Link layer and LAN Chapter 5

Communication over a link

- Information needs to be converted to a signal appropriate to the link
  - Wired, optics, wireless
  - 1s and Os to appropriate signal levels
  - Bits per second = baud x log₂(levels)

Data Link Layer Functionality

- Convert bits to signals and recover bits from received signals
  - Encoding
- Decide on a minimum unit for sending bits
  - Cannot send bit by bit (too much overhead)
  - Frame creation
- Error detection and/or correction of frames
  - Parity, CRC
- Flow control
  - ARQ, Sliding WINDOW
- Addressing
  - MAC address

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 Os, noise levels

NRZ: 0 0 1 0 1 1 1 1 0 1 0 0 0 1 0
Encodings (cont)

- Manchester encoding: +ve transition -> 0; -ve transition -> 1
- XOR(bit, clock)

<table>
<thead>
<tr>
<th>Bits</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>NRZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manchester</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Framing

- The data unit at the data link layer is called a “frame”
- A frame is a group of bits, typically in sequence
- Issues:
  - Frame creation
  - How many bits (size of frame)
  - Overhead
  - Frame delineation
  - Have meta tags
    - start and stop characters or bit sequence
    - What if the meta tags appear in the message?

Stuffing

- Character stuffing ($, #)
  - $# this prof is good $`
  - $# this prof s$s"ks $`. Meta tag in message
  - $# this prof s$s"ks $`. at sender
  - $# this prof s$s"ks $`. at receiver, remove stuffing

- Bit stuffing: have a unique bit sequence
  - 01111110 this prof is good 01111110
  - 01111110 this prof is 01111110 good 01111110
  - 01111110 this prof is 01111110 good 01111110 -- sender
  - Receiver checks for 5 1s, if next bit is 0 - stuff
  - If next bits are 10 end of frame else error

Error Control

- No physical link is perfect
- Bits will be corrupted
- We can either:
  - detect errors and request retransmission
  - or correct errors without retransmission
Error Detection
- Parity bits
  - For a fixed sequence, add a 1 bit (0/1) to have odd 1s (odd parity) or even 1s (even parity)
  - Extra bits are overhead, multiple bit errors
- Checksum
  - Divide msg into fixed size (16-bit) chunks
  - Add chunks (using 1s complement) and send check sum (complement of total) with message
  - Receiver add msg + checksum
  - Complement (result) = 0 accept else reject
- Polynomial codes or CRC
  - Divide the MSG by polynomial, add R to get CRC bits
  - Receiver : divides MSG + R, check if zero

Addressing
- Hosts need to be identified at the link layer
- MAC address
  - 48 bits unique address (permanent with adapter)
  - 24 bits: manufacturer; 24 bits Serial number
- No relationships between MAC addresses
- hosts connected by a link
  - No grouping or hierarchy possible
- Fixed length address
  - Look up is efficient but table size = number of
    hosts on the network
  - Scaling

Connecting multiple hosts
- All hosts share the same link
  - Simple, need to deal with contention
  - A pair at a time can communicate
- Each pair of hosts connected by separate link
  - Mesh connection required, complex
  - N/2 pairs can simultaneously communicate

Multiple Access Methods (5.3)
- Pure ALOHA
- Slotted ALOHA
- CSMA
  - 1-Persistent CSMA
  - Non-Persistent CSMA
  - P-Persistent CSMA
- CSMA/CD
Pure ALOHA

- Originally developed for ground-based packet radio communications in 1970
- Goal: let users transmit whenever they have something to send

The Pure ALOHA Algorithm

1. Transmit whenever you have data to send
2. Listen to the broadcast
   - Because broadcast is fed back, the sending host can always find out if its packet was destroyed just by listening to the downward broadcast one round-trip time after sending the packet
3. If the packet was destroyed, wait a random amount of time and send it again
   - The waiting time must be random to prevent the same packets from colliding over and over again

Pure ALOHA (cont’d)

- Note that if the first bit of a new packet overlaps with the last bit of a packet almost finished, both packets are totally destroyed.
  - $t$: one packet transmission time
  - Vulnerable period: $2t$

Pure ALOHA (cont’d)

- Due to collisions and idle periods, pure ALOHA is limited to approximately 18% throughput in the best case
- Can we improve this?
Slotted ALOHA

- Slotted ALOHA cuts the vulnerable period for packets from $2t$ to $t$.
- This doubles the best possible throughput from 18.4% to 36.8%.
- How?
  - Time is slotted. Packets must be transmitted within a slot.

The Slotted ALOHA Algorithm

1. If a host has a packet to transmit, it waits until the beginning of the next slot before sending.
2. Listen to the broadcast and check if the packet was destroyed.
3. If there was a collision, wait a random number of slots and try to send again.

CSMA

- We could achieve better throughput if we could listen to the channel before transmitting a packet.
- This way, we would stop avoidable collisions.
- To do this, we need "Carrier Sense Multiple Access," or CSMA, protocols.

Contention Access Methods

- Determine when to transmit, sense the channel.
- CSMA
  - 1-Persistent CSMA
    - Transmit if idle, else wait until idle and then transmit.
  - Non-Persistent CSMA
    - Transmit if idle, else wait for random time, and then repeat.
    - Spreads arrival times.
  - P-Persistent CSMA
    - Transmit with probability $p$ if idle, else wait until idle.
**CSMA/CD**

- In CSMA protocols
  - If two stations begin transmitting at the same time, each will transmit its complete packet, thus wasting the channel for an entire packet time.
- In CSMA/CD protocols
  - The transmission is terminated immediately upon the detection of a collision.
  - \( CD = \text{Collision Detect} \)
- In wired links, transceiver can send and receive simultaneously.

**Ethernet Backoff Algorithm:**

*Binary Exponential Backoff*

- If collision,
  - Choose one slot randomly from \( 2^k \) slots, where \( k \) is the number of collisions the frame has suffered.
  - One contention slot length = \( 2 \times \text{end-to-end propagation delay} \)

This algorithm can adapt to changes in network load.

**Algorithm (cont)**

- If collision...
  - jam for 32 bits, then stop transmitting frame
  - minimum frame is 64 bytes (header + 46 bytes of data)
  - Header 18 bytes: 6 bytes S, 6 bytes D, 2 bytes frame type, 4 bytes checksum
  - delay and try again
    - 1st time: 0 or 51.2us
    - 2nd time: 0, 51.2, or 102.4us
    - 3rd time: 51.2, 102.4, or 153.6us
    - \( n \)th time: \( k \times 51.2us, \text{for randomly selected} \)
    - \( k \in [0, 2^{n-1}] \)
    - give up after several tries (usually 16)
    - exponential backoff

**CSMA/CD and Ethernet**

- **Ethernet:**
  - Short end-to-end propagation delay
  - Broadcast channel
- **Ethernet access protocol:**
  - 1-Persistent CSMA/CD
  - with Binary Exponential Backoff Algorithm
- **Ethernet frame**

```
  Preamble  Source  Destination  Type  Data  Frame Checksum
  0x0800 IP
  0x0806 ARP
  0x8035 RARP
  0x8808 IPv6
```