Link Layer and LANs: Multiple Access and MAC

Nov 21, 2023

Min Suk Kang
Associate Professor
School of Computing/Graduate School of Information Security

NetS&P
Research Lab @ KAIST

KAIST

CSMA (carrier sense multiple access)

simple CSMA: listen before transmit:

- if channel sensed idle: transmit entire frame
- if channel sensed busy: defer transmission

- human analogy: don't interrupt others!

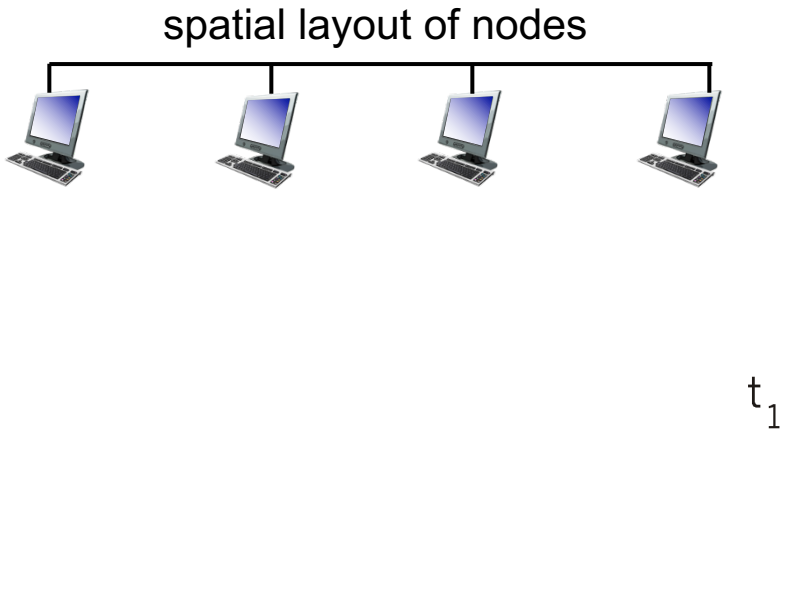
CSMA/CD: CSMA with *collision detection*

- collisions *detected* within short time
- colliding transmissions aborted, reducing channel wastage
- collision detection easy in wired, difficult with wireless

- human analogy: the polite conversationalist

CSMA: collisions

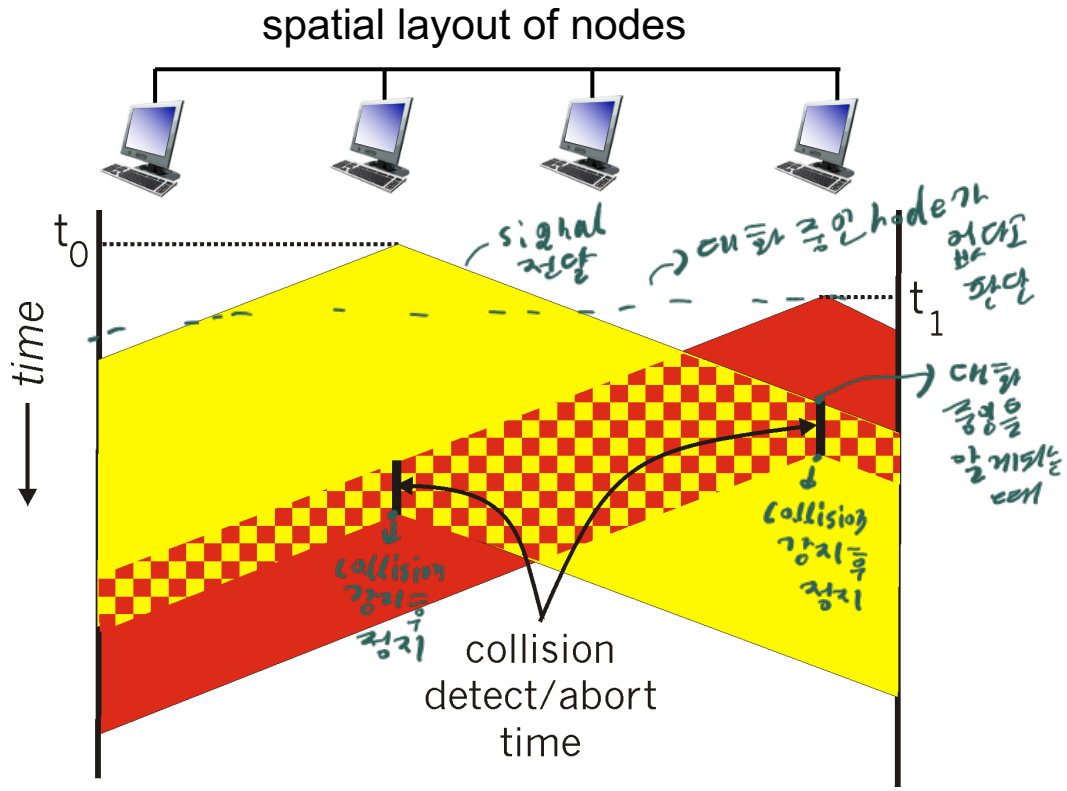
- collisions *can* still occur with carrier sensing:
  - propagation delay means two nodes may not hear each other's just-started transmission
- collision: entire packet transmission time wasted
  - distance & propagation delay play role in in determining collision probability



CSMA/CD:

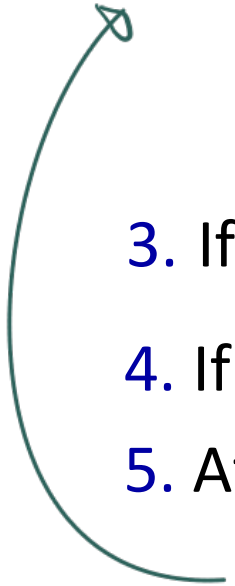
- CSMA/CS reduces the amount of time wasted in collisions
  - transmission aborted on collision detection

→ collision을 감지하고 바로 drop하여 낭비되는 시간 최소화



Ethernet CSMA/CD algorithm

1. NIC receives datagram from network layer, creates frame
2. If NIC senses channel:
   if idle: start frame transmission.
   if busy: wait until channel idle, then transmit
3. If NIC transmits entire frame without collision, NIC is done with frame !
4. If NIC detects another transmission while sending: abort, send jam signal
5. After aborting, NIC enters *binary (exponential) backoff:*
   - after *m*th collision, NIC chooses *K* at random from *{0,1,2, …, 2^m-1}*. NIC waits *K*·512 bit times, returns to Step 2
   - more collisions: longer backoff interval



ALOHA의 경우, 고정 확률 p를 사용하므로 사용자가 늘어나면 network tweaking (장애) 발생

CSMA/CD efficiency

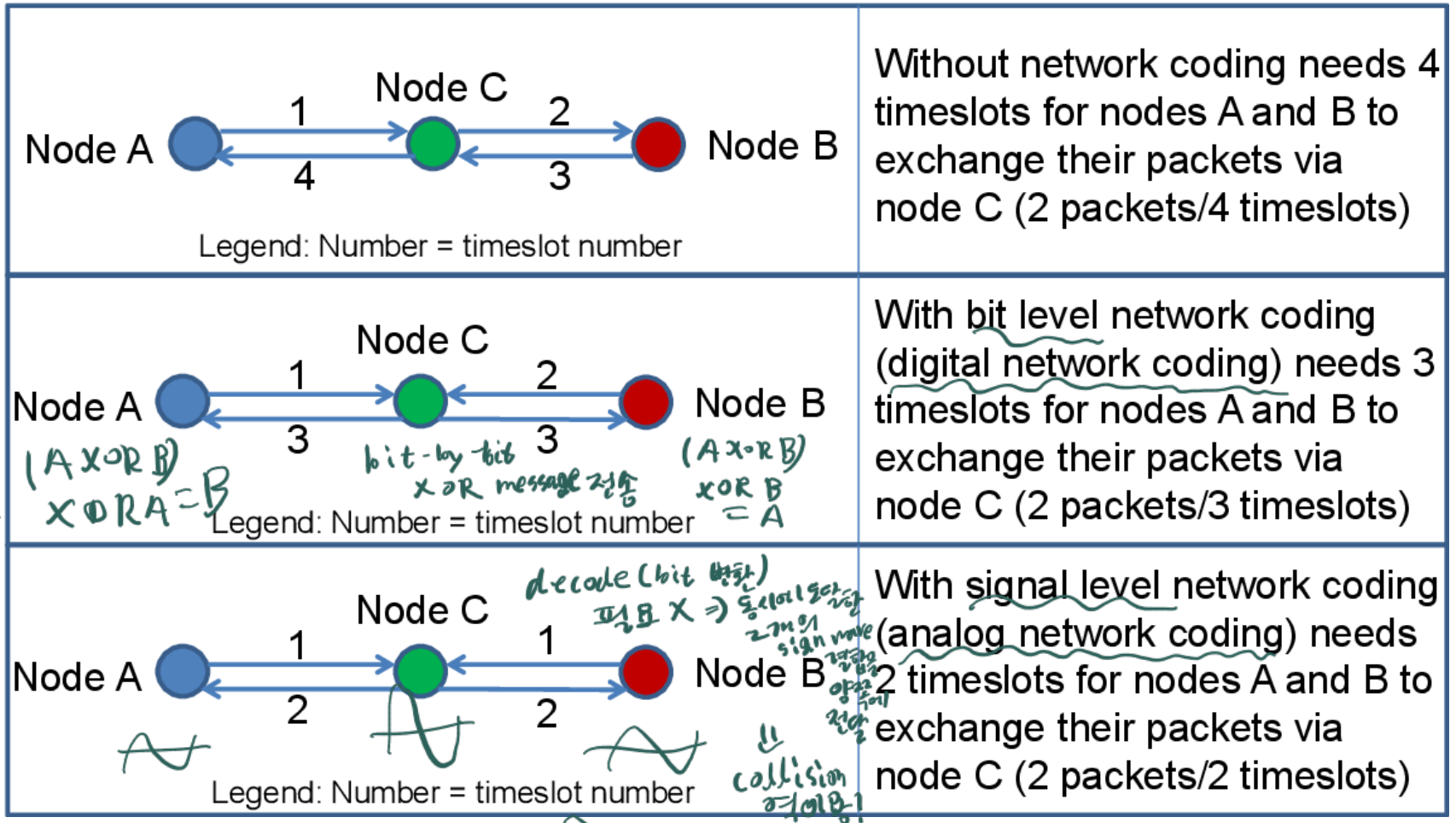
- T_{prop} = max prop delay between 2 nodes in LAN
- t_{trans} = time to transmit max-size frame

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

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

Is Collision Always Bad?

code division?



(Susanto 2015)

“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!

↳ load가 적을 경우, share channel은 낭비

random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

↳ random access는 load가 높을 경우 많은 collision으로 효율↓

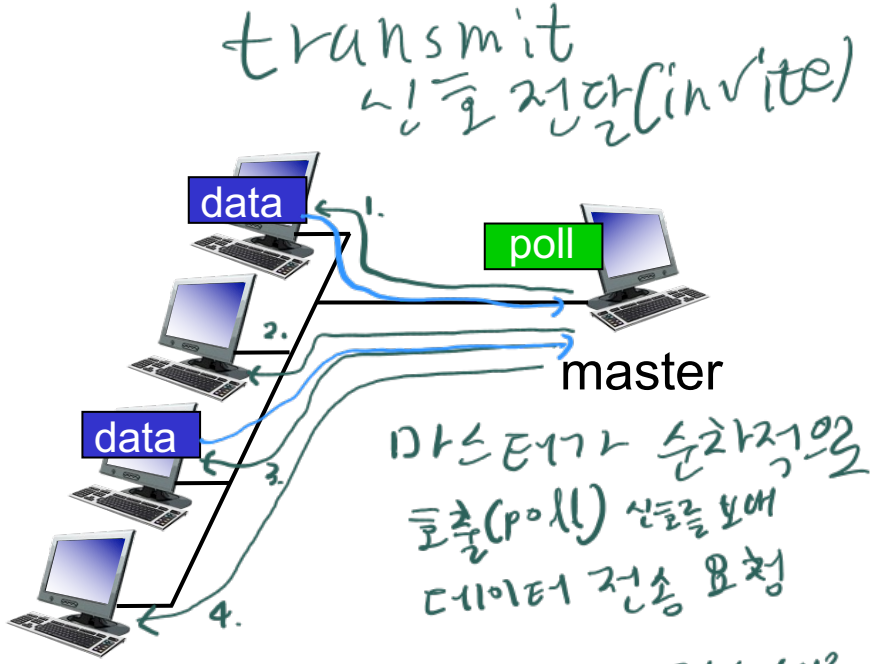
“taking turns” protocols

- look for best of both worlds!

“Taking turns” MAC protocols

polling:

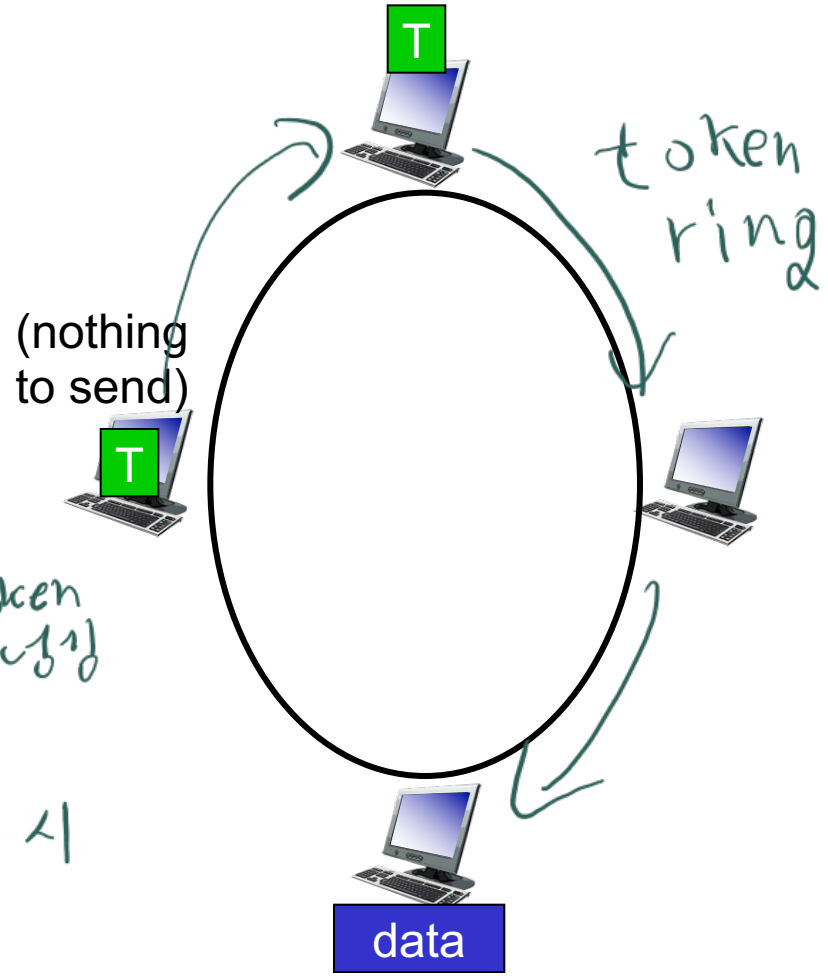
- master node “invites” other nodes to transmit in turn
- typically used with “dumb” devices → 지능이 낮음 X
- concerns:
  - polling overhead → 데이터가 없는 slave에도 poll 요청 ⇒ 자원 낭비
  - latency → 전송 순서 대기 필요
  - single point of failure (master)
    → 마스터에 의존 ⇒ master error 시 전체 error



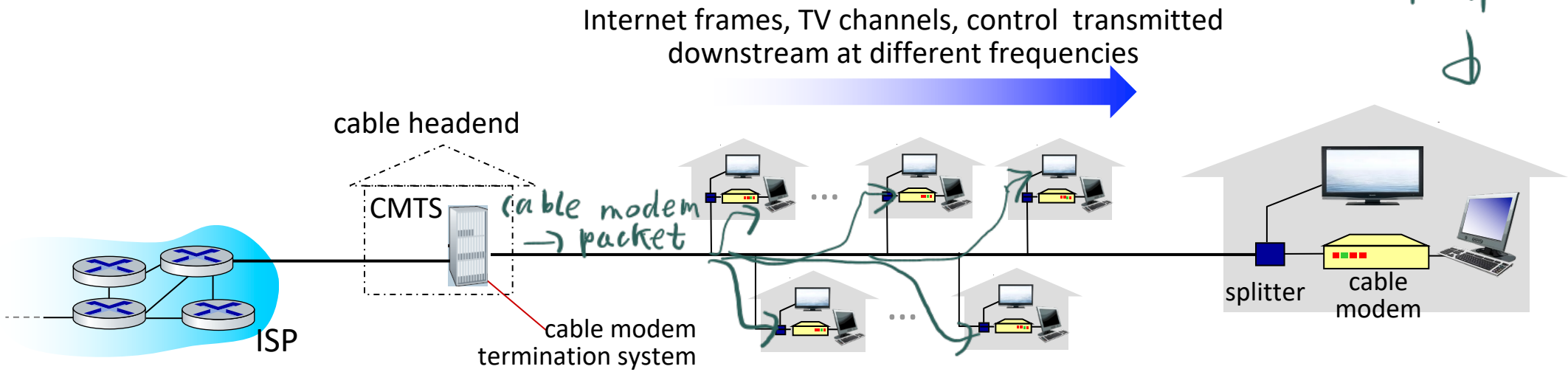
“Taking turns” MAC protocols

token passing:

- control *token* passed from one node to next sequentially.
- token message
- concerns:
  - token overhead → data 없이도 token 생성
  - latency → 기다림
  - single point of failure (token) → 중간에 error 발생 시 순환 끊김

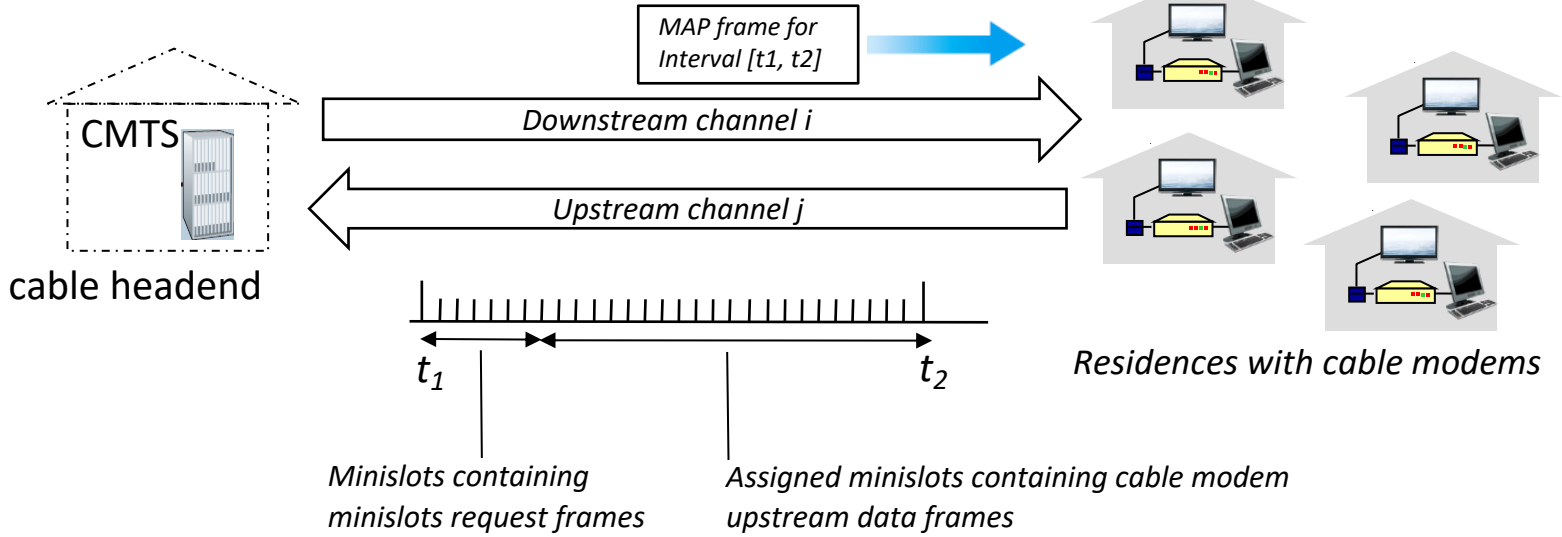


Cable access network: FDM, TDM *and* random access!



- multiple downstream (broadcast) FDM channels: up to 1.6 Gbps/channel
  - single CMTS transmits into channels
- multiple upstream channels (up to 1 Gbps/channel)
  - multiple access: all users contend (random access) for certain upstream channel time slots; others assigned TDM

Cable access network:

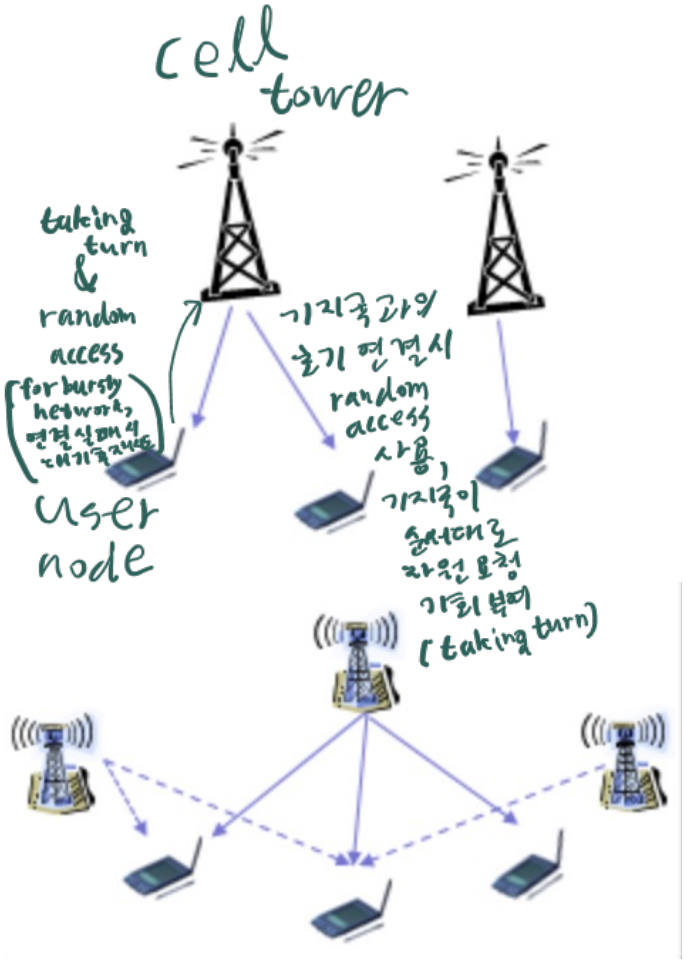


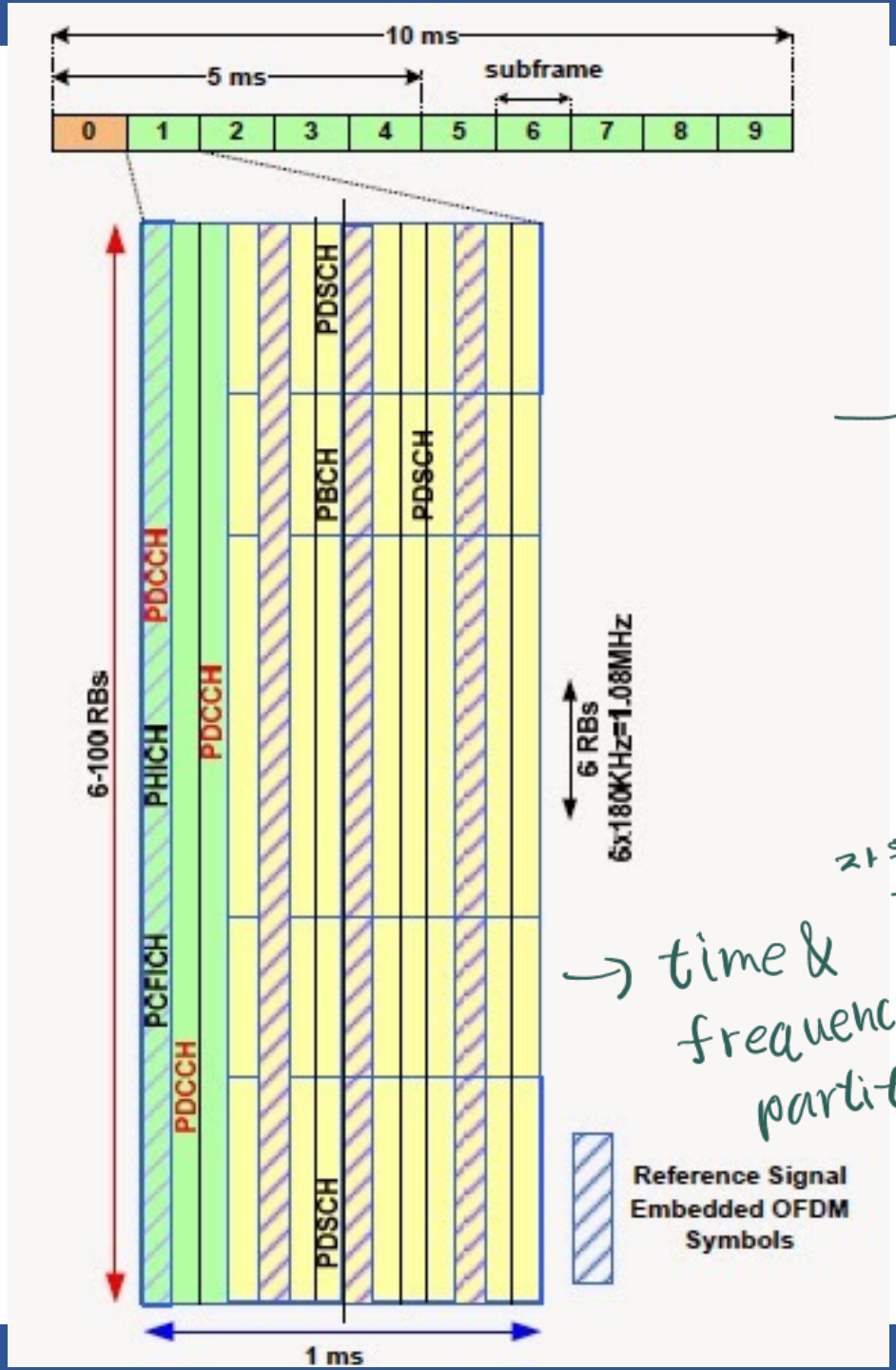
DOCSIS: data over cable service interface specificaiton

- FDM over upstream, downstream frequency channels
- TDM upstream: some slots assigned, some have contention
  - downstream MAP frame: assigns upstream slots
  - request for upstream slots (and data) transmitted random access (binary backoff) in selected slots

3개 동시 사용

Cellular network:



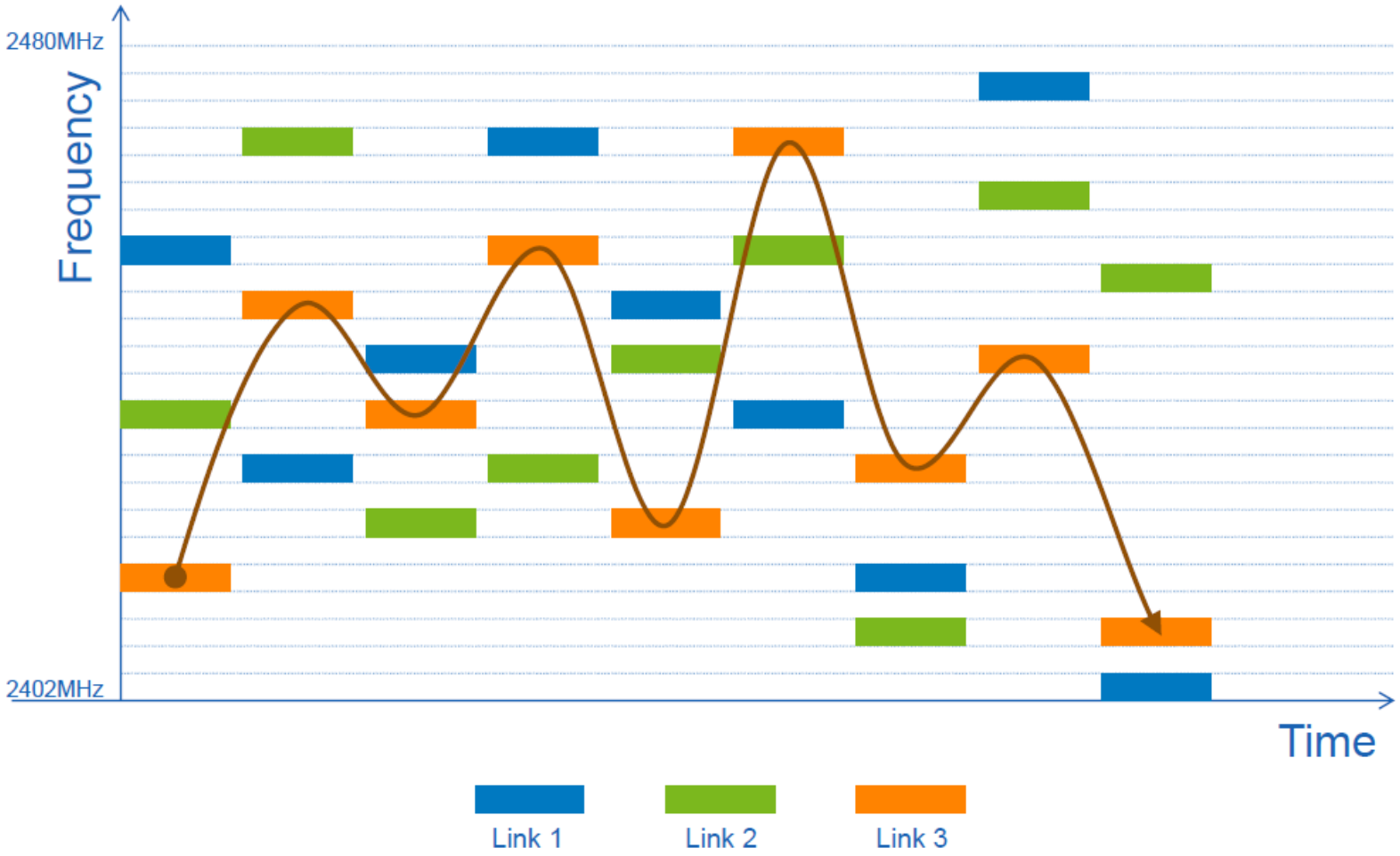


cell tower's message (1,000/s)

→ time & frequency partition

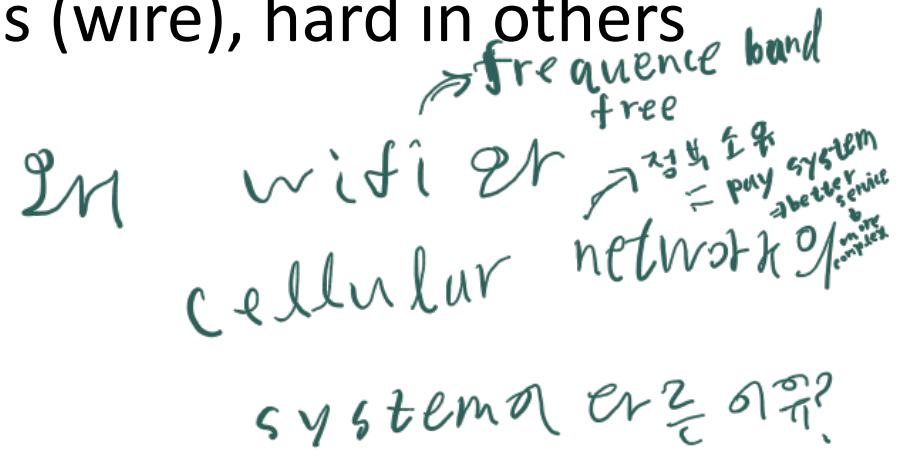
자원을 분할하여 각 cell 에 할당 ⇒ resource partition & which resources 또는 얼만큼 resource 할당 필요 (channel partitioning)

Bluetooth frequency hopping



Summary of MAC protocols

- channel partitioning, by time, frequency or code
  - Time Division, Frequency Division
- random access (dynamic),
  - ALOHA, S-ALOHA, CSMA, CSMA/CD
  - carrier sensing: easy in some technologies (wire), hard in others (wireless)
  - CSMA/CD used in Ethernet
  - CSMA/CA used in 802.11
- taking turns
  - polling from central site, token passing
  - Bluetooth, FDDI, token ring

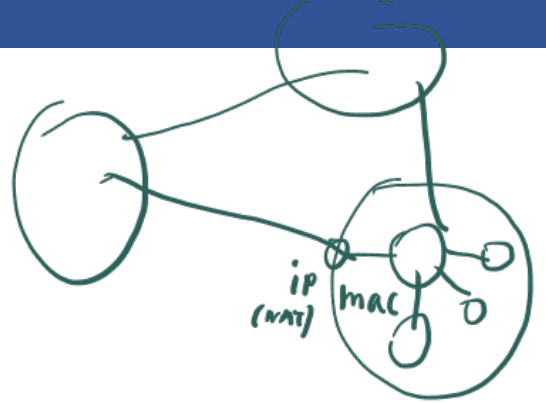


Link layer, LANs: roadmap

- introduction
- error detection, correction
- multiple access protocols
- LANs
  - addressing, ARP
  - Ethernet
  - switches
  - VLANs → skip
- link virtualization: MPLS
- data center networking

- a day in the life of a web request

MAC addresses



- 32-bit IP address:
  - *network-layer* address for interface
  - used for layer 3 (network layer) forwarding
  - e.g.: 128.119.40.136

link layer

- MAC (or LAN or physical or Ethernet) address:
  - function: used "locally" to get frame from one interface to another physically-connected interface (same subnet, in IP-addressing sense)
  - 48-bit MAC address (for most LANs) burned in NIC ROM, also sometimes software settable
  - e.g.: 1A-2F-BB-76-09-AD

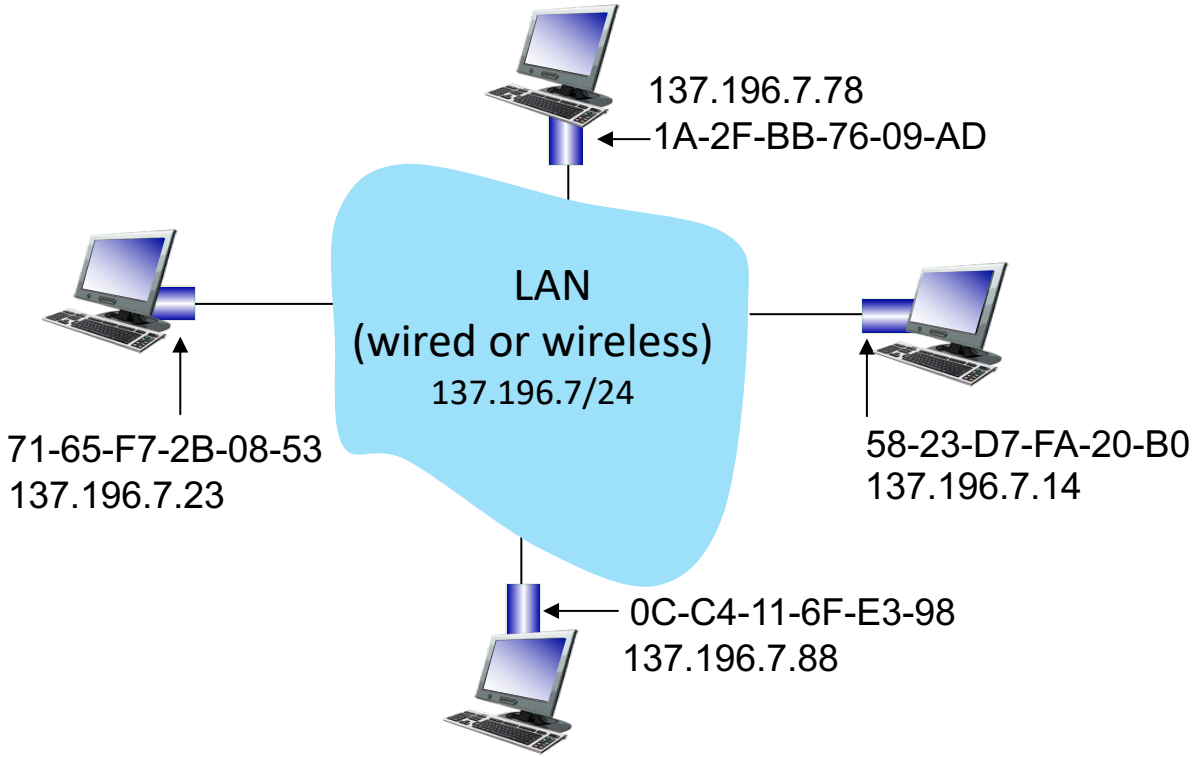
hexadecimal (base 16) notation (each "numeral" represents 4 bits)

고정값 할당하면 fixed random

MAC addresses

each interface on LAN

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

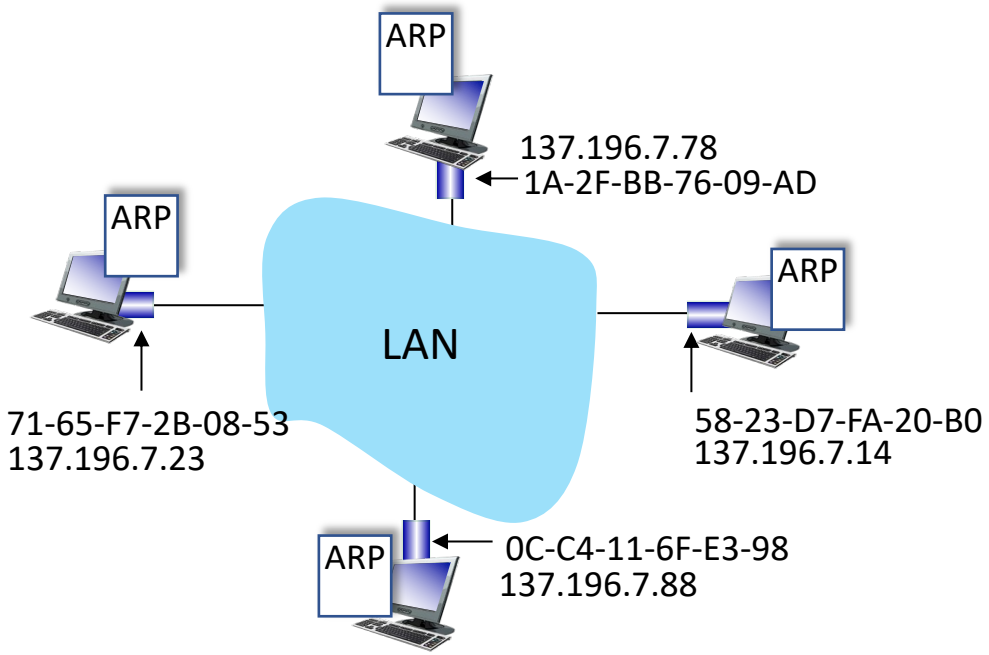


MAC addresses

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness) → MAC 주소를 구입하여 사용
- analogy:
  - MAC address: like Social Security Number ,, identity (don't change)
  - IP address: like postal address 우편 주소 (changable)
- MAC flat address: portability - 이동성
  - can move interface from one LAN to another → LAN 바꿔도 MAC 동일
  - recall IP address *not* portable: depends on IP subnet to which node is attached → LAN 바꾸면 subnet에 따라 ip 변경

ARP: address resolution protocol

Question: how to determine interface's MAC address, knowing its IP address? → gateway 같아야 ccng 사용!



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)

  삭제 전까지 유지되는 시간

ARP protocol in action

example: A wants to send datagram to B

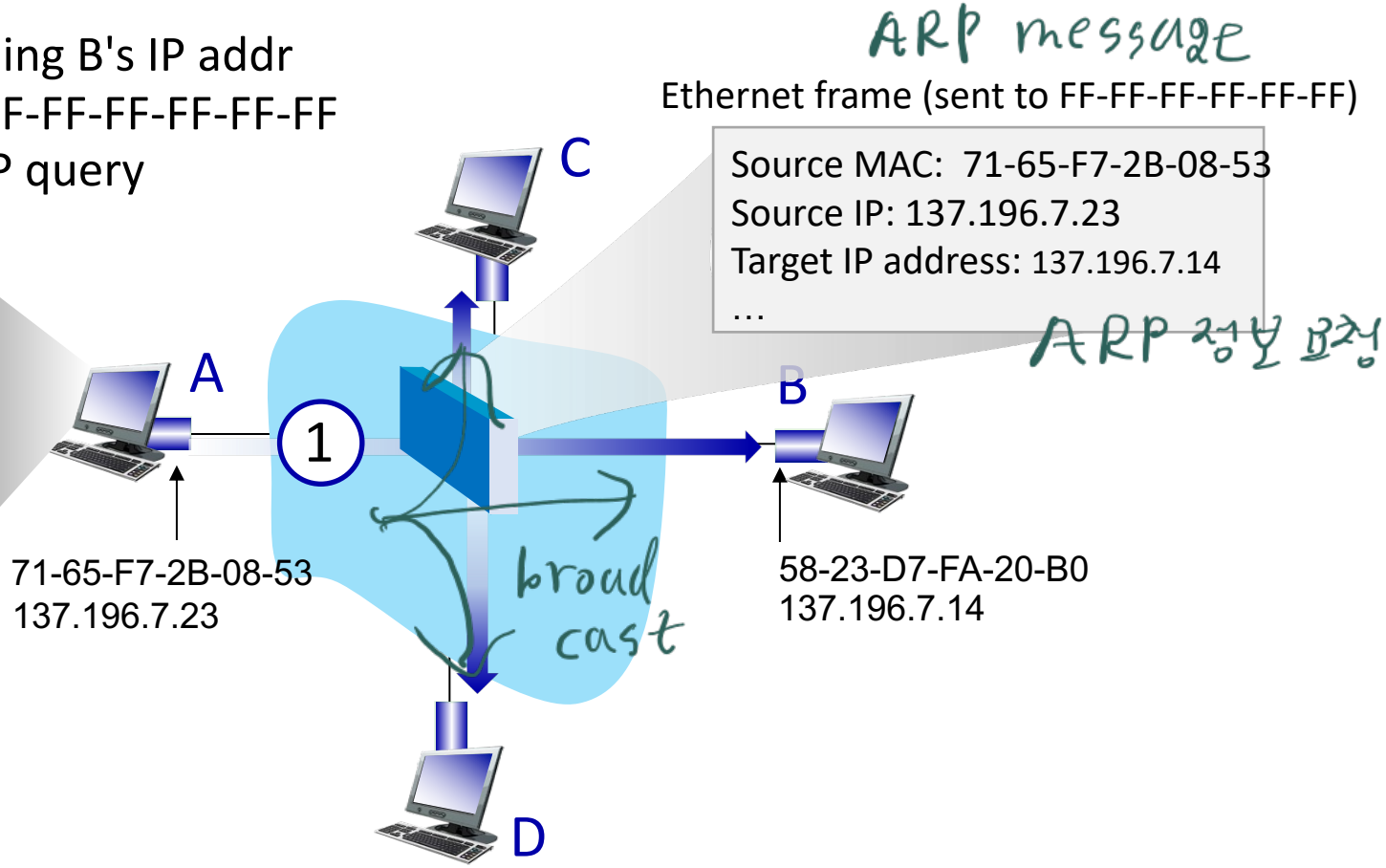
- B's MAC address not in A's ARP table, so A uses ARP to find B's MAC address

A broadcasts ARP query, containing B's IP addr

1
- destination MAC address = FF-FF-FF-FF-FF-FF
- all nodes on LAN receive ARP query

ARP table in A

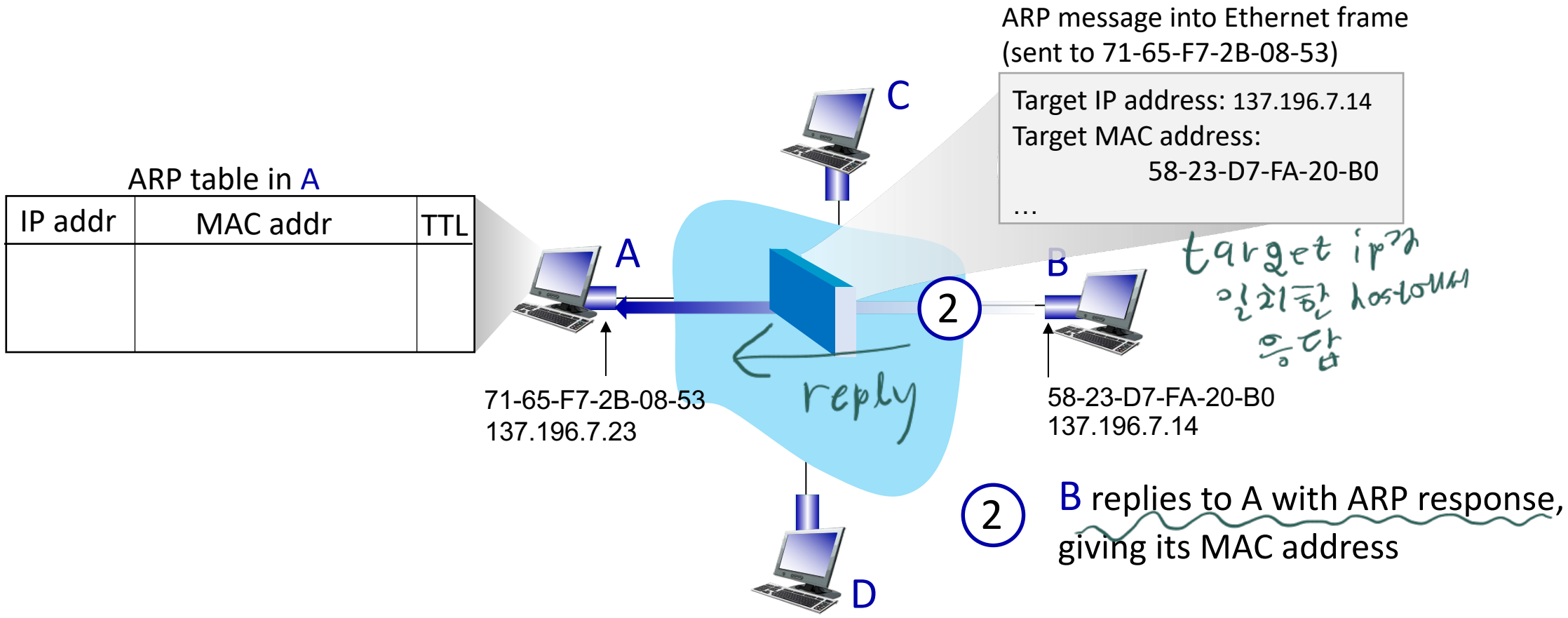
IP addr	MAC addr	TTL



ARP protocol in action

example: A wants to send datagram to B

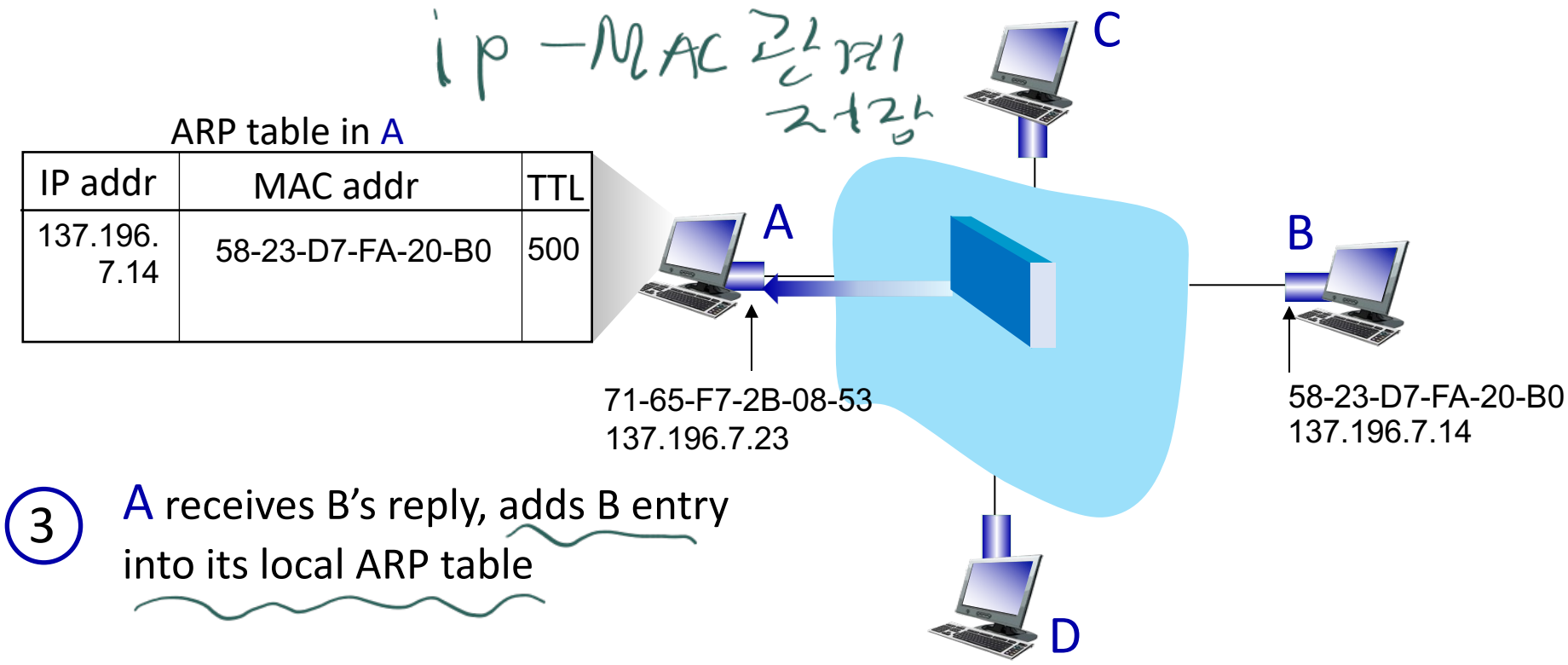
- B's MAC address not in A's ARP table, so A uses ARP to find B's MAC address



ARP protocol in action

example: A wants to send datagram to B

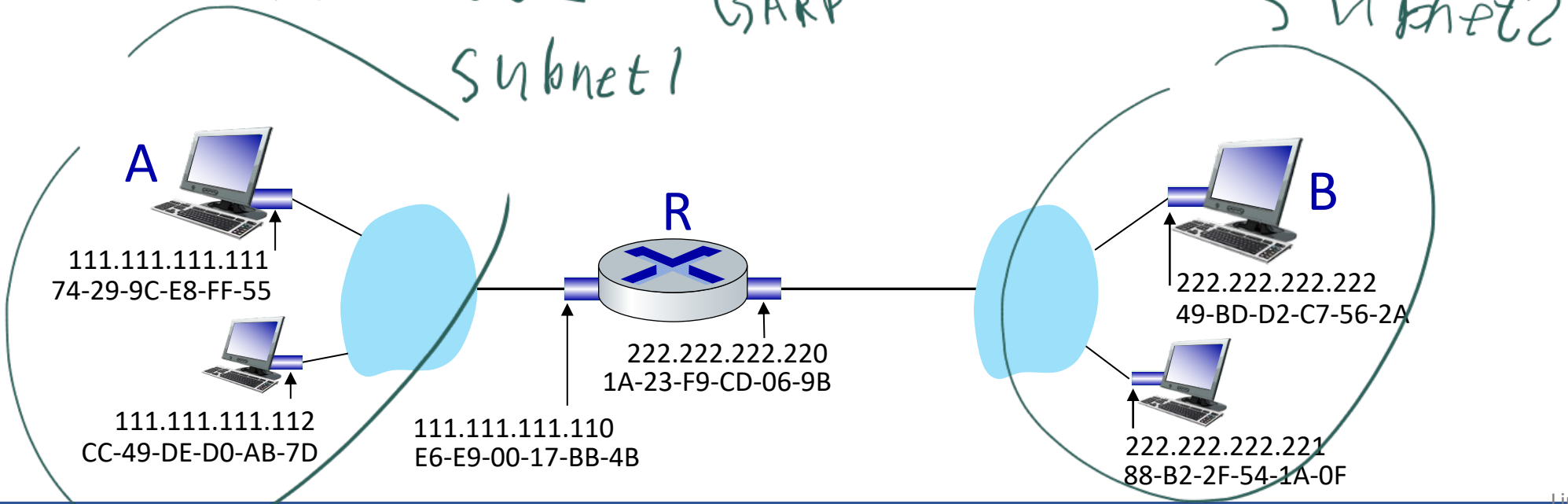
- B's MAC address not in A's ARP table, so A uses ARP to find B's MAC address



Routing to another subnet: addressing

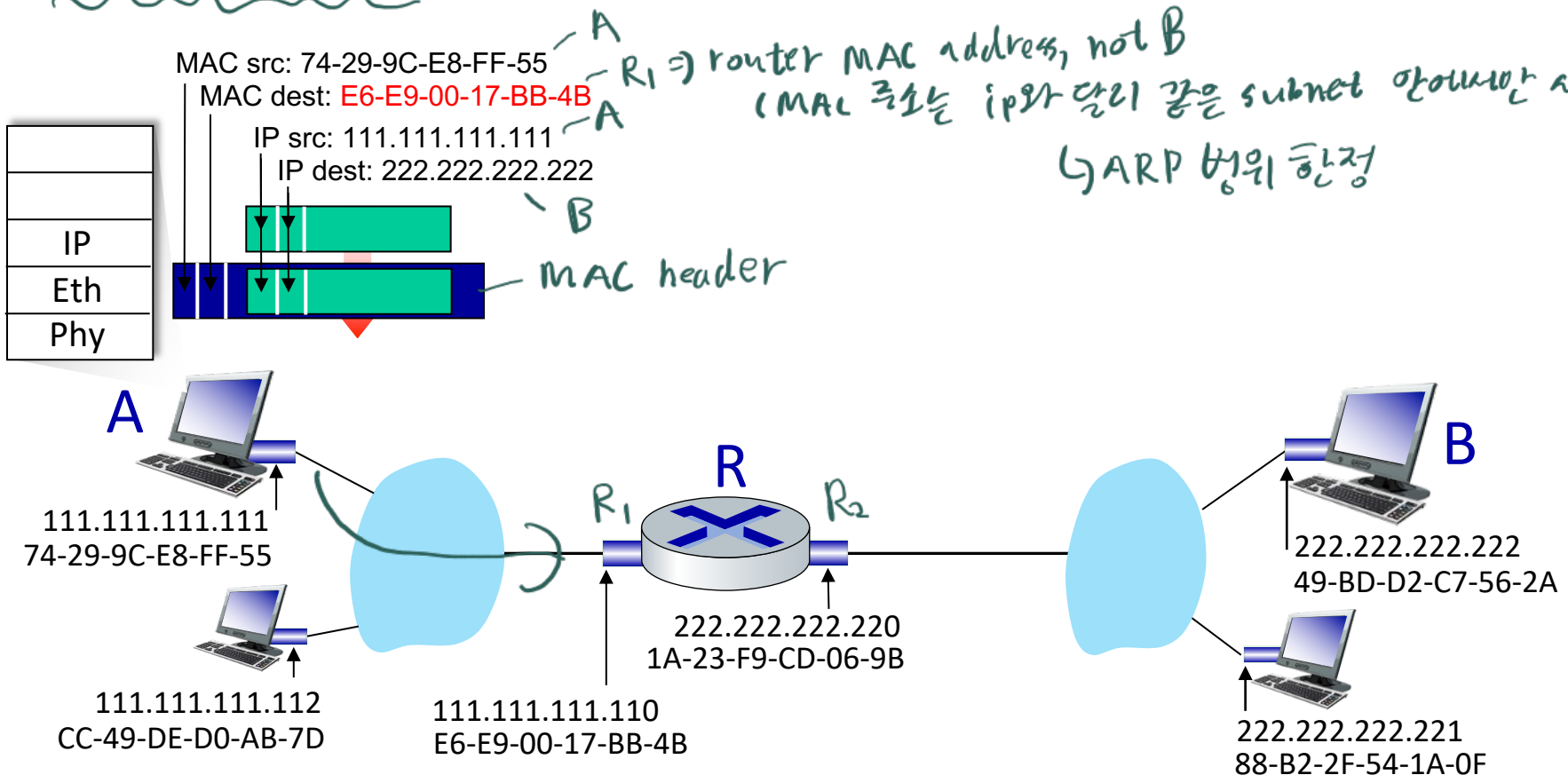
walkthrough: sending a datagram from *A* to *B* via *R*

- focus on addressing – at IP (datagram) and MAC layer (frame) levels
- assume that:
  - A knows B's IP address
  - A knows IP address of first hop router, R (how?)
  - A knows R's MAC address (how?)



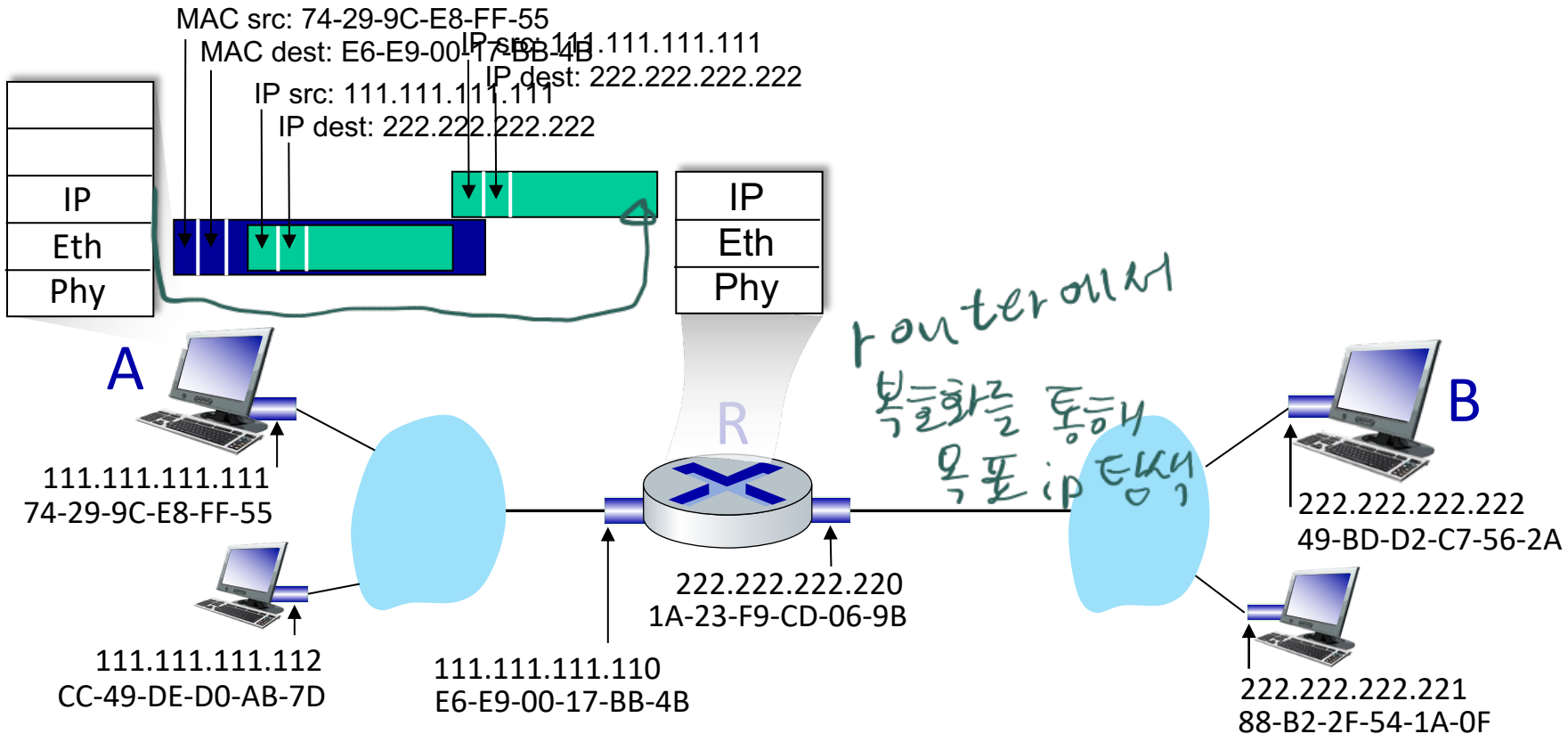
Routing to another subnet: addressing

- A creates IP datagram with IP source A, destination B
- A creates link-layer frame containing A-to-B IP datagram
  - R's MAC address is frame's destination



Routing to another subnet: addressing

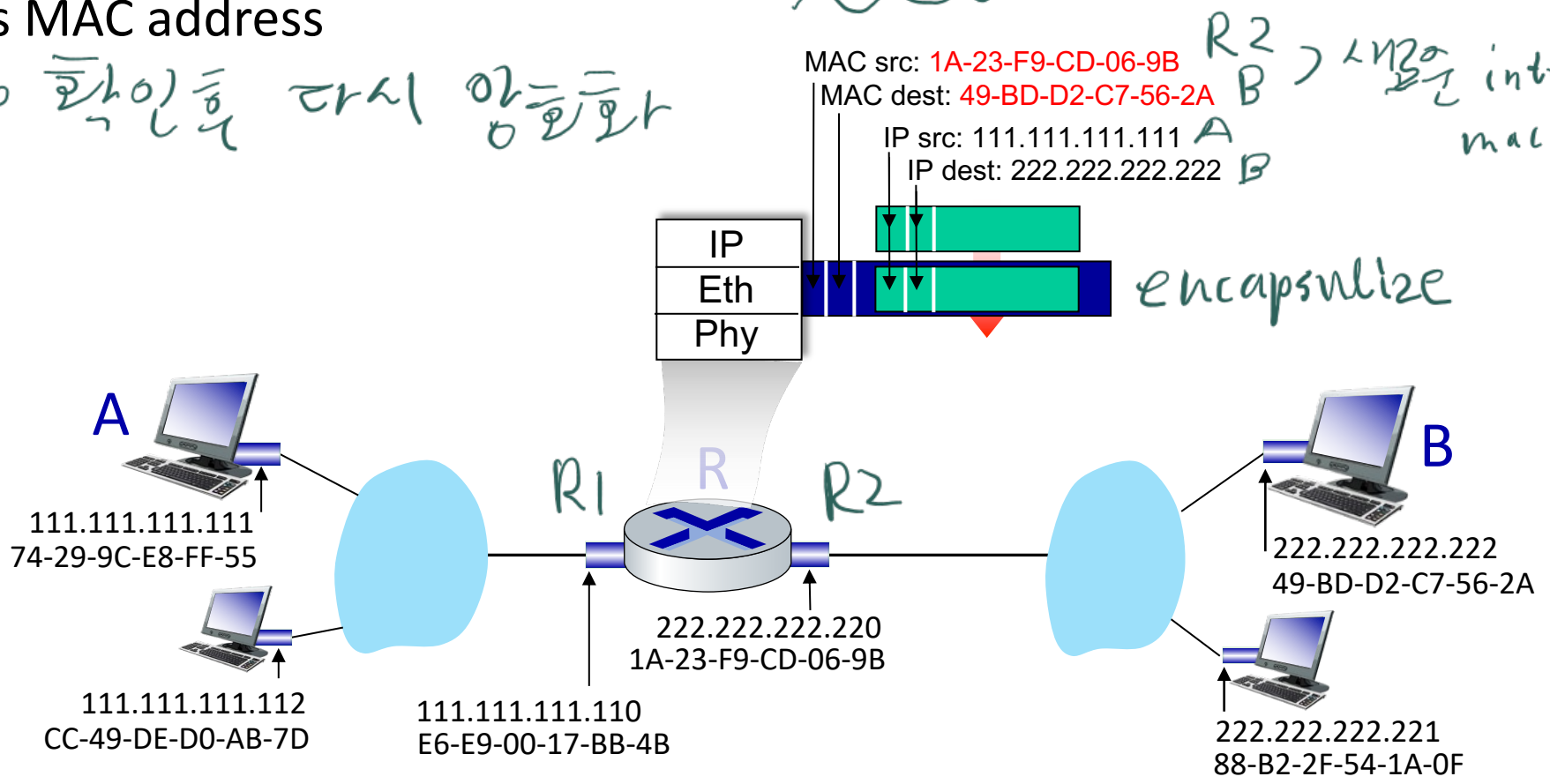
- frame sent from A to R
- frame received at R, datagram removed, passed up to IP



Routing to another subnet: addressing

- R determines outgoing interface, passes datagram with IP source A, destination B to link layer
- R creates link-layer frame containing A-to-B IP datagram. Frame destination address: B's MAC address

ip 확인후 다시 알려줌

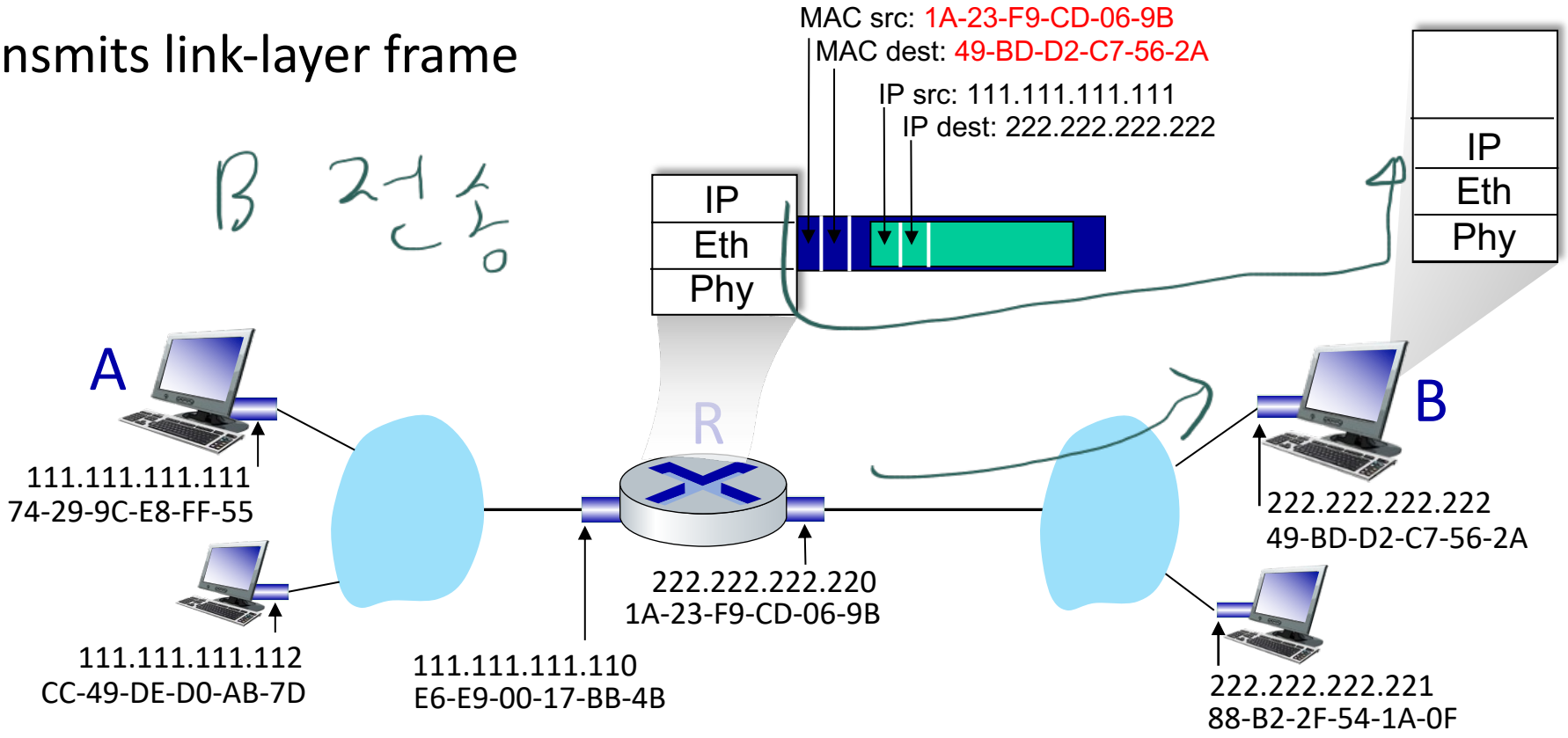


R2 가 보낼 interface의 mac주소로 변경

encapsulize

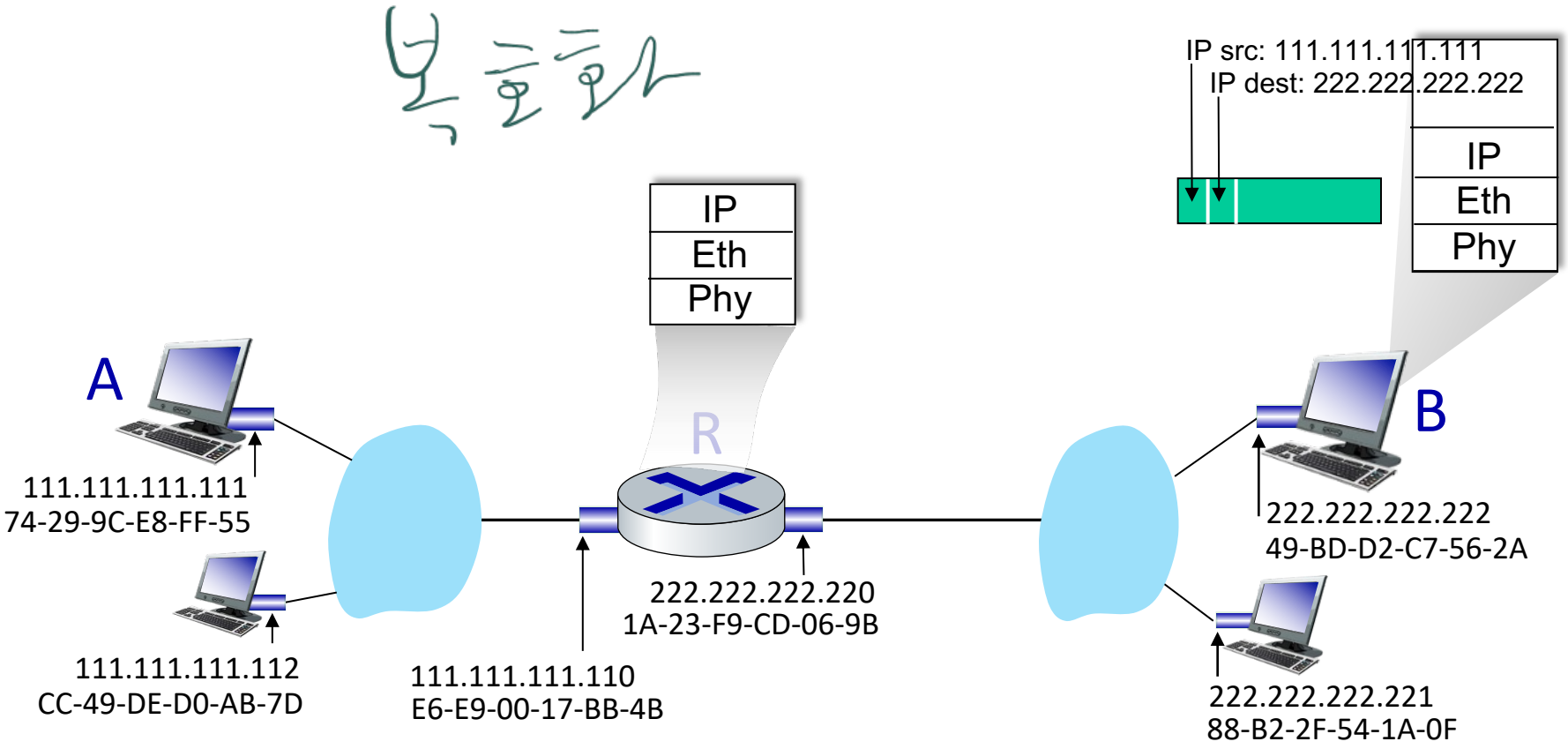
Routing to another subnet: addressing

- R determines outgoing interface, passes datagram with IP source A, destination B to link layer
- R creates link-layer frame containing A-to-B IP datagram. Frame destination address: B's MAC address
- transmits link-layer frame



Routing to another subnet: addressing

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



Link layer, LANs: roadmap

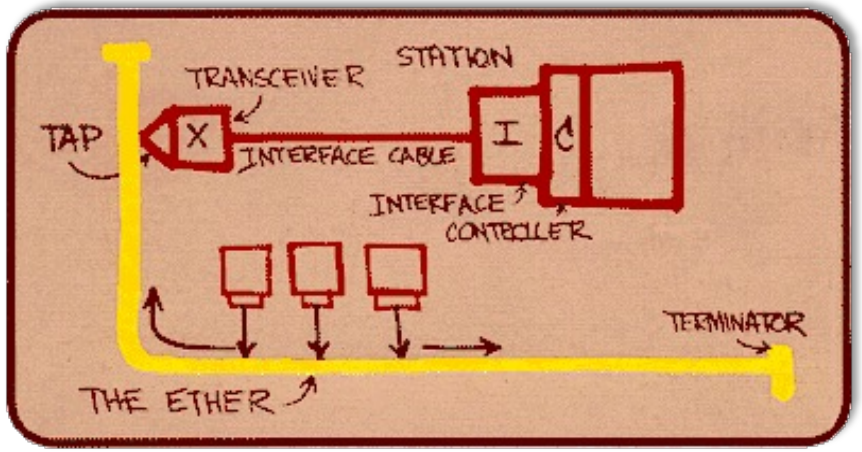
- introduction
- error detection, correction
- multiple access protocols
- LANs
  - addressing, ARP
  - Ethernet
  - switches
  - VLANs
- link virtualization: MPLS
- data center networking

- a day in the life of a web request

Ethernet

"dominant" wired LAN technology:

- first widely used LAN technology
- simpler, cheap
- kept up with speed race: 10 Mbps – 400 Gbps
- single chip, multiple speeds (e.g., Broadcom BCM5761)



Metcalfe's Ethernet sketch

https://www.uspto.gov/learning-and-resources/journeys-innovation/audio-stories/defying-doubters

ACM A.M. Turing Award Honors Bob Metcalfe for Invention, Standardization, and Commercialization of Ethernet

Metcalfe is recognized for creating the foundational technology of the Internet which supports more than 5 billion users and enables much of modern life.

ACM has named Bob Metcalfe as recipient of the 2022 ACM A.M. Turing Award for the invention, standardization, and commercialization of Ethernet. Metcalfe is an Emeritus Professor of Electrical and Computer Engineering (ECE) at The University of Texas at Austin and a Research Affiliate in Computational Engineering at the Massachusetts Institute of Technology (MIT) Computer Science & Artificial Intelligence Laboratory (CSAIL).

The ACM A.M. Turing Award, often referred to as the "Nobel Prize of Computing," carries a $1 million prize with financial support provided by Google, Inc. The award is named for Alan M. Turing, the British mathematician who articulated the mathematical foundations of computing.

Invention of The Ethernet

In 1973, while a computer scientist at the Xerox Palo Alto Research Center (PARC), Metcalfe circulated a now-famous memo describing a "broadcast communication network" for connecting some of the first personal computers, PARC's Altos, within a building. The first Ethernet ran at 2.94 megabits per second, which was about 10,000 times faster than the terminal networks it would replace.

Although Metcalfe's original design proposed implementing this network over coaxial cable, the memo envisioned "communication over an ether," making the design adaptable to future innovations in media technology including legacy telephone twisted pair, optical fiber, radio (Wi-Fi), and even power networks, to replace the coaxial cable as the "ether." That memo laid the groundwork for what we now know today as Ethernet.

Metcalfe's Ethernet design incorporated insights from his experience with ALOHAnet, a pioneering computer networking system developed at the University of Hawaii. Metcalfe recruited David Boggs (d. 2022), a co-inventor of Ethernet, to help build a 100-node PARC Ethernet. That first Ethernet was then

2022 ACM A.M. Turing Award Laureate

Robert Melancton Metcalfe is Emeritus Professor of Electrical and Computer Engineering (ECE) after 11 years at The University of Texas at Austin. He has recently become a Research Affiliate in Computational Engineering at his alma mater, the Massachusetts Institute of Technology (MIT) Computer Science & Artificial Intelligence Laboratory (CSAIL). Metcalfe graduated from MIT in 1969 with Bachelor degrees in Electrical Engineering and Industrial Management. He earned a Master's degree in Applied Mathematics in 1970 and a PhD in Computer Science in 1973 from Harvard University.

제 1회
열린 튜링 강연회

2023년 5월 26일 금요일
16:00-17:30

KAIST 학술문화관(E9) 5층
정근모 컨퍼런스홀

***강연내용**
- 2022년 튜링상(인터넷) 해설: **문수복 교수**(KAIST)
- 2018년 튜링상(인공지능) 해설: **이남훈 교수**(POSTECH)
- 2012년 튜링상(암호학) 해설: **송용수 교수**(서울대학교)

서울대학교 SEOUL NATIONAL UNIVERSITY | KAIST | Value Creating University POSTECH

제1회 열린 튜링 강연회 / The 1 st Open Turing: Turing Award Achievements Lecture

Association for Computing Machinery

A.M. TURING AWARD

2022년 수상자:
로버트 멧캐프

"이더넷 발명, 표준화, 상업화에 대한 공로"
"For the invention, standardization, and commercializaiton of Ethernet"

AWARD RECIPIENT

Robert Melancton Metcalfe

ACM A. M. Turing Award (2022)

ACM Grace Murray Hopper Award (1980)

2022 ACM A.M. Turing Award

문 수 복 교수
KAIST
2022년 튜링상(인터넷)

Prof. Sue Moon
KAIST
Lecture on the winner of 2022

서울대학교 SEOUL NATIONAL UNIVERSITY KAIST Value Creating University POSTECH

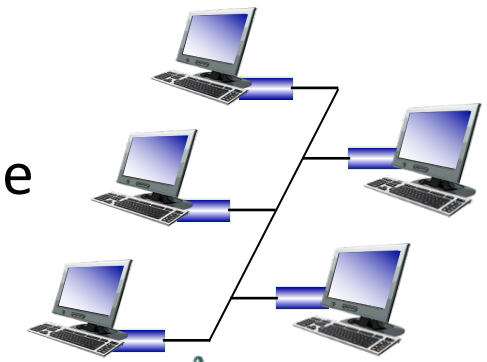
Open Turing : Lecture on the Winners of Turing Award(#1 Eng.)

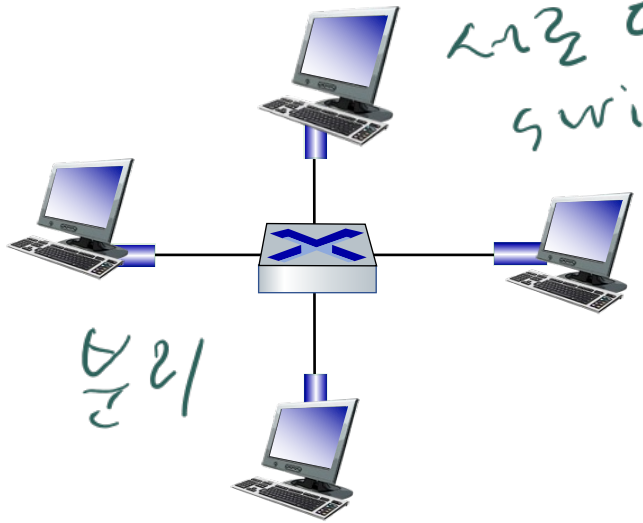
https://www.youtube.com/watch?v=pLQOO1kNgYU

Ethernet: physical topology

- bus: popular through mid 90s
  - all nodes in same collision domain (can collide with each other)
- switched: prevails today
  - active link-layer 2 *switch* in center
  - each "spoke" runs a (separate) Ethernet protocol (nodes do not collide with each other)

bus: coaxial cable

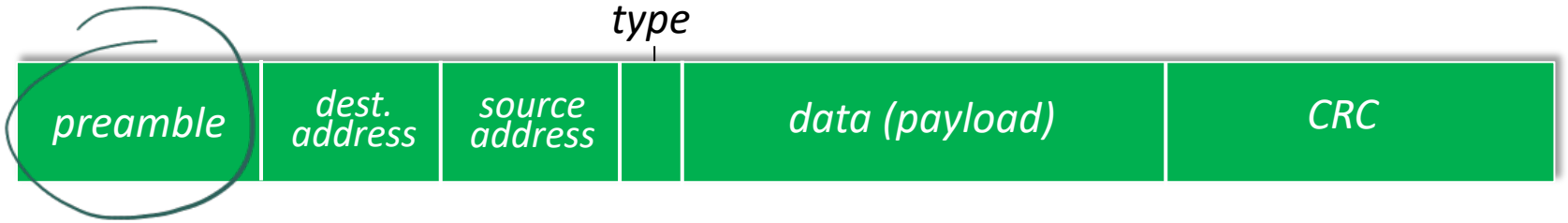




switched

Ethernet frame structure

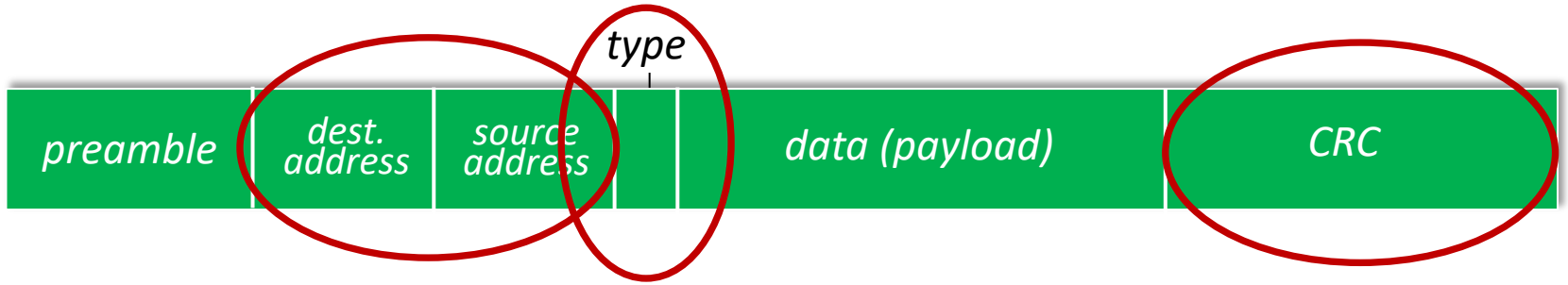
sending interface encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame



preamble:

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

Ethernet frame structure (more)



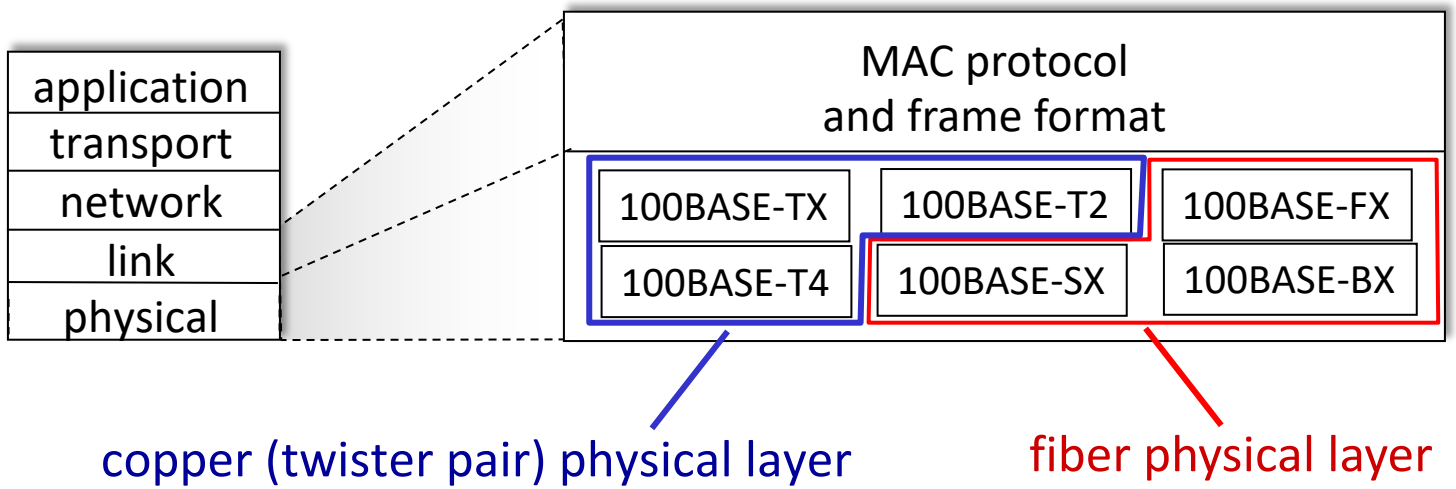
- addresses: 6 byte source, destination MAC addresses
  - if adapter receives frame with matching destination address, or with broadcast address (e.g., ARP packet), it passes data in frame to network layer protocol
  - otherwise, adapter discards frame
- type: indicates higher layer protocol
  - mostly IP but others possible, e.g., Novell IPX, AppleTalk
  - used to demultiplex up at receiver
- 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 NAKs 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, LANs: roadmap

- introduction
- error detection, correction
- multiple access protocols
- LANs
  - addressing, ARP
  - Ethernet
  - switches
  - VLANs
- link virtualization: MPLS
- data center networking

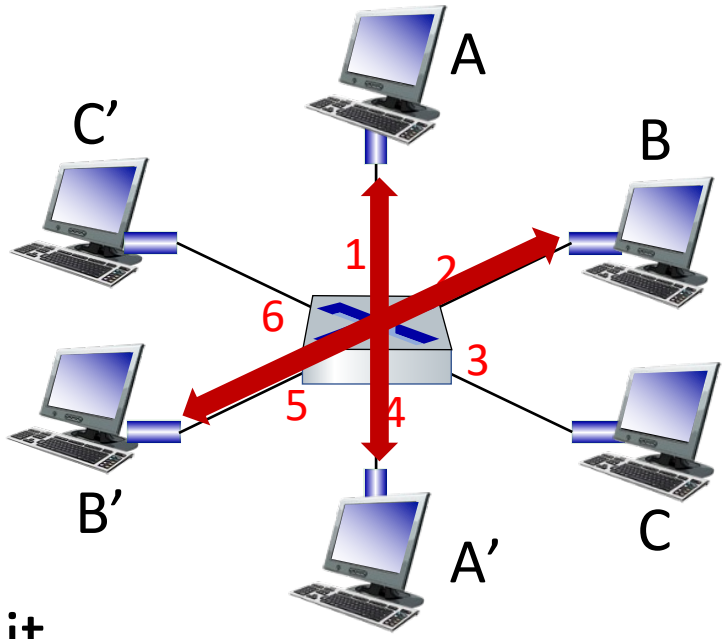
- a day in the life of a web request

Ethernet switch

- Switch is a link-layer device: takes an *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 *unaware* of presence of switches
- plug-and-play, self-learning
  - switches do not need to be configured

Switch: multiple simultaneous transmissions

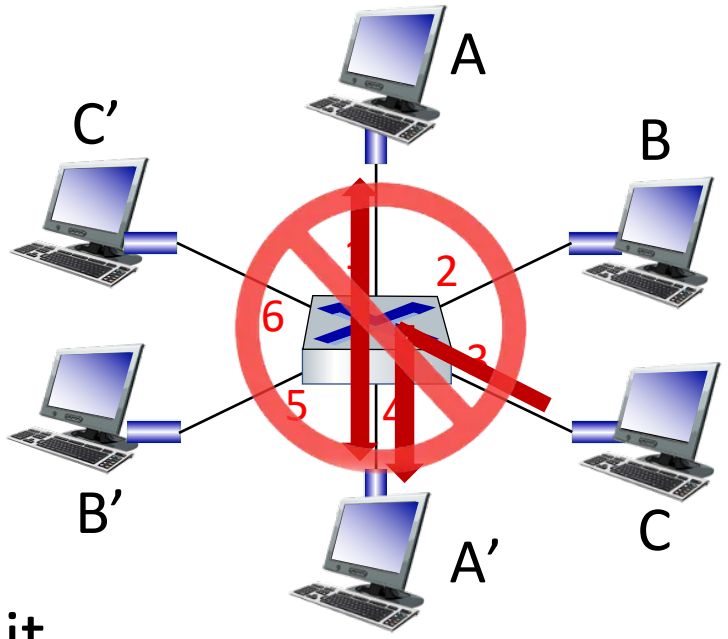
- hosts have dedicated, direct connection to switch
- switches buffer packets
- Ethernet protocol used on *each* incoming link, so:
  - no collisions; full duplex
  - each link is its own collision domain
- switching: A-to-A' and B-to-B' can transmit simultaneously, without collisions



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

Switch: multiple simultaneous transmissions

- hosts have dedicated, direct connection to switch
- switches buffer packets
- Ethernet protocol used on *each* incoming link, so:
  - no collisions; full duplex
  - each link is its own collision domain
- switching: A-to-A' and B-to-B' can transmit simultaneously, without collisions
  - but A-to-A' and C to A' can *not* happen simultaneously



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

Switch forwarding table

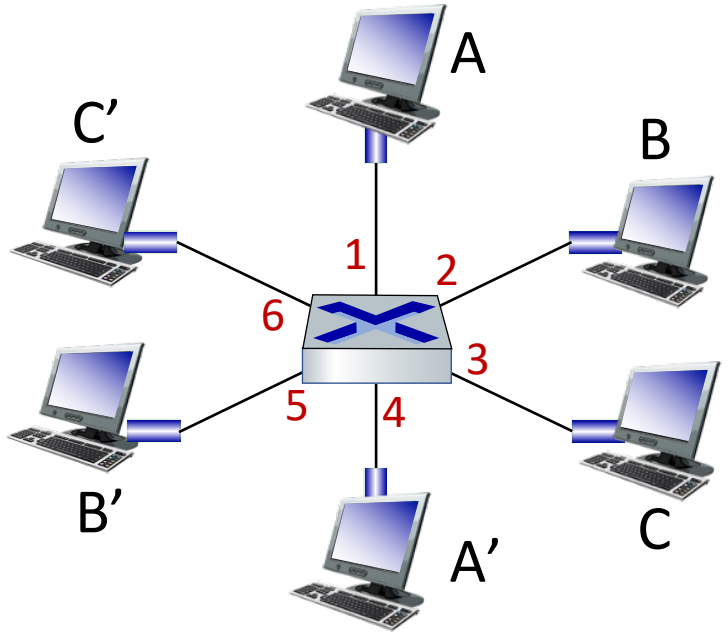
Q: how does switch know A' reachable via interface 4, B' reachable via interface 5?

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

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

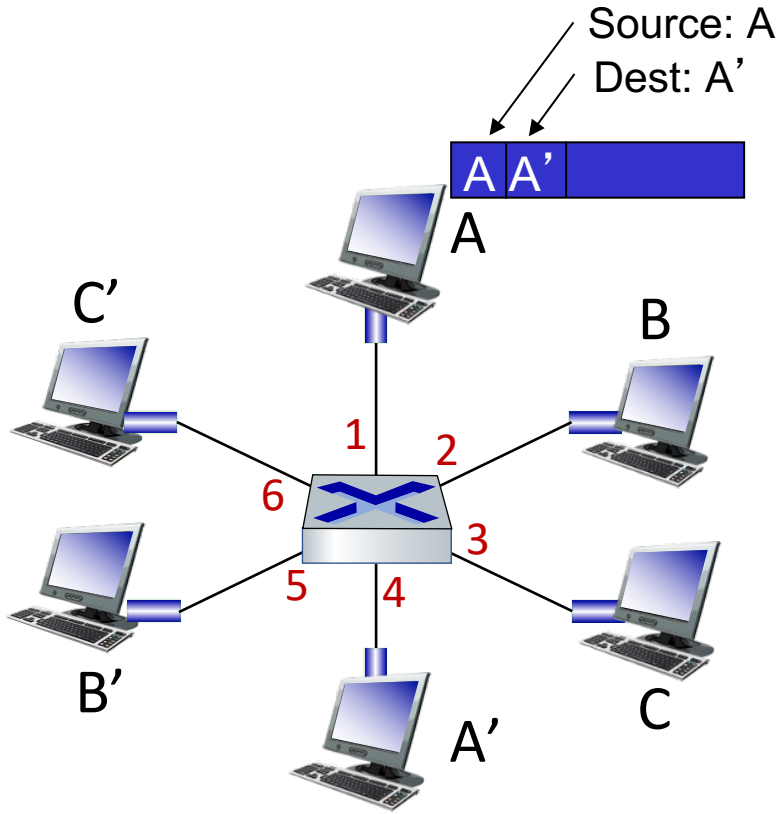
Q: how are entries created, maintained in switch table?

- something like a routing protocol?



Switch: 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)

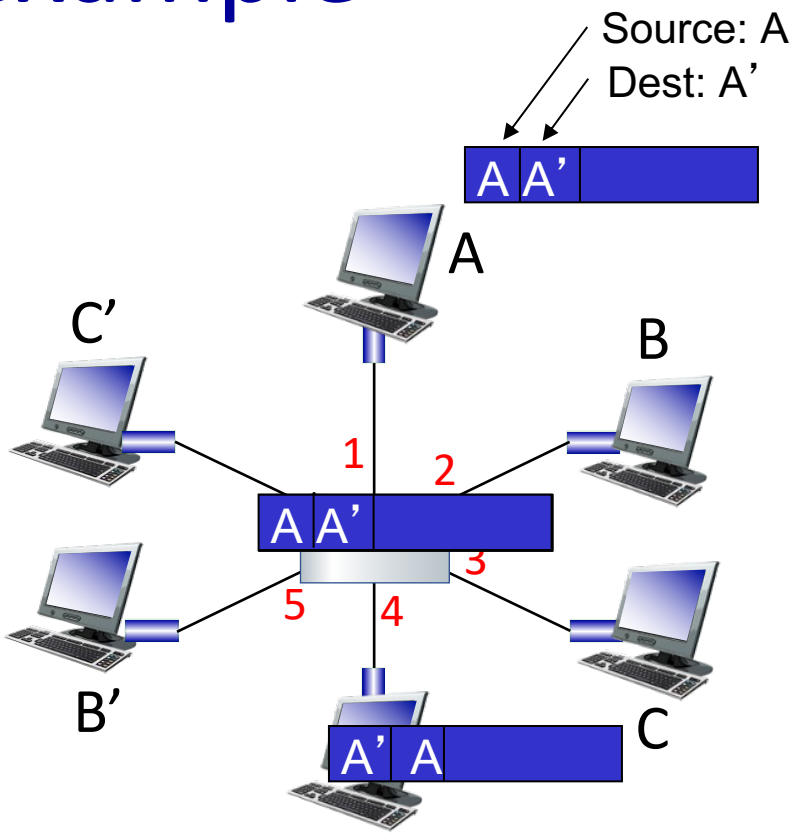
Switch: frame filtering/forwarding

when frame received at switch:

1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. 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 */

Self-learning, forwarding: example

- frame destination, A', location unknown: flood
- destination A location known: selectively send on just one link

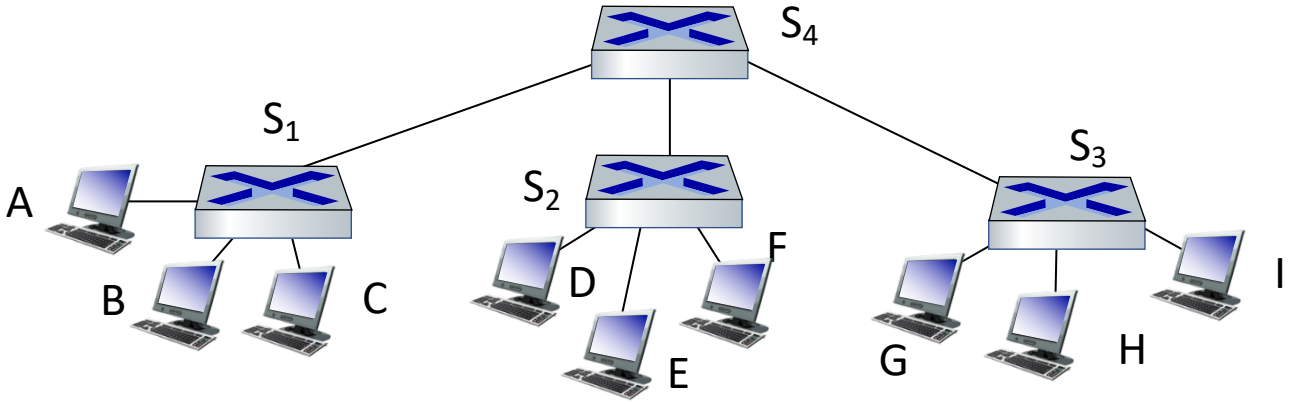


MAC addr	interface	TTL
A	*1*	*60*
A'	*4*	*60*

switch table (initially empty)

Interconnecting switches

self-learning switches can be connected together:

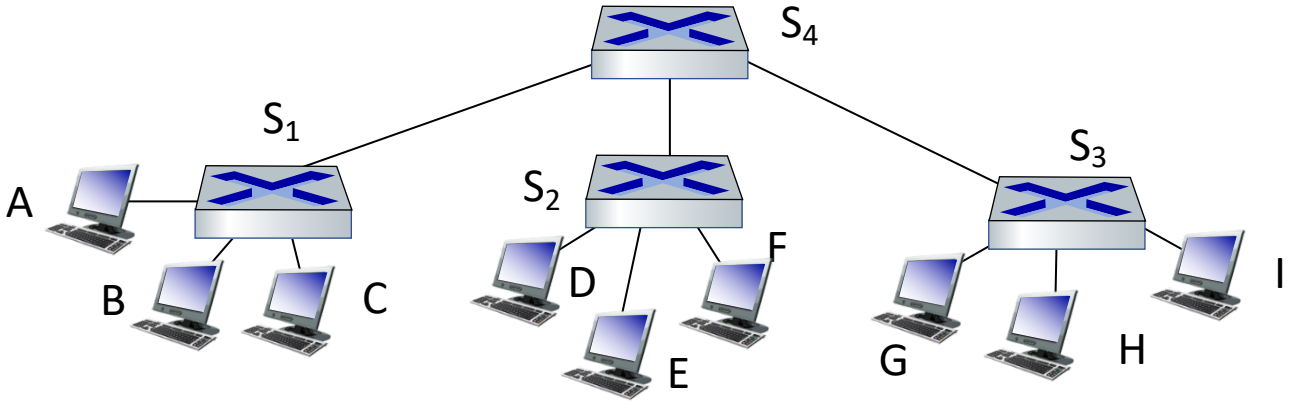


Q: sending from A to G - how does S_1 know to forward frame destined to G via S_4 and S_3?

- *A:* self learning! (works exactly the same as in single-switch case!)

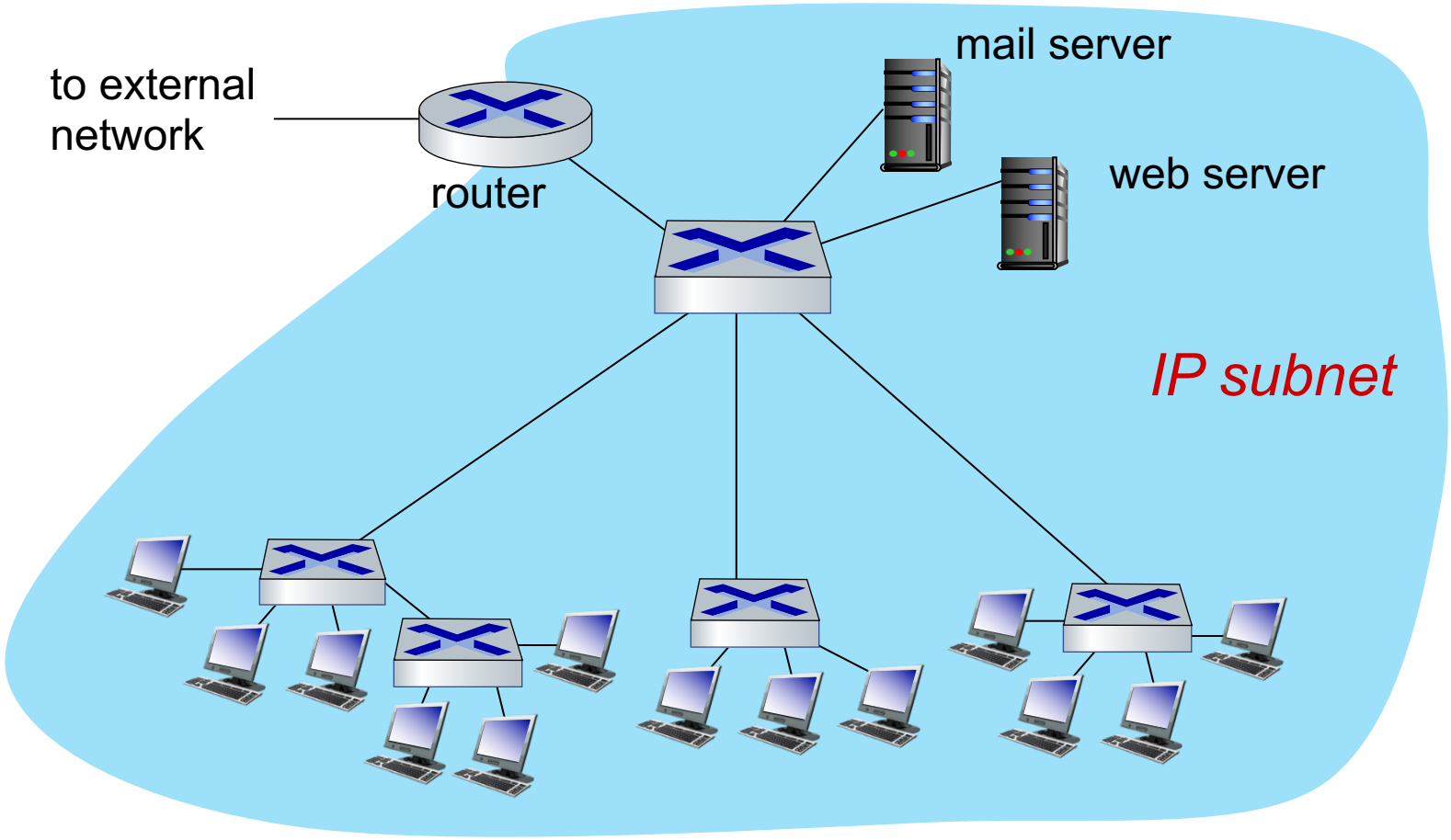
Self-learning multi-switch example

Suppose C sends frame to I, I responds to C



Q: show switch tables and packet forwarding in S_1, S_2, S_3, S_4

Small institutional network



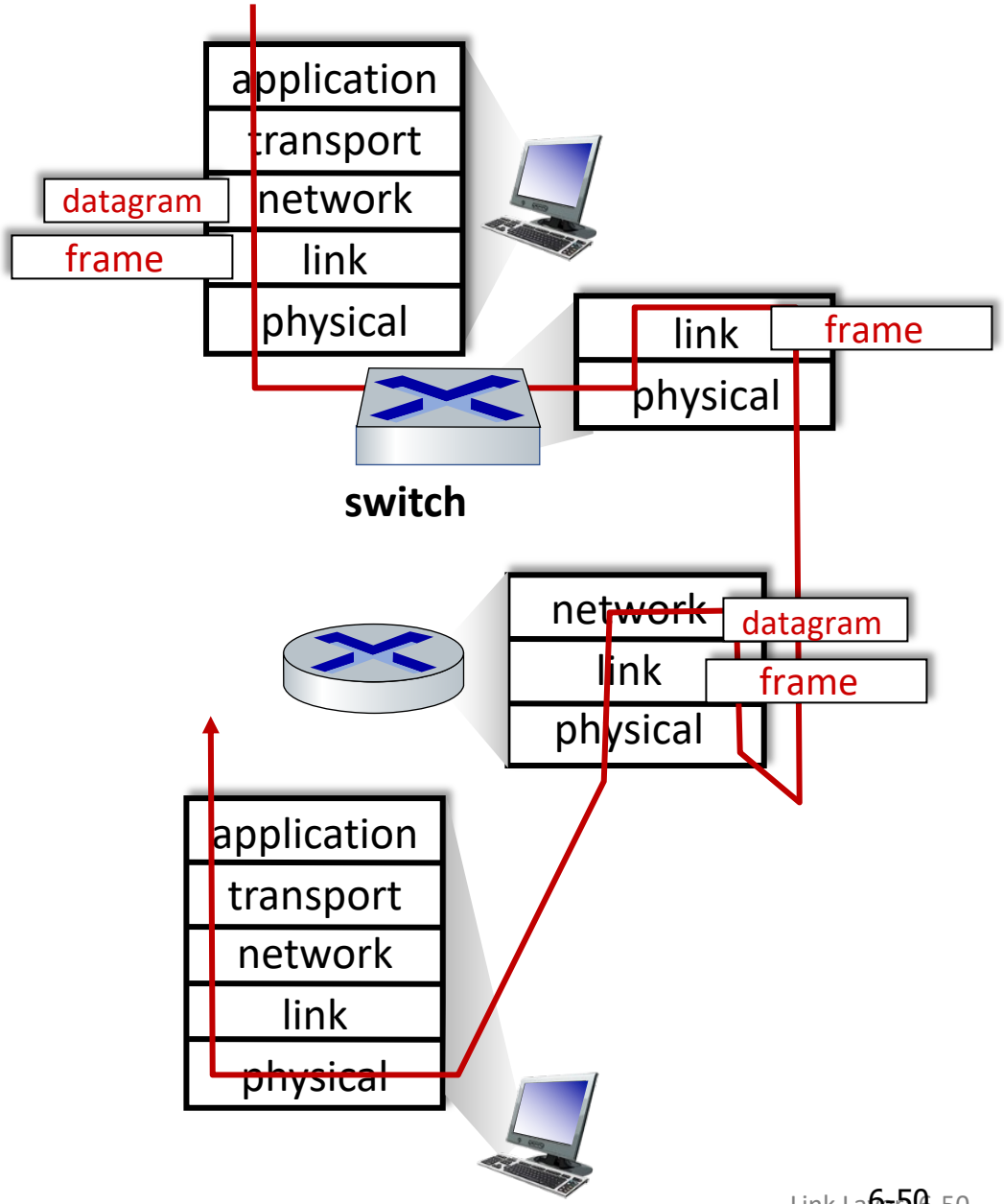
Switches vs. routers

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



Next…

- *Chapter 6.6 Data Center Networking*
- *Chapter 7.1 Introduction (Wireless and Mobile Networks)*