Chapter 5 part 1

LINK LAYER

Computer Networks
Timothy Roscoe
Summer 2007
Overview - today

• Context
• Link layer services
  – Framing
  – Error detection and correction
  – Data Link layer protocols
• Multiple access protocols and LANs
  – Channel Partitioning
  – “Taking turns”
  – Random Access
**Link Layer: setting the context**

- *two* physically connected devices
  - host-router, router-router, host-host
- unit of data is called frame

![Diagram of network layers and components]

Diagram showing the layers of the network model, with arrows indicating the flow from application to transport, network, and link layers, down to the physical layer, and back up to the frame structure.
Link Layer: setting the context
Link Layer: Implementation

- Link layer implemented in “adapter” (a.k.a. NIC)
  - e.g., PCMCIA card, Ethernet card
  - typically includes: RAM, DSP chips, host bus interface, and link interface
Link Layer Services

• Framing, link access
  – encapsulate datagram into frame, adding header, trailer
  – implement channel access if shared medium
  – ‘physical addresses’ used in frame headers to identify source, destination
    • different from IP address!

• Error Detection
  – errors caused by signal attenuation, noise
  – receiver detects presence of errors:
    • receiver signals sender for retransmission or drops frame

• Error Correction
  – receiver identifies and corrects bit error(s) without resorting to retransmission
Link Layer Services (more)

• Flow Control
  – pacing between sender and receiver

• Half-duplex and full-duplex
  – with half duplex, nodes at both ends of link can transmit, but not at same time

• Reliable delivery between two physically connected devices:
  – we learned how to do this in chapter 3
  – seldom used on low error link (fiber, some twisted pair)
  – wireless links: high error rates
  • Q: why both link-level and end-end reliability?
Aside: The End-to-End Argument

“The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the end points of the communication system.

Therefore, providing that questioned function as a feature of the communication system itself is not possible.

(Sometimes an incomplete version of the function provided by the communication system may be useful as a performance enhancement.)”

“End-to-end arguments in system design”, J.H.Saltzer, D.P. Reed and D.D. Clark.

ACM Transactions in Computer Systems
E2E and reliability

Something to ponder:

• Reliable delivery is a function
• Does reliable link-level transmission provide end-to-end reliability?

• Can it be worse than no link-level reliability?
• When is it a good thing?
A character stream.  
(a) Without errors.  (b) With one error.
Framing: bit-stuffing

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 1 0 1 1 0 0 1 0

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

Bit stuffing
(a) The original data.
(b) The data as they appear on the line.
(c) The data as they are stored in receiver’s memory after destuffing.
Framing: byte stuffing

(a) A frame delimited by flag bytes.
(b) Four examples of byte sequences before and after stuffing.
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

```
0111000110101011 0
```

d data bits → parity bit

### Two Dimensional Bit Parity
Detect *and correct* single bit errors

```
d_{1,1} ··· d_{1,j}
d_{2,1} ··· d_{2,j}
··· ··· ··· ···
d_{i,1} ··· d_{i,j}
d_{i+1,1} ··· d_{i+1,j}
```

column parity

```
\( \begin{array}{c}
101011 \\
111100 \\
011101 \\
101010 \\
\end{array} \)  \rightarrow  \( \begin{array}{c}
101011 \\
101000 \\
011101 \\
101010 \\
\end{array} \)
```

*no errors*

```
\( \begin{array}{c}
101011 \\
111100 \\
011101 \\
101010 \\
\end{array} \)  \rightarrow  \( \begin{array}{c}
101011 \\
101000 \\
011101 \\
101010 \\
\end{array} \)
```

*parity error*

```
\( \begin{array}{c}
101011 \\
111100 \\
011101 \\
101010 \\
\end{array} \)  \rightarrow  \( \begin{array}{c}
101011 \\
101000 \\
011101 \\
101010 \\
\end{array} \)
```

*correctable single bit error*
## Error correcting codes

<table>
<thead>
<tr>
<th>Char.</th>
<th>ASCII</th>
<th>Check bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>1001000</td>
<td>00110010000</td>
</tr>
<tr>
<td>a</td>
<td>1100001</td>
<td>10111001001</td>
</tr>
<tr>
<td>m</td>
<td>1101101</td>
<td>11101010101</td>
</tr>
<tr>
<td>m</td>
<td>1101101</td>
<td>11101010101</td>
</tr>
<tr>
<td>i</td>
<td>1101001</td>
<td>01101011001</td>
</tr>
<tr>
<td>n</td>
<td>1101110</td>
<td>01101010110</td>
</tr>
<tr>
<td>g</td>
<td>1100111</td>
<td>01111001111</td>
</tr>
<tr>
<td>c</td>
<td>0100000</td>
<td>10011000000</td>
</tr>
<tr>
<td>o</td>
<td>1100011</td>
<td>11111000001</td>
</tr>
<tr>
<td>d</td>
<td>1101111</td>
<td>10101011111</td>
</tr>
<tr>
<td>e</td>
<td>1100100</td>
<td>11111001100</td>
</tr>
</tbody>
</table>

Order of bit transmission
Internet checksum

**Goal**
- detect “errors” (e.g., flipped bits) in transmitted segment
- note: used at transport layer only

**Sender**
- treat segment content as sequence of 16-bit integers
- checksum: addition (1’s complement sum) of segment content
- 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?
  More later…
Cyclic Redundancy Check (CRC): Ring

- Polynomials with binary coefficients $b_k x^k + b_{k-1} x^{k-1} + \ldots + b_0 x^0$
- Order of polynomial: $\max i$ with $b_i \neq 0$

- Binary coefficients $b_i$ (0 or 1) form a field with operations “+” (XOR) and “•” (AND).

- Polynomials form ring $R$ with operations “+” and “•”:
  - $(R,+)$ is an abelian group, $(R, \cdot)$ is an associative set,
  - Distributivity: $a \cdot (b+c) = (a \cdot b) + (a \cdot c)$
    and $(b+c) \cdot a = (b \cdot a) + (c \cdot a)$ for $a,b,c \in R$.

- Example: $(x^3+1) \cdot (x^4+x+1)$
  
  $= x^3 \cdot (x^4+x+1) + 1 \cdot (x^4+x+1)$
  
  $= x^7+x^4+x^3 + (x^4+x+1)$
  
  $= x^7+x^3+x+1$
Cyclic Redundancy Check (CRC): Division

- Generator polynomial \( G(x) = x^{16} + x^{12} + x^{5} + 1 \)
- Let the whole frame (D+EDC) be polynomial \( T(x) \)

- Idea: fill EDC (CRC) field such that \( T(x) \mod G(x) = 0 \)
- How to divide with polynomials? Example with \( G(x) = x^{2} + 1 \) (=101)
  \[
  \begin{array}{c}
  11101100 \\
  101 \\
  111 \\
  100 \\
  010 \\
  \end{array}
  \]
  \[
  100 \\
  011 \\
  \]

- Idea: Fill EDC with remainder when dividing \( T(x) \) with \( EDC=00\ldots0 \) by \( G(x) \). Then calculating and testing CRC is the same operation.
CRC calculation in Hardware

- Use cyclic shift register with \( r \) registers, where \( r \) is the order of \( G(x) \)

- Example

\[
G(x) = x^3 + x^2 + 1
\]

⇒ Remainder of the division ends up in the registers
**CRCs: How to chose G(x)?**

- Typical generator polynomial \( G(x) = x^{16} + x^{12} + x^{5} + 1 \)
- Why does \( G(x) \) look like this?

- Let \( E(x) \) be transmission errors: \( T(x) = M(x) + E(x) \)
- \( T(x) \mod G(x) = (M(x) + E(x)) \mod G(x) \)
  \[ = M(x) \mod G(x) + E(x) \mod G(x) \]
- And recall \( M(x) \mod G(x) = 0 \) …
- \( \Rightarrow \) We can detect all transmission errors iff \( E(x) \) is not divisible by \( G(x) \) without remainder
**CRCs: How to choose G(x)?**

- One can show that G(x) of order r can detect
  - all single bit errors if G(x) has 2 or more coefficients
  - all bursty errors (k-bit long 1xxxx1 string) with \( k \leq r \) (note: needs G(x) to include the term 1)
  - Any error with probability \( 2^{-r} \)
**High-level Data Link Control Protocol (HDLC)**

<table>
<thead>
<tr>
<th>Bits</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>≥ 0</th>
<th>16</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>011111110</td>
<td>Address</td>
<td>Control</td>
<td>Data</td>
<td>Checksum</td>
<td>011111110</td>
</tr>
</tbody>
</table>

- Bit-oriented protocol ⇒ bitstuffing for framing
- Checksum for error detection
### HDLC (2)

<table>
<thead>
<tr>
<th>Bits</th>
<th>1</th>
<th>3</th>
<th>1</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0</td>
<td>Seq</td>
<td>P/F</td>
<td>Next</td>
</tr>
<tr>
<td>(b)</td>
<td>1</td>
<td>0</td>
<td>Type</td>
<td>P/F</td>
</tr>
<tr>
<td>(c)</td>
<td>1</td>
<td>1</td>
<td>Type</td>
<td>P/F</td>
</tr>
</tbody>
</table>

Control field of
- (a) An information frame.
- (b) A supervisory frame.
- (c) An unnumbered frame.
(Some) IEEE 802 working groups

- 802.1: Higher-layer LAN Protocols
- 802.2: Logical Link Control
- 802.3: Ethernet
- 802.4: Token Bus (what?)
- 802.5: Token Ring (almost dead)
- 802.6: Metropolitan Area Network (DQDB…)
- 802.11: Wireless LAN (WiFi)
- 802.12: Demand Priority
- 802.14: Cable Modem
- 802.15: Wireless Personal Area Network (Bluetooth)
- 802.16: Broadband Wireless Access (WiMax)
- 802.17: Resilient Packet Ring (SONET without circuits)
- Etc…
Sublayers: Link Control vs. Media Access

(a) Data link layer

- Network layer
- LLC
- MAC

(b) Physical layer

Packet

MAC LLC Packet MAC
Multiple Access Links and Protocols

Three types of “links”

- point-to-point (single wire; e.g. PPP, SLIP)
- broadcast (shared wire or medium; e.g. Ethernet, WLAN)
- switched (e.g. switched Ethernet, ATM)
**Frequency Division Multiplexing (FDM)**

- Separation of the whole spectrum into smaller frequency bands
- A channel gets a certain band of the spectrum for the whole time
  - no dynamic coordination necessary
  - works also for analog signals
    - waste of bandwidth if traffic is distributed unevenly
    - inflexible

- Example: broadcast radio
Time Division Multiplexing (TDM)

- A channel gets the whole spectrum for a certain amount of time
  + only one carrier in the medium at any time
  + throughput high even for many users
    - precise synchronization necessary

- Example: Ethernet
Time/Frequency Division Multiplexing

- Combination of both methods
- A channel gets a certain frequency band for some time
  - protection against frequency selective interference
  - protection against tapping
  - adaptive
    - precise coordination required
- Example: GSM
**Code Division Multiplexing (CDM)**

- Each channel has a unique code
- All channels use the same spectrum at the same time
  - bandwidth efficient
  - no coordination or synchronization
  - hard to tap
  - almost impossible to jam
- lower user data rates
- more complex signal regeneration
- Example: UMTS
- Spread spectrum
- U. S. Patent 2‘292‘387, Hedy K. Markey (a.k.a. Lamarr or Kiesler) and George Antheil (1942)
Cocktail party as analogy for multiplexing

- Space multiplex: Communicate in different rooms

- Frequency multiplex: Use soprano, alto, tenor, or bass voices to define the communication channels

- Time multiplex: Let other speaker finish

- Code multiplex: Use different languages and hone in on your language. The “farther apart” the languages the better you can filter the “noise”: German/Japanese better than German/Dutch. Can we have orthogonal languages?
Media Access Protocols

- Single shared communication channel
- Two or more simultaneous transmissions by nodes: interference
  - only one node can send successfully at a time
- **Media Access Control (MAC) Protocol**
  - distributed algorithm that determines how stations share channel, i.e., determine when station can transmit
  - communication about channel sharing must use channel itself!
  - what to look for in multiple access protocols
    - synchronous or asynchronous
    - information needed about other stations
    - robustness (e.g. to channel errors)
    - performance
Three broad classes

- **Channel Partitioning**
  - divide channel into smaller “pieces” (time slots, frequency)
  - allocate piece to node for exclusive use

- **“Taking turns”**
  - tightly coordinate shared access to avoid collisions

- **Random Access**
  - allow collisions
  - “recover” from collisions

- **Goals:** decentralized, efficient, simple, fair
Wavelength-Division Multiple Access

- Station
  - A: m time slots for control
  - B: n time slots for data plus 1 for status
  - C: B's control channel
  - D: C's control channel

- A's control channel is used by other stations to contact A
- B's control channel
- Used by B to transmit data
- C's control channel
- D's control channel

Time
“Taking Turns” MAC protocols

Polling
- master node “invites” slave nodes to transmit in turn
- Request to Send, Clear to Send messages
- concerns
  - polling overhead
  - latency
  - single point of failure (master)

Token passing (Token Ring)
- control token passed from one node to next sequentially
- token message
- concerns
  - token overhead
  - latency
“Taking Turns” Protocols: Round Robin

- Round robin protocol: station \( k \) sends after station \( k-1 \mod n \)
- If a station does not need to transmit data, then it sends “\( \varepsilon \)”
- There is a maximum message size \( m \) that can be transmitted
- Is this different from token ring protocol?

- Questions
  - How efficient is round robin?
  - What if a station breaks or leaves?
  - Can a new station join?

- All deterministic protocols have these (or worse) problems
  - Try randomized protocols instead!
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 $\rightarrow$ “collision”
• Random access MAC protocol specifies
  – how to detect collisions
  – how to recover from collisions
    • via delayed retransmissions
• Examples of random access MAC protocols:
  – ALOHA and variants (slotted ALOHA, adaptive ALOHA)
  – Backoff protocols (CSMA, CSMA/CD, CSMA/CA)
**Slotted Aloha**

- Time is divided into equal size slots
  - A slot is equal to the packet transmission time
- Node with new arriving packet: transmit at beginning of next slot
- If collision: retransmit packet in future slots with probability $p$, until successful

![Diagram of Slotted Aloha]

- Success (S), Collision (C), Empty (E) slots
**Slotted Aloha (slightly simplified)**

- We assume that the stations are perfectly synchronous.

- In each time slot each station transmits with probability $p$.

\[ P_1 = \Pr[\text{Station 1 succeeds}] = p(1 - p)^{n-1} \]

\[ P = \Pr[\text{any Station succeeds}] = nP_1 \]

maximize $P$:

\[
\frac{dP}{dp} = n(1 - p)^{n-2}(1 - pn) = 0 \quad \Rightarrow \quad pn = 1
\]

then,

\[ P = (1 - \frac{1}{n})^{n-1} \geq \frac{1}{e} \]

- In slotted aloha, a station can transmit successfully with probability at least $1/e$. How quickly can an application send packets to the radio transmission unit? Queuing Theory!
Queuing Theory (Remember Chapter 3?)

- Simplest M/M/1 queuing model (M=Markov):
- Poisson arrival rate $\lambda$, exponential service time with mean $1/\mu$

In our time slot model, this means that the probability that a new packet is received by the buffer is $\lambda$; the probability that sending succeeds is $\mu = 1/e$, for any time slot. To keep the queue bounded we need $\rho = \lambda/\mu < 1$, thus $\lambda < 1/e$.

In the equilibrium, the expected number of packets in the system is $N = \rho/(1-\rho)$, the average time in the system is $T = N/\lambda$. 
Slotted Aloha vs. Round Robin

- Slotted aloha uses not every slot of the channel; the round robin protocol is better in this respect.

+ What happens in round robin when a new station joins? What about more than one new station? Slotted aloha is more flexible.

- Example: If the actual number of stations is twice as high as expected, there is still a successful transmission with probability 27%. If it is only half, 30% of the slots are successful.
Adaptive slotted aloha

- Idea: Change the access probability with the number of stations
- How can we estimate the current number of stations in the system?
- Assume that stations can distinguish whether 0, 1, or more than 1 stations send in a time slot.

- Idea: Try to estimate the number of stations!
  - If you see that nobody sends, increase $p$.
  - If you see that more than one sends, decrease $p$.

- Analysis a little too tough for this course (unfortunately)
Pure (unslotted) ALOHA

- Unslotted Aloha: simpler, no synchronization
- Packet needs transmission
  - send without awaiting for beginning of slot
- Collision probability increases:
  - packet sent at $t_0$ collide with packets sent in $(t_0-1, t_0+1)$
Pure Aloha Analysis

• Partition each slot of size 1 into x “minislots” of size ε (then x = 1/ε)
• Probability to start transmission in minislot is p_ε for each station

\[
P_1 = \Pr[\text{Station 1 succeeds}] = p_\epsilon (1 - p_\epsilon)^{(2x-1)(n-1)}
\]
\[
P = \Pr[\text{any station succeeds}] = n \cdot P_1
\]

P can be maximized by choosing \( p_\epsilon = \frac{\epsilon}{\epsilon + 2n} \), which is \( \frac{\epsilon}{2n} \) for \( \epsilon \to 0^+ \). Then

\[
P = np_\epsilon (1 - p_\epsilon)^{(2x-1)(n-1)} = \frac{\epsilon}{2} \left( 1 - \frac{\epsilon}{2n} \right)^{2n/\epsilon - \ldots} \geq \frac{\epsilon}{2e}
\]

• Since there are x minislots in 1 slot, the success rate of a slot is about 1/2e, that is, half the rate of slotted aloha
Slotted Aloha vs. Pure Aloha

\[ S = \text{throughput} = \text{“goodput”} \]

\[ G = \text{offered load} = np \]

Protocol constrains effective channel throughput!
Demand Assigned Multiple Access (DAMA)

• Channel efficiency only 36% for Slotted Aloha, and even worse for Aloha.

• Practical systems therefore use reservation whenever possible. But: Every scalable system needs an Aloha style component.

• Reservation
  – a sender reserves a future time-slot
  – sending within this reserved time-slot is possible without collision
  – reservation also causes higher delays
  – typical scheme for satellite systems
Example for reservation-based protocol

Distributed Polling
- time divided into slots
- begins with N short reservation slots
  - reservation slot time equal to channel end-end propagation delay
  - station with message to send posts reservation
  - reservation seen by all stations
- after reservation slots, message transmissions ordered by known priority
Backoff Protocols

• Backoff protocols rely on acknowledgements only.
• Binary exponential backoff, for example, works as follows:
  • If a packet has collided $k$ times, we set $p = 2^{-k}$
  Or alternatively: wait from random number of slots in $[1..2^k]$

• It has been shown that binary exponential backoff is not stable for any $\lambda > 0$ (if there are infinitely many potential stations)
  – Intuition: there’s a small chance things get bad. Once they’re bad, they only get worse: more stations waiting $\Rightarrow$ more likelyhood of collisions $\Rightarrow$ more backlog…

• Interestingly when there are only finite stations, binary exponential backoff becomes unstable with $\lambda > 0.568$; Polynomial backoff however, remains stable for any $\lambda < 1$. 


CSMA: Carrier Sense Multiple Access

Idea of CSMA: listen before transmit!

- If channel sensed idle: transmit entire packet
- If channel sensed busy, defer transmission. Two variants
  - Persistent CSMA
    - retry immediately with probability p when channel becomes idle (may cause instability)
  - Non-persistent CSMA
    - retry after random interval

- Human analogy
  1. Don’t interrupt anybody already speaking
**CSMA collisions**

Collisions can occur due to propagation delay: two nodes may not hear each other’s transmission.

A collision occurs when the entire packet transmission time is wasted.

Note the role of distance and propagation delay in determining the collision probability.
CSMA/CD (Collision Detection)

CSMA/CD: carrier sensing, as in CSMA
- collisions detected within short time
- colliding transmissions aborted, reducing channel wastage
- persistent or non-persistent retransmission

* collision detection
  - easy in wired LANs: measure signal strengths, compare transmitted, received signals
  - difficult in wireless LANs: receiver shut off while transmitting

* Human analogy (the polite conversationalist)
  1. Don’t interrupt anybody already speaking
  2. If another starts speaking with you, then back off.
CSMA/CD collision detection
Summary of MAC protocols

- What do you do with a shared medium?
  - Channel Partitioning, by time, frequency or code
    - Time Division, Code Division, Frequency Division
  - Taking Turns
    - polling from a central site, token passing
  - Random partitioning (dynamic)
    - ALOHA, S-ALOHA, CSMA, CSMA/CD
    - carrier sensing
      - easy in some technologies (wire)
      - hard in others (wireless)
    - CSMA/CD used in Ethernet
Next week:

- LAN addressing
- Ethernet!
- Bridges, Hubs, Switches
- PPP
- VLANs
- More philosophy…

- After that: Wireless.