

Introduction to Computer Science

Lecture 1: DATA STORAGE

Instructor: Tian-Li Yu

Taiwan Evolutionary Intelligence Laboratory (TEIL)
Department of Electrical Engineering
National Taiwan University

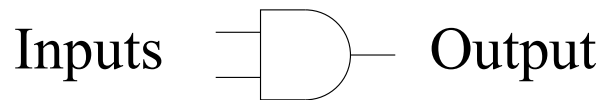
tianliyu@cc.ee.ntu.edu.tw

Slides made by Tian-Li Yu, Jay-Wie Wu, and Chu-Yu Hsu

Binary World

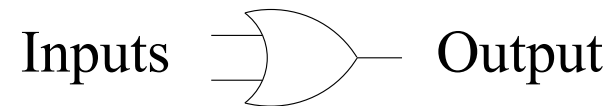
- **Bit**: binary digit (0/1)
- Simple, logical, and unambiguous
- Boolean operations & gates

AND



Inputs	Output
0 0	0
0 1	0
1 0	0
1 1	1

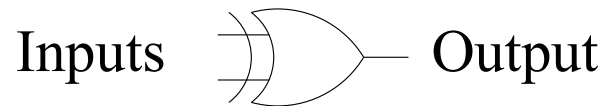
OR



Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	1

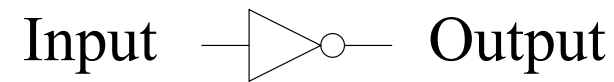
Logical Gates

XOR



Inputs	Output
0 0	0
0 1	1
1 0	1
1 1	0

NOT

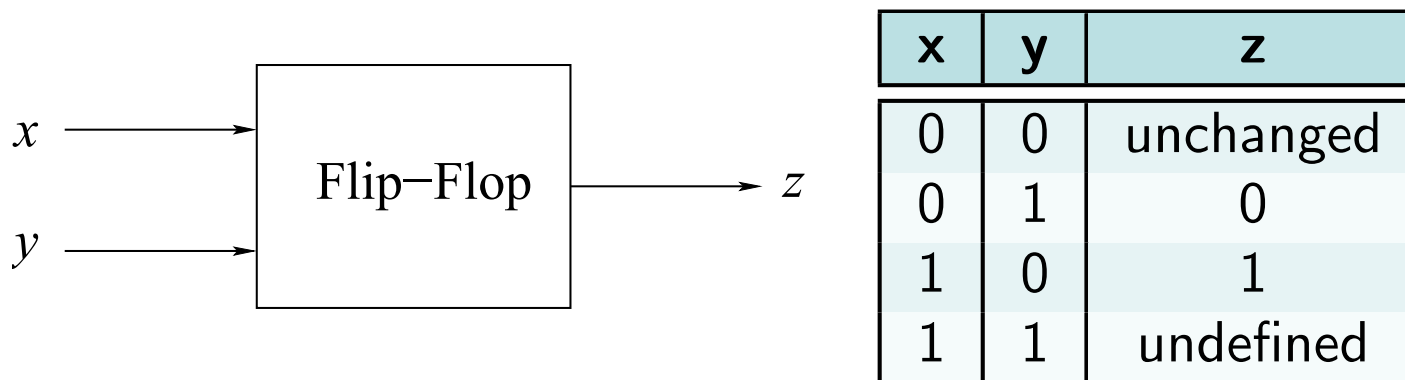


Inputs	Output
0	1
1	0

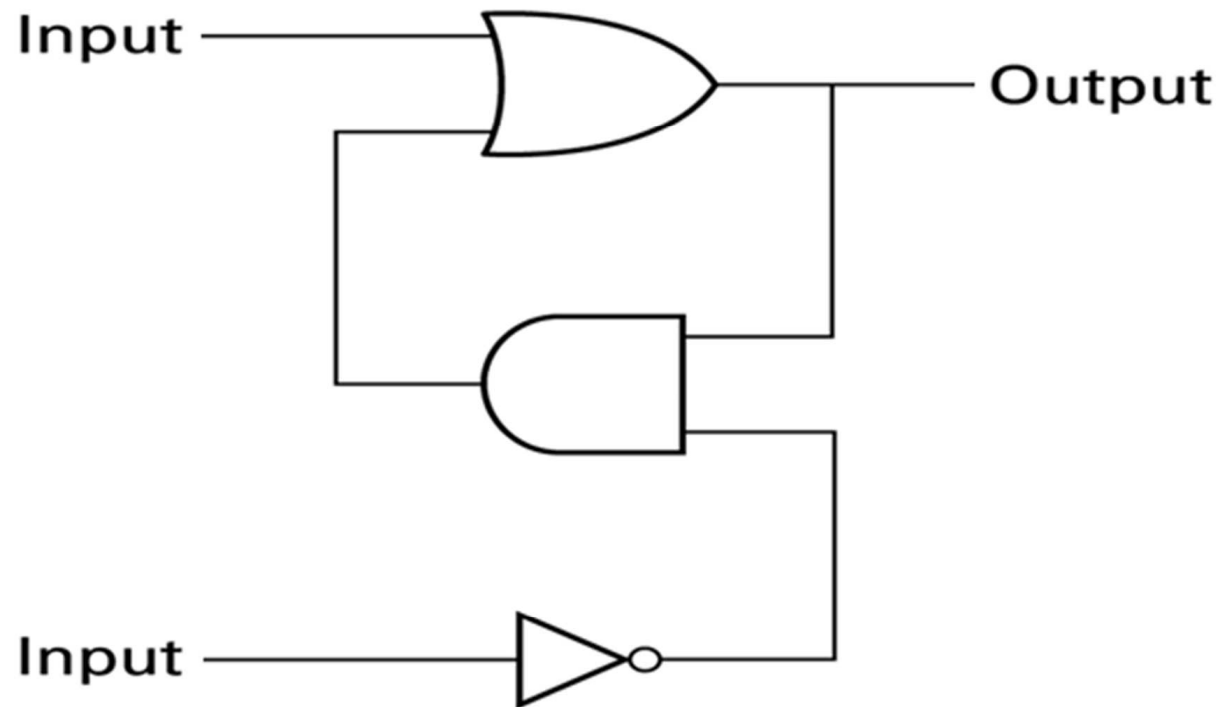
- Logical vs. real world
 - To be or not to be \rightarrow always TRUE.

Flip-Flop

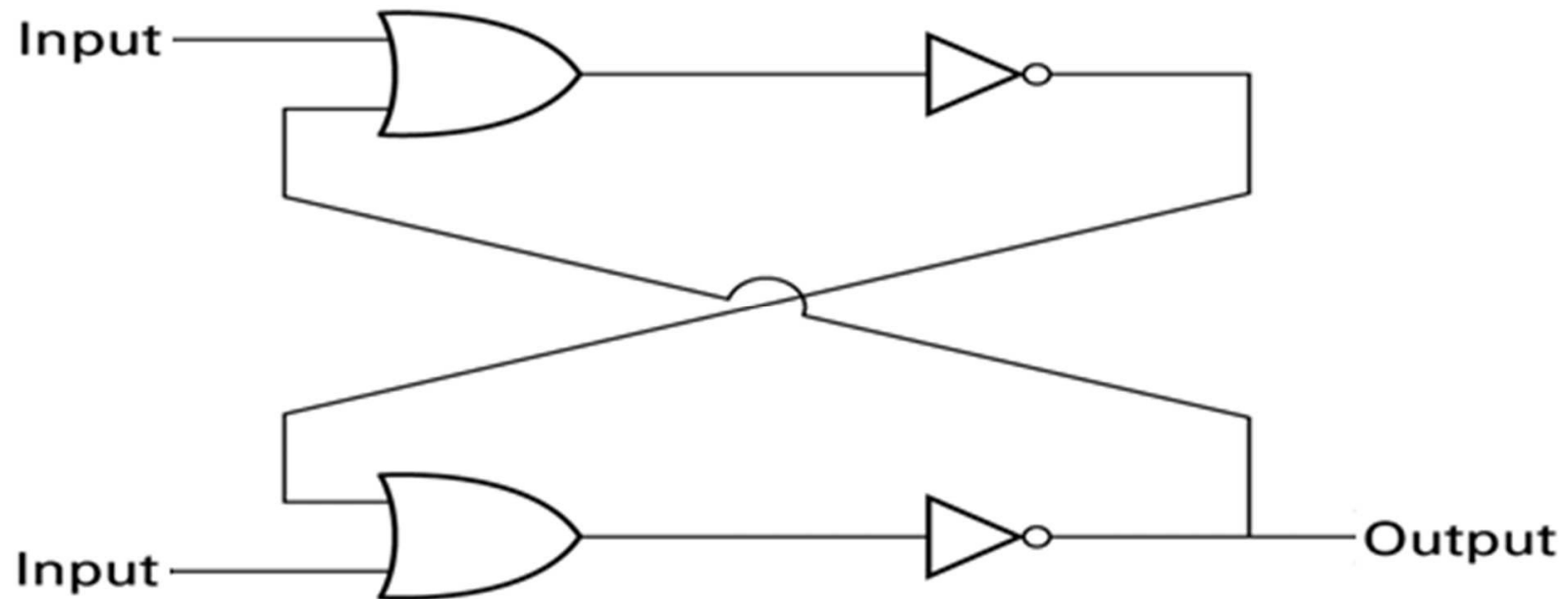
- **Purpose:** to keep the state of output until the next excitement.
- SR Flip-Flop
 - Has two input lines: set and reset.
 - One input its stored value to 1.
 - The other input sets its stored value to 0.
 - While both inputs are 0, the most recently stored value is preserved.



A Simple SR Flip-Flop Circuit



Another SR Flip-Flop Circuit



Hexadecimal Coding (Hex)

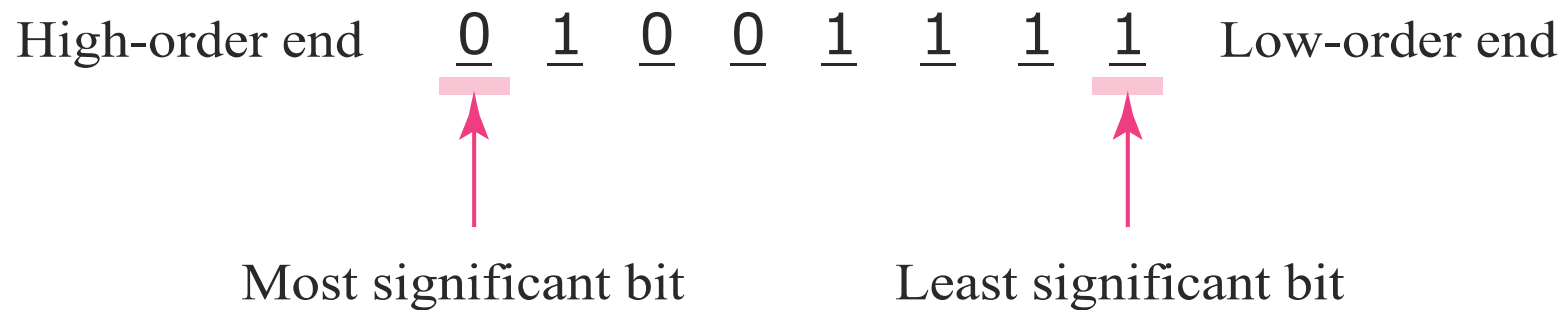
Bit pattern	Hexadecimal representation
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

- Binary is usually too long for human to remember.
- Binary to Hex is straightforward.
- 0010**1110**1011**0101**
→ 2**EB5**.

Main Memory Cells

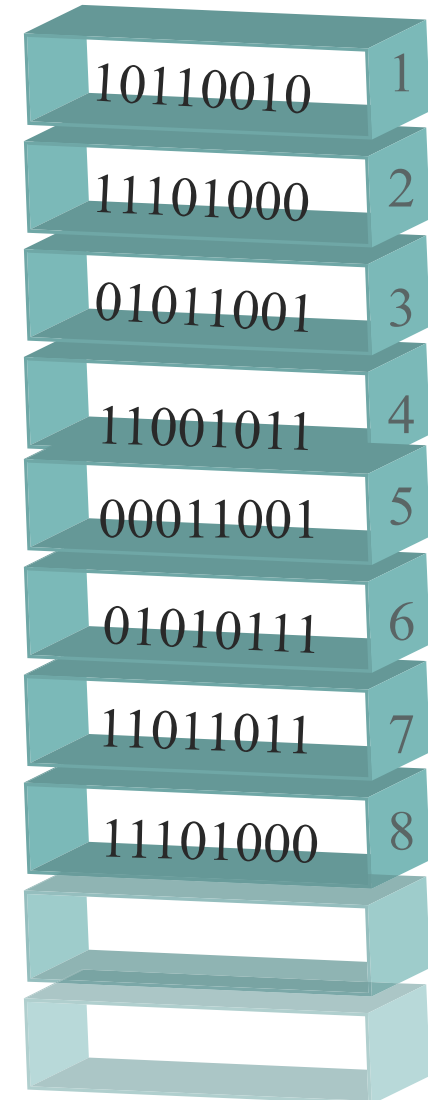
- **Cell**: A unit of main memory (typically 8 bits which is one **byte**)

Cell: 主記憶體的單位、8bits = 1 byte



Main Memory and Address

- One dimensional.
- Random accessible.
- Access the content by the **address** (practically, also in binary).
- Recall the **pointer** in C/C++.



Memory Techniques

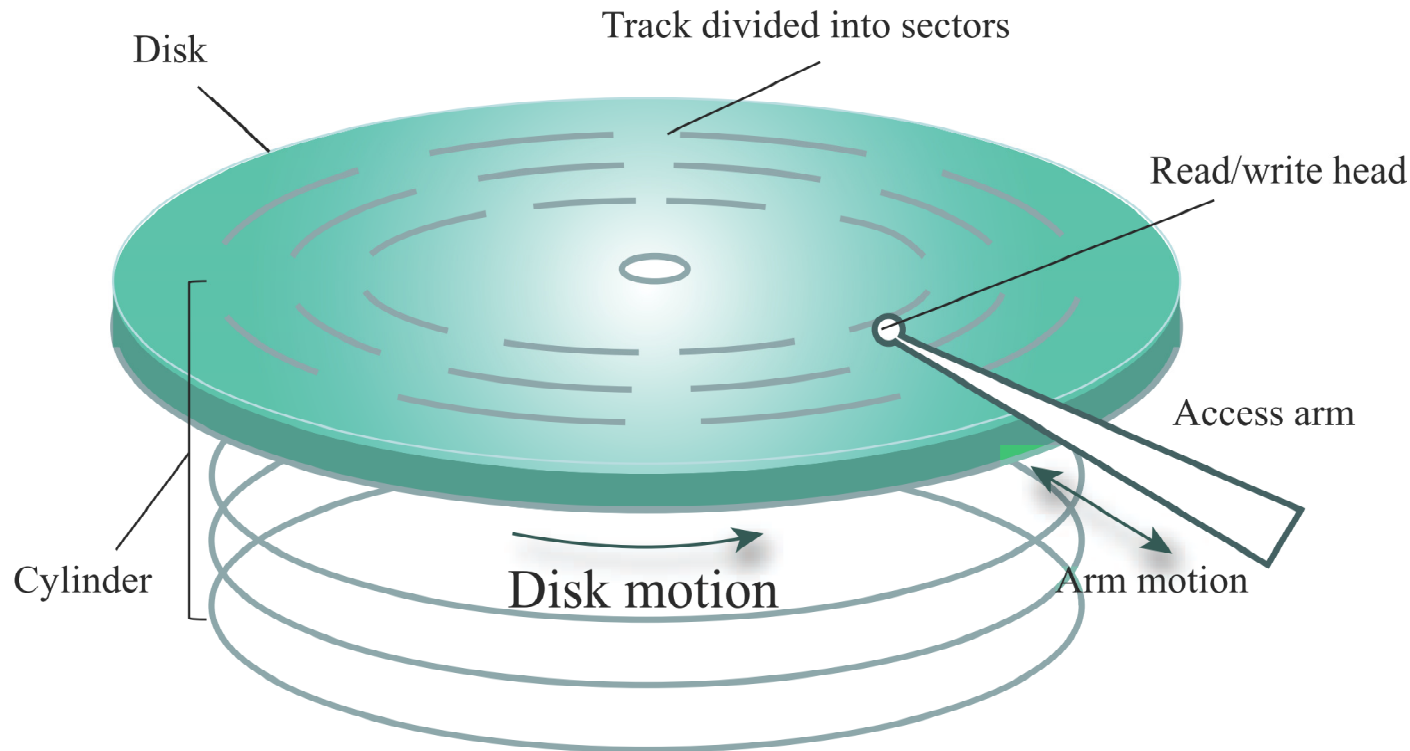
RAM: cell 中的記憶體可被任意access

- **Random Access Memory (RAM)**: Memory in which individual cells can be easily accessed in any order.
 - **Static Memory (SRAM)**: like flip-flop.
 - **Dynamic Memory (DRAM)**: Tiny capacitors replenished regularly by refresh circuit.
 - **Synchronous DRAM (SDRAM)**
 - **Double Data Rate (DDR)**
 - **Dual/Triple channel**
- **Capacity**
 - **Kilobyte**: 2^{10} bytes = 1,024 bytes $\simeq 10^3$ bytes.
 - **Megabyte**: 2^{20} bytes = 1,048,576 bytes $\simeq 10^6$ bytes.
 - **Gigabyte**: 2^{30} bytes = 1,073,741,824 bytes $\simeq 10^9$ bytes.

Mass Storage

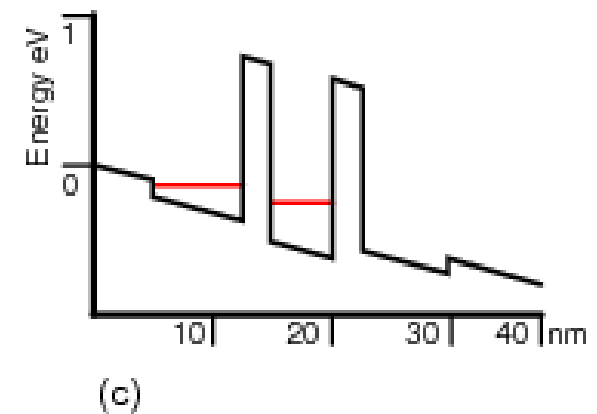
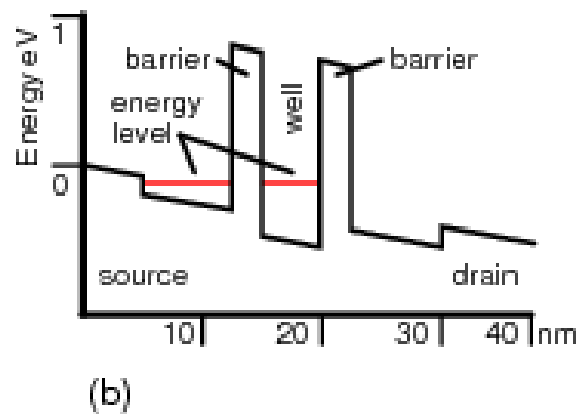
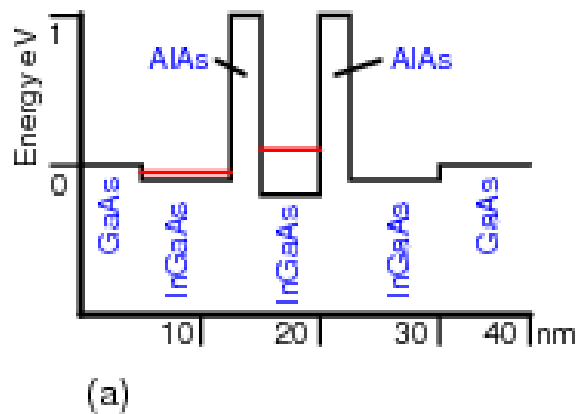
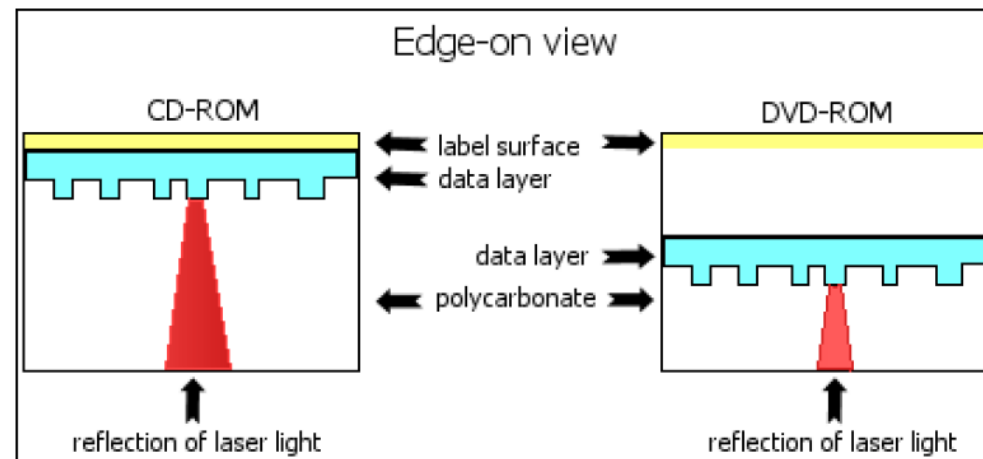
- Properties (compared with main memory)
 - Larger capacity
 - Less volatility
 - Slower
 - On-line or off-line
- Types
 - Magnetic systems (hard disk, tape)
 - Optical systems (CD, DVD)
 - Flash drives

Magnetic Disk Storage System

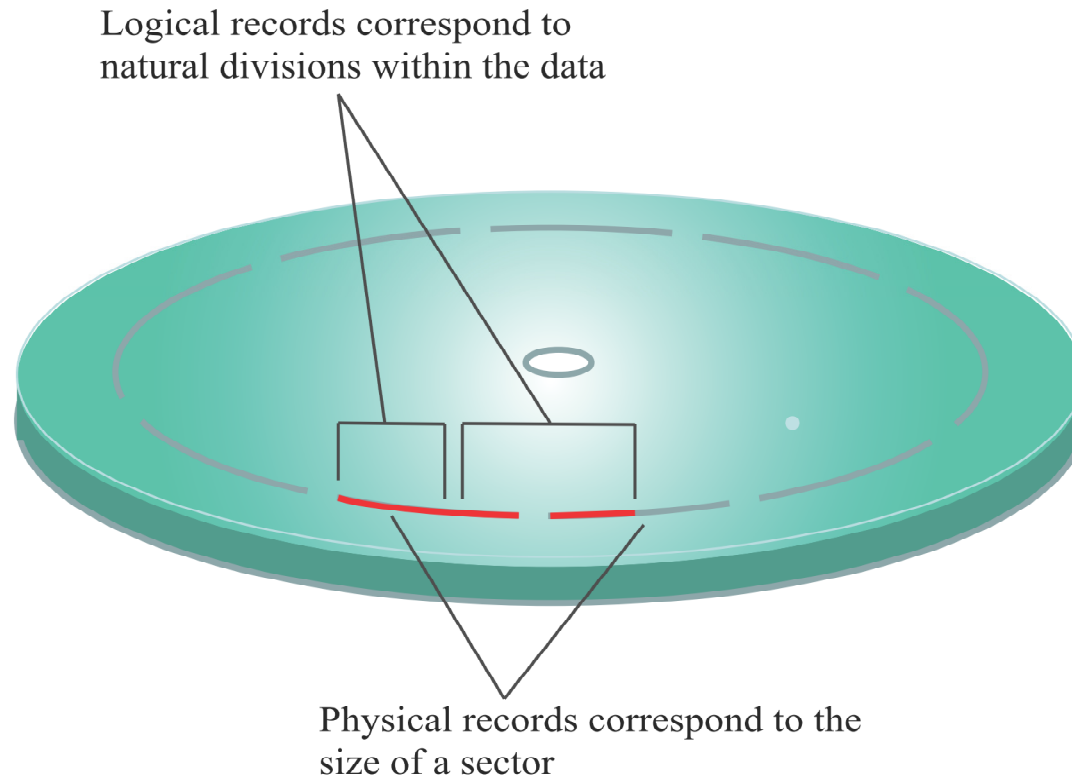


- Head, track, sector, cylinder
- Access time = seek time + rotation delay / latency time.
- Transfer rate (SATA 1.5/3/6, etc.)

Optical and Flash Storages



Physical vs. Logical Records



- Files and file systems
- Fragmentation problem
- We talk about this later in OS.

Buffer

- Purpose: To synchronize (or to make compatible) different R/W mechanisms and rates.
- A memory area used for the temporary storage of data (usually as a step in transferring the data).
- Blocks of data compatible with physical records can be transferred between buffers and the mass storage system.
- Data in buffer can be referenced in terms of logical records.

Representing Text

- **ASCII** (American standard code for information interchange by ANSI): 7 bits (or 8 bits with a leading 0).
- **Unicode**: 16 bits.
- **ISO standard** (international organization of standardization): 32 bits.

ASCII Example

ASCII Code Chart

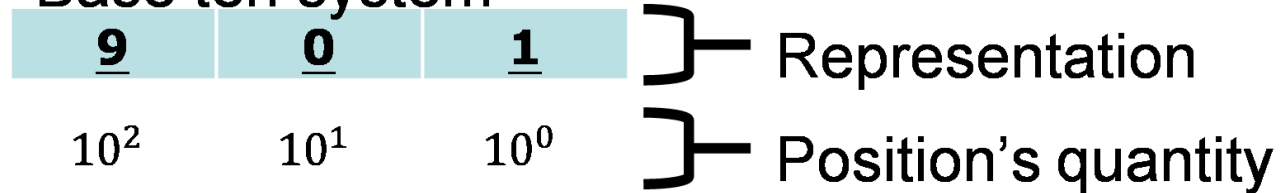
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL



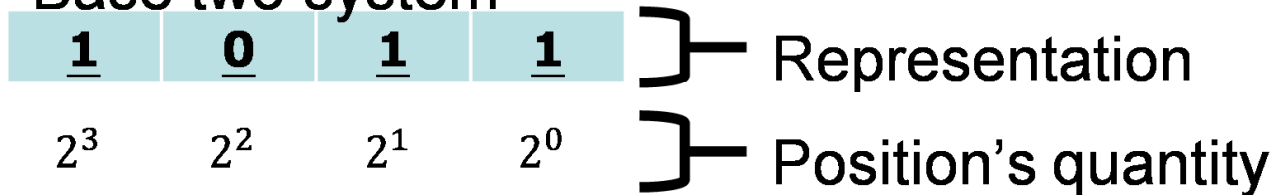
01001000	01100101	01101100	01101100	01101111	00101110
H	e	l	l	o	.

Representing Numeric Values

Base ten system



Base two system



From Binary to Decimal

Base two system

<u>1</u>	<u>0</u>	<u>1</u>	<u>1</u>
-----------------	-----------------	-----------------	-----------------

1	0	1	1
---	---	---	---

×	×	×	×
---	---	---	---

2^3	2^2	2^1	2^0
-------	-------	-------	-------

8	0	2	1
---	---	---	---

Total	11
-------	----

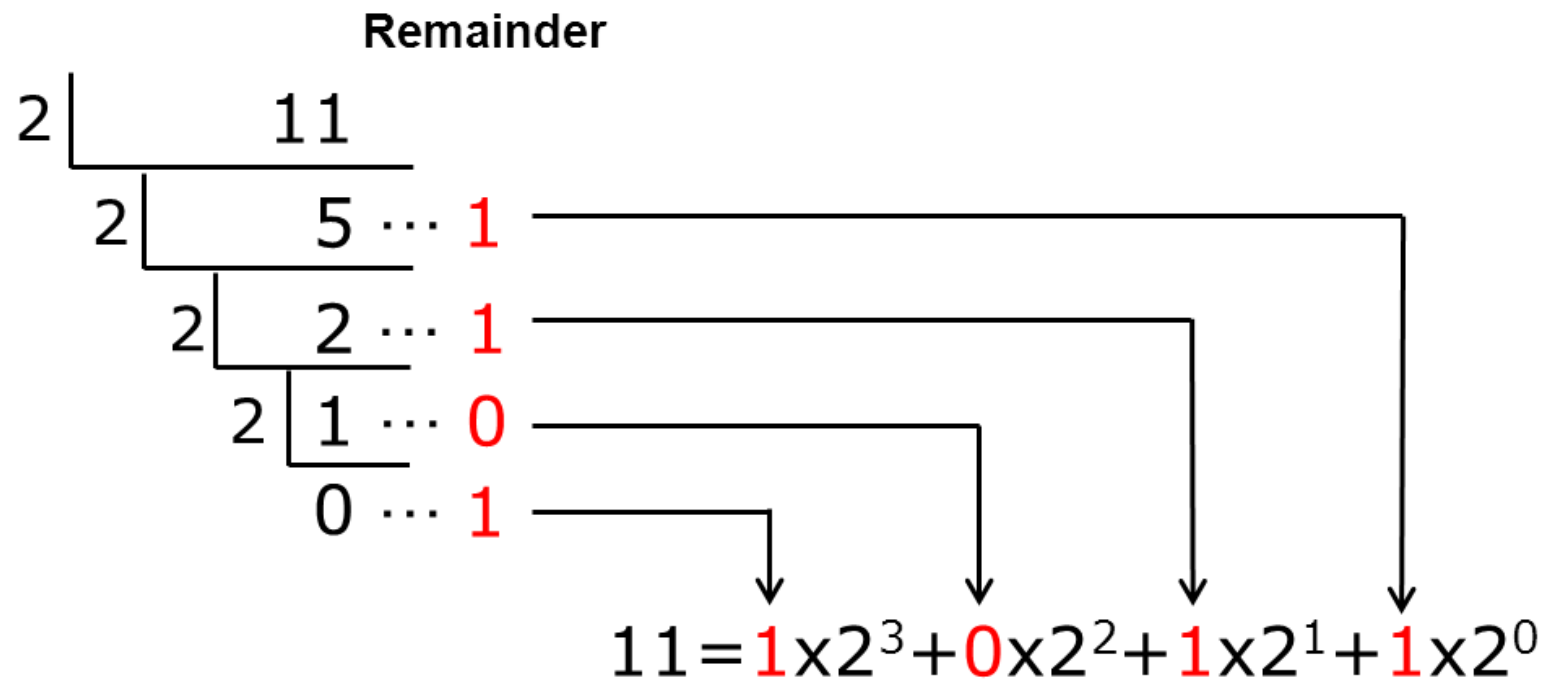
— Binary pattern

— Bit's value

— Position's quantity

From Decimal to Binary

- Just as in decimal, keep dividing the number by 2 and record the remainders.
- Be careful about the order.

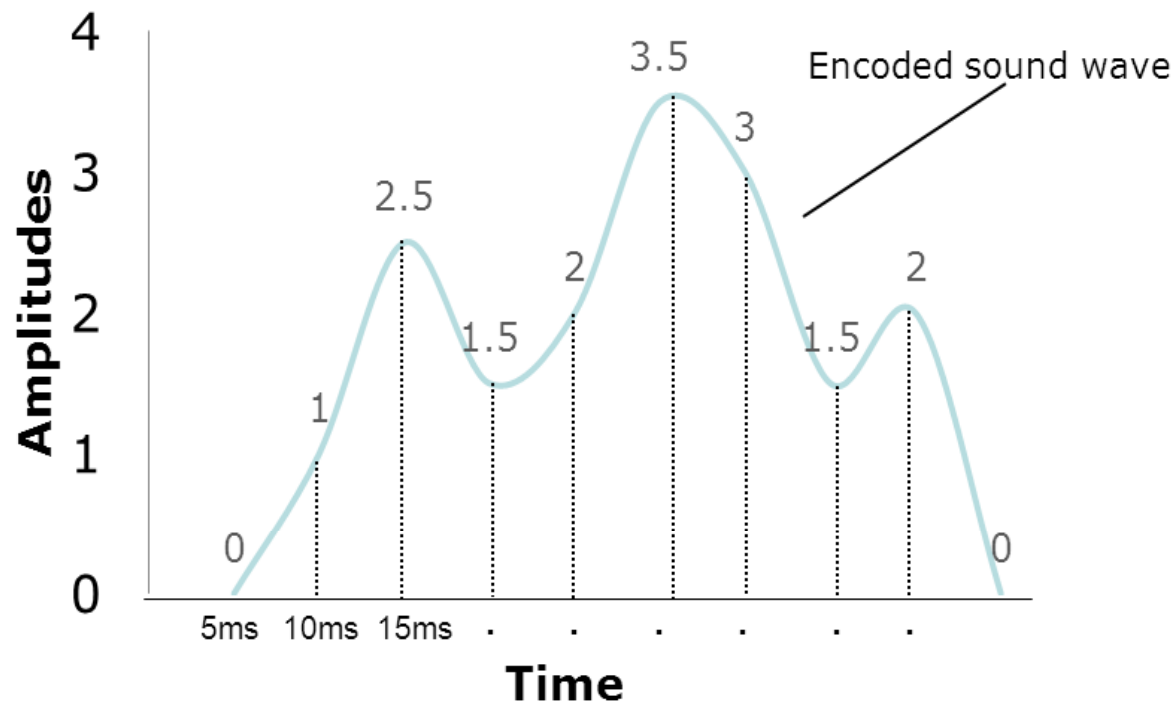


Representing Images

- Bit map techniques
 - **Pixel**: picture element.
 - Colors: RGB, HSV, etc.
 - LCD, scanner, digital cameras, etc.
- Vector techniques
 - Scalable
 - TrueType, Postscript, SVG (scalable vector graphics), etc.
 - CAD, printers.

Representing Sounds

- Sampling
 - Sampling rate
 - Bit resolution
 - Bit rate (sampling rate \times bit resolution)
- MIDI (synthesis)



Binary System Revisited

- Addition

$$\begin{array}{r} 0 \\ + 0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 0 \\ + 1 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ + 0 \\ \hline 1 \end{array}$$

$$\begin{array}{r} 1 \\ + 1 \\ \hline 10 \end{array}$$

- Subtraction?

- Let's first define negative numbers.

Two's Complement Notation

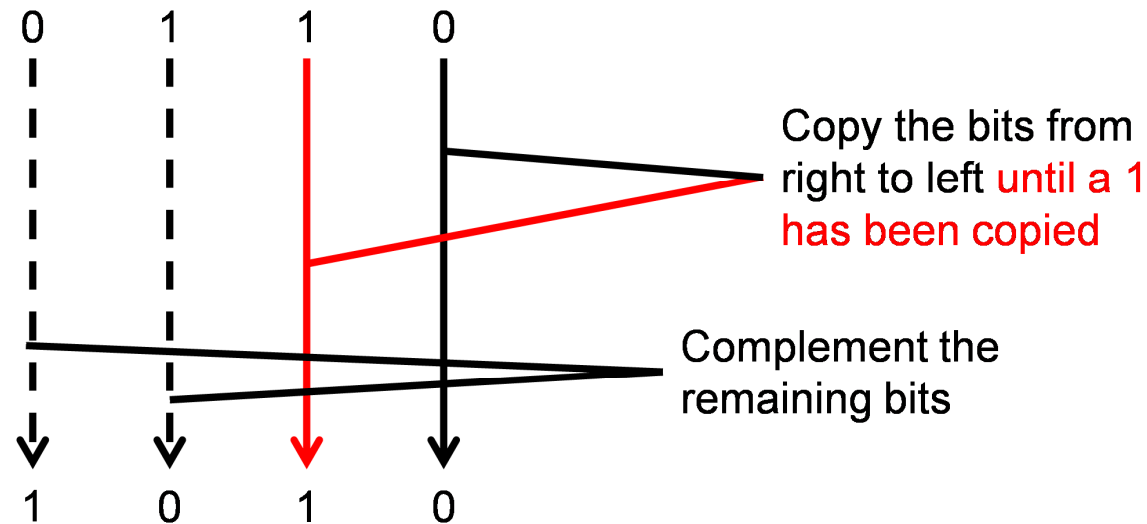
- Range: $-2^{n-1} \sim 2^{n-1} - 1$

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

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 Encoding

- Textbook's way



- My way

For positive x ,

- $x \rightarrow$ binary encoding of x .
- $-x \rightarrow$ binary encoding of $(2^n - x)$.

Subtraction in 2's Complement

- Do it as usual in binary.

$$\begin{array}{r}
 3 \\
 + 2 \\
 \hline
 ?
 \end{array}
 \Rightarrow
 \begin{array}{r}
 0011 \\
 + 0010 \\
 \hline
 0101
 \end{array}
 \Rightarrow 5$$

$$\begin{array}{r}
 -3 \\
 + -2 \\
 \hline
 ?
 \end{array}
 \Rightarrow
 \begin{array}{r}
 1101 \\
 + 1110 \\
 \hline
 1011
 \end{array}
 \Rightarrow -5$$

$$\begin{array}{r}
 7 \\
 + -5 \\
 \hline
 ?
 \end{array}
 \Rightarrow
 \begin{array}{r}
 0111 \\
 + 1011 \\
 \hline
 0010
 \end{array}
 \Rightarrow 2$$

Excess Notation

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

- Conversion

$$x \rightarrow (2^{n-1} + x) \mod 2^n$$

- Addition

$$\begin{aligned}
 x + y &\rightarrow \\
 &(2^{n-1} + (2^{n-1} + x) + (2^{n-1} + y)) \mod 2^n \\
 &= (2^{n-1} + x + y) \mod 2^n
 \end{aligned}$$

Overflow

- **Overflow** occurs when the arithmetic result is out of the range of representation.

$$\begin{array}{r} 010 \\ + 011 \\ \hline 101 \end{array}$$

- Addition of two positive numbers
 - $2 + 3 = 5 \rightarrow -3 \pmod{8}$
- Addition of two negative numbers
 - $(-2) + (-3) = -5 \rightarrow 3 \pmod{8}$

$$\begin{array}{r} 110 \\ + 101 \\ \hline 011 \end{array}$$

Fraction in Binary (Fixed-Point)

Base two system

1 **0** **1** . **1** **0** **1**

1 **0** **1** **1** **0** **1**

× × × × × ×

2^2 2^1 2^0 2^{-1} 2^{-2} 2^{-3}

4 0 1 $\frac{1}{2}$ 0 $\frac{1}{8}$

Total $5\frac{5}{8}$

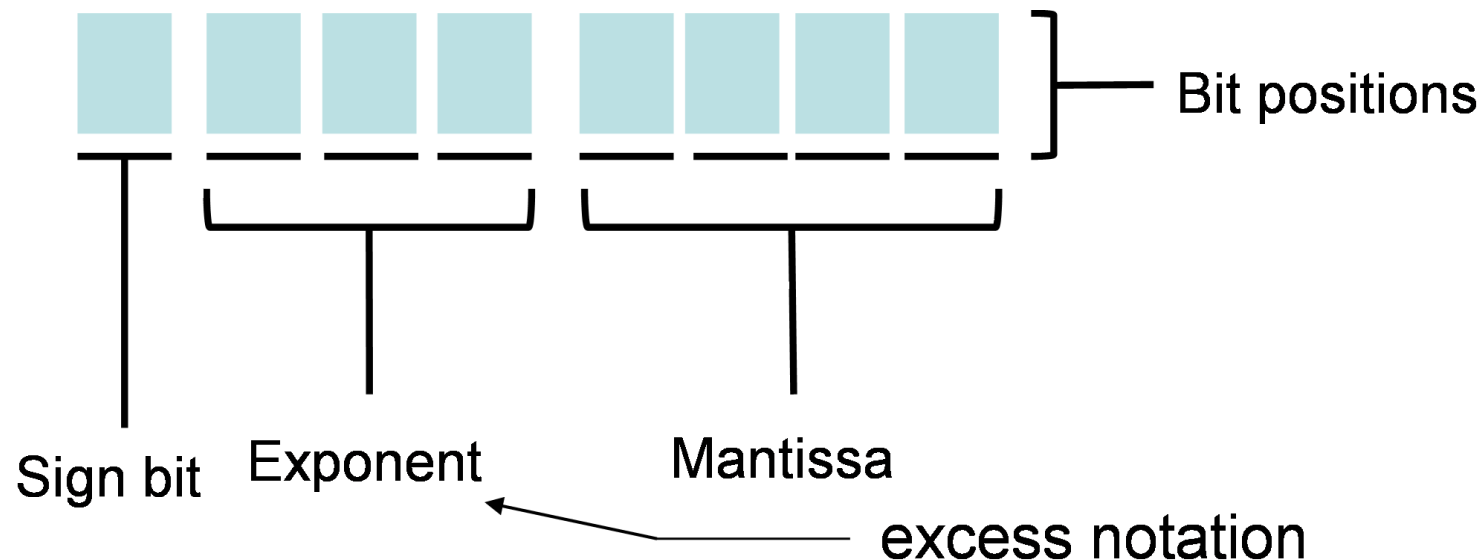
— Binary pattern

— Bit's value

— Position's quantity

Float-Point Notation

- Why? (How to represent 0.000000000000000001?)



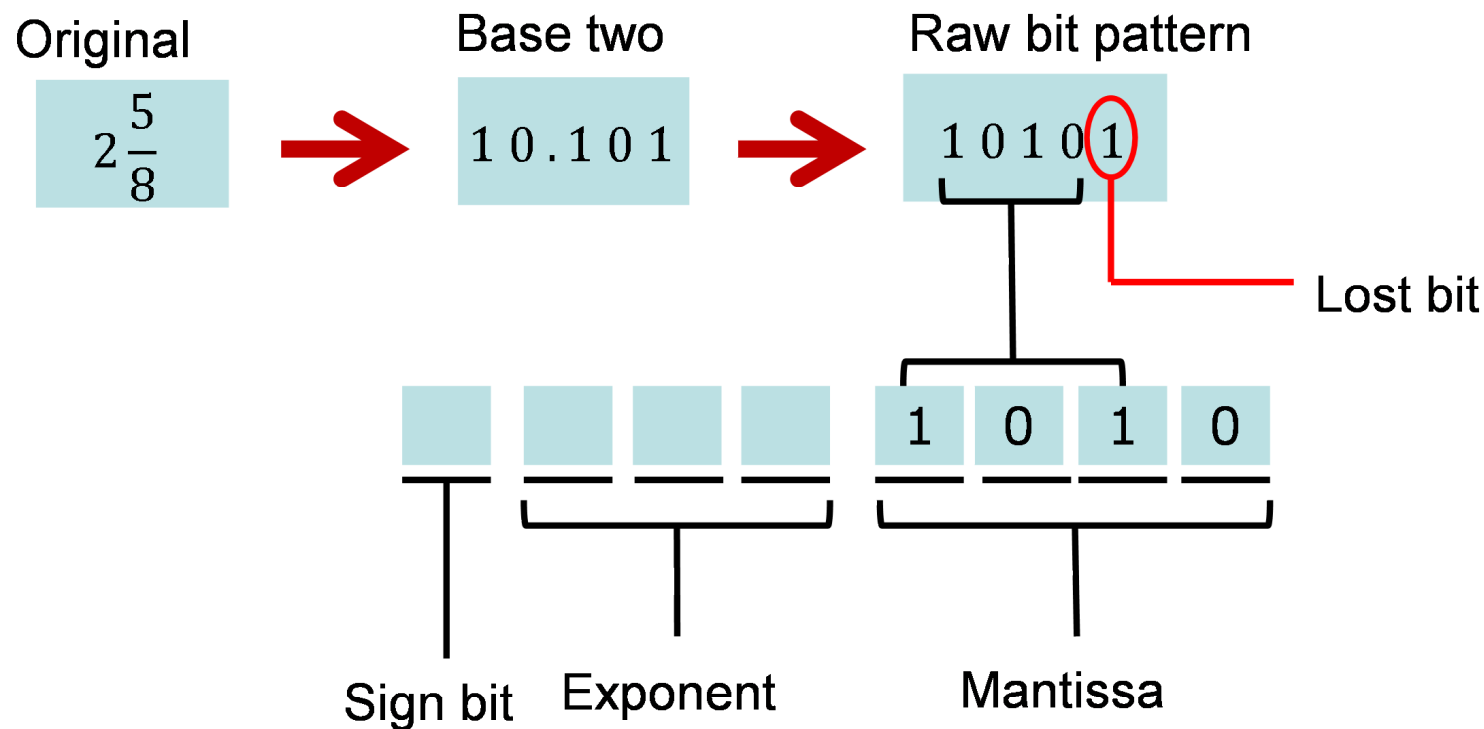
- On most current 64-bit computers, the exponent takes 11 bits, and the mantissa takes 52 bits (IEEE 754 standard).

Decoding Floating-Point

- 01101011
→ (0)(110)(1011)
→ (+)(+2)(1011)
→ $.1011 \rightarrow 10.11 \rightarrow 2 + \frac{1}{2} + \frac{1}{4} = 2\frac{3}{4}$
- 10010011
→ (1)(001)(0011)
→ (−)(−3)(0011)
→ $-.0011 \rightarrow -.0000011 \rightarrow -\left(\frac{1}{64} + \frac{1}{128}\right) = -\frac{3}{128}$

Truncation Errors

- Required precision is beyond the limitation of the mantissa.



The computer can only represent it as $2\frac{1}{2}$.

Normalized Form

- The most significant bit of mantissa is 1.
- 0's floating-point representation is all zero.
- Normalization
 - $01100011 \rightarrow (0)(110)(0011) \rightarrow .0011 \times 2^2$
 $\rightarrow .1100 \times 2^0 \rightarrow (0)(100)(1100) \rightarrow 01001100$
- IEEE standard
 - The left-most bit in mantissa is always 1 \rightarrow omit it.
 - An IEEE standard normalized form is $(s)(eee)(mmmm)$
 $\rightarrow (-1)^s \times 1.mmmm \times 2^{(eee-4)}$
 - $01100011 \rightarrow (0)(110)(0011) \rightarrow 1.0011 \times 2^{(6-4)}$

Loss of Digits

- $4 + \frac{1}{4} + \frac{1}{4}$

$$= 01111000 + 00111000 + 00111000$$

$$= 01111000 + 01110000 + 01110000$$

$$= 01111000 = 4 !!!$$

- $4 + \left(\frac{1}{4} + \frac{1}{4}\right)$

$$= 01111000 + (00111000 + 00111000)$$

$$= 01111000 + 01001000$$

$$= 01111000 + 01110001$$

$$= 01111001 = 4\frac{1}{2} !!!$$

- Just like when you use a calculator to do $10^{99} + 0.123 - 10^{99}$.

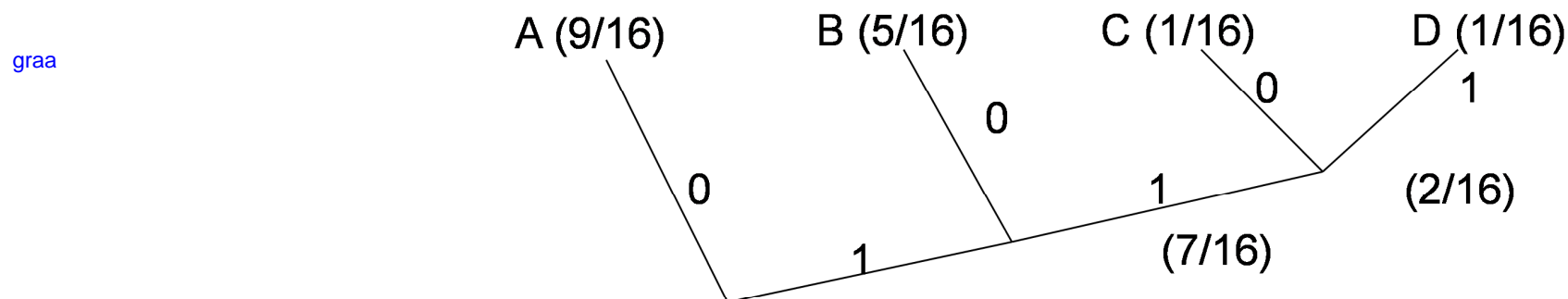
Data Compression

- Lossy vs. lossless
- Run-length encoding
- Frequency-dependent encoding
 - Huffman encoding
- Relative encoding / difference encoding
- Dictionary encoding
 - Adaptive dictionary encoding
 - LZW encoding

Huffman Encoding

AAABBBBAABCAAAABD

- Tradition encoding
 - $A \rightarrow 00$; $B \rightarrow 01$; $C \rightarrow 10$; $D \rightarrow 11$.
 - 000000010101000001100000000111 (32 bits).
- Huffman encoding
 - Count occurrences: $A(9)$; $B(5)$; $C(1)$; $D(1)$.
 - Build a Huffman tree.



- $A \rightarrow 0$; $B \rightarrow 10$; $C \rightarrow 110$; $D \rightarrow 111$.
- 0001010100010110000010111 (25 bits)

LZW Encoding

新增word給他流水號，存進dictionary

ZIP / RAR 壓縮檔的技術

- A dictionary encoding which does not need to store the dictionary.

dictionary 不必給
c.f. Huffman Encoding 要附上tree

- xyx xyx xyx xyx
- 1
- 12
- 121
- 1213 → (knowing xyx forms a word).
- 12134
- 121343434
- Decoding is similar.

Symbol	Code
x	1
y	2
space	3
xyx	4

In reality, simply use ASCII code. So no addition dictionary is needed.

Images, Audios, and Videos

- GIF: 256 colors, dictionary encoding
- JPEG
 - Lossy or lossless.
 - Discrete cosine transform. 傅立葉轉換
 - Discard high-frequency information that is insensitive to human eyes.
- MP3
 - Temporal masking 在較大的聲音訊號之後，背後的小訊號可以刪除
 - Frequency masking
- MPEG
 - Relative encoding & other techniques.

Communication Errors

- Compression
 - Remove redundancy.
- Error detection & correction
 - Add redundancy to prevent errors.
- Error detection: Check code
 - Cannot correct errors, but can check if errors occur.
 - ID numbers
 - ISBN
 - Parity code
- Error correcting
 - Can correct errors (to some degree).

Taiwan ID

$Ca_1a_2a_3a_4a_5a_6a_7a_8a_9$

- ① Convert the English letter C into a number xy :

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	1	1	1	1	1	1	1	3	1	1	2	2	2	3	2	2	2	2	2	2	2	3	3	3	3
0	1	2	3	4	5	6	7	4	8	9	0	1	2	5	3	4	5	6	7	8	9	2	0	1	3

- ② $d_1 = x + 9y$
- ③ $d_2 = \sum_{i=1}^8 (9 - i) \cdot a_i = 8 \cdot a_1 + 7 \cdot a_2 + \dots + 1 \cdot a_8$
- ④ Check code $a_9 = 10 - ((d_1 + d_2) \bmod 10)$

ISBN-10

The first 9 digits of ISBN-10 of the textbook is
0-273-75139

① Compute

$$S = 0 \cdot 10 + 2 \cdot 9 + 7 \cdot 8 + 3 \cdot 7 + 7 \cdot 6 + 5 \cdot 5 + 1 \cdot 4 + 3 \cdot 3 + 9 \cdot 2 = 193$$

② $M = S \bmod 11 = 6$

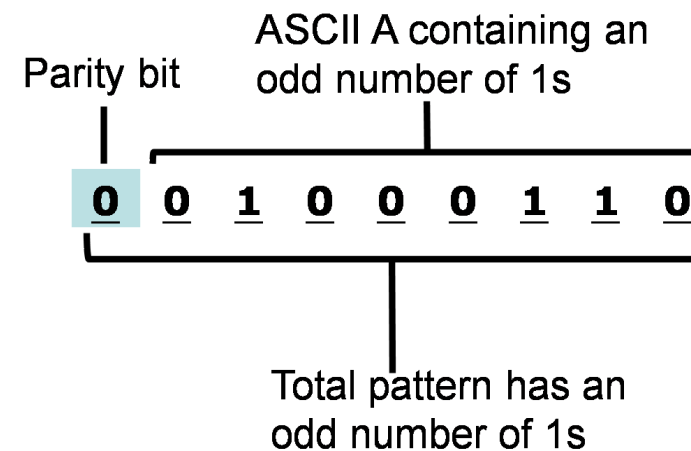
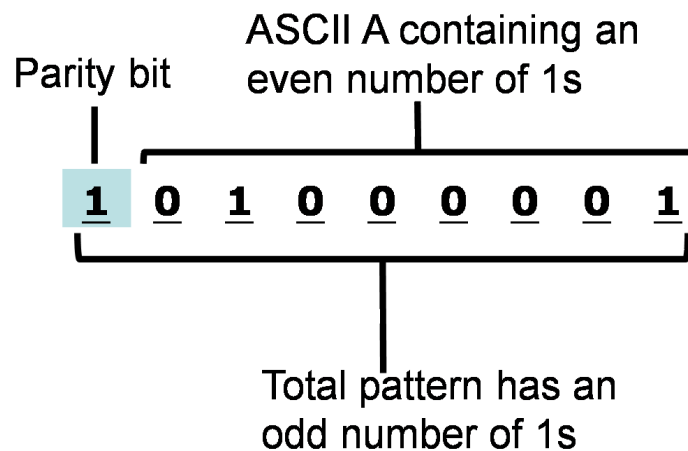
③ $N = 11 - M = 5$

- If $N = 10$, the check code is **X**.
- If $N = 11$, the check code is **0**.
- Otherwise, the check code is the number N

④ So the whole ISBN is **0-273-75139-5**.

Parity Bits

八個bits一個檢查碼



- Add an additional bit to make the whole odd number of 1s.
- Communication
- RAID (redundant array of independent disks) techniques

RAID five / six : 有問題的技術，現在是用RAID 0 + 1

An Error-Correcting Code (ECC)

奇數個code
 三個0 or 1 沒錯誤
 兩個0 or 1 直接取多數

- (3,1)-repetition code (can correct 1-bit errors)

Triplet received	Interpret as
000	0 (error free)
001	0
010	0
100	0
111	1 (error free)
110	1
101	1
011	1

Another Error-Correcting Code (ECC)

速度慢兩倍(3 bits)

- Maximized Hamming distances among symbols (at least 3).

Symbol	Code
A	000000
B	001111
C	010011
D	011100
E	100110
F	101001
G	110101
H	111010

- Received 010100.

Symbol	Distance
A	2
B	4
C	3
D	1
E	3
F	5
G	2
H	4

- 010100 → D.