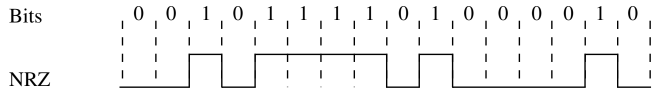
Point-to-Point Links

- Outline
  - Encoding
  - Framing
  - Error Detection
  - Sliding Window Algorithm

Encoding

- Signals propagate over physical medium
  - modulate electromagnetic waves
  - e.g., vary voltage over copper
- Encode binary data onto signals
  - e.g., 0 as low signal and 1 as high signal
  - known as Non-Return to Zero (NRZ)

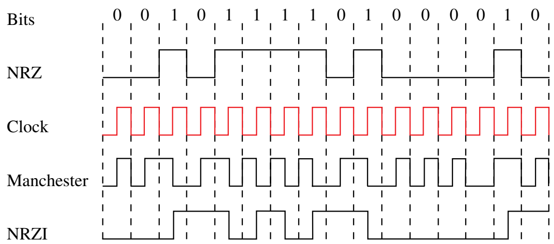


- Problem: Consecutive 1s or 0s
  - Low signal (0) may be interpretted as no signal
  - High signal (1) leads to baseline wander
  - Unable to recover clock

Alternative Encodings

- Non-return to Zero Inverted (NRZI): Make a transition from the current signal to encode a one, and stay at the current signal to encode a zero; solves the problem of consecutive ones.

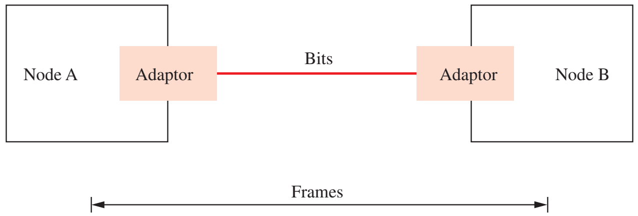
- Manchester: Transmits the XOR of the NRZ encoded data and the clock; only 50% efficient.



- 4B/5B: Every 4 bits of data is encoded in a 5-bit code, with the 5-bit codes selected to have no more than one leading 0 and no more than two trailing 0 (i.e., never get more than three consecutive 0s). Resulting 5-bit codes are then transmitted using the NRZI encoding. Achieves 80% efficiency.

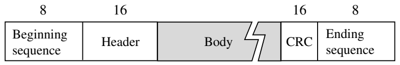
Framing

- Breaking sequence of bits into a frame
- Typically implemented by network adaptor

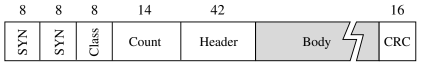


Approaches

- Sentinel-Based
  - delineate frame with a special pattern: 01111110
  - e.g., HDLC, SDLC, PPP



  - problem: special pattern appears in the payload
  - solution: bit stuffing
    * sender: insert 0 after five consecutive 1s
    * receiver: delete 0 that follows five consecutive 1s
- Counter-Based
  - include payload length in header
  - e.g., DDCMP



  - problem: count field corrupted
  - solution: catch when CRC fails.
- Clock-Based
  - each frame is 125μs long
  - e.g., SONET: Synchronous Optical Network
  - STS-n (STS-1 = 51.84 Mbps)

Cyclic Redundancy Check

- Add k bits of redundant data to an n-bit message.
  - want $k << n$
  - e.g., $k = 32$ and $n = 12,000$ (1500 bytes)
- Represent n-bit message as an $n - 1$ degree polynomial
  - e.g., MSG=10011010 as $M(x) = x^7 + x^4 + x^3 + x^1$
- Let k be the degree of some divisor polynomial $C(x)$
  - e.g., $C(x) = x^3 + x^2 + 1$
- Transmit polynomial $P(x)$ that is evenly divisible by $C(x)$
  - shift left k bits, i.e., $M(x)x^k$
  - subtract remainder of $M(x)x^k/C(x)$ from $M(x)x^k$
- Receive polynomial $P(x) + E(x)$
  - $E(x)$=0 implies no errors
- Divide $(P(x) + E(x))$ by $C(x)$; remainder zero if:
  - $E(x)$ was zero (no error), or
  - $E(x)$ is exactly divisible by $C(x)$

Selecting $C(x)$

- All single-bit errors, as long as the x^k and x^0 terms have non-zero coefficients.
- All double-bit errors, as long as $C(x)$ has a factor with at least three terms.
- Any odd number of errors, as long as $C(x)$ contains the factor $(x + 1)$.
- Any 'burst' error (i.e sequence of consecutive errored bits) for which the length of the burst is less than k bits.
- Most burst errors of larger than k bits can also be detected.

Common polynomials for $C(x)$:

CRC	$C(x)$
CRC-8	$x^8 + x^2 + x^1 + 1$
CRC-10	$x^{10} + x^9 + x^5 + x^4 + x^1 + 1$
CRC-12	$x^{12} + x^{11} + x^3 + x^2 + 1$
CRC-16	$x^{16} + x^{15} + x^2 + 1$
CRC-CCITT	$x^{16} + x^{12} + x^5 + 1$
CRC-32	$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} +$
	$x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

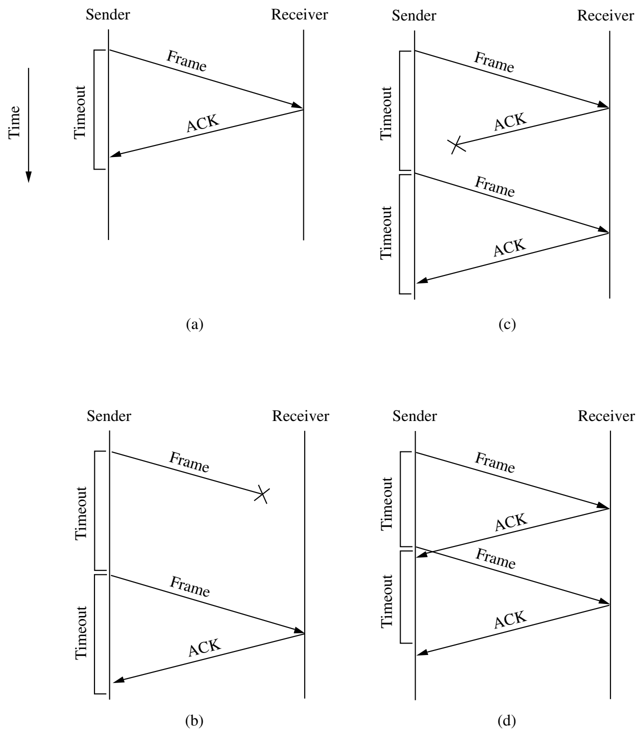
Internet Checksum Algorithm

Idea: view message as a sequence of 16-bit integers. Add these integers together using 16-bit ones complement arithmetic, and then take the ones complement of the result. That 16-bit number is the checksum.

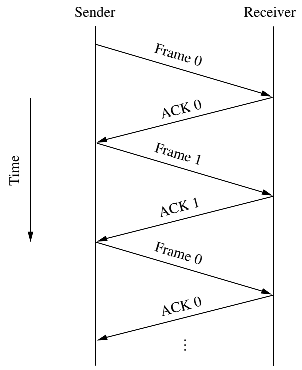
```
u_short
cksum(u_short *buf, int count)
{
    register u_long sum = 0;

    while (count--)
    {
        sum += *buf++;
        if (sum & 0xFFFF0000)
        {
            /* carry occurred, so wrap around */
            sum &= 0xFFFF;
            sum++;
        }
    }
    return ~(sum & 0xFFFF);
}
```

Acknowledgements & Timeouts



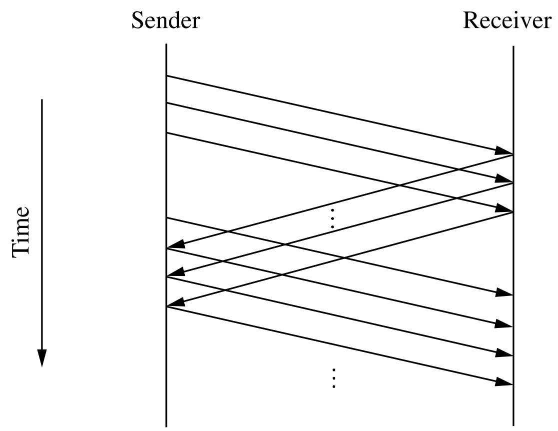
Stop-and-Wait



- Problem: keeping the pipe full.
- Example
  - 1.5Mbps link $\times$ 45ms RTT = 67.5Kb (8KB)
  - 1KB frames imples 1/8th link utilization

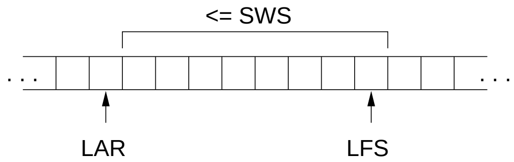
Sliding Window

- Allow multiple outstanding (un-ACKed) frames
- Upper bound on un-ACKed frames, called *window*



SW: Sender

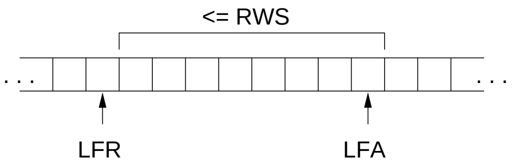
- Assign sequence number to each frame (`SeqNum`)
- Maintain three state variables:
  - send window size (`SWS`)
  - last acknowledgment received (`LAR`)
  - last frame sent (`LFS`)
- Maintain invariant: `LFS - LAR` $\leq$ `SWS`



- When ACK arrives, advance `LAR`, thereby opening window
- Buffer up to `SWS` frames

SW: Receiver

- Maintain three state variables:
  - receive window size (`RWS`)
  - largest frame acceptable (`LFA`)
  - last frame received (`NFE`)
- Maintain invariant: `LFA - LFR` $\leq$ `RWS`



- Frame `SeqNum` arrives:
  - if `LFR` $<$ `SeqNum` $\leq$ `LFA` $\longrightarrow$ accept
  - if `SeqNum` $\leq$ `LFR` or `SeqNum` $>$ `LFA` $\longrightarrow$ discarded
- Send cumulative ACK
- Variations
  - selective acknowledgements
  - negative acknowledgements (NAK)

Sequence Number Space

- `SeqNum` field is finite; sequence numbers wrap around
- Sequence number space must be larger than number of outstanding frames
- `SWS` $\leq$ `MaxSeqNum-1` is not sufficient
  - suppose 3-bit `SeqNum` field (`0..7`)
  - `SWS=RWS=7`
  - sender transmit frames `0..6`
  - arrive successfully, but ACKs lost
  - sender retransmits `0..6`
  - receiver expecting `7,0..5`, but receives second incarnation of `0..5`
- `SWS` $<$ `(MaxSeqNum+1)/2` is correct rule
- Intuitively, `SeqNum` "slides" between two halves of sequence number space

Concurrent Logical Channels

- Multiplex several logical channels over a single point-to-point link; run stop-and-wait on each logical channel.
- Maintain three bits of state for each channel:
  - boolean saying whether the channel is currently busy
  - sequence number for frames sent on this logical channel
  - next sequence number to expect on this logical channel
- ARPANET supported eight logical channels over each ground link (16 over each satellite link).
- Header for each frame included a 3-bit channel number and a 1-bit sequence number, for a total of 4 bits; same number of bits as the sliding window protocol requires to support up to eight outstanding frames on the link.
- Separates reliability from *reliable delivery* and *frame order*.