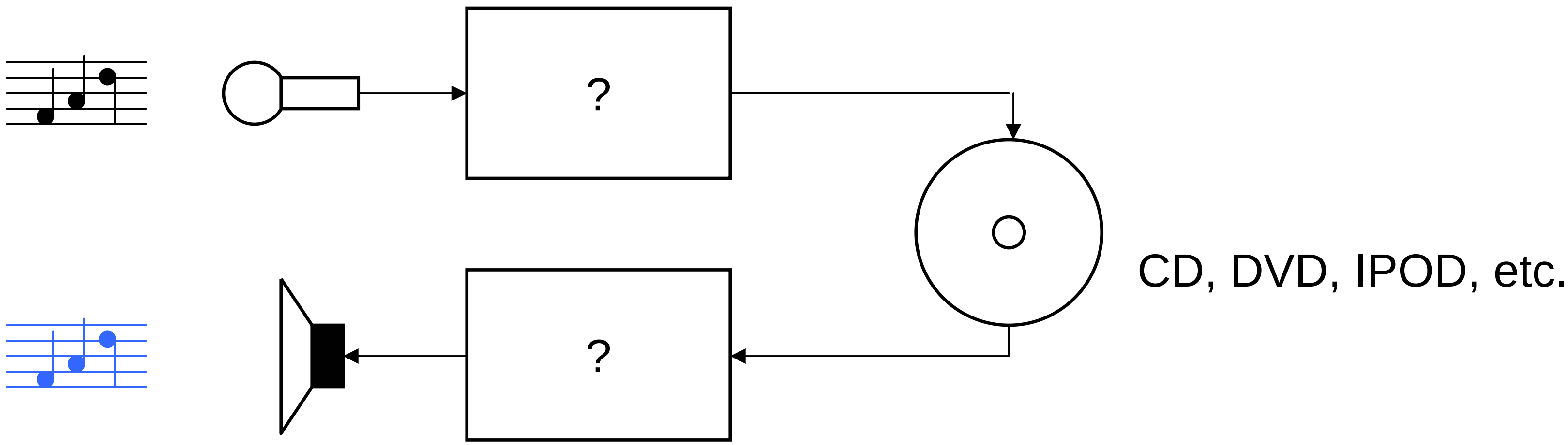
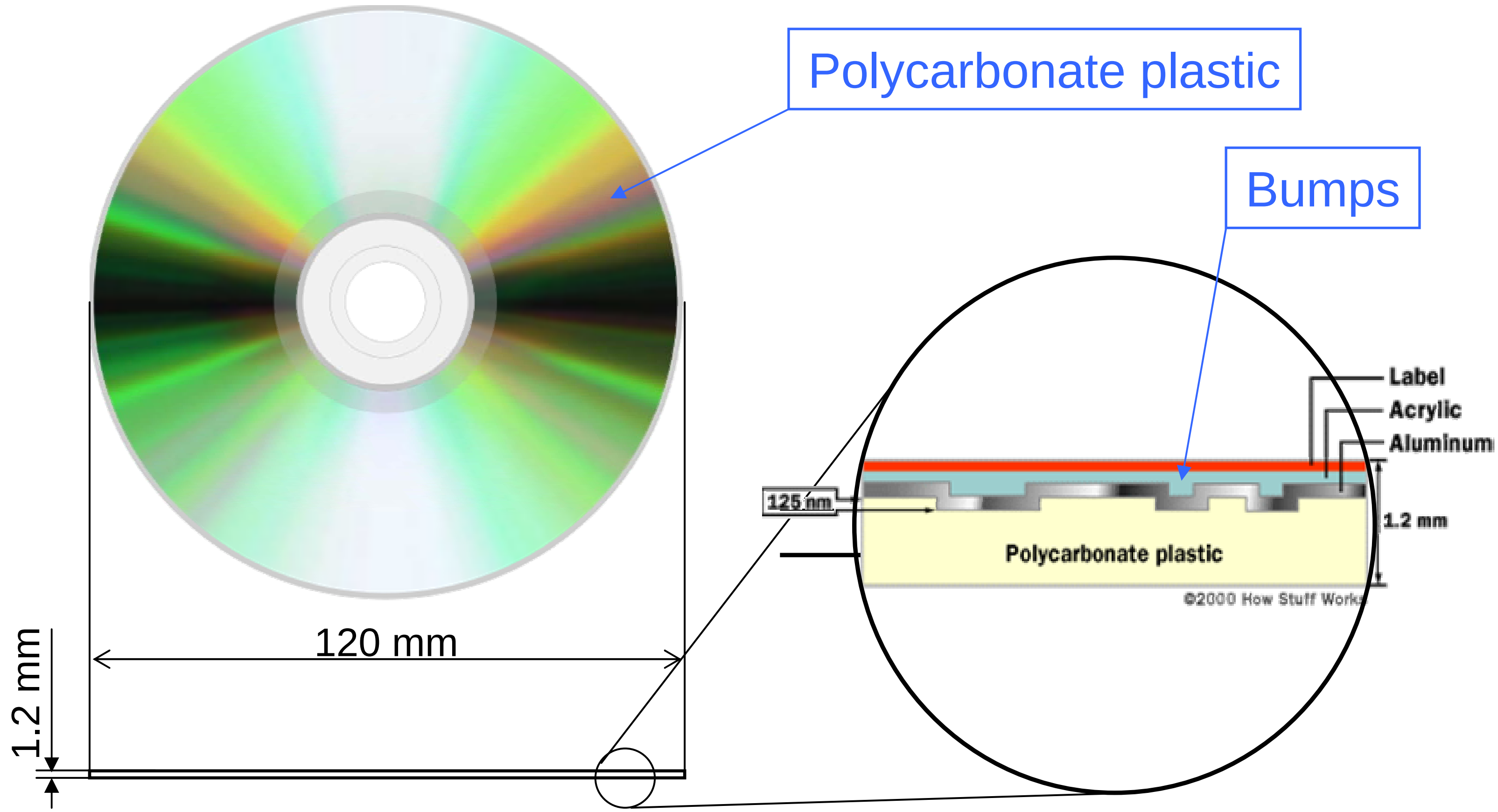
Information Sciences: Introduction

- We will consider a system with the following structure:

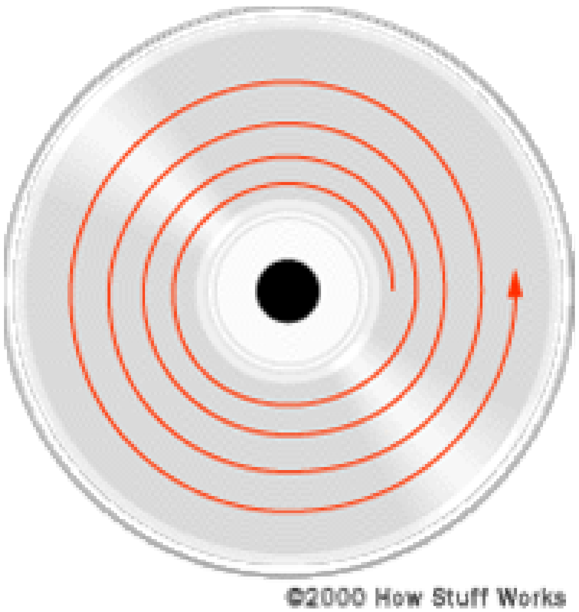


- Sound is acquired with a microphone and reproduced with a loudspeaker
- We record sound on a digital support, such as a CD. A digital support can record symbols that take only two states, call them 0 and 1
- What are the blocks “?” and why do we need them? This is the goal of this course…

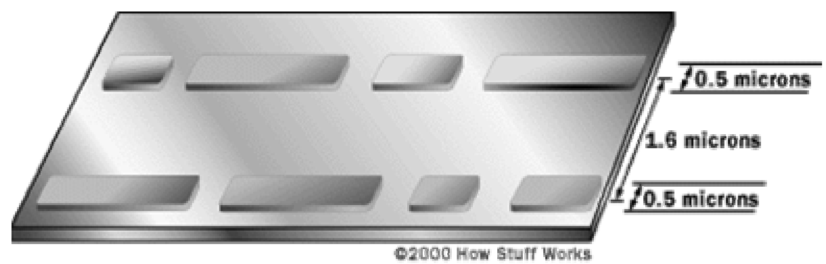
- Let’s analyze first the CD:



- A CD has a single spiral track of data, circling from the inside of the disc to the outside



- The data track is 0.5 μm wide with 1.6 microns separating one track from the next
- The bumps that make up the track are a minimum of 0.83 microns long and 125 nm high



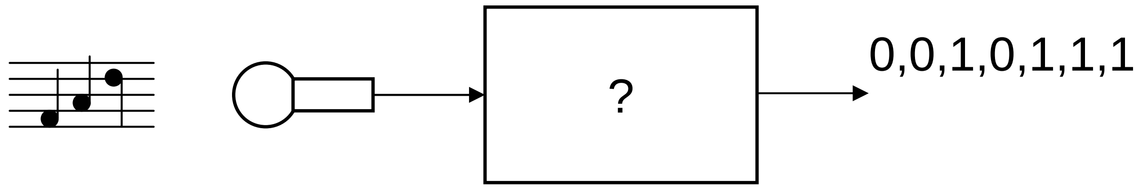
- How many bits can we write on a CD?

  Length of the spiral = $2\pi r_{average}\,(r_{max}-r_{min})/step$= 5000 m

  N_{CD} = Length / pitsize = 19 Gb
- Unfortunately we can not use all this information:
  - Size of one hair: 100 μm (100 times the pit size!)
  - Dust: 100 μm
  - Scratches on the polycarbonate

  We need to spend a part of the total number of bits to protect information

Signal Processing Module

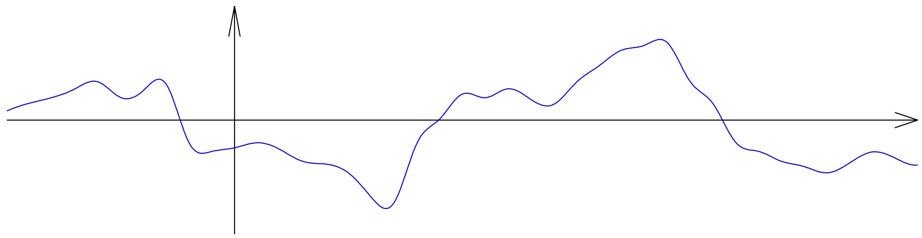


- On the CD we can record 0s and 1s, but sound pressure takes all possible values, we have to do a conversion
- We will see that we can sample the sound pressure and still be able to reconstruct it continuously. Also, each sample has to be represented with a finite number of bits
- On a audio CD, 44100 samples per second are recorded for 2 channels. Each sample is represented with 16 bits. To record 74 minutes of music, we need

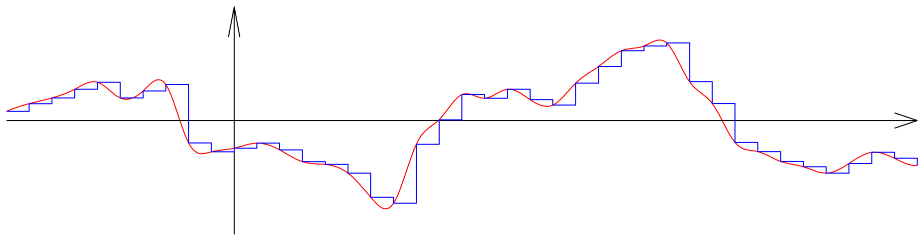
  N_{bits}= 2 44100 16 74 60 = 6.2 Gbit

- How do we do the conversion from music to bits?

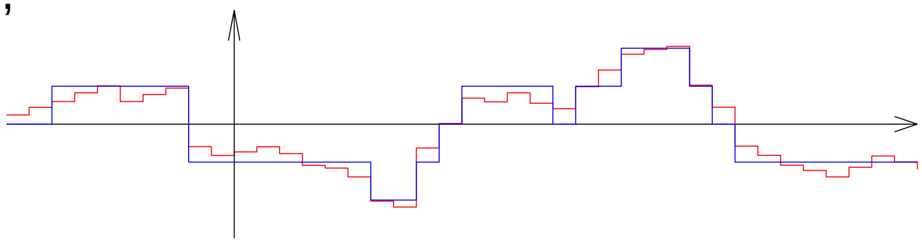
Sound is the perception of acoustic pressure. This is a quantity that varies over time…



We measure acoustic pressure at a regular rate, 44100 times per second...



Each sample is a real number, we have to approximate it (e.g. with the closest integer) and represent it with 16 bits (2^{16} possible combinations)

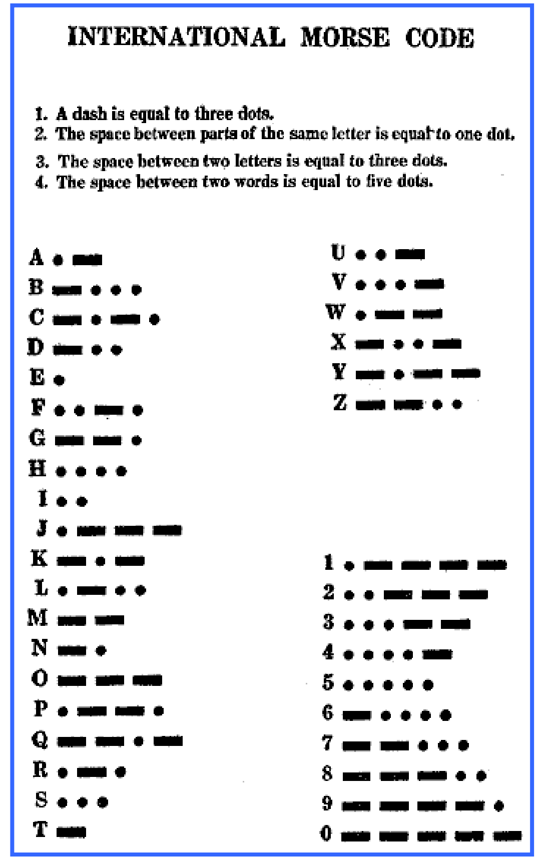


Source Coding

- Using 16 bits for each sample is a waste! If there is silence we spend the same number of bits per second that a wonderful guitar solo
- Solution: save bits by
    - Representing with short words the sequences that occur more often
    - Representing better the most important parts (e.g. guitar solo) and worse the less important parts
- E.g. of 1, the Morse code:

  Hello World

  .... . .-.. .-.. --- .-- --- .-. .-.. -..



INTERNATIONAL MORSE CODE

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to five dots.

A .-
B -...
C -.-.
D -..
E .
F ..-.
G --.
H ....
I ..
J .---
K -.-
L .-..
M --
N -.
O ---
P .--.
Q --.-
R .-.
S ...
T -
U ..-
V ...-
W .--
X -..-
Y -.--
Z --..
1 .----
2 ..---
3 ...--
4 ....-
5 .....
6 -....
7 --...
8 ---..
9 ----.
0 -----

- In typing SMS, people use abbreviations to reduce the number of characters. Can you say what the abbreviations CU, 4, THX, 2DAY mean?

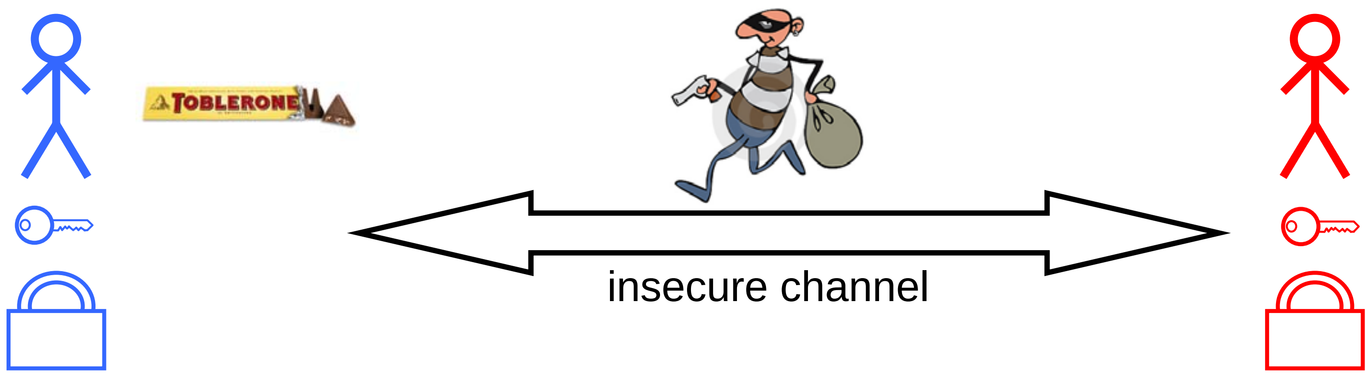
- How can we measure the minimum amount of information that a message conveys? We can remove randomly some characters until we are not able to understand the message. Let’s see an example:

Message: 20% of the characters

????n?n? ?s t?? a?qu????io? and d?velo?me?t ?? ??mo?ies ??? ?e????ors, in?ludi?? s?il??? k??wl????, u??????andin?? v?lue?, ?nd ??????? ?? i? t?? g?a? ?? ?du??????, ??? ??? ????u?? of ?x???i?n??. ?e??n??? r???es fr?? ???p?e ??rm? ???? a? h?b?tu?tio? ?o ???? ???p?e? ?o?ms su?h ?? p??y (a?t?vi?y)? ??e? on?? ?n ????e ?????????es

Cryptography

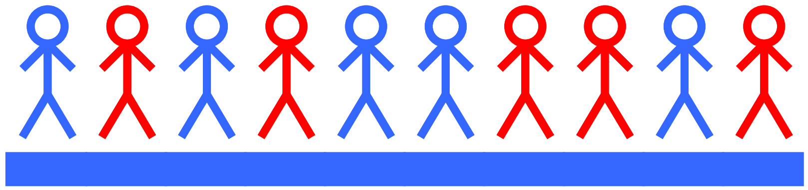
- Did you pay for the MP3 that you have in your IPOD?
- Maybe not, but one day you may want to sell audio files and be sure that everyone pays the bill. How can you send something through an insecure communication system?



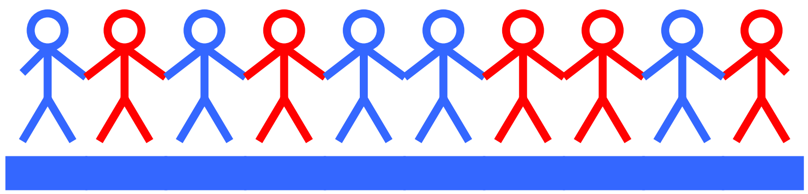
- How can A send a piece of chocolate to B through an “insecure channel”? A and B have padlocks, but no way to send securely the key…

Error Correcting Codes

- A CD can store theoretically 19 Gbits, but the probability of errors is very high (a scratch can damage many bits)
- We need only 6.2 Gbits to record 74 minutes of music, but they have to be error-free. Can we transform many unreliable bits in few reliable bits?



Some bits are lost, but we don’t know which ones



The values are interrelated so that removing some bits still allows to recognize the sequence

- How can we implement this in an easy way? Consider the following scheme:
    - A bit equal to 0 is represented by 3 consecutive 0s
    - A bit equal to 1 is represented by 3 consecutive 1s
- In this way the 6.2 Gbits are expanded into 18.6 Gbits! However, we are able to correct some errors. How?
- When we read the disk, we apply the decoding scheme
    - The sequences 000, 001, 010, 100 are decoded as 0
    - The sequences 111, 110, 101, 011 are decoded as 1
- If a scratch causes a single error on a group of 3 bits, we can correct it. 2 or 3 errors cannot be corrected!