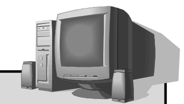
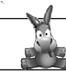
Data Representation

9/16/2009

Che-Rung Lee

Two different worlds



What we see/hear		Inside computers
Text	a,b,c	01100001,01100010,01100011
Number	1,2,3	00000001,00000010,00000011
Sound		01001100010101000110100…
Image		10001001010100000100111...
Video		00110000001001101011001…

Binary system

- Computers uses 0 and 1 to represent and store all kinds of data.
- Why binary?
  - We need to find physical objects/phenomenon to store, transmit, and process data. Binary is the most straightforward representation.

有無 上下 黑白 真偽 勝負

北南 負正 錯對 陽陰 關開 10

Some jargons

- **Bit:** a binary digit (0 or 1)
- **Byte**: 8 bits
  - Basic storage unit in computer system
- **Hexadecimal notation:**
  - Represents each 4 bits by a single symbol
  - Example: A3 denotes 1010 0011

Bit pattern	Hexadecimal representation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

More jargons

- **Kilobyte:** 2^{10} bytes = 1024 bytes ≈ 10^3 bytes
  - Example: 3 KB ≈ 3 × 10^3 bytes
- **Megabyte:** 2^{20} bytes ≈ 10^6 bytes
  - Example: 3 MB ≈ 3 × 10^6 bytes
- **Gigabyte:** 2^{30} bytes ≈ 10^9 bytes
  - Example: 3 GB ≈ 3 × 10^9 bytes
- **Terabyte:** 2^{40} bytes ≈ 10^{12} bytes
  - Example: 3 TB ≈ 3 × 10^{12} bytes

Outline

- Data representation in bit patterns
- Binary operations and logic gates
- Data storage and transmission
- Data processing

Data Representation in Bit Patterns

Text, number, image, and sound

Text data

- Each character is assigned a unique bit pattern.
- ASCII code
  - **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange
  - Uses **7-bits** to represent most symbols used in English text

01001000	01100101	01101100	01101100	01101111	00101110
H	e	l	l	o	.

  - Quiz: how many different bit patterns can be represented by 7 bits?

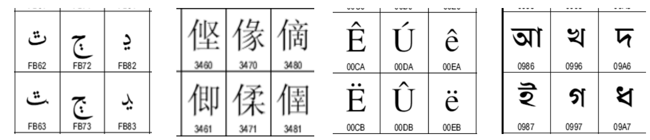
Big5 code

- For Chinese character encoding
- Uses 16 bits to represent a character
  - But does not use all (A140-F9FF)
- Example

我	身	騎	白	馬
A7DA	A8AD	C34D	A5D5	B0A8

Unicode

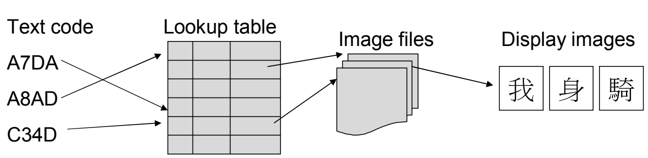
- Uses 16-bits to represent the major symbols used in languages world side



Arabic char CJK char Latin char Indic char

Display characters

- Computer doesn’t show the codes directly to us. It displays what we can read.



- Those images for displaying characters are called fonts.
  - We will talk about images later.

Numbers

- We can use 4 bits to represent decimal digits 0,1,2,3,4,5,6,7,8,9
  - This is called “Binary-coded decimal” (BCD) representation
- Problems
  - We waste last 6 bit-patterns of 4 bits
    - Quiz: How many different numbers can be represented by 4 bits?
  - Difficult to do calculation (+-*/)

	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

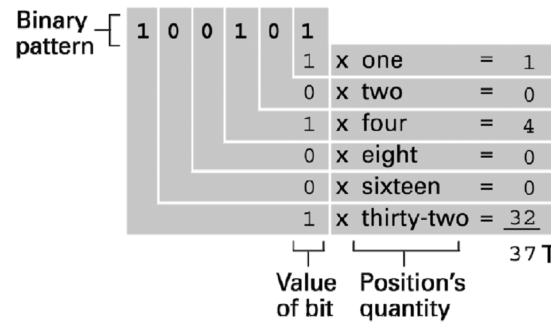
Binary numeral system

- Uses bits to represent a number in base-2

- We put a subscript b to a number for binary, and a subscript d for decimal.
  – 10_d is number ten, and 10_b is number two.

Binary to decimal

- What is the decimal number of 100101_b?



2^0 2^1 2^2 2^3 2^4 2^5

Decimal to binary

- What is the binary number of 13_d?
  – First, how many bits we need for 13.
    - Since $13<16=2^4$, 4 bits can represent 13.

$$13 = \boxed{b_3}\boxed{b_2}\boxed{b_1}\boxed{b_0} = b_3\times8+b_2\times4+b_1\times2+ b_0\times1$$

  – Second, decide b_0 is 0 or 1.
    - Since 13 is odd, b_0 must be 1.
  – Then? How to decide b_1.
    - You can do $(13-b_0)/2 = 6 = b_3\times4+b_2\times2+b_1\times1$.
    - Since 6 is even, b_1 must be 0.

– We can use the same way to decide b_2 and b_3.
  - $(6-b_1)/2 = 3= b_3\times2+b_2\times1$ is odd, so b_2 is 1.
  - $(3-b_2)/2 = 1= b_3\times1$, b_3 must be 1.

– So, $13_d = 1101_b$

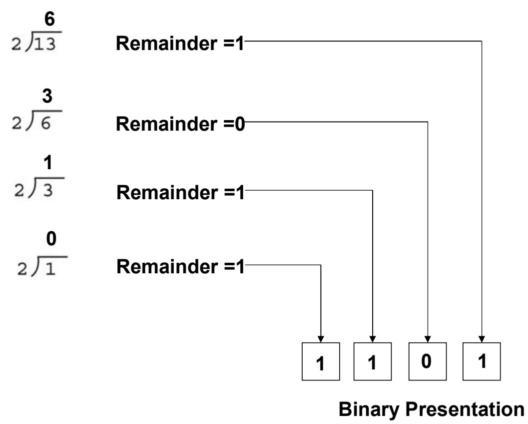
- You have your first algorithm here

Step 1. Divide the value by two and record the remainder.

Step 2. As long as the quotient obtained is not zero, continue to divide the newest quotient by two and record the remainder.

Step 3. Now that a quotient of zero has been obtained, the binary representation of the original value consists of the remainders listed from right to left in the order they were recorded.

Running the algorithm

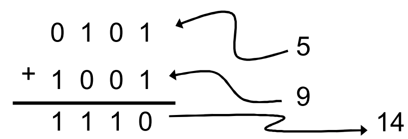


Binary number calculations

- Binary number is easy for calculations
- For example, the one bit addition

```
  0     1     0     1
+ 0   + 0   + 1   + 1
  0     1     1    10
```

- So, what is 5_d+9_d in binary number form?



Another example

```
   111 1
   00111010
+  00011011
   01010101
```

0 +0 0	1 +0 1	0 +1 1	1 +1 10

The Binary Addition Facts

Negative numbers

- How to represent -1, -2, … on a computer?
- Solution 1: use an extra bit to represent the negatives sign.
  - It is called *the sign bit*, in front of numbers.
  - Usually, 0 is for positives; 1 is for negatives.
  - Example: 1 0001 is -1 and 0 0100 is +4
- But how can we to do the addition (-1) + (4) efficiently?

Solution 2

- The negative sign "–" just means the "opposite" or the "inverse".
  - For example, the opposite of east is west. (why is not south or north?)
  - For addition, the inverse of a number d, denoted I(d), has the property: I(d)+d=0.
  - We can use this to define negative numbers.

- If we use four bits to represent a number, zero is 0000, and one is 0001. What is -1?
  - Find b_3, b_2, b_1, b_0 such that

$$\begin{array}{r} b_3\ b_2\ b_1\ b_0 \\ +\ 0\ 0\ 0\ 1 \\ \hline \boxed{1}\ 0\ 0\ 0\ 0 \end{array}$$

This 1 will be "truncated" since it is a 4 bits numbering system.

  - The solution is 1111
  - You can use the same method to find other numbers.
  - Observe: the leading bit is 1 for negative values ➔ *sign bit*

b. Using patterns of length four

Bit pattern	Value represented
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	−1
1110	−2
1101	−3
1100	−4
1011	−5
1010	−6
1001	−7
1000	−8

Two's complement

- A simple algorithm to find the inverse*
  1. Change each bit 0 to 1 and bit 1 to 0
  2. Add 1.

$6_d = 0110_b \rightarrow 1001_b$

$$\begin{array}{r} 1001_b \\ +\ 0001_b \\ \hline 1010_b = -6_d \end{array} \qquad \begin{array}{rr} 0110_b & (\ 6) \\ +\ 1010_b & (-6) \\ \hline 1\ 0000_b & (\ 0) \end{array}$$

truncated

- This number representation is called the "two's complement".
- *Textbook uses a different algorithm

Exercises

- What are the decimal numbers for the following 2's complement representations?

(a) 00000001 (b) 01010101 (c) 11111001
(d) 10101010 (e) 10000000 (f) 00110011

- Find the negative value represented in 2's complement for each number

Calculation with 2's complement

- Calculation can be made easily for two's complement representation.
  - Example

Problem in base ten		Problem in two's complement		Answer in base ten
3 + 2	→	0011 + 0010 = 0101	→	5
−3 + −2	→	1101 + 1110 = 1011	→	−5
7 + −5	→	0111 + 1011 = 0010	→	2

Overflow

- What is 5+4?

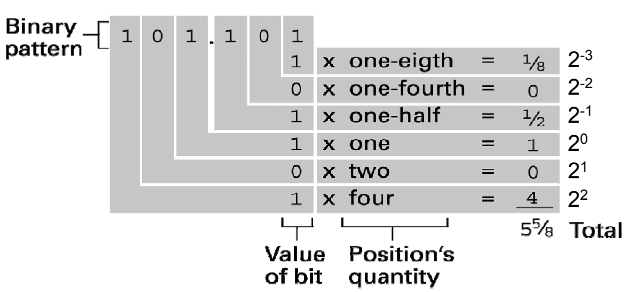
  $5_d+4_d=0101_b + 0100_b=1001_b$
- This is called overflow
  - Adding two positive numbers results a negative number; or adding two negative numbers results a positive number.
  - A 4 bits 2's complement system can only represent 7~ –8

b. Using patterns of length four

Bit pattern	Value represented
0111	7
0110	6
0101	5
0100	4
0011	3
0010	2
0001	1
0000	0
1111	−1
1110	−2
1101	−3
1100	−4
1011	−5
1010	−6
1001	−7
1000	−8

Fractions

- The binary number of fractions.
  - Problem: where to put the decimal point?

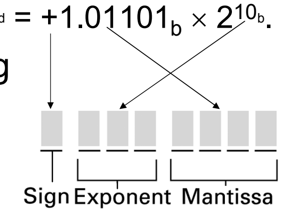


Floating point

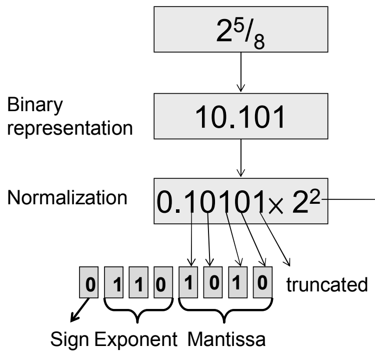
- To represent a wide range of numbers, we allow the decimal point to "float".

  $40.1_d = 4.01_d \times 10^1 = 401_d \times 10^{-1} = 0.401_d \times 10^2$
  - It is just like the scientific notation of numbers.

  $101.101_b = +1.01101_b \times 2^{2_d} = +1.01101_b \times 2^{10_b}$.
- This is called the floating point representation of fractions.



Coding the value of $2^5/_8$



- Exponent uses excess notation

Bit pattern	Value represented
111	3
110	2
101	1
100	0
011	−1
010	−2
001	−3
000	−4

Truncation error

- Mantissa field is not large enough
  - $2^5/_8 = 2.625 \Rightarrow$ **2.5 + round off error (0.125)**
- Nonterminating representation
  - $0.1 = {}^1/_{16} + {}^1/_{32} + {}^1/_{256} + {}^1/_{512} + \ldots$
  - Change the unit of measure
- Order of computation:
  - $2.5 + 0.125 + 0.125 \Rightarrow$ **2.5 + 0 + 0**

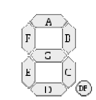
Exercises

- What are the fractions for the following floating number representations?
  - Suppose 1 bit for sign, 3 bits for exponent (using excess notation), 4 bits for mantissa

  (a) 01001010 (b) 01101101 (c) 11011100 (d) 10101011
- If direct truncation is used, what are the ranges of their possible values?

Images

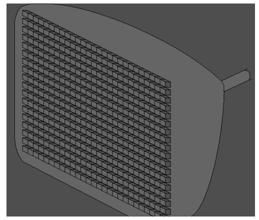
- Image representation depends on what the output device can display.
  - For example, an image on the seven segment can be represented by 7 bits.

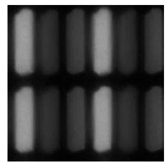


No	Img	Repre.						
0		1111110	4		0110011	8		1111111
1		0110000	5		1011011	9		1111011
2		1101101	6		1011111	A		1110111
			3		1111001	7		1100000

Common output devices

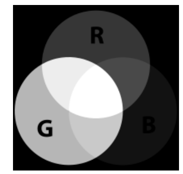
- The cathode ray tube (CRT) uses raster scan.
- The liquid crystal display (LCD) is consisted of an array of crystal molecules.
- Most printers use dots to compose images.

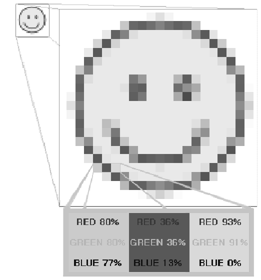




Raster image (bitmap)

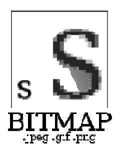
- Represent an image by a rectangular grid of pixels (short for "picture element")
- Each pixel is composed by three values: R, G, B.





Vector graph image

- When scaled up, a bitmap image shows the zigzag effect.
- Vector graph images store the mathematical formula for lines, shapes and colors of the objects in an image.
  - Example: TrueType font
- Rasterisation:
  - a process converting vector graph to raster image.



Courier New AAAAA

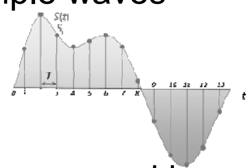
Courier AAAAA

Sound

- Sound is an acoustic wave
  - A simple wave can be characterized by amplitude and frequency.
    - The larger amplitude the louder the sound
    - The higher frequency the higher pitch
  - All sound can be composed by simple waves.
- MIDI file
  - Represent sounds by the amplitude and frequency of composed simple waves.

Sampled sound

- The sound composed by simple waves may not sound real.
- Alternatively, sampling the real sound and **record** it



- Quality of sampled sound is measured by
  - Sampling rate: how often to do the sampling
  - Bit depth: bits used for one sample
  - CD music has sampling rate 44.1kHZ and uses 16 bits for each sample.

Video

- Digital video is composed by a sequence of "continuous" images and synchronized sound tracks
  - Each image is called a "frame"
  - Each frame is flashed on a screen for a short time (1/24 seconds or 1/30 seconds)

Binary Operations and Logic Gate

Basic operations for binary data and the physical devices to implement them.

○ Logic data

- Logic data: either true or false.
- Logic operation
  - If the room is dark and someone is in the room, turn on the light.
- Use binary (0/1) representation

Room is dark { Yes (1) / No (0) }

Someone in the room { Yes (1) / No (0) }

Light is on { Yes (1) / No (0) }

The AND function

- We can use the AND function to represent the statement

Room is dark A	Someone in the room B	Light is on A .AND. B
0	0	0
0	1	0
1	0	0
1	1	1

Input Output

Boolean operators

- The AND function is a Boolean operator.
- Boolean operator is an operation that manipulates one or more 0/1 values.
- Other common Boolean operations

OR

Input		Output
0	0	0
0	1	1
1	0	1
1	1	1

XOR (exclusive or)

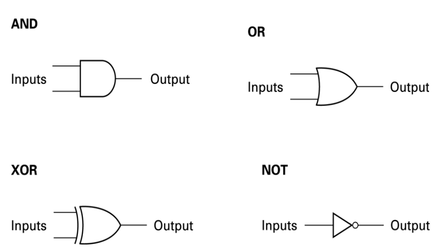
Input		Output
0	0	0
0	1	1
1	0	1
1	1	0

NOT

Input	Output
0	1
1	0

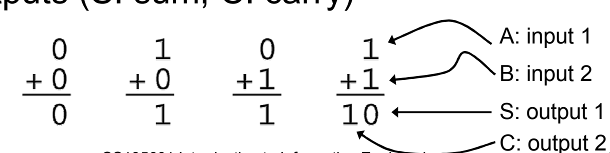
Logic gate

- We call a device that implements a Boolean operation a gate
- Pictorial representation of gates



Example

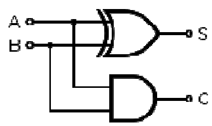
- Almost all operations of computers can be carried out by logic gates
  - The textbook uses flip-flop as an example.
  - We will use "one bit adder" as an example.
- One bit adder has two inputs and two outputs (S: sum, C: carry)



Implementation of one bit adder

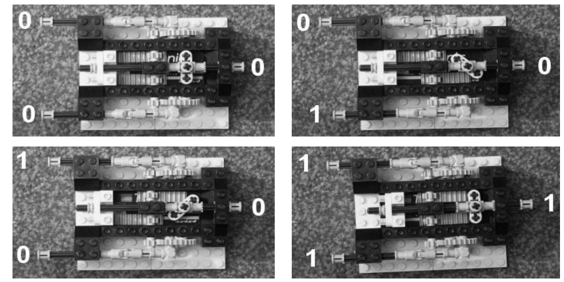
- The truth table of an one-bit adder
- Compare it to the truth table of Boolean function AND, OR, XOR, NOT
  - S = A .XOR. B
  - C = A .AND. B

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

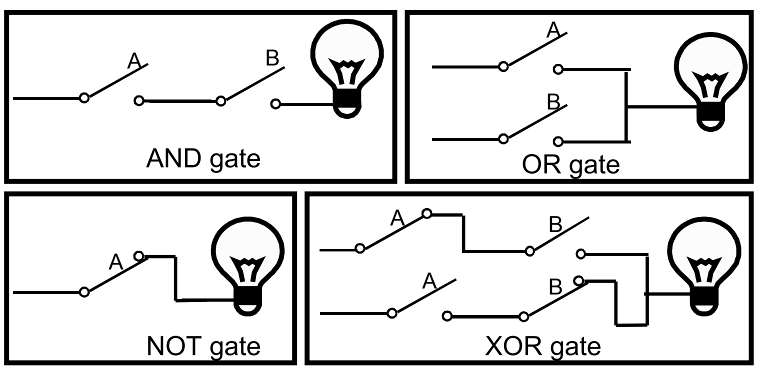


What can be a gate?

- LEGO's "mechanical gates"
  - The AND gate



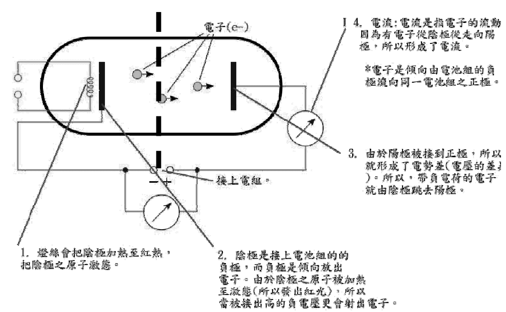
Circuit gates

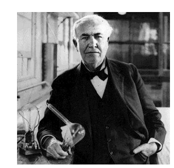


- Can we flip the switches without hands?

Electronic switch

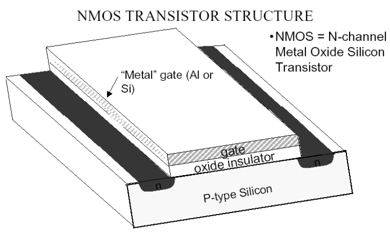
- The earliest one is the vacuum tube
  - 1884, Thomas Edison





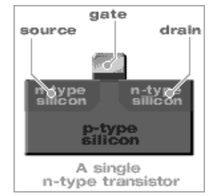
Transistor

- The problems of vacuum tubes are slow, large, expensive, easy to break.
- Transistor can make it faster, smaller, and more robust.



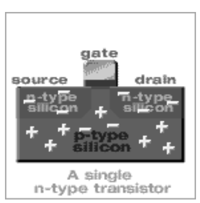
How transistor works (1/5)

- Transistors consist of three terminals; the source, the gate, and the drain:



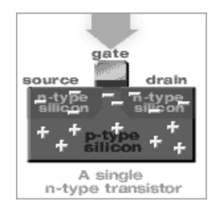
How transistor works (2/5)

- In the n-type transistor, both the source and the drain are negatively-charged and sit on a positively-charged well of p-silicon.



How transistor works (3/5)

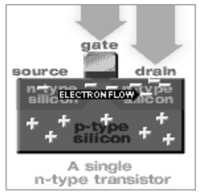
- When positive voltage is applied to the gate, electrons in the p-silicon are attracted to the area under the gate forming an electron channel between the source and the drain.



How transistor works (4/5)

- When positive voltage is applied to the drain, the electrons are pulled from the source to the drain.
  In this state the transistor is on.

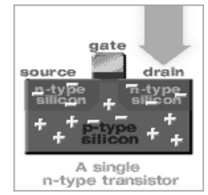
開



How transistor works (5/5)

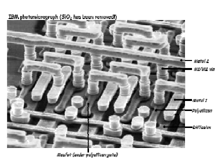
- If the voltage at the gate is removed, electrons are not attracted to the area between the source and drain. The pathway is broken and the transistor is turned off.

關



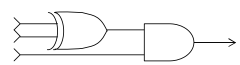
Integrated circuit (IC)

- An electronic circuit consisted of transistors and other components in the thin substrate of semiconductor material.
- Also known as **IC**, **microchip**, or **chip**.
- Invented by Jack Kilby and Robert Noyce
  - 2000 Nobel Prize in Physics
- **VLSI**: Very-Large-Scale IC
  - More than million transistors

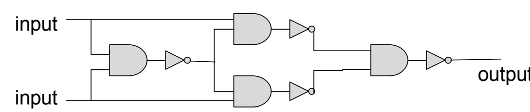


Exercises

- What input bit patterns will cause the following circuit to output 1? And output 0?



- What Boolean operation does the circuit compute?



Data storage and transmission

Memory, RAM, address

CD/DVD, hard disk, flash memory

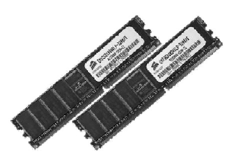
signal, communication media

○ Storage media

- Physical objects that can store bits and retrieve them can be a storage media.
- Volatile (temporary) memory:
  - DRAM, SRAM, SDRAM
- Non-volatile storage (massive storage)
  - Optical Systems: CD, DVD
  - Magnetic Systems: Hard disk, tape
  - Flash drives: iPod, Cell Phone, USB drivers…

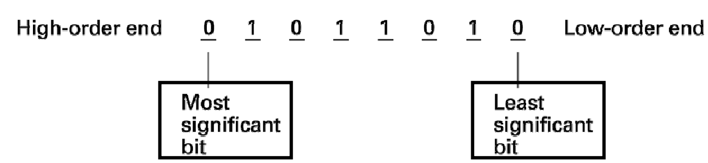
○ Memory

- Memory is used inside computers for temporary storages.
- They are often called RAMs
  - **R**andom **A**ccess **M**emory: data can be accessed in any order
  - Dynamic RAM (DRAM):
  - Synchronous DRAM (SDRAM)
  - Static RAM (SRAM)



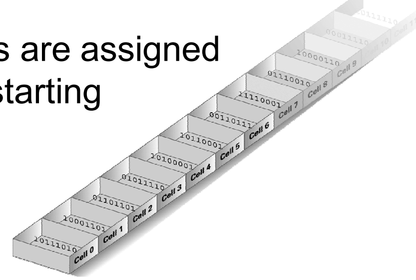
○ Data storage unit

- To efficiently access data, computers use 8 bits (a byte) as a smallest storage unit.
- Some jargons for a byte
  - **Most significant bit:** at the high-order end
  - **Least significant bit:** at the low-order end



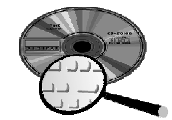
Memory address

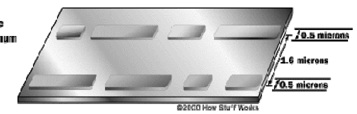
- Each storage unit in memory is numbered by an address so that data can be stored and loaded.
  - These numbers are assigned consecutively starting at zero.

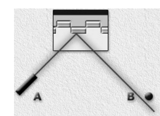


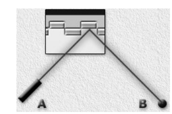
CD/DVD

- CD: Compact Disk
- DVD: Digital Video Disk
  - Use bumps to represent 0/1



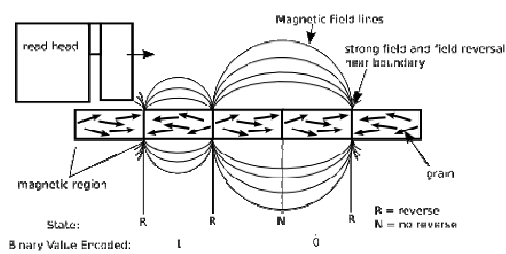




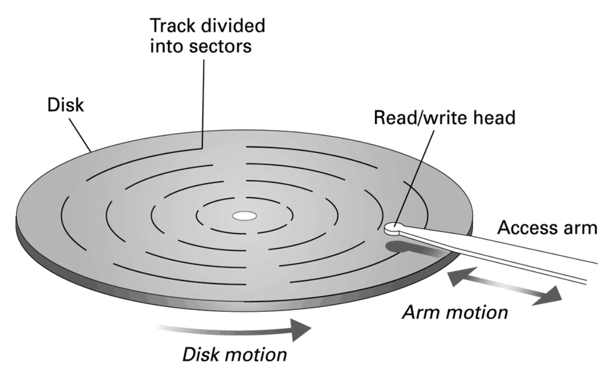


Hard disks (HDD)

- A hard platter holds the magnetic medium
  - Use magnetic field to represent 0/1

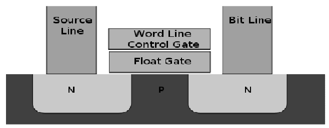


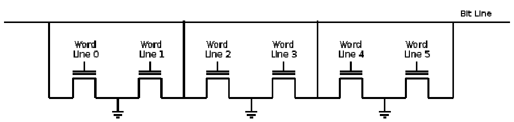
Some terms of hard disk

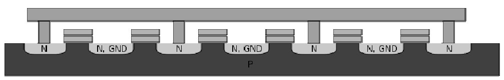


Flash memory

- Use electrical charge to represent 0/1





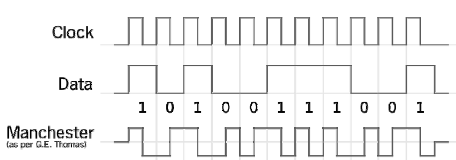
Files

- File is the basic storage unit in massive storages that contains related data.
  – Text documents, photos, mp3,…
- A file is associated with many attributes
  – File name, file name extension
  – Size, modified date, read only, etc.
- It requires a system to store, retrieve, and organize files.*

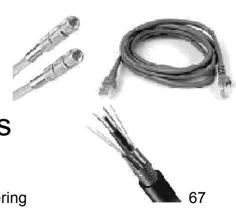
*We will study the operating system in chapter 3.

Data transfer

- Many media can transfer binary data
  – Voltage
  – Voltage change



  – Voice: telephone line (modem)
  – Electromagnetic wave: radio
  – Light: infrared, laser, fiber optics



Data communication rates

- Measurement units
  – Bps: Bits per second
  – Kbps: Kilo-bps (1,000 bps)
  – Mbps: Mega-bps (1,000,000 bps)
  – Gbps: Giga-bps (1,000,000,000 bps)
- Multiplexing: make single communication path as multiple paths
- Bandwidth: maximum available rate

Data Processing

Compression, error correction, encryption

Data compression

- Purpose: reduce the data size so that data can be stored and transmitted efficiently.
- For example,
  - 0000000000111111111 can be compressed as (10,0,9,1)
  - 123456789 can be compressed as (1,1,9)
  - AABAAAABAAC can be compressed as 11011111011100, where A, B, C are encoded as 1, 01, and 00 respectively.

Many compression techniques

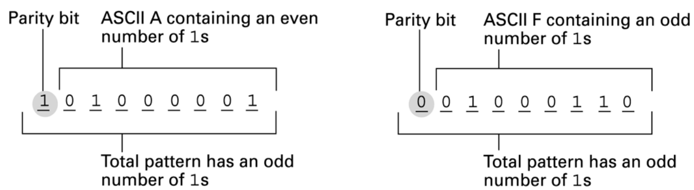
- Lossy versus lossless
- Run-length encoding
- Frequency-dependent encoding (Huffman codes)
- Relative encoding
- Dictionary encoding (Includes adaptive dictionary encoding such as LZW encoding.)

Different data has different compression methods

- Image data
  - GIF: Good for cartoons
  - JPEG: Good for photographs
  - TIFF: Good for image archiving
- Video: MPEG
  - High definition television broadcast
  - Video conferencing
- Audio: MP3
  - Temporal masking, frequency masking

Error detection

- During transmission, error could happen.
  - For example, bit 0 →1 or bit 1 →0.
- How could we know there is an error?
  - Adding a parity bit (even versus odd)



Error correction

- Can we find a way that not only detects an error, but also corrects errors?
- Yes, by carefully design the code

Character	Code	Pattern received	Distance between received pattern and code
A	0 0 0 0 0 0	0 **1** 0 **1** 0 0	2
B	0 0 1 1 1 1	0 **1 0** 1 **0 0**	4
C	0 1 0 0 1 1	0 1 0 **1 0 0**	3
D	0 1 1 1 0 0	0 1 **0** 1 0 0	**1** ← Smallest distance
E	1 0 0 1 1 0	**0 1** 0 1 **0** 0	3
F	1 0 1 0 0 1	**0 1 0 1** 0 **0**	5
G	1 1 0 1 0 1	**0** 1 0 1 0 **0**	2
H	1 1 1 0 1 0	**0** 1 **0 1 0** 0	4

Exercises

- Using the error correction code table to decode the following message

001111 100100 001100 010001
000000 001011 011010 110110
100000 011100

Character	Code
A	0 0 0 0 0 0
B	0 0 1 1 1 1
C	0 1 0 0 1 1
D	0 1 1 1 0 0
E	1 0 0 1 1 0
F	1 0 1 0 0 1
G	1 1 0 1 0 1
H	1 1 1 0 1 0

- The following bytes are encoded using odd parity. Which of them definitely has an error

(a) 10101101 (b) 10000001 (c) 11100000 (d) 11111111

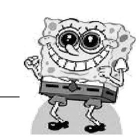
Data encryption

- Suppose Alice wants to send a secret message, 10110101, to Bob
  - If they both know a key, 00111011, that no one else knows.
  - Alice can send the encrypted message to Bob using XOR, and Bob can decrypt it the same way

```
    10110101
XOR 00111011
------------
    10001110
```

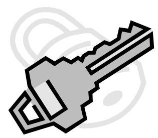
→

```
    10001110
XOR 00111011
------------
    10110101
```

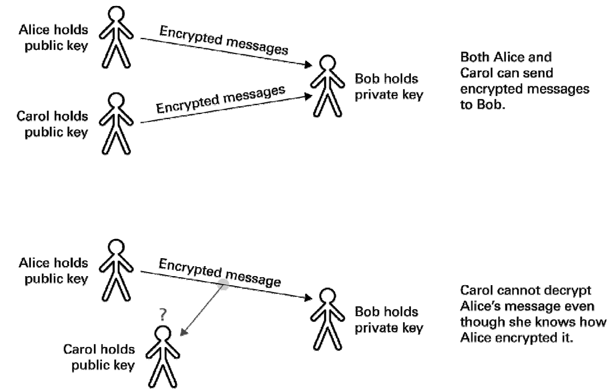


Secret key encryption

- This is called the secret key encryption.
- If no one else knows the secret key and the key is generated randomly and used only once, this is a very good encryption algorithm
- Problems:
  – the key can be used only once
  – Alice and Bob both need to know the key



Public key encryption



Related courses

- Data storage, representation, processing
  – 計算機結構
  - Data transfer 計算機網路概論
- Gates, transistors
  – 數位邏輯設計, 電子電路, 積體電路設計簡介
- Data compression, correction,
  – 影像處理，資訊檢索，多媒體技術概論
- Data encryption
  – 離散數學，離散結構專題，密碼與網路安全概論

References

- http://computer.howstuffworks.com/
- http://en.wikipedia.org/
- http://www.weethet.nl/english/
- http://goldfish.ikaruga.co.uk/logic.html
- http://www.mandarinpictures.com/stephenzinn/images/a a-raster-1.gif
- Textbook: most materials are from chapter 1
  – Communication media is in 2.5
  – Vector graph and rasterization are in 10.4