Computer Networks
COL 334/672

Link Layer

Tarun Mangla

Slides adapted from KR

Sem 1, 2024-25

Quiz on Moodle

Password: wattlebird

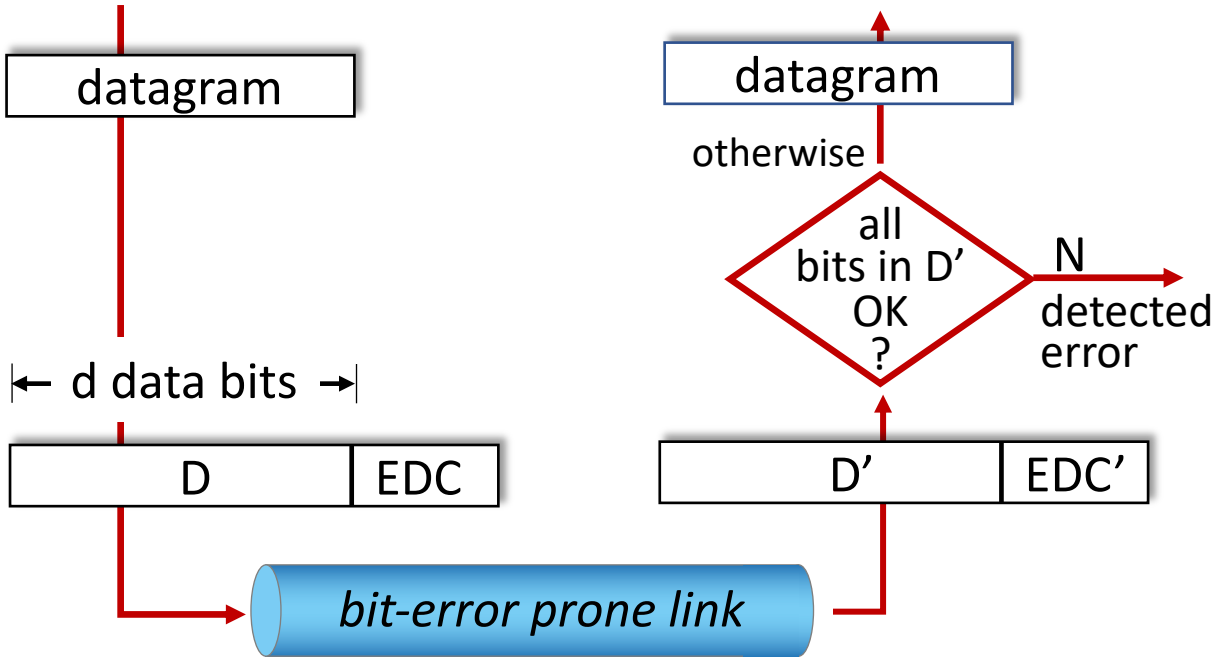
Link Layer: Services

- Framing
- **Error detection**
- Reliability
- Link access

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

Goal: maximize probability of detecting errors using only a small number of redundant bits

Cyclic Redundancy Check

$M(x)$: $1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0 \leftarrow 1011$

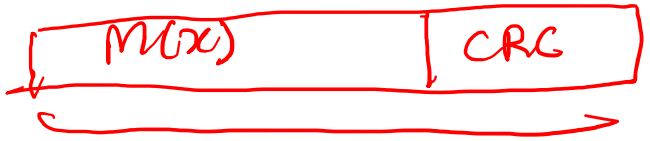
$n+1$ - bit message as a polynomial of degree n

$C(x)$: divisor polynomial of degree k

$P(x)$: degree $n+1+k$

$C(x) \mid P(x)$

↳ exactly divides



$P(x)$

↓

How do you generate this?

Receiver :

$C(x) \mid P'(x)$

Cyclic Redundancy Check (CRC)

- Based on *finite fields*
- A message of *n+1*-bits can be represented as polynomial of degree n
- Consider:
  - M(x), a n+1 bits message to be sent
  - C(x), a divisor polynomial of degree k that is known to both sender and receiver
- Key Idea: Send P(x), n+1+k bits such that C(x) divides P(x)
  - At the receiver, if P'(x) is exactly divisible by C(x) then less likelihood of error, otherwise there is error
- How do you construct P(x) using M(x)?

Some facts [for this course!]

- Any polynomial $B(x)$ can be divided by a divisor polynomial $C(x)$ if $B(x)$ is of higher degree than $C(x)$
- Any polynomial $B(x)$ can be divided once by a divisor polynomial $C(x)$ if $B(x)$ is of the same degree as $C(x)$
- The remainder obtained when $B(x)$ is divided by $C(x)$ is obtained by performing the exclusive OR (XOR) operation on each pair of matching coefficients

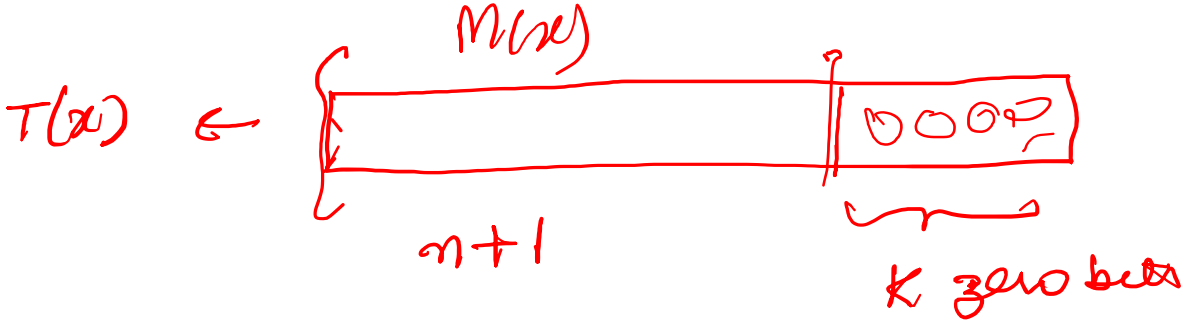
x^3+1 by x^3+x^2+1

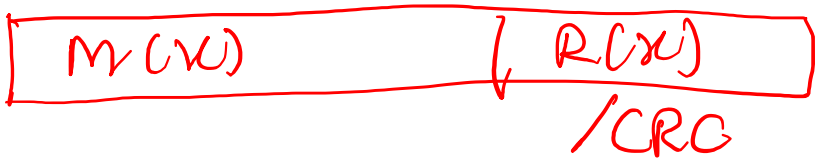
1001 1101

1101) 1001 (1
1101
0100

Algorithm to Obtain CRC Bits

1. Multiply $M(x)$ by x^k; that is, add k zeros at the end of the message. Call this zero-extended message $T(x)$.
2. Divide $T(x)$ by $C(x)$ and find the remainder.
3. Subtract the remainder from $T(x)$.





R(x) : K−1 degree or K bits

P(x) = T(x) − R(x)

T(x) = C(x) · G(x) + R(x)

T(x) − R(x) = C(x) · G(x)

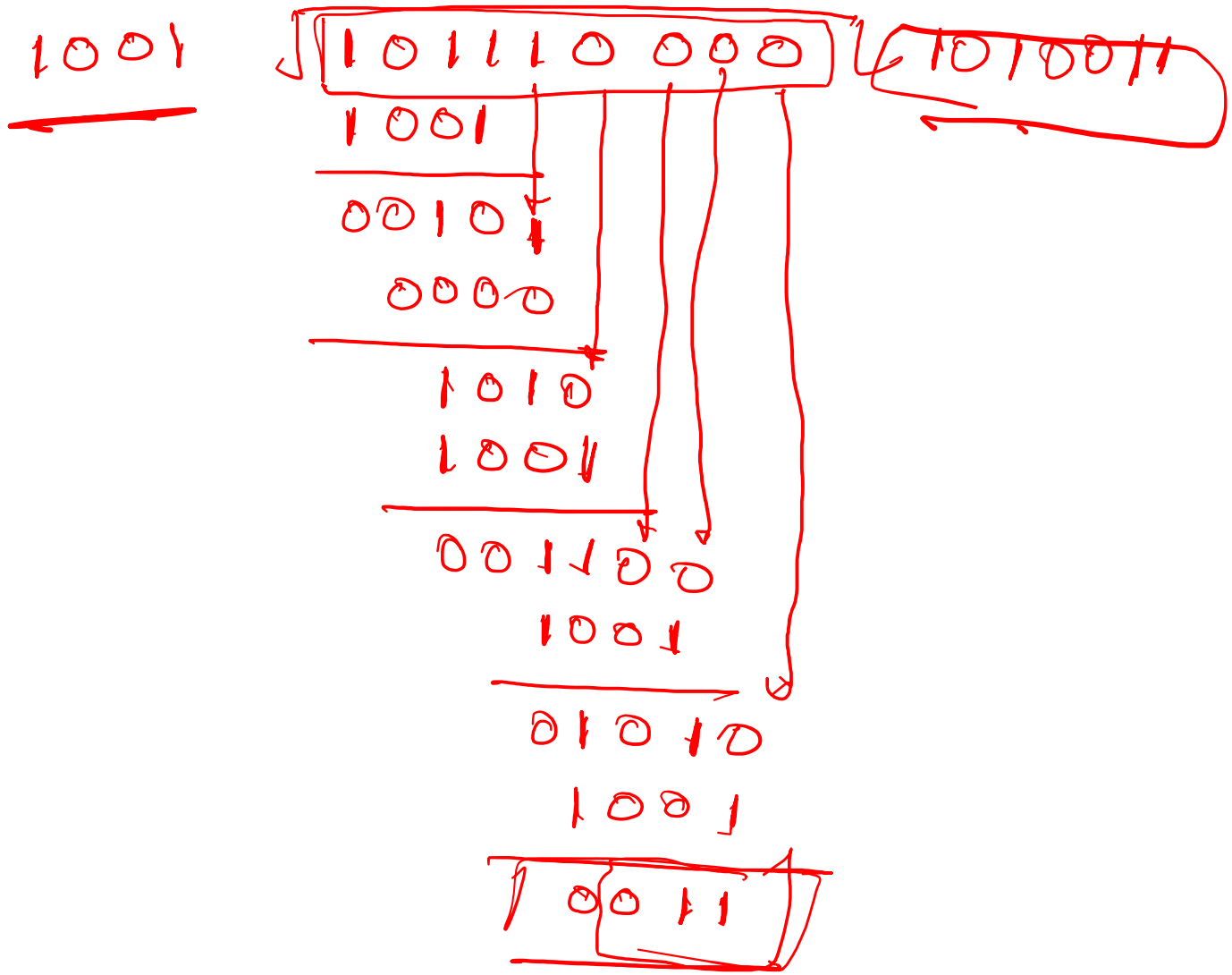
Cyclic Redundancy Check (CRC): Example

- M(x) = 101110
- C(x) = 1001
- What is P(x)?

$T(x) = M(x) \cdot x^3$

$P(x) = T(x) - R(x)$

$= 101110011$



Cyclic Redundancy Check (CRC)

$P(x) + E(x) = P'(x)$

- How to pick C(x)?
  - Transmitted message: P(x) + E(x)
  - For errors to go undetected, E(x) should be divisible by C(x)
  - Pick C(x) such that above is unlikely to happen for common errors
  - Example, all single-bit errors, as long as the x^k and x^0 terms in C(x) have nonzero coefficients

1101 + 0001

P(x)/C(x) + E(x)/C(x)

Claim :

$E(x) = x^i$ where $i \in \{0, \ldots n-k-1\}$

Cyclic Redundancy Check (CRC)

- How to pick C(x)?
  - Transmitted message: P(x) + E(x)
  - For errors to go undetected, E(x) should be divisible by C(x)
  - Pick C(x) such that above is unlikely to happen for common errors
  - Example, all single-bit errors, as long as the x^k and x^0 terms in C(x) have nonzero coefficients

- Ethernet protocol
  - Uses a 32-bit error check

$$\text{CRC-32} = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

- Where is CRC implemented? Hardware

Link Layer: Services

- Framing
- Error detection
- **Reliability**
- Link access

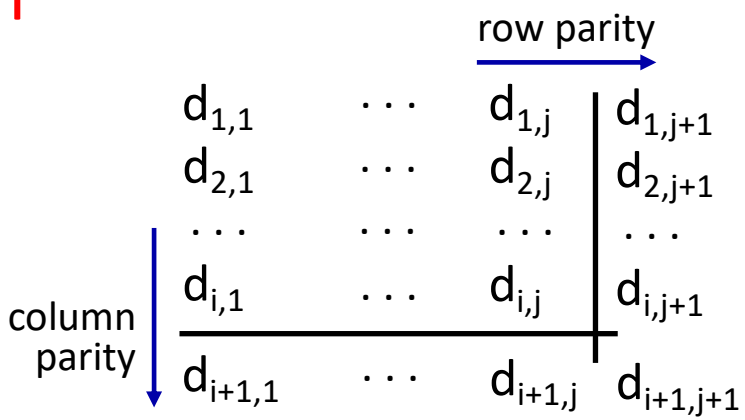
Reliability

- Error correction codes
- Acknowledgements and timeouts or Automatic Repeat request (ARQ)

Error correction code

- Also known as Forward Error Correction
- Using 2D parity

  Can detect *and* correct errors (without retransmission!)
  - detect *and correct* single bit errors
- Always useful?
  - When cost of retransmissions are high
  - When there are frequent bit errors



no errors:

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

detected and correctable single-bit error:

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

parity error

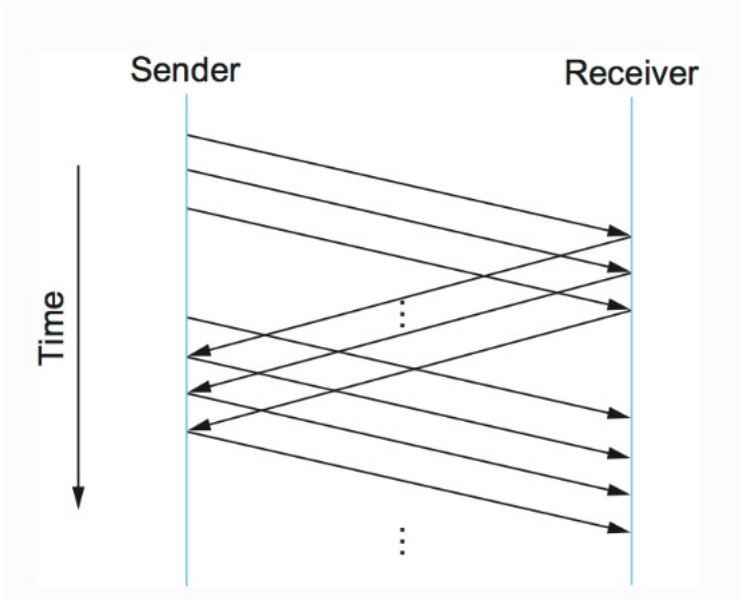
ARQ Protocol: Stop and Wait

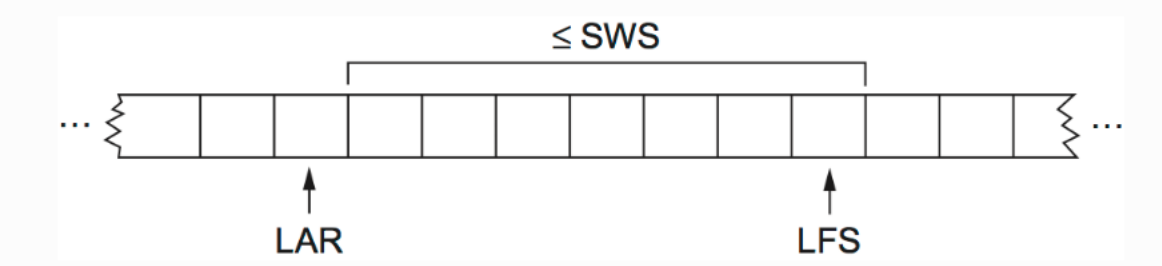
- Transmit one frame, wait for an acknowledgement
  - If no ack and timer expires, resend

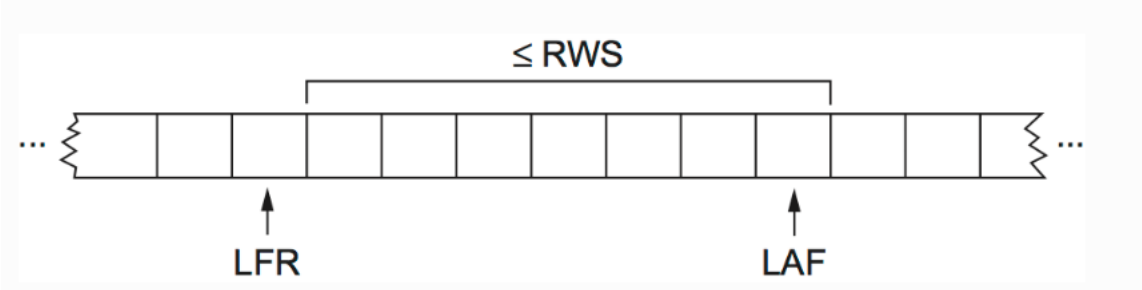
Stop and Wait

- Transmit one frame, wait for an acknowledgement
  - If no ack and timer expires, resend
- How to handle duplicate frames?
  - Sequence numbers for duplicate frames
- Any limitation?
  - Under-utilization of link
  - Example, 4 Mbps link, RTT – 10ms, Frame size – 1 KB
  - How to achieve full-link utilization?
    - Bandwidth delay product

Sliding Window Protocol







Link Layer: Services

- Framing
- Error detection
- Reliability
- **Next class: link access**

Attendance