Introduction to Framing

- Framing, or frame synchronization problem: enabling receiver to identify frame boundaries, i.e. beginning and end of strings bits/bytes
  - Turn a symbol channel into a frame channel
  - Multiplexing several continuous streams of symbols together
- Problem occurs at different levels in the stack
  - Data-link
    - Stream of symbols arriving over channel (think marble pipe)
    - Because of physical limits, first few symbols may be corrupted ⇒ need to ensure receiver can identify first symbol in frame
  - Transport
    - Stream of bytes arriving over channel
    - No loss/corruption, but need to unambiguously identify the end of each frame
Framing for Multiplexed Streams

Example: T-1 carrier system
- Used in **circuit-switched** telephony
- 24 digital voice channels carried over 2-wires
  - Time-division multiplexing at 1,544,000 bits/sec
- Each channel: 8000 8-bit samples/second
- Fixed frame size: one sample from each of 24 channels = 192 bits
- Problem: identify first bit in frame

8-bit voice samples

Framing bit: always 1

8 x 24 + 1 = 193 bits/frame
Framing Streams in T-1

Scenario:
- Receiver startup (e.g. after power outage)
- Bits arriving continuously at 1,544,000/second
- How to identify framing bit?

Solution:
- Wide shift register checks for $k$ consecutive 1's, 193 bits apart
- If this condition occurs every 125 microsec → indicate "Frame Lock" acquired

Probability of false Frame Lock?
Variable-length Framing

• **Beginning of frame**: Indicate with a *unique marker*
  - A pattern of bits that normally *occurs rarely* in data
    • Depending on protocol and underlying channel, may or may not be able to *guarantee* that the marker does not occur in data

• **Two approaches to indicating end of frame**:
  1. *Unique marker* in stream
    • Must not occur in data (i.e. between start and end markers)
  2. Transmit *length* of data in header
Example: Bisync

IBM's Binary Synchronous Communication
- 1960's data link protocol
- Designed for character-oriented communication between mainframes and terminals
- Designed for half-duplex modems at 1200-9600 bps

• Beginning: unique marker = SYN SYN SOH
  = 00010110 00010110 00000001 (in ASCII*)
• Data marked by STX (start text)
  = 00000010
• End: unique marker = ETX (end text)
  = 00000011

*Bisync actually used EBCDIC, a different code
Example: Bisync

- **Problem:** ETX might occur in data
- **Solution:** **Byte-stuffing** (or "character stuffing")
  - **Sender:** insert a special escape character DLE before any occurrence of ETX in Data portion of frame
    - Data Link Escape (DLE) = 00010000
    - For consistency, must also "escape" any occurrences of DLE in the data
  - **Receiver:** while looking for ETX, if DLE is encountered, throw it away and treat the following character as data

```
user data       A  ETX  DLE
xmitted frame   SYN  SYN  SOH  Hdr  STX  A  DLE  ETX  DLE  DLE  ETX
```
Example: HDLC

- High-Level Data Link Control
  - 1970's **bit-oriented** data link protocol
    - Concepts from IBM's SDLC originally
    - Designed to be implemented in **hardware**

- Beginning and end of frame: **same marker**
  \[= 01111110 \ (0x7E)\]

- Problem: 01111110 must not occur in transmitted data
Example: HDLC

• Solution: **bit-stuffing**
  
  - **Sender:**
    
    • Insert a single 0 bit after any string of 5 consecutive 1's in transmitted data
  
  - **Receiver:**
    
    • After receiving 11111:
      
      - if next bit is 0 → discard it and continue
      - if next bits are 10 → recognize end of frame
      - if next bits are 11 → framing error

User data: 10111111100010100011111010101

Txmitted frame: 0111110 1011111 1100010100011111 010101 01111110
Example: IEEE 802.5 Token Ring

- Principle: Use "code violations" as unique markers to avoid need for stuffing
- 802.5 uses **Differential Manchester** signaling
  - Transition in the middle of each data bit
  - Another way to look at it:
    - 2 signal elements (1/0) per bit
    - 4 possible symbols: Data 0, Data 1, **Nondata J**, **Nondata K**
    - "Nondata" symbols have no transition in the middle—code violation!

**802.5 Frame Format:**

![Frame Format Diagram]
Example: IEEE 802.5 Token Ring

- Start delimiter (SD) and End delimiter (ED) formed using non-data symbols J and K
  - These symbols never occur in data portion of frame
  - No need for stuffing!

- This technique makes use of the redundancy already present in the Differential Manchester coding
Length-Based Framing

- Often used in higher-level protocols (e.g. IP)
  - Other methods required to locate beginning of first frame
- **Fixed-size length field** near beginning of frame indicates number of bytes (or bits) in frame
  - Size of field determines maximum frame length supported
- Receiver **counts bytes** after beginning of frame to detect end of frame

![Diagram of frame structure]

- SYN
- SYN
- k+4
- k bytes of data

- Redundant BoF indicator
- 2-byte length field
- max length = 65535
Example: Internet Protocol

• IP header includes a 16-bit "datagram length" field

• Why? IP is designed to run on top of all kinds of link layers
  – Designers assumed all would have some way to detect frame beginning
  – But some might not detect end-of-frame reliably
Framing in Application Protocols

- FTP: unique marker = \langle end-of-stream \rangle
- SMTP: unique marker = \langle CRLF \rangle.\langle CRLF \rangle
- HTTP: two methods
  - length-based for some files
  - unique marker (= \langle end-of-stream \rangle) for other files
    - Used when length is not known in advance (e.g. live audio)