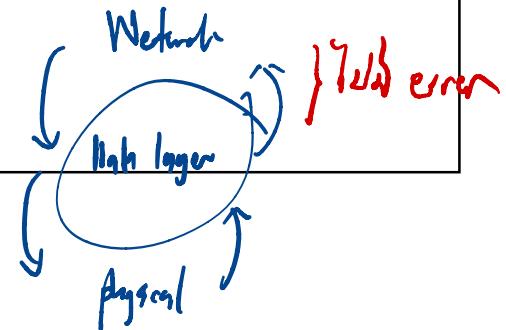


Chapter 6

The Link Layer and LANs

OSI : layer 2

2



Link layer and LANs: our goals

- understand principles behind link layer services:
 - error detection, correction
 - sharing a broadcast channel: multiple access
-
- link layer addressing
 - local area networks: Ethernet, VLANs

- instantiation, implementation of various link layer technologies



Link Layer: 6-4

4

Link layer, LANs: roadmap

- introduction
 - error detection, correction
 - multiple access protocols
 - LANs
 - addressing, ARP
 - Ethernet
 - switches
 - VLANs



- a day in the life of a web request

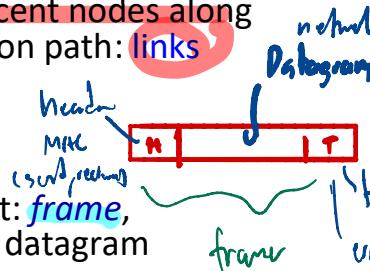
Link Layer: 6-7

7

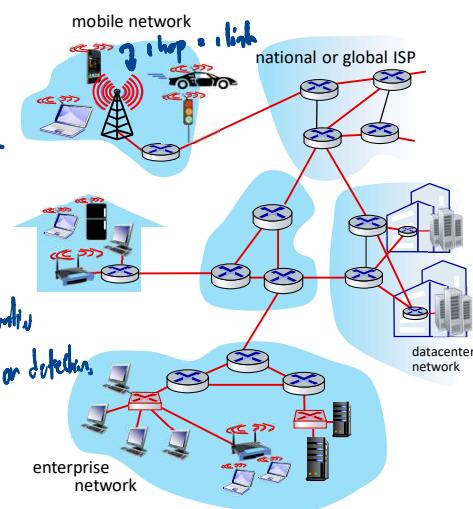
Link layer: introduction

terminology:

- hosts and routers: nodes
- communication channels that connect adjacent nodes along communication path: links
 - wired
 - wireless
 - LANs
- layer-2 packet: frame, encapsulates datagram



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



Link Layer: 6-8

8

Link layer: context

- datagram transferred by different link protocols over different links:
 - e.g., WiFi on first link, Ethernet on next link
- each link protocol provides different services
 - e.g., may or may not provide reliable data transfer 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

Link Layer: 6-9

9

Datalink: Hop → Hop Ex: router → router.

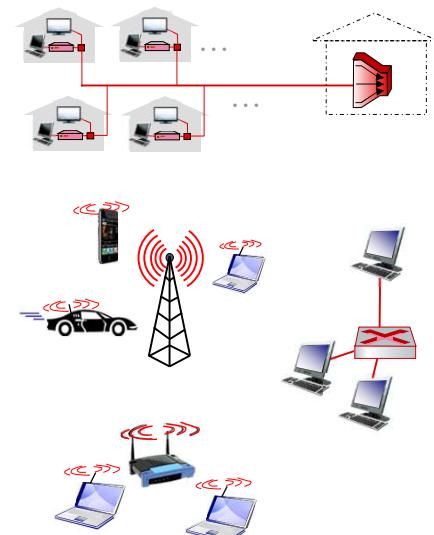
Network: Source → Dest. Ex: client → host.

Link layer: services

- framing, link access:
 - encapsulate datagram into frame, adding header, trailer
 - channel access if shared medium
 - “MAC” addresses in frame headers identify source, destination (different from IP address!)
- reliable delivery between adjacent nodes
 - we already know how to do this!
 - seldom used on low bit-error links
 - wireless links: high error rates
 - Q: why both link-level and end-end reliability?
 - noise

Illustrations and notes:

- Physical Address
- data link layer header uses frame for source + dest.
- Data Com.
 - Go back END
 - Selective ACK
 - Repeat



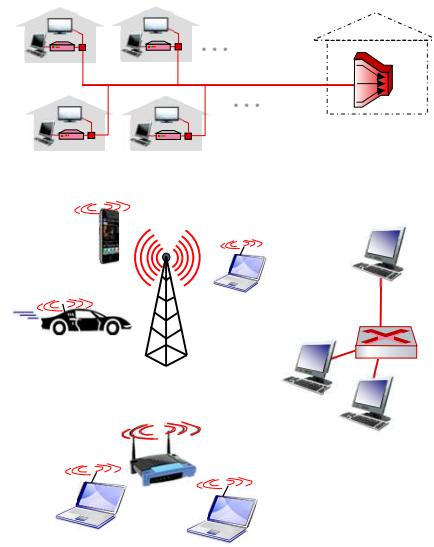
Link Layer: 6-10

10

Link layer: services (more)

- **flow control:**
 - pacing between adjacent sending and receiving nodes
- **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**
- **half-duplex and full-duplex:**
 - with half duplex, nodes at **both ends of link**, but not at same time

လိပ်ချက်မှာ မြတ်စွမ်းမည့်



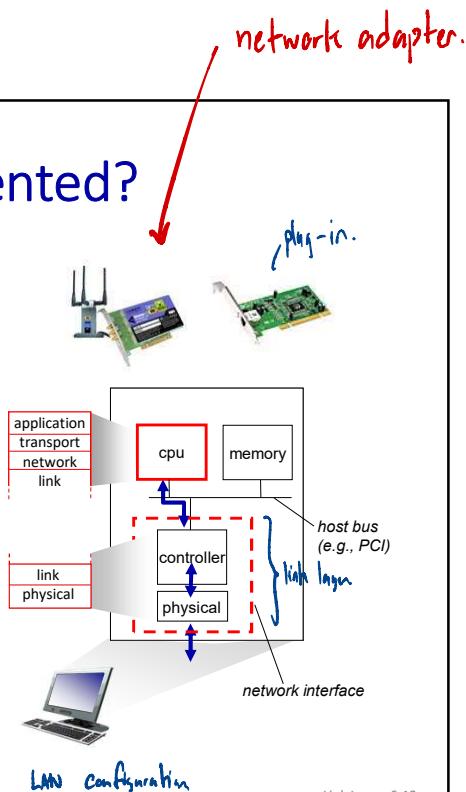
Link Layer: 6-11

11

လိပ်ချက် + အောင်

Where is the link layer implemented?

- in each-and-every host
- link layer implemented in **network interface card** (NIC) or **on a chip**
 - Ethernet, WiFi card or chip
 - implements link, physical layer
- attaches into host's system **buses**
- combination of hardware, software, firmware



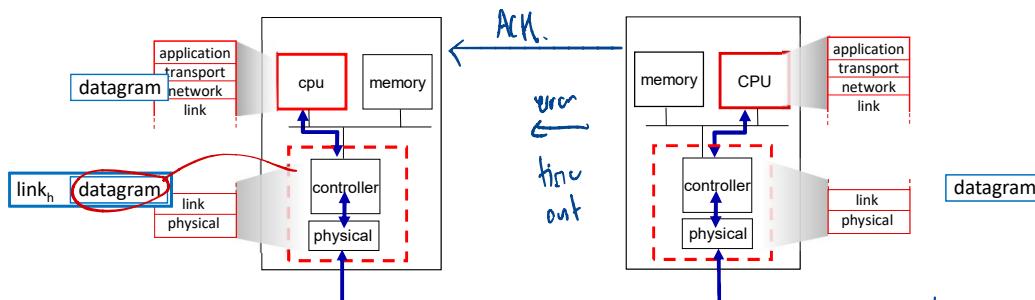
Link Layer: 6-12

12



Interfaces communicating

Flow control in Data com.



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

detect error

Link Layer: 6-13

13

Link layer, LANs: roadmap

- introduction
- error detection, correction
- multiple access protocols
- LANs
 - addressing, ARP
 - Ethernet
 - switches
 - VLANs
- link virtualization: MPLS
- a day in the life of a web request



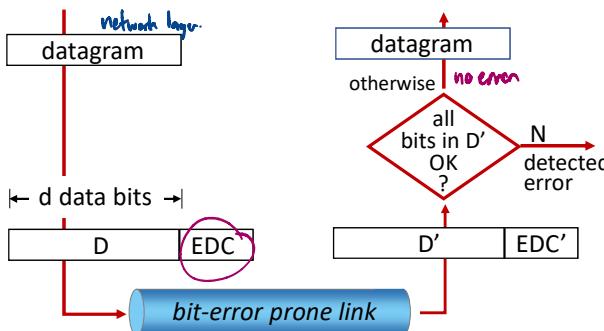
Link Layer: 6-15

15

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

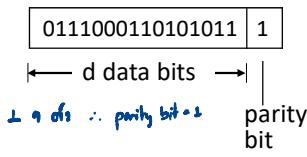
Link Layer: 6-16

16

Parity checking

single bit parity:

- detect single bit errors

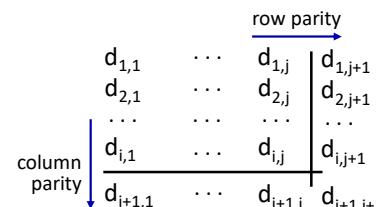


Even parity: set parity bit so there is an even number of 1's



two-dimensional bit parity:

- detect and correct single bit errors



even check:	
in data (s)	Parity bit (s)
1 0 1 0 1 1	1

no errors:	
1 0 1 0 1 1	1
1 1 1 1 0 0	0
0 1 1 1 0 1	1
1 0 1 0 1 0	0

detected and correctable single-bit error:
1 0 1 0 1 1
1 0 1 1 0 0

parity error

Link Layer: 6-18

18

Internet checksum (review)

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

sender:

- treat contents of UDP segment (including UDP header fields and IP addresses) as sequence of 16-bit integers
- **checksum:** addition (one's complement sum) of segment content
- checksum value put into UDP checksum field

receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
 - not equal - error detected
 - equal - no error detected. *But maybe errors nonetheless? More later*

Transport Layer: 3-19

19

Cyclic Redundancy Check (CRC)

- more powerful error-detection coding
- **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 (mod 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](#))

Remainder = 0 → ✓

Link Layer: 6-20

20

Cyclic Redundancy Check (CRC): example

We want:

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

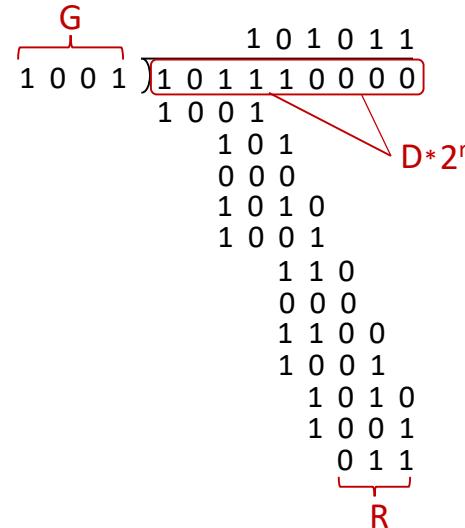
or equivalently:

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

or equivalently:

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

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



* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Link Layer: 6-21

21

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

Link Layer: 6-22

22

Multiple access links, protocols

two types of “links”:

- **point-to-point**
 - point-to-point link between Ethernet switch, host
 - PPP for dial-up access
- **broadcast (shared wire or medium)**
 - old-fashioned Ethernet
 - upstream HFC in cable-based access network
 - 802.11 wireless LAN, 4G/4G, satellite



Link Layer: 6-23

23

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

Link Layer: 6-24

24

An ideal multiple access protocol

given: multiple access channel (**MAC**) of rate R bps

desiderata: 

1. when one node wants to transmit, it can send at rate R . R (1 nodes)
2. when M nodes want to transmit, each can send at average rate R/M R/M (> 1 nodes)
3. fully decentralized:
 - no special node to coordinate transmissions → 
 - no synchronization of clocks, slots
4. simple

CPU ռայընթեռ անուղղական
մասնագիր համար աշխատավոր
է աշխատավոր աշխատավոր
աշխատավոր

Link Layer: 6-25

25

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

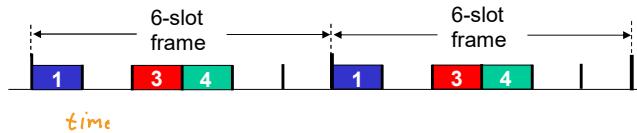
Link Layer: 6-26

26

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** → empty - in.
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle



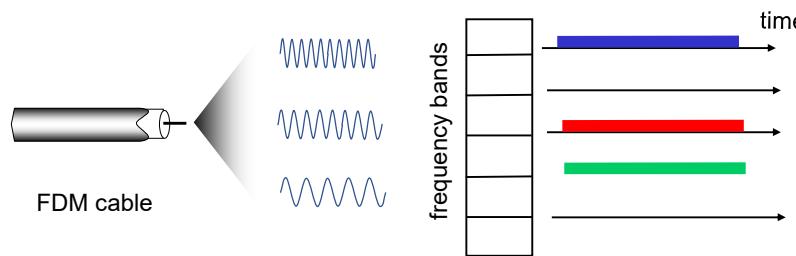
Link Layer: 6-27

27

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



Link Layer: 6-28

28

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:
 - ALOHA,
 - slotted ALOHA
 - CSMA, → *no collision detection*
 - CSMA/CD, ↳ *collision detection*
 - CSMA/CA

Link Layer: 6-29

29

Slotted ALOHA

go into next slot immediately.

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:

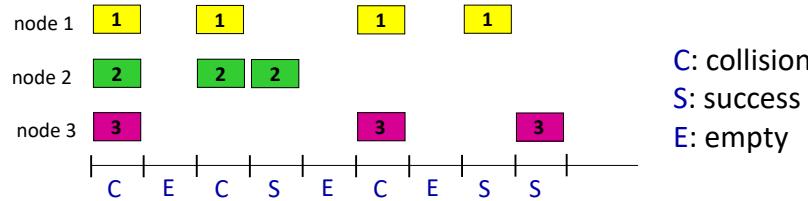
- when node obtains fresh frame, transmits in next slot
 - if *no collision*: node can send new frame in next slot
 - if *collision*: node **retransmits** frame in each **subsequent slot** with **probability p** until success

randomization – why?

Link Layer: 6-30

30

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

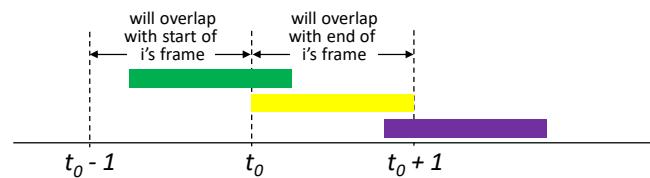
$$\text{max efficiency} = 1/e = 0.37$$

Link Layer: 6-31

31

Pure ALOHA

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



- pure Aloha efficiency: 18% !

Link Layer: 6-33

33

CSMA (carrier sense multiple access)

simple CSMA: listen before transmit:

- if channel sensed idle: transmit entire frame no collision.
- if channel sensed busy: defer transmission

CSMA/CD: CSMA with *collision detection*

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

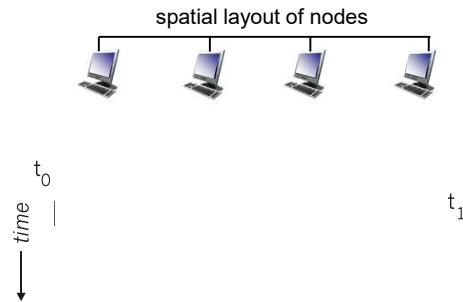
hidden terminal problem

Link Layer: 6-35

35

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 determining collision probability

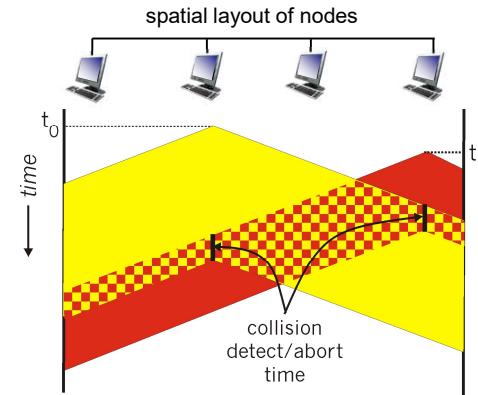


Link Layer: 6-36

36

CSMA/CD:

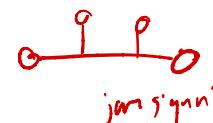
- CSMA/CD reduces the amount of time wasted in collisions
 - transmission aborted on collision detection



Link Layer: 6-37

37

Ethernet CSMA/CD algorithm

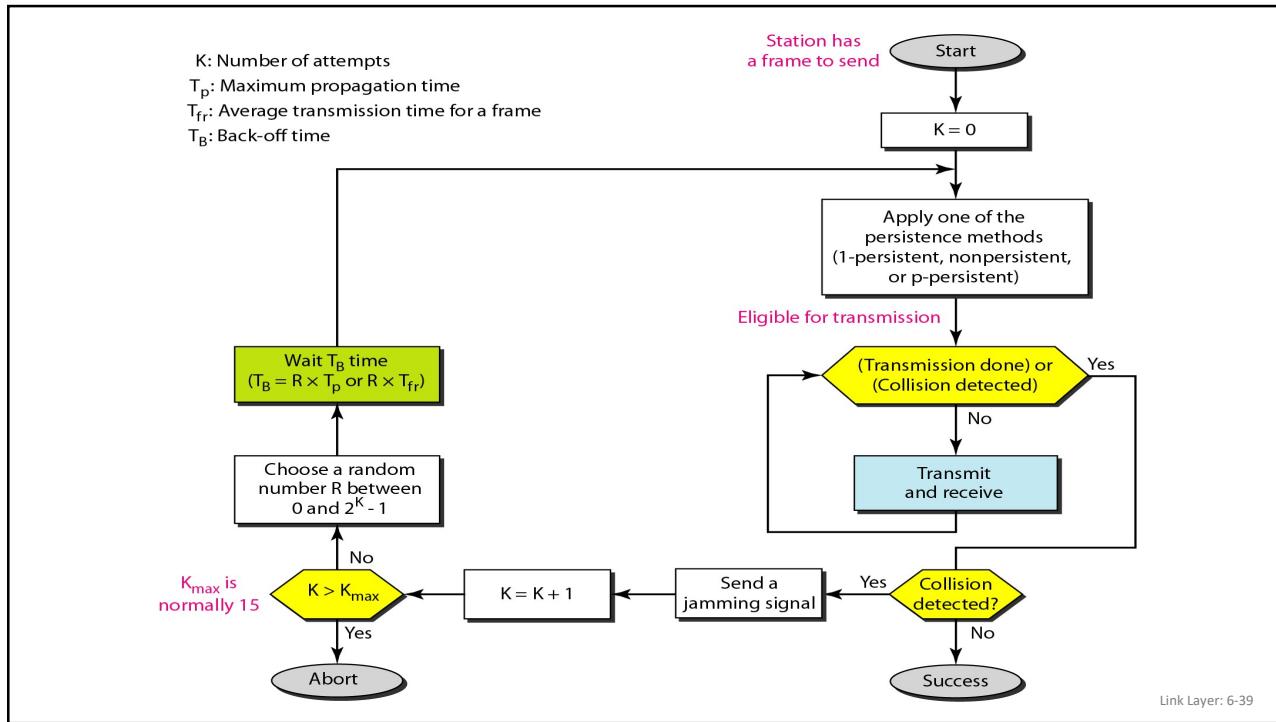


1. NIC receives datagram from network layer, creates frame
2. If NIC senses channel:
 - if idle: start frame transmission.
 - if busy: wait until channel idle, then transmit
3. If NIC transmits entire frame without collision, NIC is done with frame !
4. If NIC detects another transmission while sending: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

yellow
on air
repeat

Link Layer: 6-38

38



39

“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!

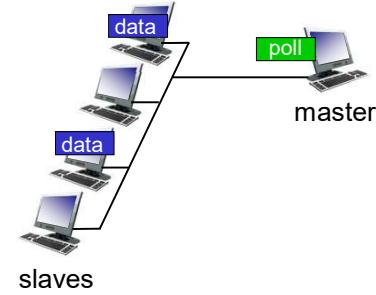
Link Layer: 6-41

41

“Taking turns” MAC protocols

polling:

- master node “invites” other nodes to transmit in turn
- typically used with “dumb” devices
- concerns:
 - polling overhead
 - latency *delay*
 - single point of failure (master)



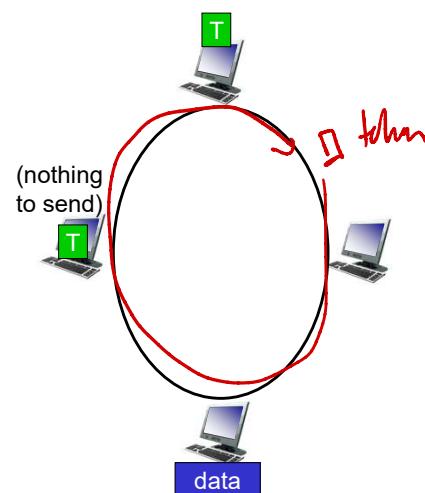
Link Layer: 6-42

42

“Taking turns” MAC protocols

token passing:

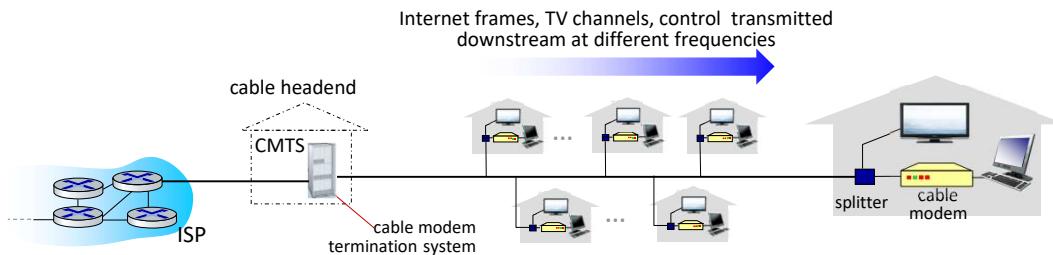
- control *token* passed from one node to next sequentially.
- token message
- concerns:
 - token overhead
 - latency
 - single point of failure (token)



Link Layer: 6-43

43

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

Link Layer: 6-44

44

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 *Lan*
 - CSMA/CA used in 802.11 *Wlan*
- **taking turns**
 - **polling** from central site, **token passing**
 - Bluetooth, FDDI, token ring



Link Layer: 6-46