Lecture 4:
Layers & Framing

CSE 123: Computer Networks
Alex C. Snoeren

Proj 1 out Today, due 11/05
Lecture 4 Overview

- Finish encoding schemes
  - Manchester, NRZ, NRZI, etc.

- Layering
  - Focus on Link Layer

- Framing
  - Stuffing
Non-Return to Zero (NRZ)

- Signal to Data
  - High $\Rightarrow$ 1
  - Low $\Rightarrow$ 0

- Comments
  - Transitions maintain clock synchronization
  - Long strings of 0s confused with no signal
  - Long strings of 1s causes *baseline wander*
    » We use average signal level to infer high vs low
  - Both inhibit clock recovery

Bits: 0 0 1 0 1 1 1 1 0 1 0 0 0 0 1 0
NRZ:  

Courtesy Robin Kravets
NRZ Inverted (NRZI)

- Signal to Data
  - Transition $\Rightarrow 1$
  - Maintain $\Rightarrow 0$

- Comments
  - Solves series of 1s, but not 0s
**Manchester Encoding**

- **Signal to Data**
  - XOR NRZ data with senders clock signal
  - High to low transition ⇒ 1
  - Low to high transition ⇒ 0

- **Comments**
  - Solves clock recovery problem
  - Only 50% efficient (½ bit per transition)
  - Still need preamble (typically 0101010101… trailing 11 in Ethernet)

![Diagram showing NRZ, Clock, and Manchester encoding](image_url)
4B/5B (100Mbps Ethernet)

- Goal: address inefficiency of Manchester encoding, while avoiding long periods of low signals
- Solution:
  - Use five bits to encode every sequence of four bits
  - No 5 bit code has more than one leading 0 and two trailing 0’s
  - Use NRZI to encode the 5 bit codes
  - 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>11110</td>
<td>1000</td>
<td>10010</td>
</tr>
<tr>
<td>0001</td>
<td>01001</td>
<td>1001</td>
<td>10011</td>
</tr>
<tr>
<td>0010</td>
<td>10100</td>
<td>1010</td>
<td>10110</td>
</tr>
<tr>
<td>0011</td>
<td>10101</td>
<td>1011</td>
<td>10111</td>
</tr>
<tr>
<td>0100</td>
<td>01010</td>
<td>1100</td>
<td>11010</td>
</tr>
<tr>
<td>0101</td>
<td>01011</td>
<td>1101</td>
<td>11011</td>
</tr>
<tr>
<td>0110</td>
<td>01110</td>
<td>1110</td>
<td>11100</td>
</tr>
<tr>
<td>0111</td>
<td>01111</td>
<td>1111</td>
<td>11101</td>
</tr>
</tbody>
</table>
**Encoding Summary**

- **Two basic tasks: send and receive**
  - The trouble is the channel distorts the signal

- **Transmission modulates some physical carrier**
  - Lots of different ways to do it, various efficiencies

- **Receiver needs to recover clock to correctly decode**
  - All real clocks drift, so needs to continually adjust
  - The encoding scheme can help a lot
Layering: A Modular Approach

- Sub-divide the problem
  - Each layer relies on services from layer below
  - Each layer exports services to layer above

- Interface between layers defines interaction
  - Hides implementation details
  - Layers can change without disturbing other layers

- Interface among peers in a layer is a protocol
  - If peers speak same protocol, they can interoperate
Protocol Standardization

- Communicating hosts speaking the same protocol
  - Standardization to enable multiple implementations
  - Or, the same folks have to write all the software

- Internet Engineering Task Force
  - Based on working groups that focus on specific issues
  - Produces “Request For Comments” (RFCs)
    - Rough consensus and running code
    - After enough time passes, promoted to Internet Standards

- Other standards bodies exist
  - ISO, ITU, IEEE, etc.
TCP/IP Protocol Stack

- **Application Layer**
  - HTTP

- **Transport Layer**
  - TCP

- **Network Layer**
  - IP

- **Link Layer**
  - Ethernet interface
  - SONET interface

**Routing**
- Between IP layers
- Between Network and Transport layers

CSE 123 – Lecture 4: Framing
Encapsulation

HTTP

TCP

IP

Ethernet interface

Payload

Headers

HTTP

TCP

IP

Ethernet interface

CSE 123 – Lecture 4: Framing
Internet Protocol Suite

The Hourglass Model

Applications
Transport
Data Link
Physical

“Thin Waist”

FTP
HTTP
NV
TFTP
TCP
UDP
IP
NET_1
NET_2
...
NET_n
Physical layer

2.4Ghz Radio
DS/FH Radio
(1-11Mbps)

802.11b Wireless Access Point

Cat5 Cable (4 wires)
100Base TX Ethernet
100Mbps

Ethernet switch/router

To campus backbone

62.5/125um 850nm MMF
1000BaseSX Ethernet
1000Mbps

CSE 123 – Lecture 4: Framing
TCP/IP Protocol Stack

Application Layer
- HTTP

Transport Layer
- TCP
- IP
- Ethernet interface
- SONET interface

Network Layer
- IP
- router

Link Layer
- Ethernet interface

CSE 123 – Lecture 4: Framing
(Data) Link Layer

- **Framing**
  - Break stream of bits up into discrete chunks

- **Error handling**
  - Detect and/or correct errors in received frames

- **Media access**
  - Arbitrate which nodes can send frames at any point in time
  - Not always necessary; e.g. point-to-point duplex links

- **Multiplexing**
  - Determine appropriate destination for a given frame
  - Also not always required; again, point-to-point
Today’s Focus: Framing

- Break down a stream of bits into smaller, digestible chunks called **frames**

- Allows the physical media to be shared
  - Multiple senders and/or receivers can time multiplex the link
  - Each frame can be separately addressed

- Provides manageable unit for error handling
  - Easy to determine whether something went wrong
  - And perhaps even to fix it if desired
What’s a Frame?

- Wraps payload up with some additional information
  - Header usually contains addressing information
  - Maybe includes a trailer (w/checksum—next lecture)
- Basic unit of reception
  - Link either delivers entire frame payload, or none of it
  - Typically some maximum transmission unit (MTU)
- Some link layers require absence of frames as well
  - I.e., minimum gaps between frames
Identifying Frames

- First task is to delineate frames
  - Receiver needs to know when a frame starts and ends

- Several different alternatives
  - Fixed length (bits) frames
  - Fixed duration (seconds) frames
  - Explicit delimiters indicate start and/or end of each frame
    - Alternatively, just start and a length field

- Misidentifying frame start and end leads to errors
  - Topic for next lecture…
Fixed-Length Frames

● Easy to manage for receiver
  ◆ Well understood buffering requirements

● Introduces inefficiencies for variable length payloads
  ◆ May waste space (padding) for small payloads
  ◆ Larger payloads need to be fragmented across many frames

● Requires explicit design tradeoff
  ◆ ATM uses 53-byte frames (cells)
Length-Based Framing

- To avoid overhead, we’d like variable length frames
  - Each frame declares how long it is
  - E.g. DECNet DDCMP

- What’s the issue with explicit length field?
  - Must correctly read the length field
    » Need to decode *while* receiving
  - Still need to identify the beginning…
Flags/Sentinels

- Allows for variable frame lengths

- Start (and end) each frame with a special indicator
  - Could be a special byte/bit pattern in header/trailer
  - Unique (non-data) physical-layer symbol

- Reserving physical layer symbols reduces efficiency
  - Can’t use that symbol to transmit data!

- But using bits/bytes in data stream is problematic
  - What if sentinel happens to appear in the data?
Stuffing

- Insert bytes/bits into data stream to make sure that sentinel (flag) does not appear in payload
For Next Class

- Read 2.4
- Keep going on Homework 1
- Get started on Project 1
  - Discussion section MONDAY at 3pm