Overview

- Last Lecture
  - Data Compression
- This Lecture
  - Data Integrity 1
  - Source : Sections 10.1, 10.3
- Next Lecture
  - Data Integrity 2
  - Source: Sections 10.2, 10.5

Data Integrity Checks

jfasecurity.com

jfasecurity.com

morninglifemagazine.wordpress.com

Data Integrity Definition

- We want the data received to be the same as the data sent.
- Different to data security.
- How to do it?

I heard from your brother in Wellington. Plan to meet there tomorrow.

I heald from your brother in Wellington. Plan to meet there tomorrow.

r - 0111 0010 l - 0110 1100

Two Concepts of Data Integrity

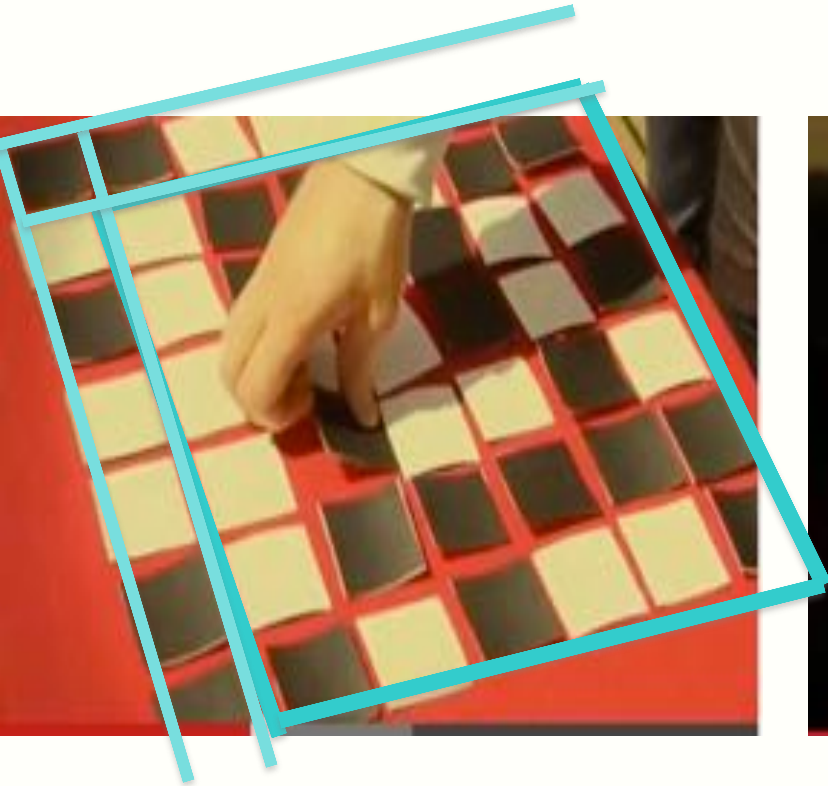
- Error Detection
  - Error detection is the ability to determine that data has an error.
- Error Correction
  - Error correction is the ability to correct the erroneous data.

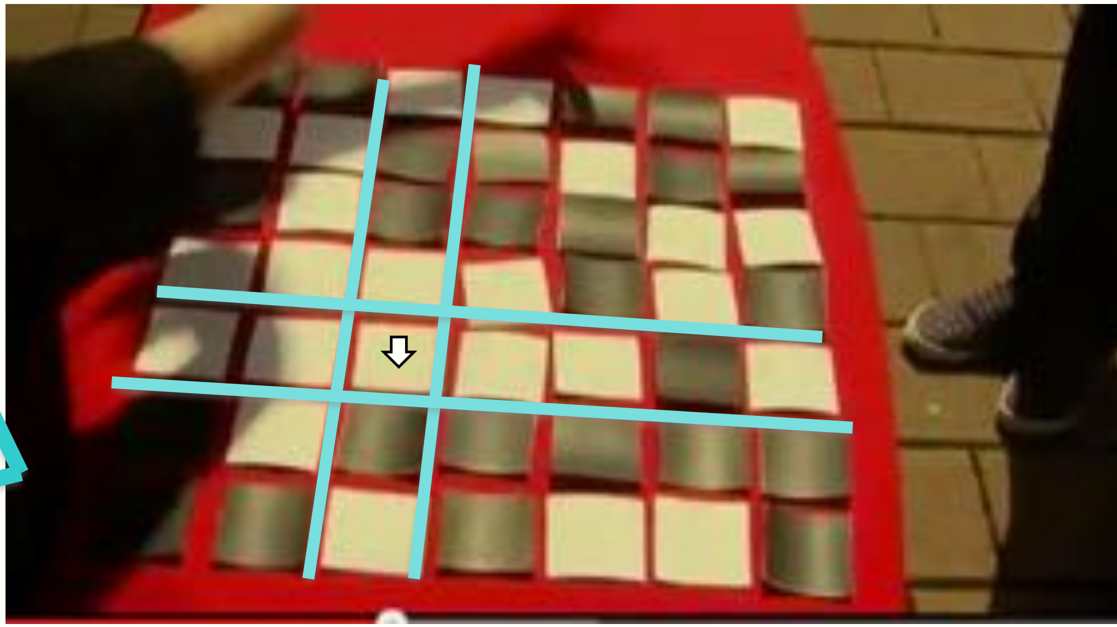
Error Detection

- Types of Errors:
  - Single bit error/Burst error (many bit errors)
- Detection Schemes:
  - Simple Parity checking (detect single error)
  - CRC (detect burst error)
- Most techniques require us to send additional bits whose value depends upon the data that is sent.
- If the data is corrupted, the additional bits will no longer match the (incorrect) data.

Parity Checking

Video: https://www.youtube.com/watch?v=OXz64qCjZ6k&feature=player_embedded





Parity Checking

- Simplest scheme
- Add an extra bit (parity bit) to the data, such that the total number of 1 bits is even (even parity) or odd (odd parity)
- Even parity
  - Make the number of 1's in a bit string an even number
- Odd parity
  - Make the number of 1's in a bit string an odd number
- Check at the receiver

Even parity	Odd Parity
1101 0100 0	1101 0100 1

Parity Checking (cont.)

- Sender: Suppose the sender wants to send the word "*world*". In ASCII the five characters are coded as

1110111 1101111 1110010 1101100 1100100

1110111**0** 1101111**0** 1110010**0** 1101100**0** 1100100**1**

- Receiver: counts the 1s in each character

(6, 6, 4, 4, 4) – data are accepted

(7, 6, 5, 4, 4) – data are corrupted, discard them and asks for retransmission

Parity Checking (cont.)

- Limitations of Parity Checking
  - Cannot detect an even number of bit errors
  - Does this make it useless?
    - Forms a basis for an error correction scheme (next lecture)
  - Burst errors (Many bits are damaged)
    - Consider (64 Kbps, 0.001 second power surge or static electricity, 64 bits may be damaged)

Cyclic Redundancy Checks

0 45496 88001 9

Check

Cyclic Redundancy Checks (cont.)

- CRC is based on binary division (polynomial division)
- What?

The bit string

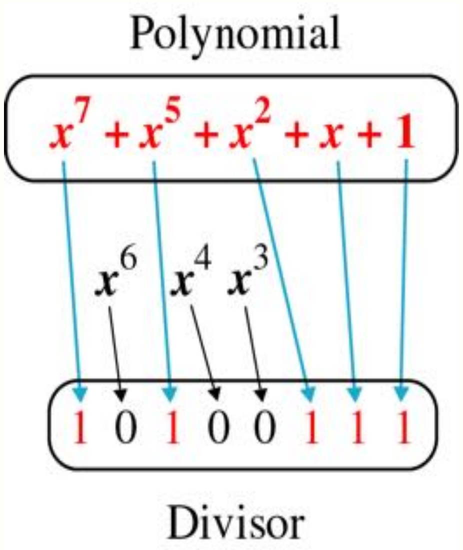
$b_{n-1}\ b_{n-2}\ .\ .\ .\ b_2\ b_1\ b_0$

is interpreted as

$b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + . . . + b_2x^2 + b_1x + b_0$

For example, 10100111 is interpreted as:

$x^7 + x^5 + x^2 + x + 1$



Cyclic Redundancy Checks (cont.)

Polynomial Division (Modulo 2 Style)

$$
\begin{array}{r|llllllll}
 & & & & x^6 & & + x^3 & & + x \\
\cline{2-9}
x^4 + x^3 + 1 & x^{10} + x^9 & + x^7 & & + x^5 + x^4 & & & & \\
 & x^{10} + x^9 & & + x^6 & & & & & \\
\cline{2-6}
 & & x^7 & + x^6 & + x^5 + x^4 & & & & \\
 & & x^7 & + x^6 & & & + x^3 & & \\
\cline{3-8}
 & & & & x^5 + x^4 & & + x^3 & & \\
 & & & & x^5 + x^4 & & & & + x \\
\cline{5-9}
 & & & & & & x^3 & & + x
\end{array}
$$

Calculation of $(x^{10} + x^9 + x^7 + x^5 + x^4) / (x^4 + x^3 + 1)$

Cyclic Redundancy Checks (cont.)

Modulo 2 Addition and Subtraction

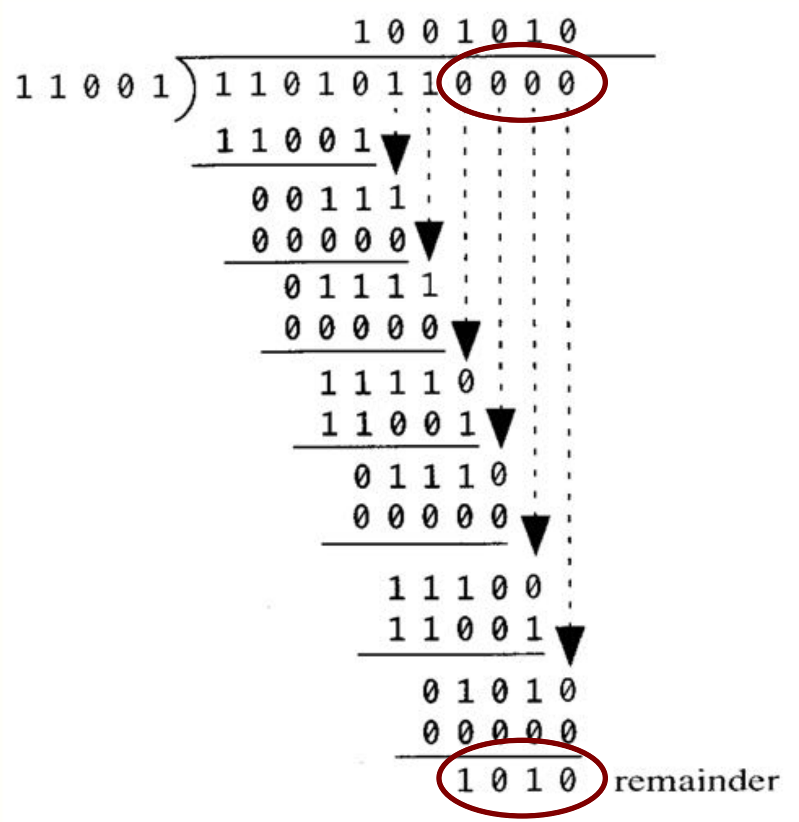
0 + 0 = 0	0 - 0 = 0
1 + 0 = 1	1 - 0 = 1
0 + 1 = 1	0 - 1 = 1
1 + 1 = 0	1 - 1 = 0

What Boolean function does this look like?
(minus and plus have the same results)

Cyclic Redundancy Checks

- Suppose original data is 110 1011
- Generator: CRC generator (divisor) is most often represented not as a string of 1s and 0s, but as an algebraic polynomial.
  - Let $G(x) = x^4 + x^3 + 1$ (11001)
- Append 4 0s (degree of $G(x)$) to the original data 110 1011 <u>0000</u>
- Divide 110 1011 <u>0000</u> by 11001, and derive the remainder 1010

Cyclic Redundancy Checks (cont.)

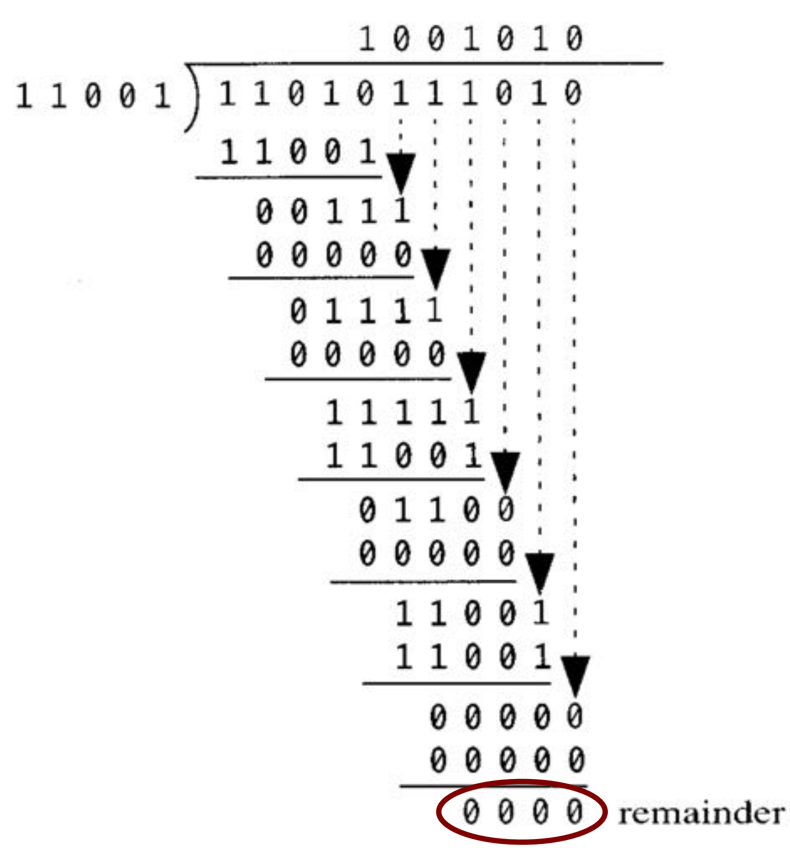


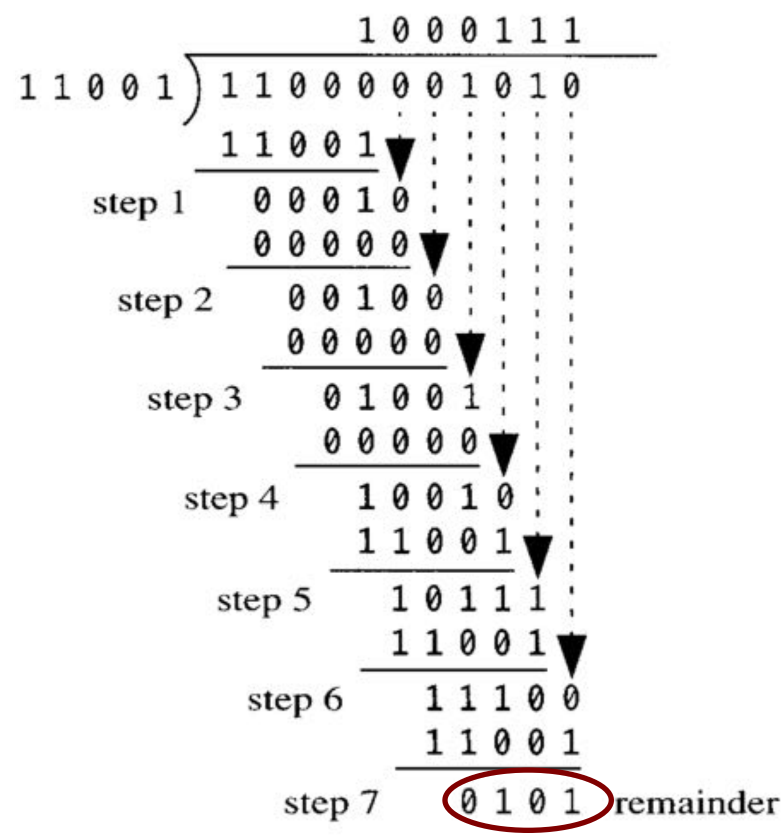
Cyclic Redundancy Checks (cont.)

- Sender: what to send?
  - Send 110 1011 1010 (This is the original data with the remainder on the end.)
- Receiver: how to check?
  - Suppose 110 1011 1010 is received. Divide it by 11001. If no errors, the remainder is 0.
  - Suppose 110 0000 1010 is received. Divide it by 11001. If any errors, the remainder is not 0.

Cyclic Redundancy Checks (cont.)

Any Error?





Cyclic Redundancy Checks (cont.)

- CRC Method:
  - Assume a generator bit string G, which has m bits. G is known to the sender and receiver. The bit string to be transmitted is B.
  - Append m -1 0's to the end of B, yielding B'.
  - Divide B' by G to get the remainder R.
  - Define $T = B' - R$. (Append the remainder R to the end of B). Send T.
  - T' is what is received. Divide T' by G.
  - If the remainder is 0, no error is detected. $T = T'$.
  - Original B is what is left after the last m-1 bits are removed from T'.

Cyclic Redundancy Checks

- $G(x)$ is very important in CRC
- Standard Generator Polynomials:
  - CRC-12: $x^{12} + x^{11} + x^3 + x^2 + x + 1$
  - CRC-16: $x^{16} + x^{15} + x^2 + 1$
  - CRC-CCITT: $x^{16} + x^{12} + x^5 + 1$
  - CRC-32: $x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$
- For CRC-32, 99.99999998% accuracy rate

Summary

- Parity Check
- CRC

What you should learn from this lecture:

- The basic process of simple parity check
- The basic process of CRC check