COMP/ELEC 429/556
Introduction to Computer Networks

Encoding and Framing

Some slides used with permissions from Edward W. Knightly, T. S. Eugene Ng, Ion Stoica, Hui Zhang
What’s Inside the Network?

• Let’s go beyond what we’ve seen at the client server hosts, which are
  – Application level (hostnames, application data formatting)
  – OS level (getaddrinfo, socket, send, recv, sliding window, error detection, etc.)
Let’s Begin with the Most Primitive Network
A Real Data Link Example

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Wire color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$V_{BUS}$</td>
<td>Red (or Orange)</td>
<td>+5 V</td>
</tr>
<tr>
<td>2</td>
<td>D−</td>
<td>White (or Gold)</td>
<td>Data−</td>
</tr>
<tr>
<td>3</td>
<td>D+</td>
<td>Green</td>
<td>Data+</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td>Black (or Blue)</td>
<td>Ground</td>
</tr>
</tbody>
</table>

Voltage signal in the differential pair

Differential decoding

NRZI decoding

Packet format

Start of packet / clock sync
Packet ID (LSB first, 1010 = NAK)
End of packet
Encoding

1. codifying

Tree →

2. sending the message

3. decodifying

Tree →
Framing
Typical Services for Point-to-Point Communications

- **Encoding**
  - How to encode 1’s and 0’s?
- **Framing**
  - How to create frames/packets that have explicit beginning and end?
- **Addressing**
  - How to identify the communication parties?
- **Access control**
  - How to coordinate multiple senders who share one data link?
- **Error detection**
  - How to detect if bits have been flipped accidentally?
- **Error recovery**
  - How to correct detected errors?
- **Flow control**
  - How to ensure sender doesn’t send faster than the receiver can handle?
Bit Stream Encoding

• Specify how bits are represented in the analog signal
• Challenges:
  – Efficiency: ideally, bit rate is maximized
  – Robustness: minimize the chance of mis-interpreting received signal
Assumptions

• We use two discrete signals, high and low, to encode 1 and 0
• There is a clock used to pace transmission; there is a clock used to sample the received signal
• Sender and receiver do not have perfect clocks
  – i.e. clocks can drift!
Non-Return to Zero (NRZ)

• 1 → high signal; 0 → low signal

• Efficiency?
  • Assumption: 1 bit of information conveyed per sampling period is considered 100% efficient

• Robustness?
  • When there is a long sequence of 1’s or 0’s
    Sensitive to clock skew, i.e., difficult to do clock recovery
Non-Return to Zero Inverted (NRZI)

• 1 → make transition; 0 → stay at the same level
• Solve problem of long sequences of 1’s, but not long sequences of 0’s
Manchester

- 1 → high-to-low transition; 0 → low-to-high transition
- Addresses clock recovery problems
- Disadvantage: signal transition rate doubled
  - I.e. useful data rate on same physical medium halved
  - Efficiency of 50%
4-bit/5-bit (100Mb/s Ethernet)

- Goal: address inefficiency of Manchester encoding, while avoiding long periods of low signals
- Solution:
  - Use 5 bits to encode every sequence of 4 bits such that no string of encoded bits will have more than 3 consecutive 0’s
  - Use NRZI to transmit the encoded bits
  - Efficiency is 80%

<table>
<thead>
<tr>
<th>4-bit</th>
<th>5-bit</th>
<th>4-bit</th>
<th>5-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td></td>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>0001</td>
<td></td>
<td>1001</td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td></td>
<td>1010</td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td></td>
<td>1011</td>
<td></td>
</tr>
<tr>
<td>0100</td>
<td></td>
<td>1100</td>
<td></td>
</tr>
<tr>
<td>0101</td>
<td></td>
<td>1101</td>
<td></td>
</tr>
<tr>
<td>0110</td>
<td></td>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>0111</td>
<td></td>
<td>1111</td>
<td></td>
</tr>
</tbody>
</table>
Framing

- Specify how **blocks** of data are transmitted between two nodes connected on the same physical media
- Decide when a frame starts/ends
Byte-Oriented Protocol: Sentinel Approach (e.g. PPP)

- STX – start of text
- ETX – end of text
  - STX and ETX can be the same
- Problem: what if ETX appears in the data portion of the frame?
- Solution
  - If ETX appears in the data, introduce a special character DLE (Data Link Escape) before it
  - If DLE appears in the text, introduce another DLE character before it
  - Like in C programming, “Say \\
Hello\”, (\ is the escape character)
Bit-Oriented Protocol (e.g. HDLC)

- Both start and end sequence can be the same
  - E.g., 01111110 in HDLC (High-level Data Link Protocol)
- Sender: in data portion inserts a 0 after five consecutive 1s
  - “Bit stuffing”
- Receiver: when it sees five 1s makes decision on the next two bits
  - If next bit 0 (this is a stuffed bit), remove it
  - If next bit 1, look at the next bit
    - If 0 this is end-of-frame (receiver has seen 01111110)
    - If 1 this is an error, discard the frame (receiver has seen 01111111)
Byte Counting Approach

• Instead of using an end of frame sequence...
• Sender: insert the length of the data (in bytes) at the beginning of the frame, i.e., in the frame header
• Receiver: extract this length and decrement it every time a byte is read. When this counter becomes zero, we are done
A Real Data Link Example

USB 1.x/2.0 standard pinout

<table>
<thead>
<tr>
<th>Pin</th>
<th>Name</th>
<th>Wire color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VBUS</td>
<td>Red (or Orange)</td>
<td>+5 V</td>
</tr>
<tr>
<td>2</td>
<td>D−</td>
<td>White (or Gold)</td>
<td>Data−</td>
</tr>
<tr>
<td>3</td>
<td>D+</td>
<td>Green</td>
<td>Data+</td>
</tr>
<tr>
<td>4</td>
<td>GND</td>
<td>Black (or Blue)</td>
<td>Ground</td>
</tr>
</tbody>
</table>

Voltage signal in the differential pair

Differential decoding

NRZI decoding

Packet format

(transition for 0, stuff a 0 after 6 1’s)