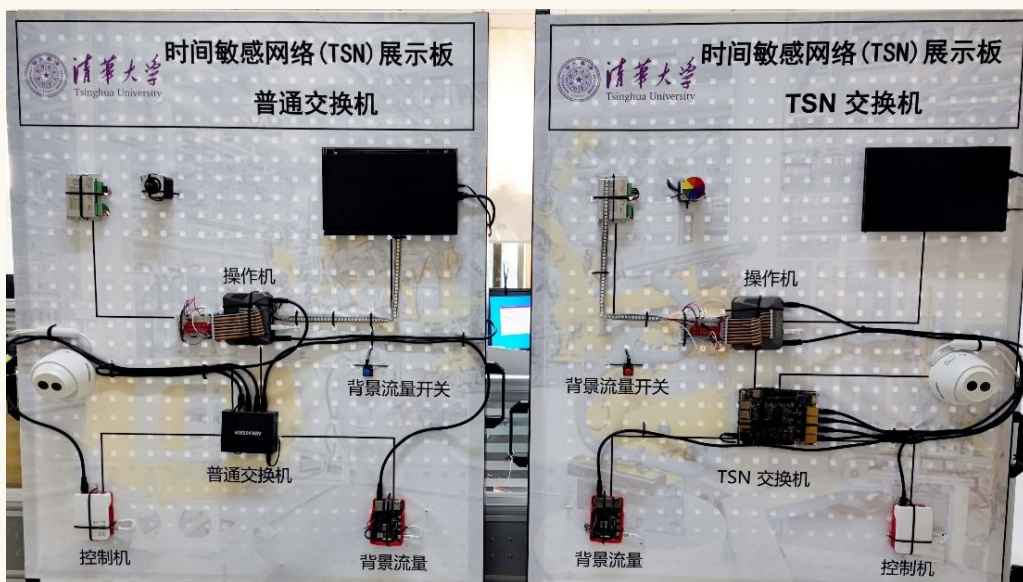
时间敏感网络 TSN

Time Sensitive Network TSN

时间敏感网络TSN

TSN Industrial Network Software and Switch

时间敏感网络 (Time Sensitive Networking, TSN) 是工业互联网的核心网络技术，利用时间同步、流量整形、管理控制等机制，保证网络的**实时可靠与灵活高效**，实现**OT网络与IT网络的融合**。



确定性转发

超低时延传输

网络控制管理

SMT流量调度算法

开发实验平台

TSN交换机：工业互联网核心设备

TSN Industrial Network Software and Switch

知更Ziggo系列TSN交换机：

- 全面支持**802.1AS**、**Qav**、**Qbv**、**Qcc**
- 支持**IT流量**与**OT流量**的共网传输
- 实现关键数据的**确定性低时延**
- 同时支持**时间触发**和**事件触发**关键流量
- 基于**SDN**、**OPC UA**的控制管理接口

纳秒

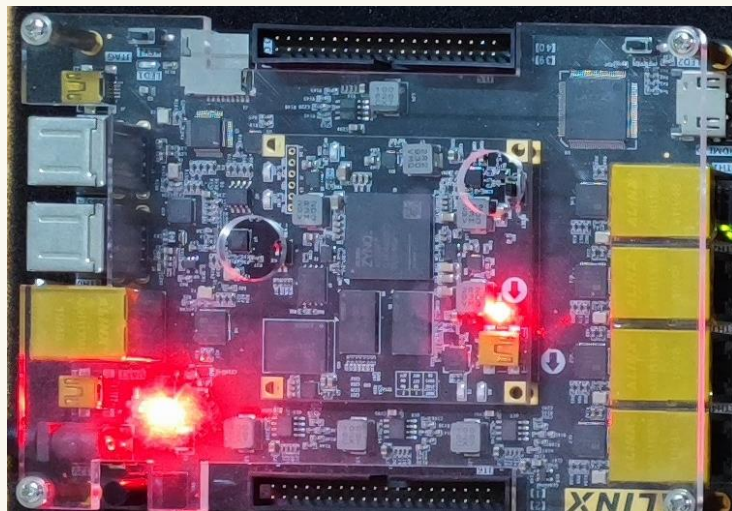
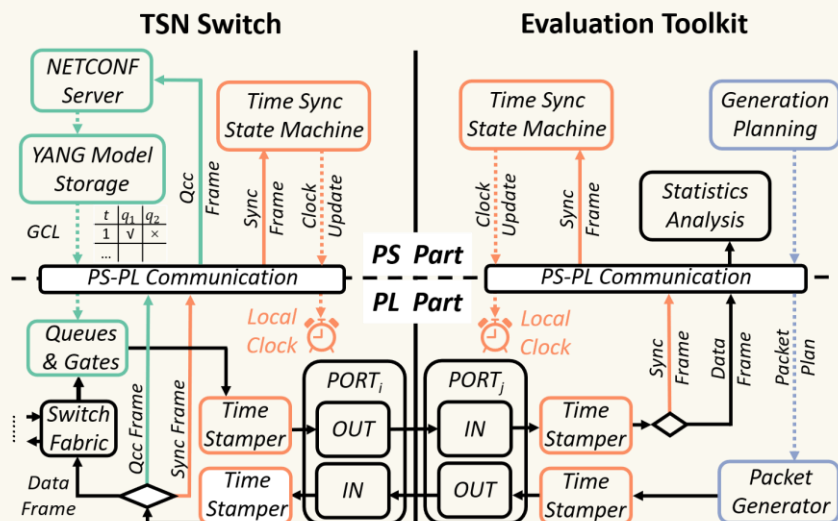
时间同步精度达到**纳秒级**

微秒

每跳时延抖动小于**一微秒**

万兆

带宽最大支持**万兆以太网**



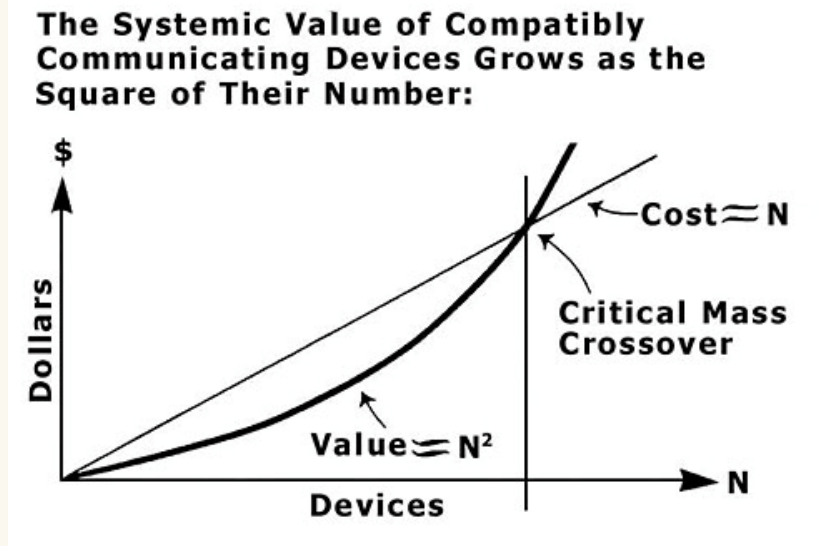
TSN交换机：工业互联网核心设备



镇界三定律之梅特卡夫定律

Metcalfe's law

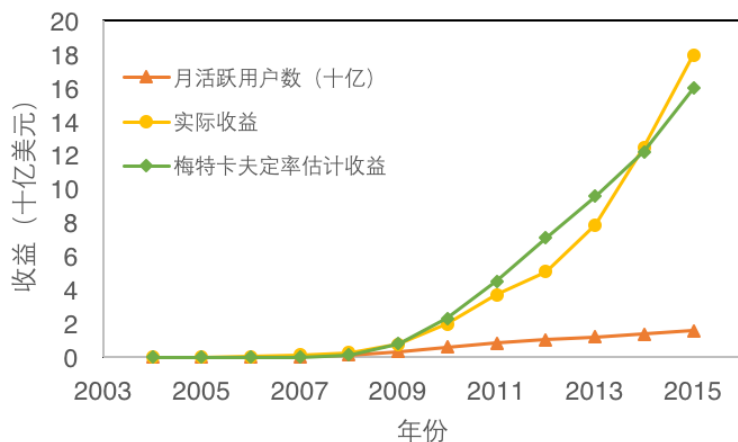
- **梅特卡夫定律** (Metcalfe's Law) 指的是，网络的价值与网络使用者数量的平方成正比。即网络的价值 $V = K \times N^2$ (K为价值系数，N为用户数量)
- 该定律被很多人质疑，因为它认为所有网络节点都是对等的，而忽略了不同节点和连接之间的差异性。



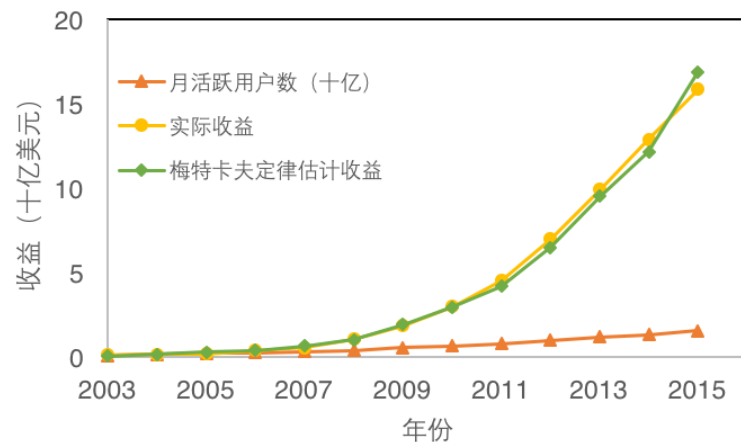
梅特卡夫定律得以验证

Metcalfe's law

Facebook数据验证梅特卡夫定律



腾讯数据验证梅特卡夫定律



2014年梅特卡夫发表一篇文章，用Facebook的数据对梅特卡夫定律做验证，发现**Facebook的收入和其用户数的平方成正比**。随后，中国有学者采用相同方法验证了腾讯收入和其用户数的平方成正比。

梅特卡夫定律提出者

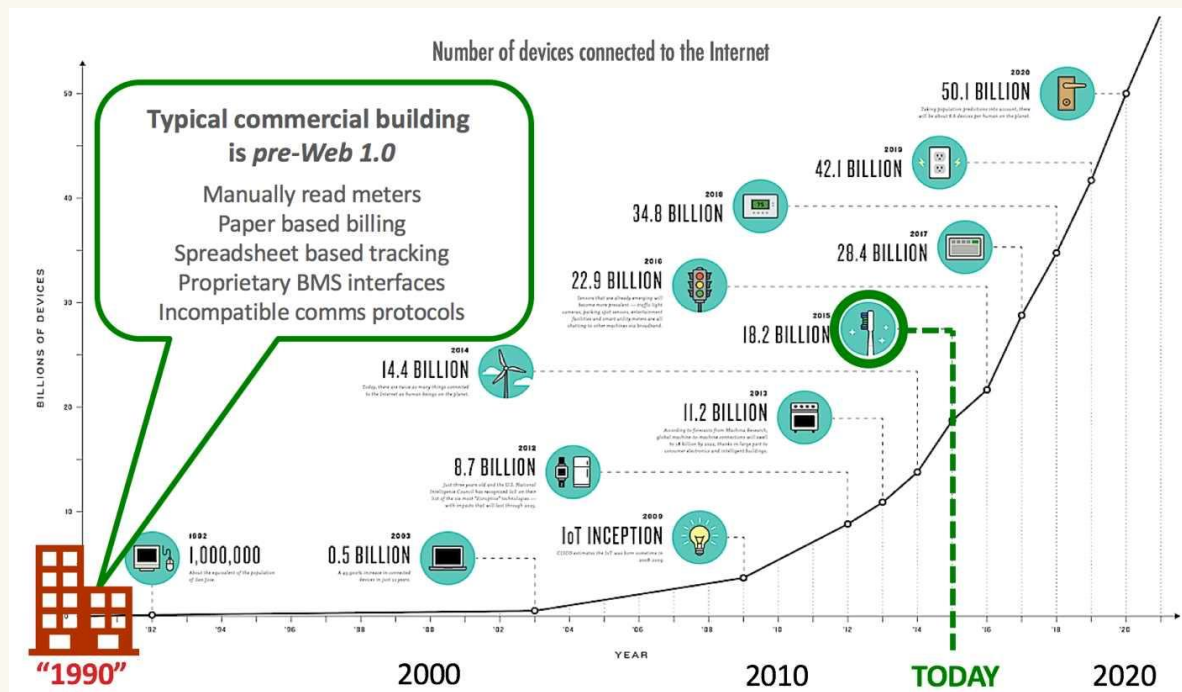
The Proposer of Metcalfe's law

- 提出者**罗伯特·梅特卡夫**（Robert Metcalfe），在1980年提出、1993年被正式规范定义的。
- **以太网发明人**，其创立的3Com公司为IBM生产了世界上第一块网卡。
- 获得**美国国家科技奖**（National Medal of Technology），该奖的含金量在美国几乎相当于诺贝尔奖。



物联网——未来世界的现实

Network of Things——Reality of the Future World



- 据思科公司估计，2015年全球已经有**超过150亿产品**接入互联网，到2020年，这个数字至少达到**300亿**。
- 现在我们**身处一个前所未有的时代**：无处不在的**设备**，无时无刻的**网络**，产生着无可估量的**数据**，也蕴含了无可比拟的**价值**。

Chapter 5: Summary

- ❑ principles behind data link layer services:
 - error detection, correction
 - sharing a broadcast channel: multiple access
 - link layer addressing
- ❑ instantiation and implementation of various link layer technologies
 - Ethernet
 - switched LANS
 - PPP
 - virtualized networks as a link layer: ATM, MPLS

Chapter 5: let's take a breath

- ❑ journey down protocol stack *complete*
(except PHY)
- ❑ solid understanding of networking principles,
practice
- ❑ could stop here but *lots* of interesting
topics!
 - wireless
 - multimedia
 - security
 - network management

思考题

- 交换机和路由器的区别？
 - 网络层次
 - 网络拓扑
 - 处理速度
 - 网络隔离
- 假设，你要为所在的机构（例如清华大学）搭建网络，该选择用交换机还是路由器连接所有主机？

时间敏感网络的设计与实现

- ❑ 欢迎有意愿参与科研工作的同学，加入并参与对时间敏感网络（**TSN**）的研究
- ❑ 深刻理解计算机网络原理
- ❑ 高性能网络交换
- ❑ 先进的流量调度算法
- ❑ **FPGA**开发平台
- ❑ 高水平论文