

Data Link Layer: Overview; Error Detection

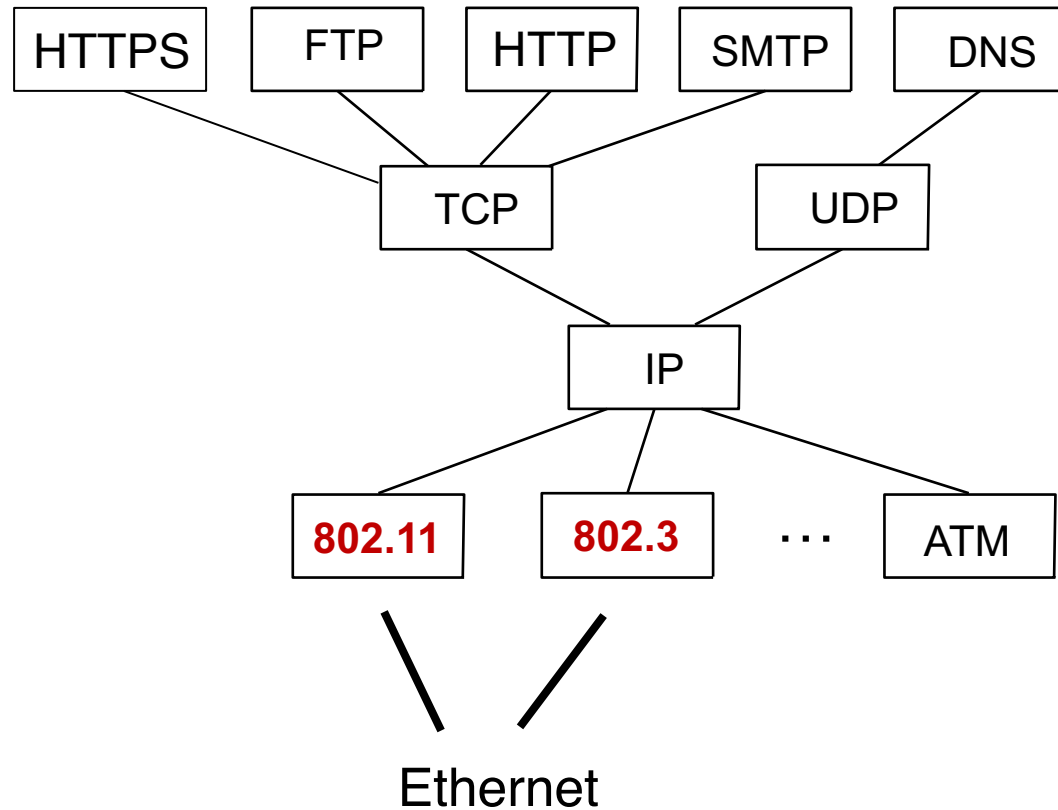
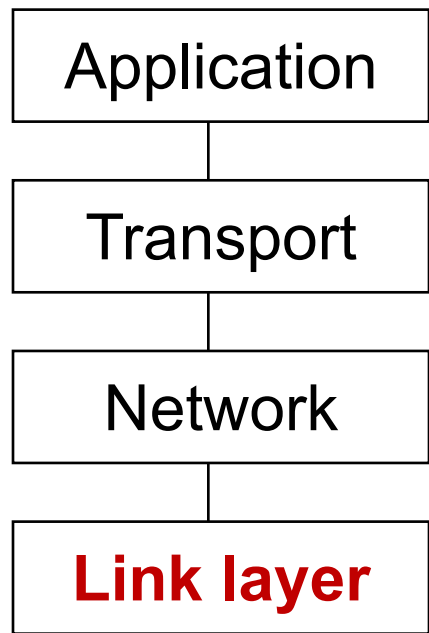
CS 352, Lecture 14

<http://www.cs.rutgers.edu/~sn624/352-S19>

Srinivas Narayana

(heavily adapted from slides by Prof. Badri Nath and the textbook authors)

The Link layer

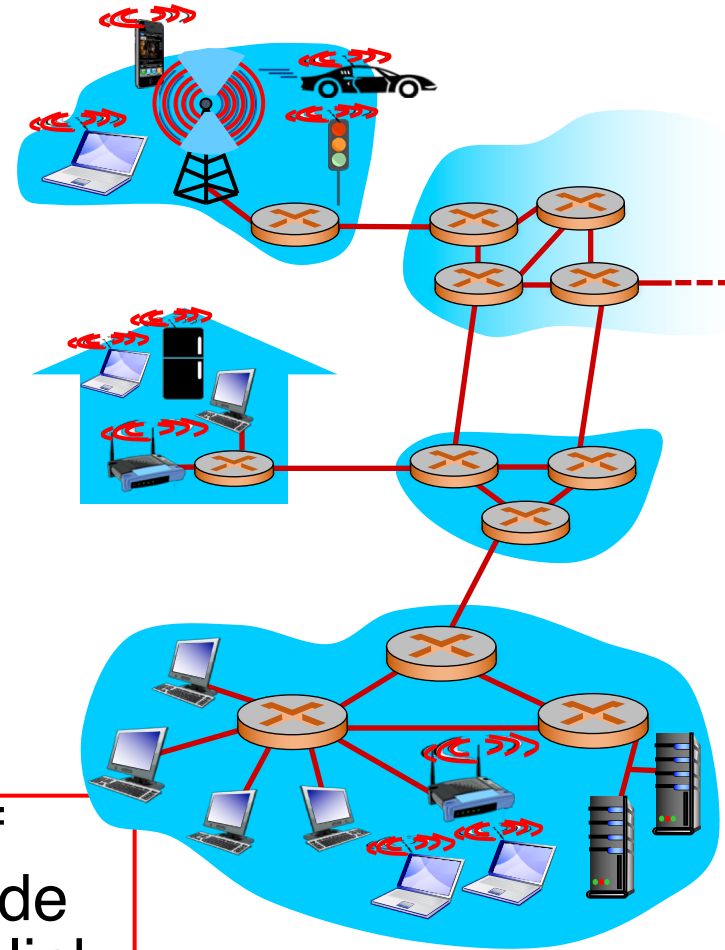


Link layer: introduction

terminology:

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

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



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:

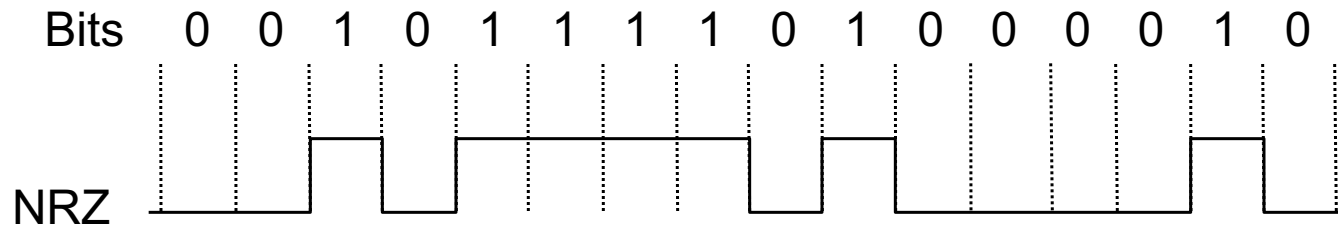
- trip from Princeton to Lausanne
 - limo: Princeton to JFK
 - plane: JFK to Geneva
 - train: Geneva to Lausanne
- tourist = **datagram**
- transport segment = **communication link**
- transportation mode = **link layer protocol**
- travel agent = **routing algorithm**

Link layer services

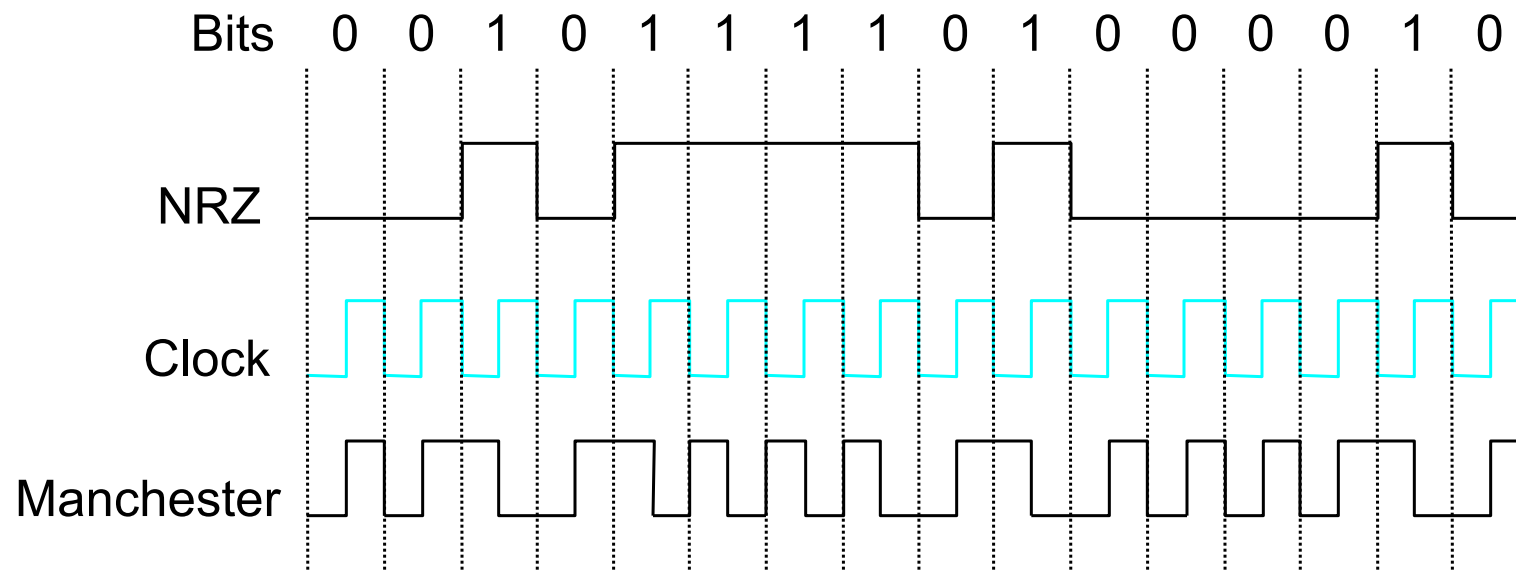
- *Encoding*
 - convert bits to signals and recover bits from received signals
- *framing, link access:*
 - encapsulate datagram into frame, adding header, trailer
 - channel access if shared medium
 - “MAC” addresses used in frame headers to identify source, destination
 - different from IP address!
- *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?

Encoding

- 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)
 - Problem: consecutive 1s and 0s, noise levels



Encodings (cont'd)



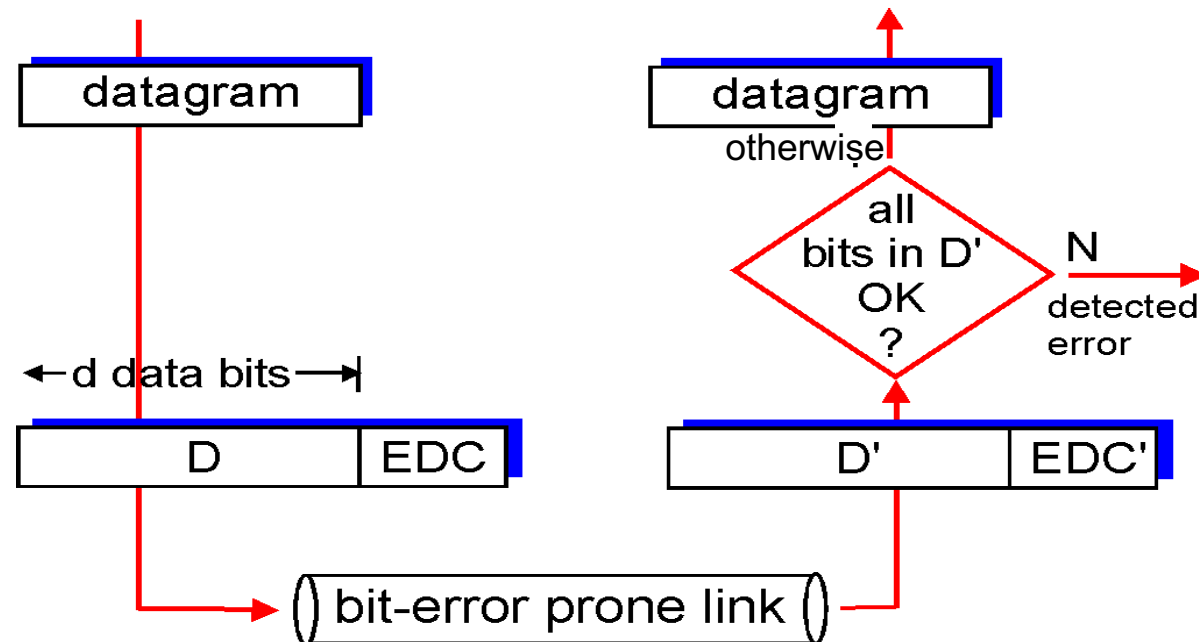
- Manchester encoding: +ve transition \rightarrow 0; -ve transition \rightarrow 1
- $\text{XOR}(\text{bit}, \text{clock})$

Error detection

EDC = Error Detection and Correction bits (redundancy)

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

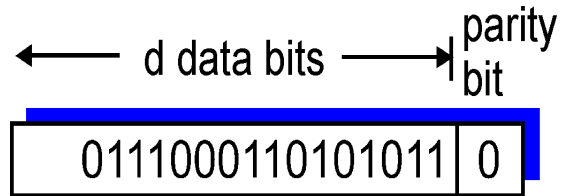
- Error detection not 100% reliable!
 - protocol may miss some errors, but rarely
 - larger EDC field yields better detection and correction



Parity checking

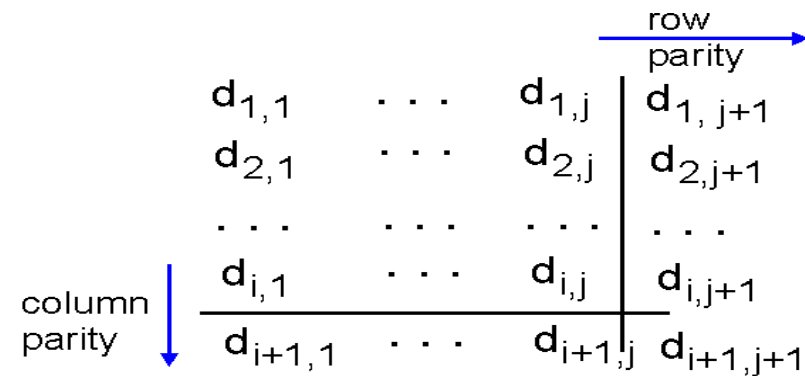
single bit parity:

- detect single bit errors



two-dimensional bit parity:

- detect and correct single bit errors



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

no errors

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

parity error

*correctable
single bit error*

Internet checksum (review)

goal: detect “errors” (e.g., flipped bits) in transmitted packet (note: used at transport layer only)

sender:

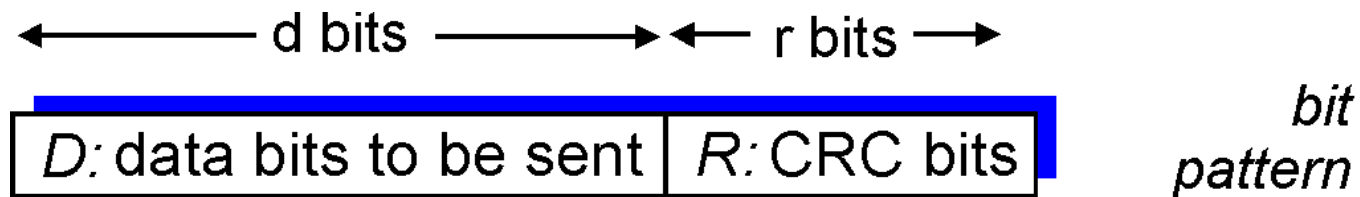
- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP checksum field

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?*

Cyclic redundancy check

- more powerful error-detection coding
- 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:

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

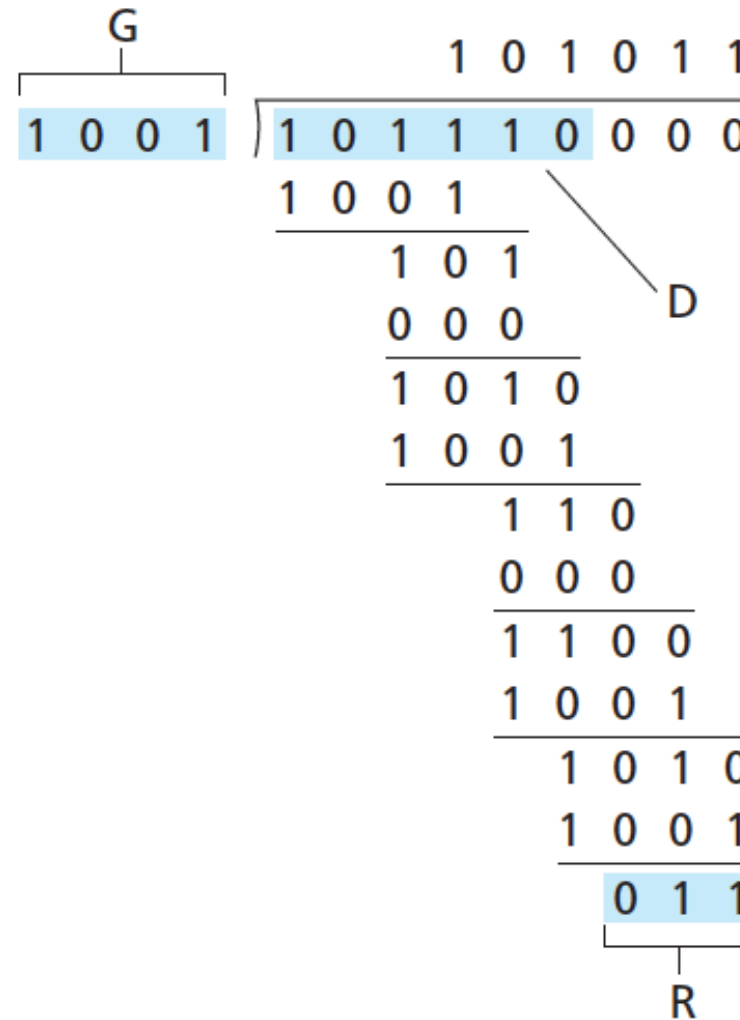
equivalently:

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

equivalently:

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

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



ARP

How to get a MAC address for an IP address?

ARP: Address Resolution Protocol

- By default, NICs only pass on packets destined to their destination MAC address to the higher layers
- In a broadcast-based LAN, each source needs to know its destination's MAC address
- After a packet reaches a router, the link layer header needs to be added to reflect the destination host on that link
- ARP returns a link layer address when given an Internet address
- Communication requires IP → MAC address translation

ARP packet format

Internet Protocol (IPv4) over Ethernet ARP packet		
Octet offset	0	1
0	Hardware type (HTYPE)	
2	Protocol type (PTYPE)	
4	Hardware address length (HLEN)	Protocol address length (PLEN)
6	Operation (OPER)	
8	Sender hardware address (SHA) (first 2 bytes)	
10	(next 2 bytes)	
12	(last 2 bytes)	
14	Sender protocol address (SPA) (first 2 bytes)	
16	(last 2 bytes)	
18	Target hardware address (THA) (first 2 bytes)	
20	(next 2 bytes)	
22	(last 2 bytes)	
24	Target protocol address (TPA) (first 2 bytes)	
26	(last 2 bytes)	

Hardware type: ex: Ethernet (1)

Hardware address length: 6 octets

Protocol Type: ex: IPv4 0x0800
(requesting IPv4 addr)

Protocol address length: 4 octets

Opcode ARP request: 1

Opcode ARP reply: 2

ARP operation

