Digital Data Communication Techniques

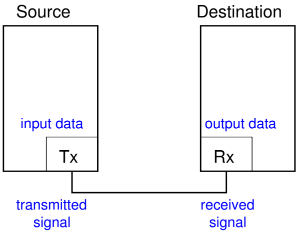
ITS323: Introduction to Data Communications
CSS331: Fundamentals of Data Communications

Sirindhorn International Institute of Technology
Thammasat University

Prepared by Steven Gordon on 3 August 2015
ITS323Y15S1L05,
Steve/Courses/2015/s1/its323/lectures/digital-data-communication-techniques.tex, r3920

Contents

Challenges with Link Communications



- How to convert information into transmittable signals?
- What are the characteristics of signals?
- What transmission media to use?
- How to efficiently encode data as signals?
- How to know who is at other end?
- How to deal with errors?
- How to share media amongst two or more transmitters?

Physical and Data Link Layer

- Researchers, designers, standards, implementations often separate functionality into layers

  Physical Converting data (e.g. bits) into signals to be sent across the link

  Data Link Ensuring link is ready for data transmission, reliable/efficient transmission of data

- See "Networks and Protocol Architectures" topic

Digital Data

- Many communication systems today carry digital data
  - Analog data often converted to digital: voice, video
  - Analog or digital signals
- Challenges for digital data communications:
  - How to split data up?
  - How to deal with errors?
  - How to deal with different types of devices?
- Solutions are often independant of how physical signals transmitted: Data Link layer

Contents

Framing

- Communication protocols group data into separate pieces
  - What is a protocol? Rules to define how two or more entities communicate, including format of messages
  - Why group into pieces? faster recovery from errors, fairer sharing of medium amongst multiple users, . . .
- At the data link layer the pieces commonly called frames
- (See lesson on Packets)
- Information in a frame often separated into parts:

Header	control information at start of frame; used to support protocol operation
Payload	actual data
Trailer	control information at end of frame; used to support protocol operation

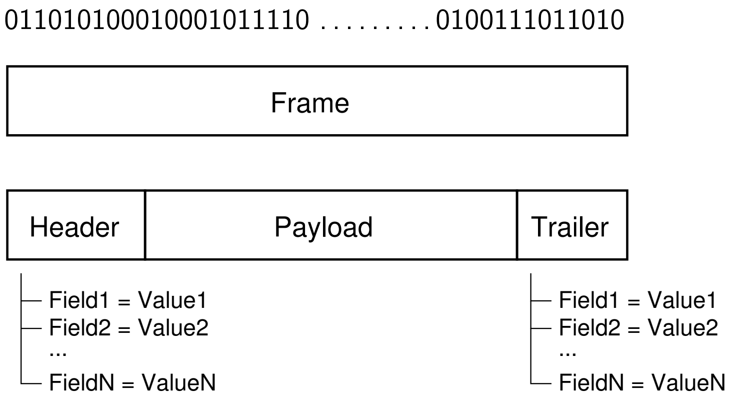
- Not all parts in all frame, e.g. Header + Payload; Header + Payload + Trailer; Header only

Frame Header (and Trailer)

What is Purpose of Header?

- Contains information to support protocol operation
- Sender includes information in header so receiver can correctly process the data and optionally respond
- Information often split into fields; each field has a value
- Number, meaning and size of fields defined in standard
  - IEEE 802.11 defines wireless LAN frame header and trailer fields
- Many protocols have default, fixed size header, with optional extra fields
  - IEE 802.11 MAC Data: typically 24 byte header and 4 byte trailer; other sizes possible

General Frame Structure



Frame Header (and Trailer)

Example Header Fields

- Source and destination addresses, e.g. MAC address
- Frame, payload, header lengths
- Sequence numbers, e.g. data sequence, ACK number
- Protocol version
- Checksums, error detection codes
- Frame types, e.g. DATA, ACK, Beacon
- Flags
  - Single bit values
  - 1: flag is set/true, e.g. feature is on
  - 0: flag is unset/false, e.g. feature is off

Contents

Performance Metrics

- Metrics: Ways to measure the performance of communication systems
- How do we use metrics?
  - Measure the actual performance of real systems
  - Calculate/estimate to predict performance of planned systems
- Represented using different statistics:
  - Instantaneous
  - Average (mean) over some time
  - Maximum (peak), minimum, standard deviation, variance, . . .
- Some metrics we have seen already: bandwidth (Hz), SNR (dB), data rate/capacity (b/s)
- Following slides show common metrics in digital data communications

Data Rate

Definition

Rate at which data is delivered from one point to another

Other/Related Names

Bit rate, capacity, signalling rate, bandwidth

Units

bits per second

Examples

- My computer LAN card can send 100Mb every second; all bits arrive at destination: Data rate = 100Mb/s

Delay

Definition

Time it takes to get data from one point to another

Other/Related Names

Latency; Response time, Round Trip Time

Units

seconds

Examples

- I send an email at 10:00am; it arrives at destination at 10:03am: Delay = 3 minutes
- At time 1.4s I click on a webpage link; at time 2.6s the webpage is fully displated on my browser: Response Time = 1.2s

Error Rate

Definition

Fraction of data sent that doesn't get delivered to destination

Other Names

Bit Error Rate (BER), Frame Error Rate (FER), Packet Error Rate (PER), Loss rate

Units

none (fraction, percentage)

Examples

- I send a copy of an email to 100 students; 5 students do not receive the email: Error rate $= 0.05 = 5\%$
- For every 1,000 bits sent across a link, on average 23 bits arrive in error: BER $= 0.023 = 2.3\%$

Overhead

Definition

Amount of additional data needed in order to deliver useful data

Other Names

-

Units

bits

Examples

- For every 8 bits of data, a 2-bit parity check is added: Overhead = 2b
- A packet contains 1000B of data, a 25B header and 25B trailer: Overhead = 50B

Throughput

Definition

Rate at which useful data (payload) is delivered to destination

Other Names

Goodput, Bandwidth

Units

bits per second

Example

- Downloading a 12MB file from website takes 26 seconds: Throughput = 6Mb/s
- WiFi link has data rate of 54Mb/s. For every 500 Bytes of data sent, there is additional 200 Bytes of overhead plus 20us spent not sending. Throughput = 32.3Mb/s

Efficiency

Definition

Fraction of time spent using a resource for intended purpose

Other Names

Utilization

Units

none (fraction, percentage)

Example

- I pay 1000 Baht per month for 10Mb/s home Internet. On average, each month I download at 2Mb/s: Efficiency = 0.2 = 20%
- WiFi link has data rate of 54Mb/s, but throughput of 20Mb/s: Efficiency = 0.37 = 37%
- For every 1000B of data sent, there is an overhead of 200B: Efficiency = 0.83 = 83%

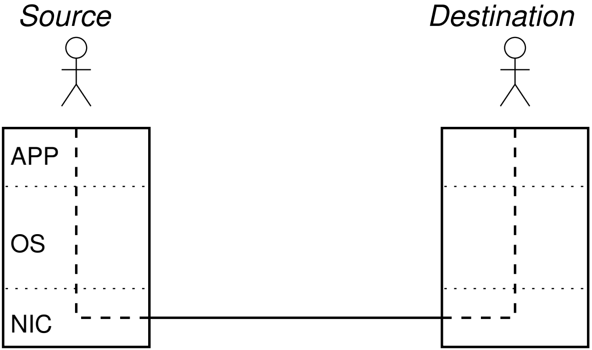
Performance Examples

Contents

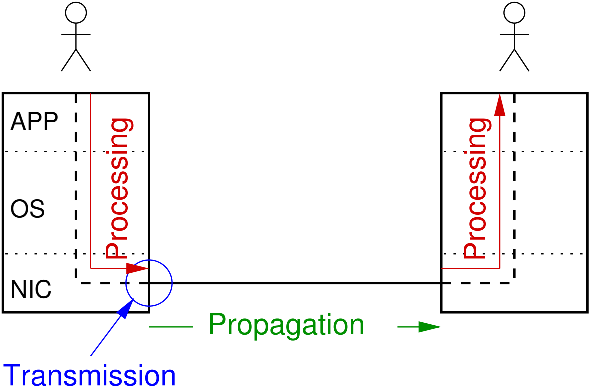
Delay

- Time it takes to get data from one point to another
- Delay is additive
- Four components that contribute to total delay:
  1. Transmission delay: time to transmit data on to link
  2. Propagation delay: time for a signal element (or bit) to propagate across link
  3. Processing delay: time for device to process data
  4. Queuing delay: time data spent waiting in queue (memory) inside device

Delay Components in a Link



Delay Components in a Link



Determining Delay

Transmission Delay

- Number of bits to send, b [bits]
- Link data rate, r [bits per second]
- Transmission delay, $trans = \frac{b}{r}$

Propagation Delay

- Link distance, d [metres]
- Speed of signal propagation, s [metres per second]
- Propagation delay, $prop = \frac{d}{s}$
  - Unless otherwise stated, $s = c = 3 \times 10^8$ m/s

Determining Delay

Processing Delay

- Depends on amount of data to process, software implementation, computer hardware, and other activities of computer
- Often very small compared to transmission and propagation delay
- Unless otherwise stated, assume $proc = 0$ s

Queuing Delay

- Depends on amount of data arriving from other users and leaving device, and queuing scheme (e.g. FIFO, priority)
- Can be significant in large networks, e.g. the Internet
- Unless otherwise stated, assume $queue = 0$ s

Delay Examples

Contents

Dealing with Errors

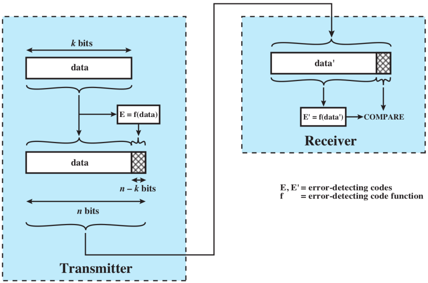
- Transmission impairments can lead to bit errors
- Error types at receiver:
  - One or more bit errors in payload (damaged frame)
  - One or more bit errors in header/trailer (damaged frame)
  - Frame not received (lost frame)
  - Frame received out-of-order
- Error detection
  - Attach extra information to data (in header or trailer) to allow receiver to check if received data is correct (Error Detection)
  - Include sequence numbers in header to identify if frames received in correct order (ARQ)
- Error correction
  - Attach extra information or transform data to allow receiver to check and correct bit errors (Forward Error Correction)
  - Receiver asks transmitter to re-transmit lost/damaged frame (ARQ)

Error Detection Example: Odd-Parity Check

- Odd-parity check: append parity bit to block of data; resulting set of bits has odd number of ones
- Receiver detects an error if receiver bits has unexpected number of ones (transmitter and receiver both know parity scheme being used)
- Assume character *S* is to be sent using odd-parity check. What is transmitted? What happens if the last bit is corrupted? What about the last two bits? What is the overhead?

Error Detection Concept

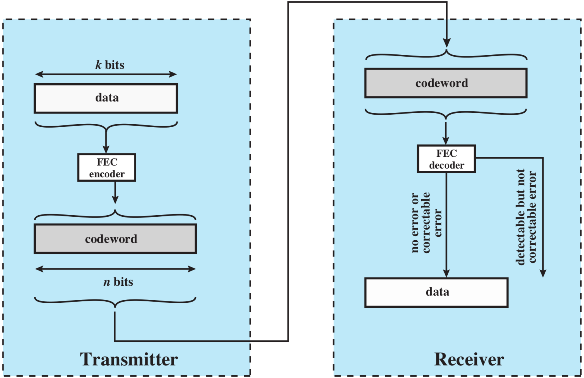
- Transmitter adds extra information to transmitted data, i.e. an error–detecting code
- Receiver recalculates the error–detecting code from received data; compares to received error-detecting code
- If the same, good. If not, then error (in data or code). Still a chance that an error is not detected



- Detection capability depend on algorithm & code length
- Cyclic Redundancy Check (CRC) very common

Forward Error Correction

- Sender sends a codeword (instead of data); codeword chosen such that if error detected, receiver can correct the error without retransmission
- Depending on encoding scheme and pattern of errors, receiver may: detect and correct errors; detect, but not correct errors; not detect errors



Example: FEC with Hamming Distance

Hamming Distance

- Number of bits of two n-bit sequences that differ
- $v_1 = 011011$, $v_2 = 110001$: $d(v1, v2) = 3$

Example FEC Encoder

- 2-bits of data mapped to 5-bit codeword $(k = 2, n = 5)$

Data	*Codeword*
00	00000
01	00111
10	11001
11	11110

- If received codeword invalid, assume valid codeword that is unique minimum Hamming distance from received codeword was transmitted

Example: FEC with Hamming Distance

What does the receiver do in the following cases? Are errors detected/corrected? What is the efficiency?

1. Data to send: 01; no transmission error
2. Data to send: 01; 3rd bit transmitted is in error
3. Data to send: 01; 1st and 4th bit transmitted in error

Contents

Flow Control

- What if transmitter sends frames too fast for receiver to process?
- Receiver will save frames in memory (buffer) and eventually drop frames
- Overflowing a receiver is bad due to lost frames
- Solution: receiver controls rate at which transmitter sends
- Flow control protocols closely related to ARQ (next topic)