CSE 461: Introduction to Computer Communications Networks
Autumn 2010

Module 2.5
Bit Encoding and Errors

Ivayla Dermendjieva
iva@cs.washington.edu

Some material borrowed from slides by Jeremy Elson and Ben Greenstein. Thanks!

This Module's Topics

- Bit Encoding
  - NRZ
  - NRZI
  - Manchester
  - 4B/5B

- Error Detection
  - Parity
  - Internet Checksums
  - CRCs

- Error Correction
  - Hamming Distance

Bit Encoding

- Want to send messages, not just bits
- So break up bit stream into discrete chunks (frames)
- Synchronize both end points on frame boundaries – how?
  - Transmit clock on a separate channel – very expensive
  - Integrate clock in data stream

NRZ

- Simplest form of bit encoding
  - Signal is high to encode a '1'
  - Signal is low to encode a '0'

- What is a problem with this encoding?

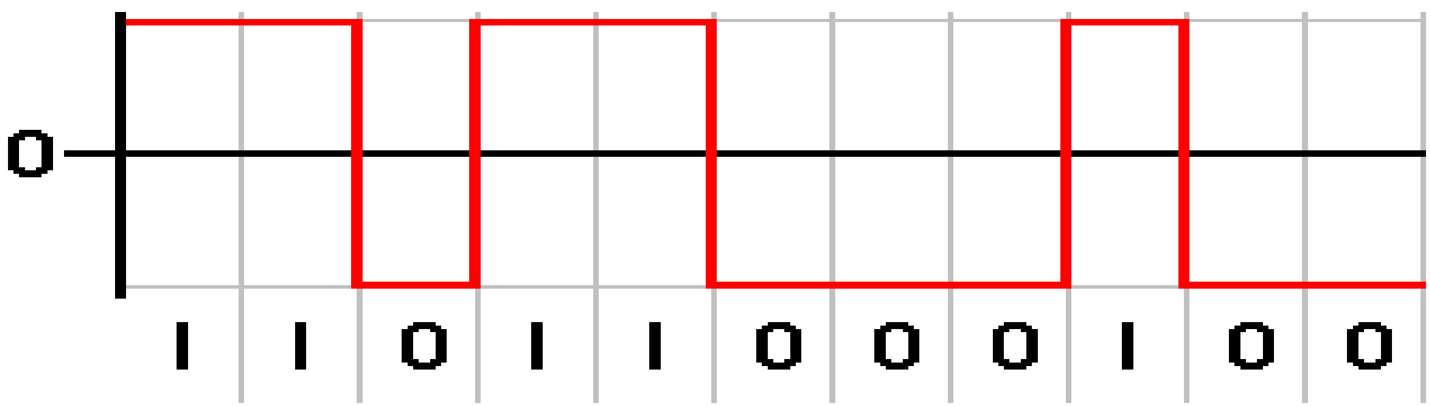


Image source: Wikipedia

NRZI

- NRZ has a problem: difficult to distinguish stream of consecutive 0's or 1's

- NRZI attempts to alleviate this
  - A signal change (i.e. high to low)encodes a '1'
  - No change in signal encodes a '0'

- This fixes the problem of sending consecutive 1's but not consecutive 0's

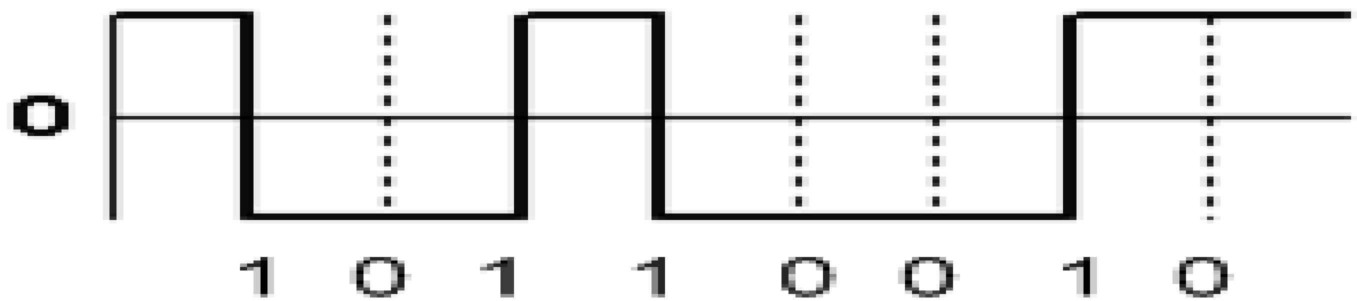


Image source: Wikipedia

Manchester

- Signal Change on every bit
  - Low-to-high encodes a '0'
  - High-to-low encodes a '1'

- Manchester unambiguously integrates clock and data. What is the downside?

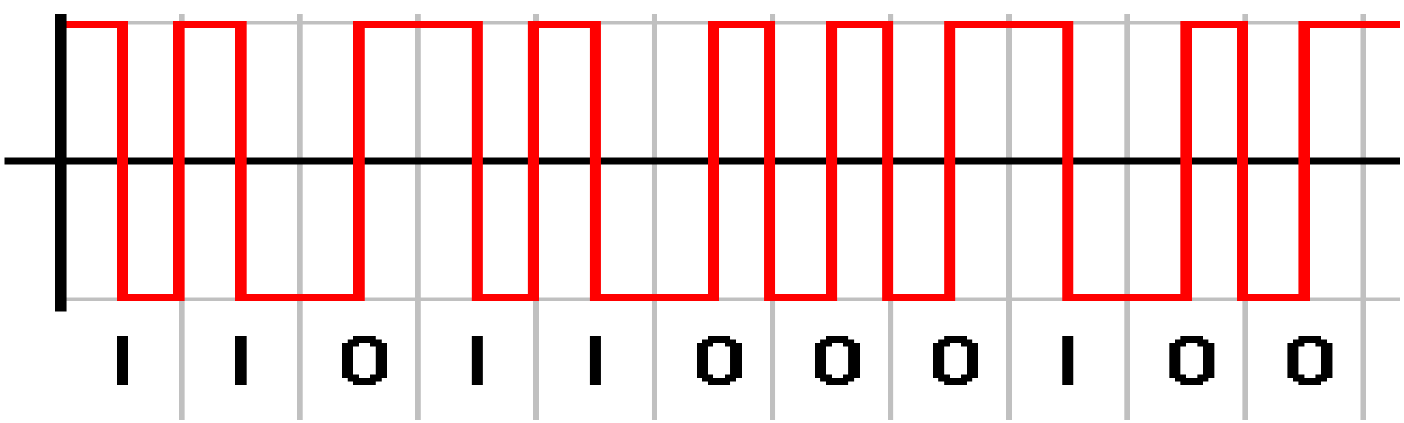


Image source: Wikipedia

4B/5B

- So far, the schemes we have looked at have had shortcomings:
  - NRZ and NRZI run the risk of holding the signal steady when transmitting consecutive bits
  - Manchester reduces efficiency of link

- Want a scheme which toggles the signal often enough, without significant efficiency overhead

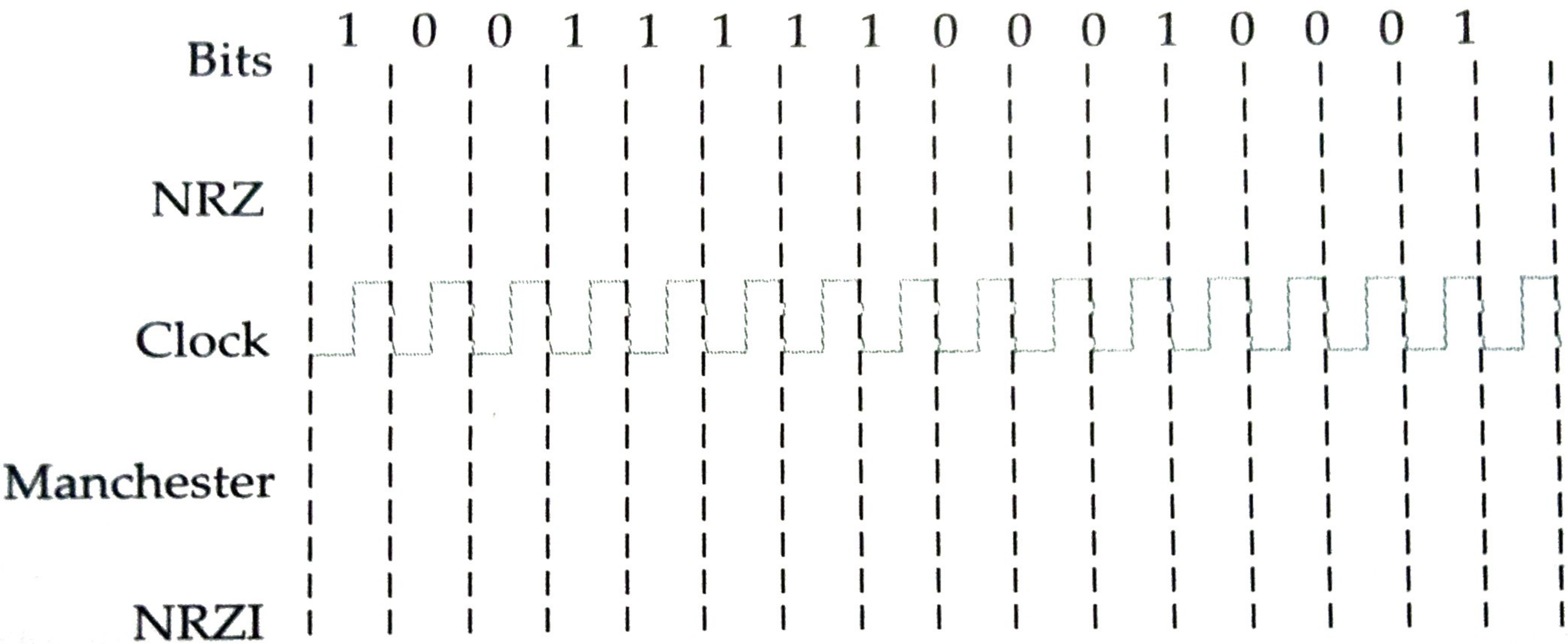
- 4B/5B encoding assigns a mapping between 4 bit data and 5-bit code words
  - There is a 5 bit code for each possible 4 bit sequence
  - Some 5 bit codes are invalid

4B/5B

4-Bit Data Symbol	5-Bit Code
0000	11110
0001	01001
0010	10100
0011	10101
0100	01010
0101	01011
0110	01110
0111	01111
1000	10010
1001	10011
1010	10110
1011	10111
1100	11010
1101	11011
1110	11100
1111	11101

Table 2.4 4B/5B encoding.

Encoding Exercise



Show the NRZ, Manchester and NRZI encodings for the above pattern. Assume NRZI signal starts out low

Error Detection and Correction

- How can we know if the frames we received have not been corrupted in transit?

- Add additional information to frames
  - Sender computes some property of data and sends it along with the data
  - Receiver computes same property and compares

- Various methods for determining frame integrity
  - Parity
  - Internet Checksum
  - Cyclic Redundancy Check

Parity

- Start with n bits and add another bit so that the total number of '1's is even (even parity)
  - i.e. 0110010 $\longrightarrow$ 01100101
  - Easy to compute as XOR of all input bits

- Will detect an odd number of bit errors
  - But not an even number

- Does not correct any errors

- Overhead of parity is proportional to data length (i.e. 1 parity bit for every 32 bits of data)

2D Pairity

- Add pairity row/column to array of bits

- How many simultaneous bit errors can it detect?

- Which errors can it correct?

- What is an example of a 4-bit error that would not be detected by a two-dimensional parity?

0101001	1
1101001	0
1011110	1
0001110	1
0110100	1
1011111	0
1111011	0

Checksums

- Used in Internet Protocols (IP, ICMP, TCP, UDP)

- Basic Idea: Add up the data and send it along with sum

- Algorithm:
  - Add 16 bit chunks using 1s complement
  - Take ones complement of the result

- Fixed overhead, independent of length of data

Cyclic Redundancy Check (CRC)

- Stronger protection than checksums
  - Used widely in practice (i.e. Ethernet CRC-32)
  - Implemented in hardware (XORs and shifts)

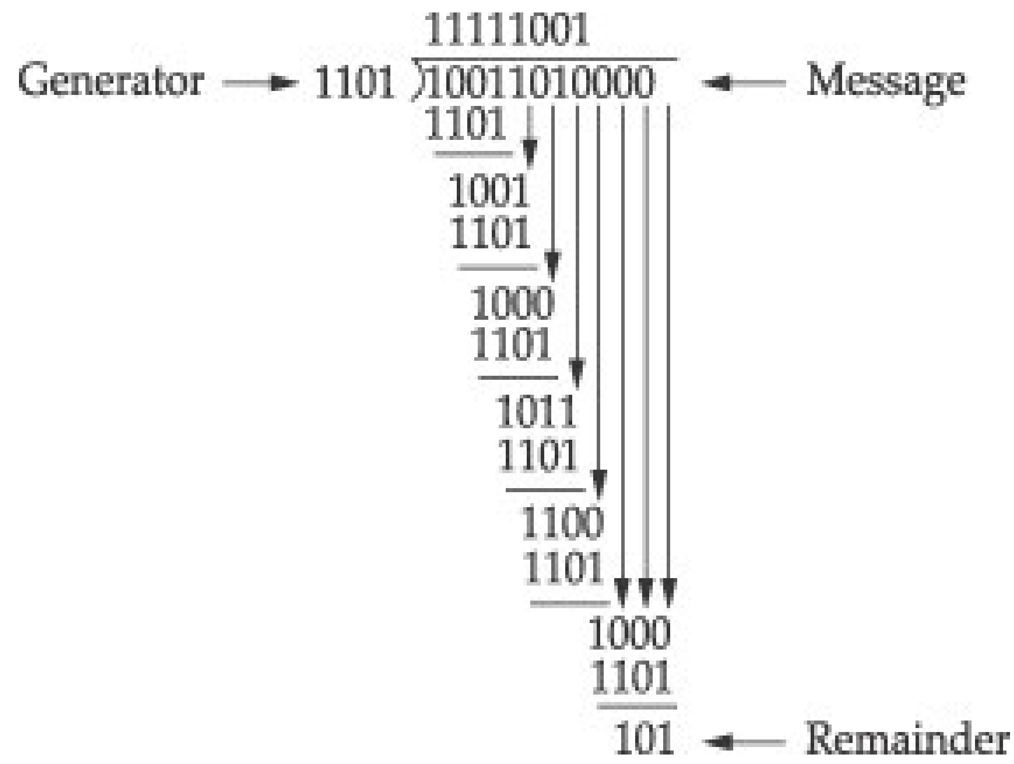
- Fixed overhead (independent of data size)

- Algorithm: Given n bits of data, generate a k bit check sequence that gives a combined n + k bits that are divisible by a chosen divisor $C(x)$

- Based on mathematics of finite fields
  - "numbers" correspond to polynomials, use modulo arithmetic
  - i.e. interpret 10011010 as $x^7 + x^4 + x^3 + x^1$

CRC Example

- Extend message with *k* 0's, when using a *k*-degree generator
- Divide message by generator (XOR)
- Discard result
- Subtract remainder from original message
- On reception, check that message is divisible by generator



CRC Example

Suppose we want to transmit the message 11001001 and protect it from errors using the CRC polynomial $x^3 + 1$. Determine the message that should be transmitted

Hamming Distance

- Errors must not turn one valid codeword into another valid codeword, or we cannot detect/correct them.
- Hamming distance of a code is the smallest number of bit differences that turn any one codeword into another
  – e.g, code 000 for 0, 111 for 1, Hamming distance is 3
- For code with distance d+1:
  – d errors can be detected, e.g, 001, 010, 110, 101, 011
- For code with distance 2d+1:
  – d errors can be corrected, e.g., 001 → 000