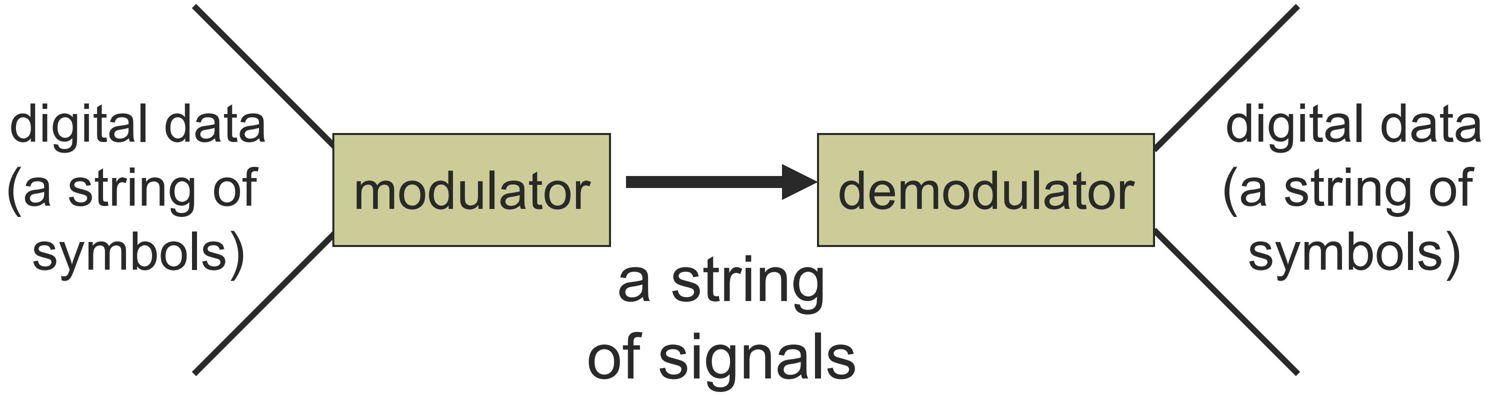
Direct Link Networks - Framing

Reading: Peterson and Davie, Chapter 2

Framing



- Encoding translates symbols to signals
- Framing demarcates units of transfer
  - Separates continuous stream of bits into frames
  - Marks start and end of each frame

Benefits of framing

- Synchronization recovery
  - Breaks up continuous streams of unframed bytes
  - Recall RS-232 start and stop bits
- Link multiplexing
  - Multiple hosts on shared medium
  - Simplifies multiplexing of logical channels
- Efficient error detection
  - Per-frame error checking and recovery

Framing

- Demarcates units of transfer
- Goal
  - Enable nodes to exchange blocks of data
- Challenge
  - How can we determine exactly what set of bits constitute a frame?
  - How do we determine the beginning and end of a frame?

Framing

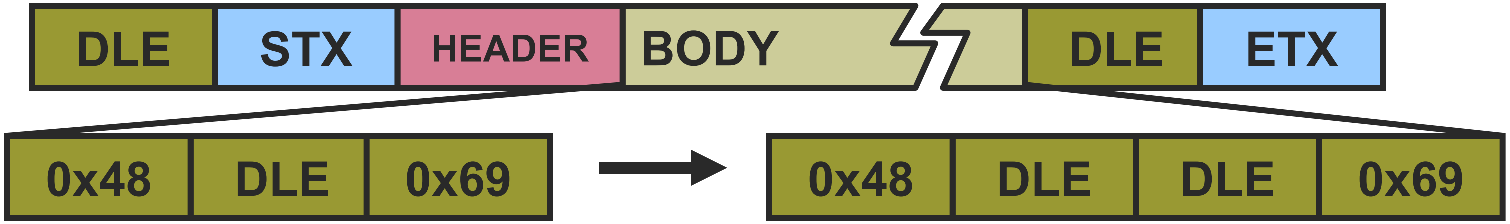
- Approaches
  - Sentinel: delimiter at end of frame (like C strings)
  - Length-based: length field in header (like Pascal strings)
  - Clock-based: periodic, time-based
- Characteristics
  - Bit- or byte-oriented
  - Fixed or variable length
  - Data-dependent or data-independent length

Sentinel-Based Framing

- End of Frame
  - Marked with a special byte or bit pattern
    - Frame length is data-dependent
  - Challenge
    - Frame marker may exist in data
    - Requires stuffing
- Examples
  - BISYNC, HDLC, PPP, IEEE 802.4 (token bus)

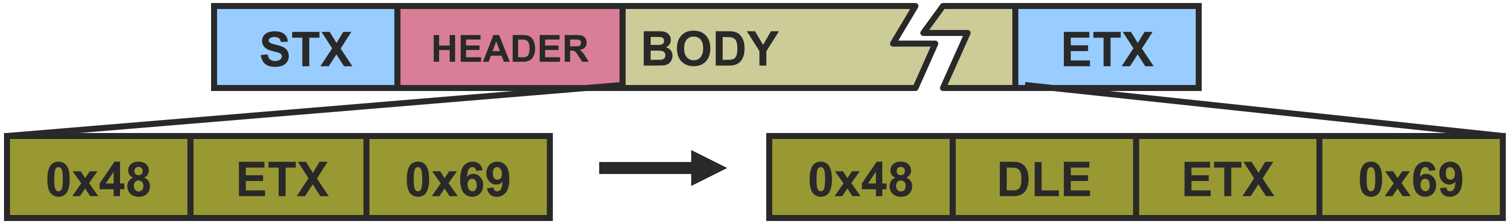
ARPANET IMP-IMP

- Interface Message processors (IMPs)
  - Packet switching nodes in the original ARPANET
  - Byte oriented, Variable length, Data dependent
  - Frame marker bytes
    - STX/ETX start of text/end of text
    - DLE data link escape
  - Byte Stuffing
    - DLE byte in data sent as two DLE bytes back-to-back



BISYNC

- BInary SYNchronous Communication
  - Developed by IBM in late 1960's
  - Byte oriented, Variable length, Data dependent
  - Frame marker bytes:
    - STX/ETX start of text/end of text
    - DLE data link escape
  - Byte Stuffing
    - ETX/DLE bytes in data prefixed with DLE's



Byte Stuffing: BISYNC

0000 0011 1110 0111 1111

1110 0001 0000 0001 1111

- ETX/DLE bytes in data prefixed with DLE's
  - DLE = 16; STX = 2; ETX = 3

- Ans: **0000 0010** **0001 0000** 0000 0011
  1110 0111 1111 1110 **0001 0000**
  0001 0000 0001 1111 **0000 0011**

Byte Stuffing: Efficiency

0000 0011 1110 0111 1111

1110 0001 0000 0001 1111

- Frame: **0000 0010** **0001 0000** 0000 0011
  1110 0111 1111 1110 **0001 0000**
  0001 0000 0001 1111 **0000 0011**
- Efficiency:
  - 72 bits were sent for 40 bits of data
  - Efficiency is 40/72 = 55.6%

High-Level Data Link Control Protocol (HDLC)

- Bit oriented, Variable length, Data-dependent
- Frame Marker
  - 01111110
- Bit Stuffing
  - Insert 0 after pattern 011111 in data
  - Example
    - 01111110   end of frame
    - 01111111   error! lose one or two frames

Bit Stuffing: HDLC

0000 0011 1110 0111 1111

1110 0001 0000 0001 1111

- Insert 0 after pattern 011111 in data

- Ans: **0111 1110** 0000 0011 111**0**0 0111
  11**0**11 1110 0001 0000 0001 1111**0**
  0111 1110

Bit Stuffing: Efficiency

0000 0011 1110 0111 1111

1110 0001 0000 0001 1111

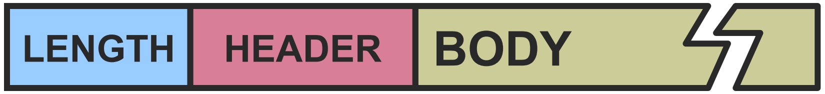
- Frame: **0111 1110** 0000 0011 111**0**0 0111
  11**0**11 1110 0001 0000 0001 1111**0**
  0111 1110
- Efficiency
  - 59 bits were sent for 40 bits of data
  - Efficiency = 67.8%

IEEE 802.4 (token bus)

- Alternative to Ethernet (802.3) with fairer arbitration
- End of frame marked by encoding violation,
  - i.e., physical signal not used by valid data symbol
  - Recall Manchester encoding
    - low-high means "0"
    - high-low means "1"
    - low-low and high-high are invalid
- IEEE 802.4
  - byte-oriented, variable-length, data-independent
- Another example
  - Fiber Distributed Data Interface (FDDI) uses 4B/5B
- Technique also applicable to bit-oriented framing

Length-Based Framing

- End of frame
  - Calculated from length sent at start of frame
  - Challenge
    - Corrupt length markers
- Examples
  - DECNET's DDCMP
    - Byte-oriented, variable-length
  - RS-232 framing
    - Bit-oriented, implicit fixed-length

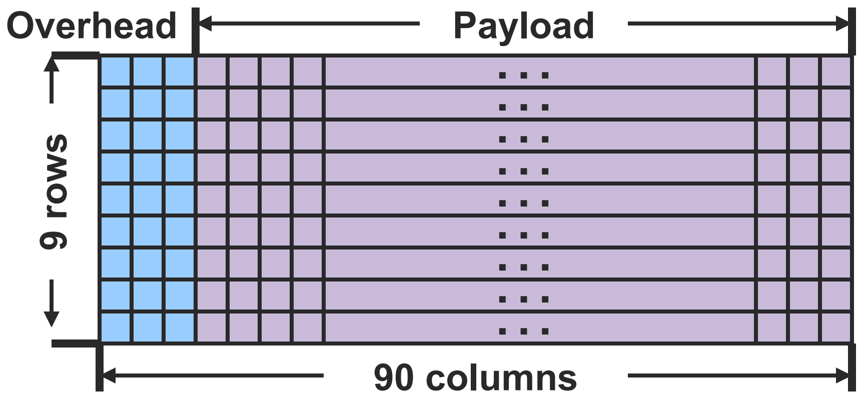


Clock-Based Framing

- Continuous stream of fixed-length frames
  - Clocks must remain synchronized
- STS-1 frames - 125μs long
  - No bit or byte stuffing
- Example
  - Synchronous Optical Network (SONET)
- Problems
  - Frame synchronization
  - Clock synchronization

SONET

- Frames (all STS formats) are 125 µsec long
  - Ex: STS-1 – 51.84 Mbps = 90 bytes
- Frame Synchronization
  - 2-byte synchronization pattern at start of each frame

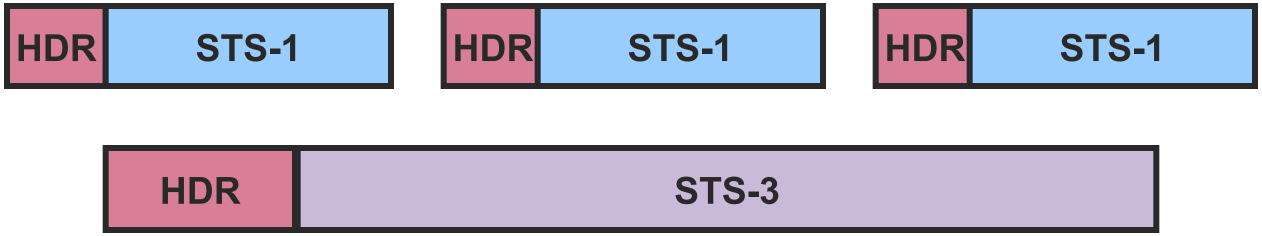


SONET: Challenges

- How to recover frame synchronization
  - Synchronization pattern unlikely to occur in data
    - Wait until pattern appears in same place repeatedly
- How to maintain clock synchronization
  - NRZ encoding
    - Data scrambled (XOR'd) with 127-bit pattern
    - Creates transitions
    - Also reduces chance of finding false sync. pattern

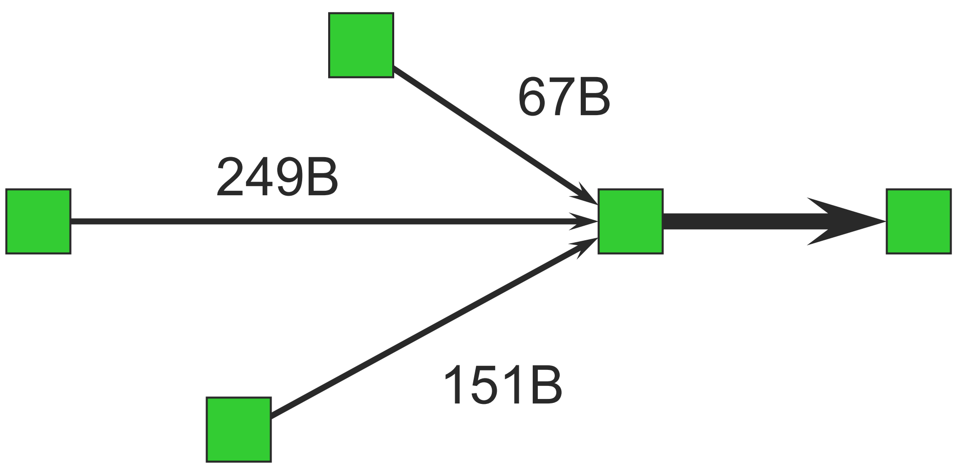
SONET

- A single SONET frame may contain multiple smaller SONET frames
- Bytes from multiple SONET frames are interleaved to ensure pacing



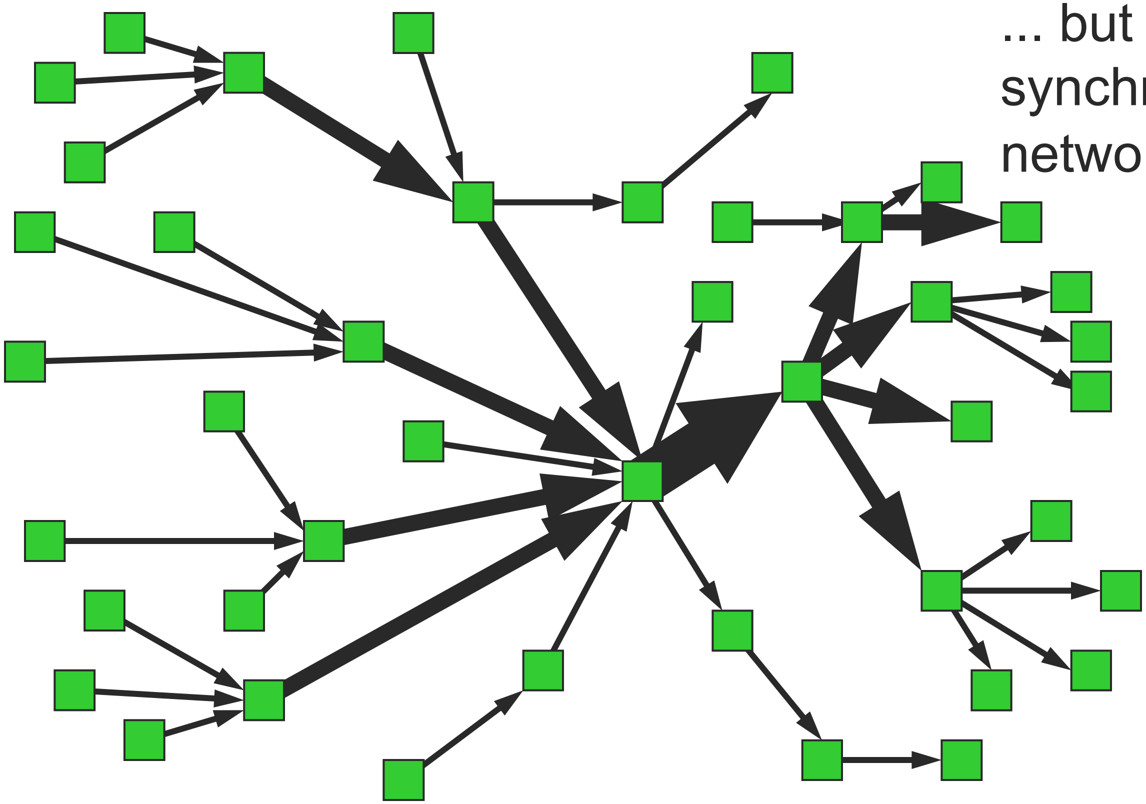
SONET

- STS-1 merged bytewise round-robin into STS-3
- Unmerged (single-source) format called STS-3c
- Problem: simultaneous synchronization of many distributed clocks



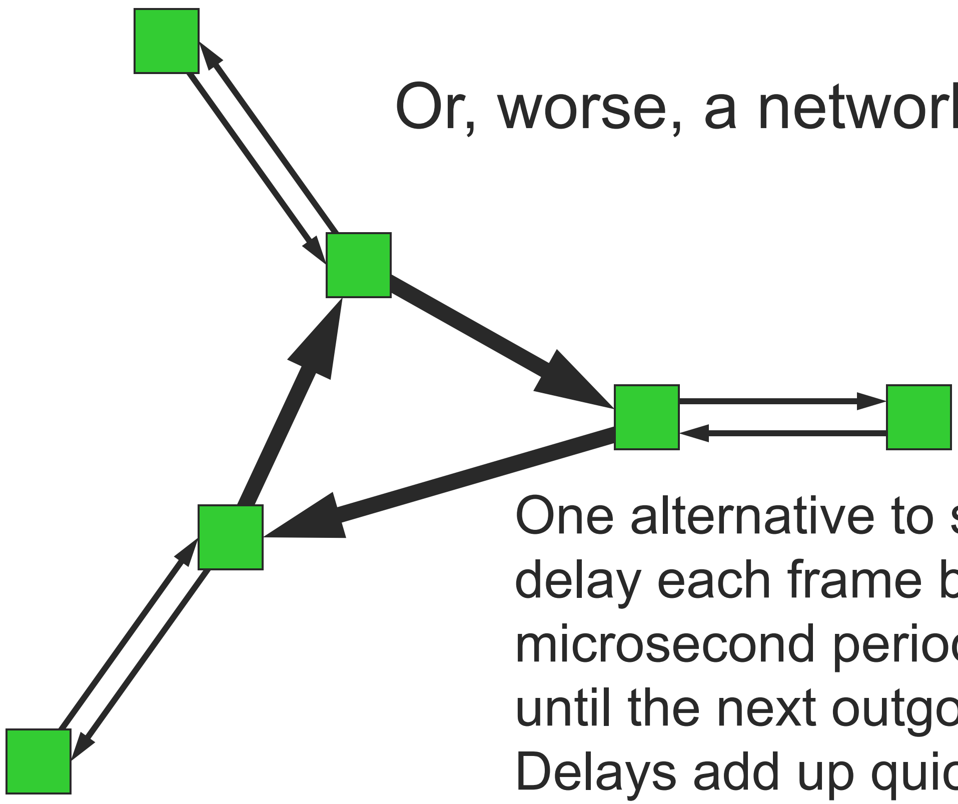
not too difficult to synchronize clocks such that first byte of all incoming flows arrives just before sending first 3 bytes of outgoing flow

SONET



... but now try to synchronize this network's clocks

SONET

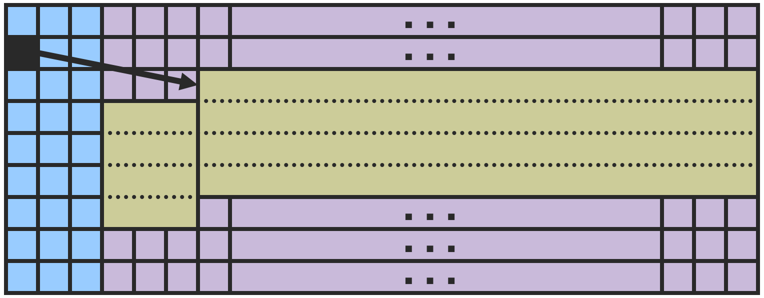


Or, worse, a network with cycles.

One alternative to synchronization is to delay each frame by some fraction of a 125 microsecond period at each switch (i.e., until the next outgoing frame starts). Delays add up quickly...

SONET

- Problem
  - Clock synchronization across multiple machines
- Solution
  - Allow payload to float across frame boundaries
  - Part of overhead specifies first byte of payload



Framing Summary

- Technique
  - Demarcate units of transfer
- Benefits
  - Synchronization recovery
  - Link multiplexing
  - Efficient error detection
- Approaches
  - Sentinel
  - Length-based
  - Clock based
- Characteristics
  - Bit- or byte-oriented
  - Fixed or variable length
  - Data-dependent or data-independent length