Introduction to Computer Science
Lecture 1: Data Storage

Instructor: Tian-Li Yu

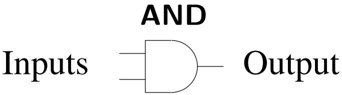
Taiwan Evolutionary Intelligence Laboratory (TEIL)
Department of Electrical Engineering
National Taiwan University

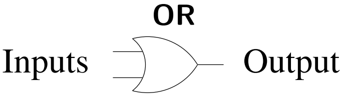
tianliyu@cc.ee.ntu.edu.tw

Slides made by Tian-Li Yu, Jie-Wei Wu, and Chu-Yu Hsu

Binary World

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



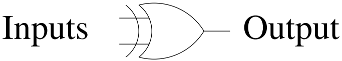


Inputs	Output
0 0	0
0 1	0
1 0	0
1 1	1

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

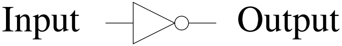
Logical Gates

XOR



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

NOT

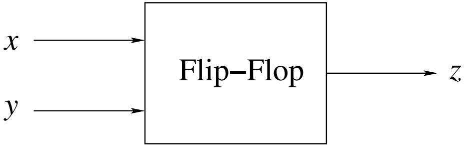


Inputs	Output
0	1
1	0

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

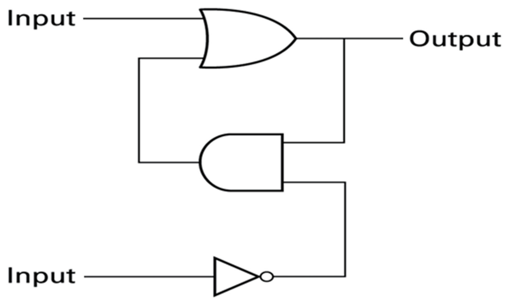
Flip-Flop

- 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.

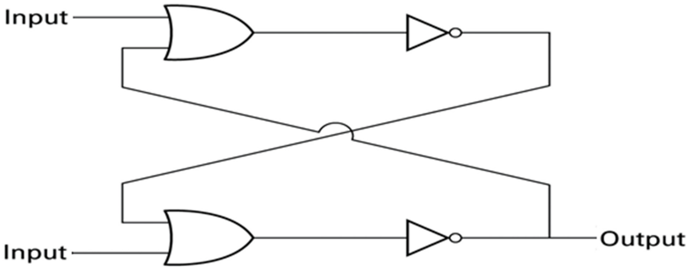


x	y	z
0	0	unchanged
0	1	0
1	0	1
1	1	undefined

A Simple SR Flip-Flop Circuit



Another SR Flip-Flop Circuit



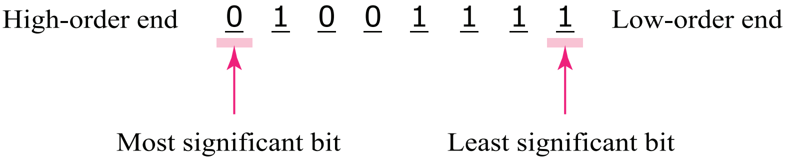
Hexadecimal Coding (Hex)

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.
- 0010111010110101 $\rightarrow$ 2EB5.

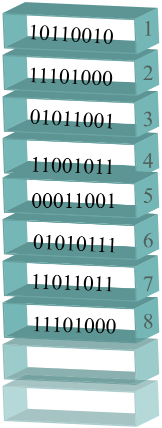
Main Memory Cells

- Cell: A unit of main memory (typically 8 bits which is one 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++.



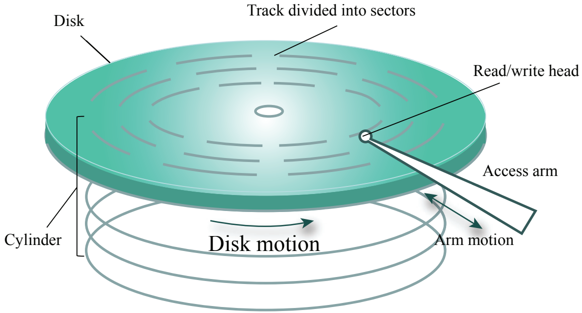
Memory Techniques

- 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

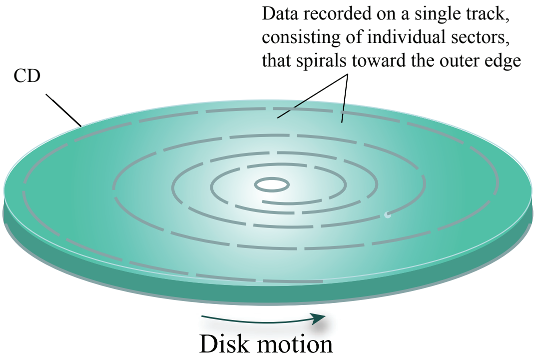
- Properties (compared with main memory)
  - Larger capacity
  - Less volatility
  - Slower
  - On-line or off-line
- Types
  - Magnetic systems (hard disk, tape)
  - Optical systems (CD, DVD)
  - Flash drives

Magnetic Disk Storage System

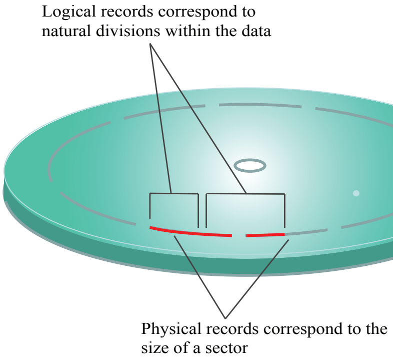


- Head, track, sector, cylinder
- Access time = seek time + rotation delay / latency time.
- Transfer rate (SATA 1.5/3/6, etc.)

Optical Storage



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

9	**0**	**1**	Representation
10^2	10^1	10^0	Position's quantity

Base two system

1	**0**	**1**	**1**	Representation
2^3	2^2	2^1	2^0	Position's quantity

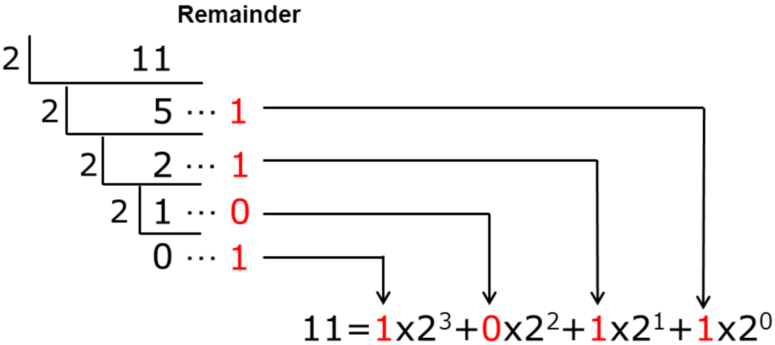
From Binary to Decimal

Base two system

	1	0	1	1
Binary pattern	1	0	1	1
Bit's value	1	0	1	1
	$\times$	$\times$	$\times$	$\times$
Position's quantity	2^3	2^2	2^1	2^0
	8	0	2	1
			Total	11

From Decimal to Binary

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

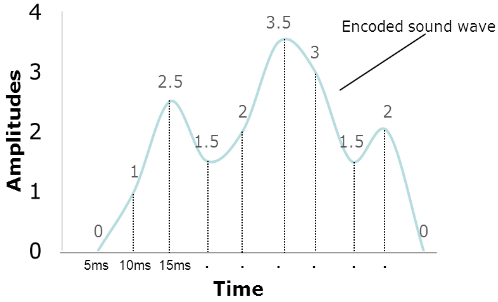


Representing Images

- Bit map techniques
  - Pixel: picture element.
  - Colors: RGB, HSV, etc.
  - LCD, scanner, digitcal 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

$$\begin{array}{r} 0 \\ +\quad 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 0 \\ +\quad 1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +\quad 0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +\quad 1 \\ \hline 10 \end{array}$$

- Subtraction?
  - Let's first define negative numbers.

Two's Complement Notation

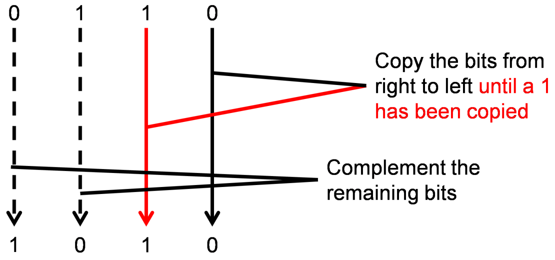
- Range: $-2^{n-1} \sim 2^{n-1} - 1$

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.

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

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

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

Excess Notation

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

- Conversion

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

- Addition

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

Overflow

- Overflow occurs when the arithmetic result is out of the range of representation.
- Addition of two positive numbers
  - $2 + 3 = 5 \rightarrow -3 \pmod 8$
- Addition of two negative numbers
  - $(-2) + (-3) = -5 \rightarrow 3 \pmod 8$

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

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

Fraction in Binary (Fixed-Point)

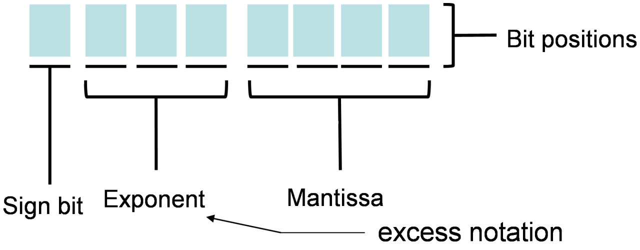
Base two system

1	**0**	**1**	**.**	**1**	**0**	**1**	— Binary pattern
1	**0**	**1**		**1**	**0**	**1**	— Bit's value
×	×	×		×	×	×	
2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	— Position's quantity
4	0	1		$\frac{1}{2}$	0	$\frac{1}{8}$	

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

Float-Point Notation

- Why? (How to represent 0.000000000000001?)



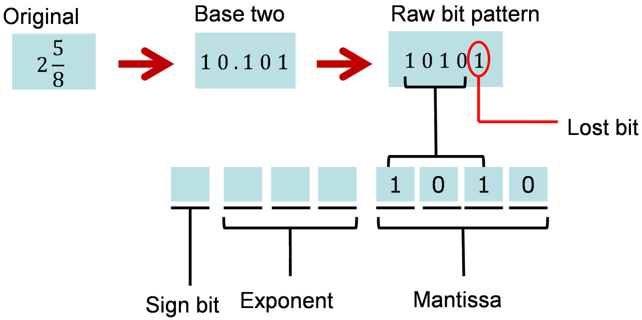
- On most current 64-bit computers, the exponent takes 11 bits, and the mantissa takes 52 bits (IEEE 754 standard).

Decoding Floating-Point

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

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

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

$$\begin{aligned} &= 01111000 + 00111000 + 00111000 \\ &= 01111000 + 01110000 + 01110000 \\ &= 01111000 = 4 \text{ !!!} \end{aligned}$$

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

$$\begin{aligned} &= 01111000 + (00111000 + 00111000) \\ &= 01111000 + 01001000 \\ &= 01111000 + 01110001 \\ &= 01111001 = 4\frac{1}{2} \text{ !!!} \end{aligned}$$

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

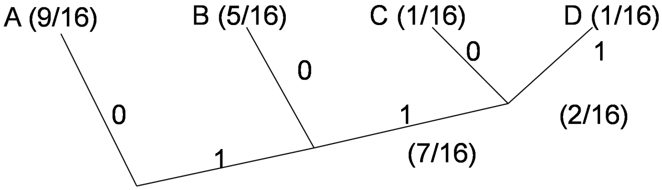
Data Compression

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

Huffman Encoding

AAABBBAABCAAAABD

- Tradition encoding
  - A $\rightarrow$ 00; B$\rightarrow$ 01; C$\rightarrow$ 10; D$\rightarrow$ 11.
  - 00000001010100000110000000000111 (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

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

- 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.
  - Discard high-frequency information that is insensitive to human eyes.
- MP3
  - Temporal masking
  - 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
10	11	12	13	14	15	16	17	34	18	19	20	21	22	35	23	24	25	26	27	28	29	32	30	31	33

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

National Taiwan University OpenCourseWare 臺大開放式課程

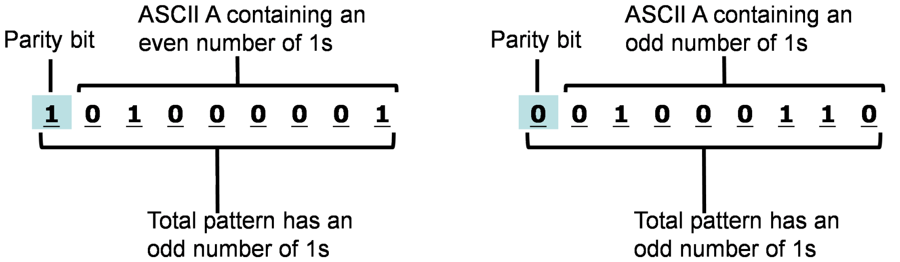
ISBN-10

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. $M = S \mod 11 = 6$
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.

National Taiwan University OpenCourseWare 臺大開放式課程

Parity Bits



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

An Error-Correcting Code (ECC)

- (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)

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

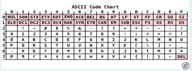
Symbol	Code
A	000000
B	001111
C	010011
D	011100
E	100110
F	101001
G	110101
H	111010

- Received 010100.

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

- 010100 $\rightarrow$ D.

License

Page	File	Licensing	Source/ author
17			Wikimedia.,Author:Anomie, Source: http://commons.wikimedia.org/wiki/File:ASCII_Code_Chart.svg, Date:2012/02/05, This file is ineligible for copyright and therefore in the public domain