Class Topics (클래스 홈페이지 참조)

- Part 1: Fundamental concepts and principles
  1) Invention of computers and digital logic design
  2) Abstractions to deal with complexity
  3) Data (versus code)
  4) Machines called computers
  5) Underlying technology and evolution since 1945
- Part 2: 빠른 컴퓨터를 위한 설계 (ISA design)
- Part 3: 빠른 컴퓨터를 위한 구현 (pipelining, cache)

Machines Called Computers

Part 3: Data

(Related to Textbook Chapter 3)

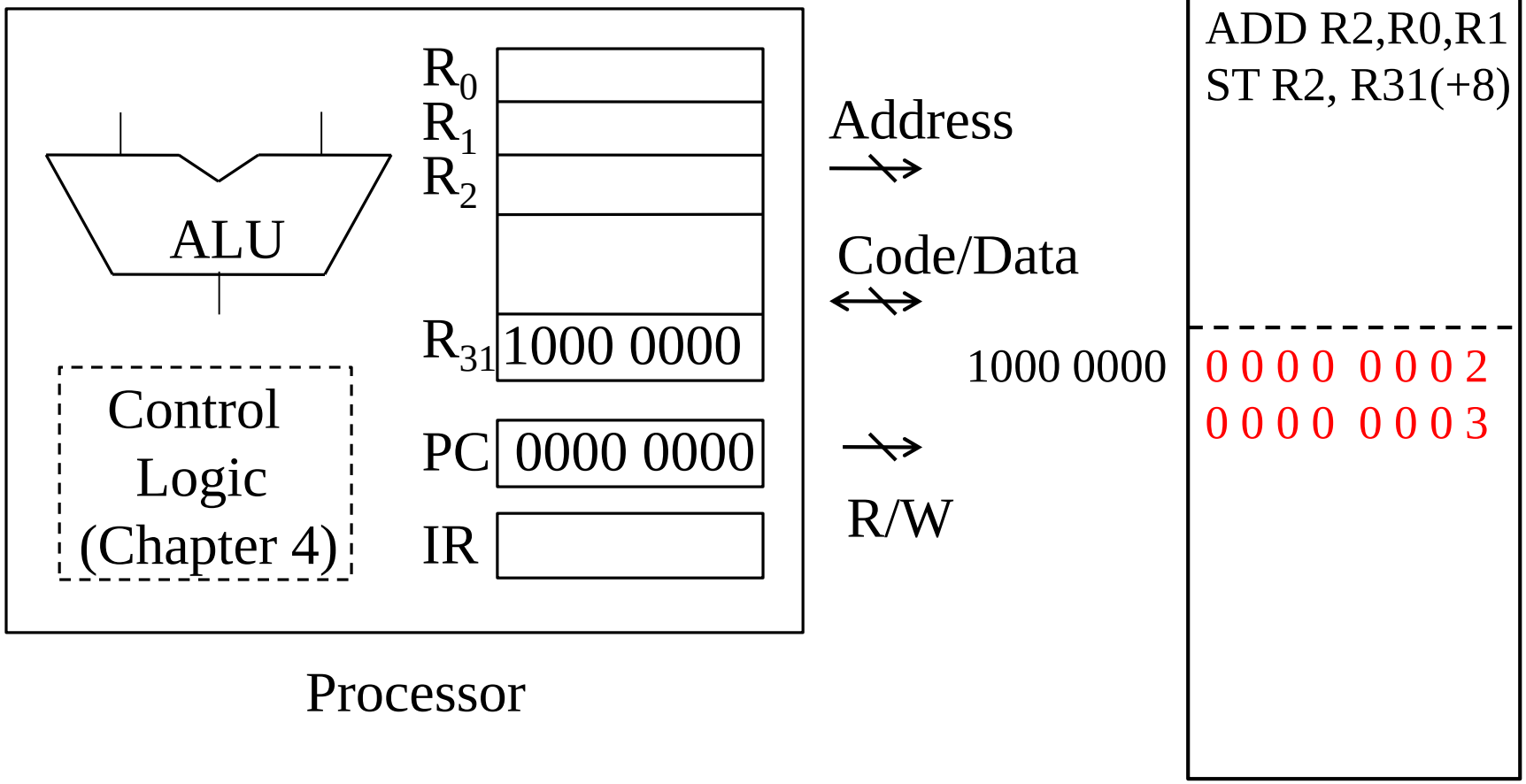
(비교적 가볍게 다룸)

Reference:

1. Fundamentals of Computer Science, Forouzan and Mosharraf

Machines Called Computers

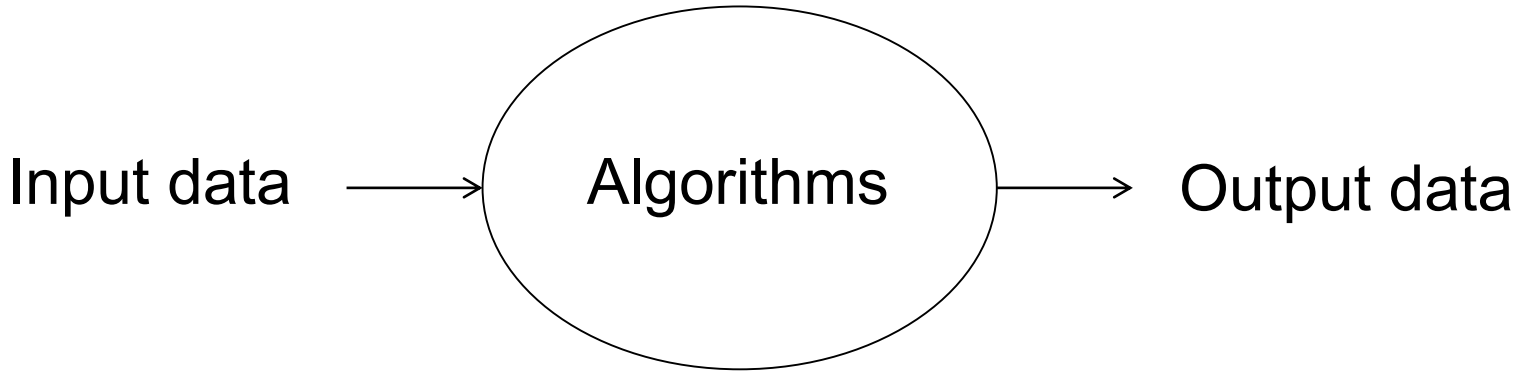
- Data in binary form
  - Programs and address as well



† I/O devices are just like memory

Data vs. Processing

- Problem solving by programming
  - How to represent information (data)
  - How to manipulate information (processing)



Evolution of Data

- Scientific computing
  - Solve differential equations
  - Data: numbers
    - Integers, floating-point numbers
- Business computing (e.g., IBM) since 1945
  - Database, data: characters or text
- Internet applications (since 1990s)
  - Multimedia data: audio, image, video

† Learn from data (corporate data as well as public data)

How do we use computers?

- Processors (계산)
  - Execute programs, compute
- Memory (저장)
  - Programs and data, files and folders
  - Data centers
- Input and output (I/O; 접속)
  - Human interactions, Internet connections
  - Internet, mobile commerce

† Why do you buy computers? (계산, 저장 또는 접속)

† 사람의 계산 속도 및 기억 능력과 비교 (실수, 피로 없음)

Data, Programs, Address in Binary

- Focus on data
  - Numeric data first
    (meaning of data in 1945)
    - Integers first

Number Systems

- Positional number systems
  - Position of a symbol determines value it represents
  - Decimal number system
    - {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
      - Concept more than 5000 years old
  - Binary number system
    - {0, 1}
      - Binary digit or bit
      - Concept more than 2000 years old

Decimal Integers

	10^2	10^1	10^0	
	3	2	5	Number

N = $3 \times 10^2 + 2 \times 10^1 + 5 \times 10^0$ Value

Binary Integers

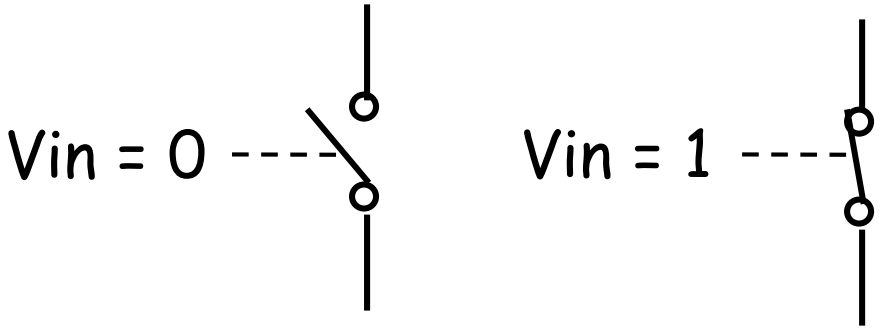
2^4	2^3	2^2	2^1	2^0	Weight
1	0	1	1	0	Number

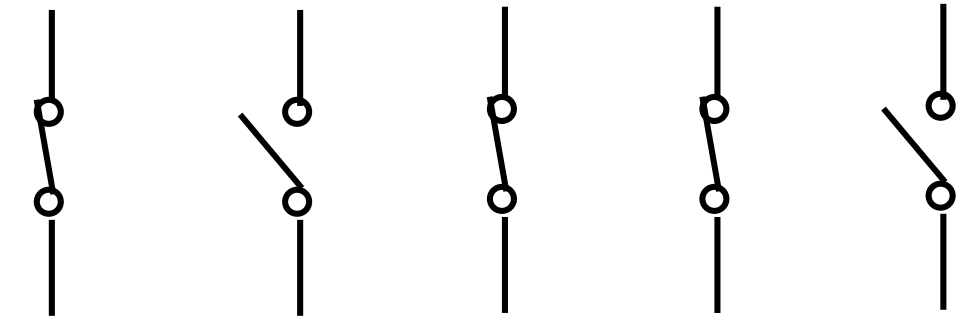
$N = 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$ Value

- "10110" in binary: 5-bit notation
  - Equivalent decimal value N = 16 + 4 + 2 = 22
- As natural as decimal notation
  - Only need more digits

Why binary numbers in computers?

- Transistors (underlying hardware)
  - 3-terminal digital switch: two stable states (ON, OFF)





1 0 1 1 0

Why we don't see binary numbers?

- Software translation between human users and machines (abstraction)

Human Users

Human-friendly form (decimal, characters)

Software (OS, Applications)

Binary form

Machines

(Unsigned) Binary Integers

- Range of expression: simply need more bits than decimal
  - 1-bit: 0 to 1
  - 2-bit: 00 to 11 (0 to 3 in decimal)
  - 4-bit: 0000 to 1111 (0 to 15 in decimal)
  - 8-bit ("byte"): 0000 0000 to 1111 1111 (0 to 255_{10})
  - 16-bit: 0 to $2^{16} - 1$
    - 64K (65,636)
  - 32-bit: 0 to $2^{32} - 1$
    - 4G (4,294,967,296)

Standard Prefixes

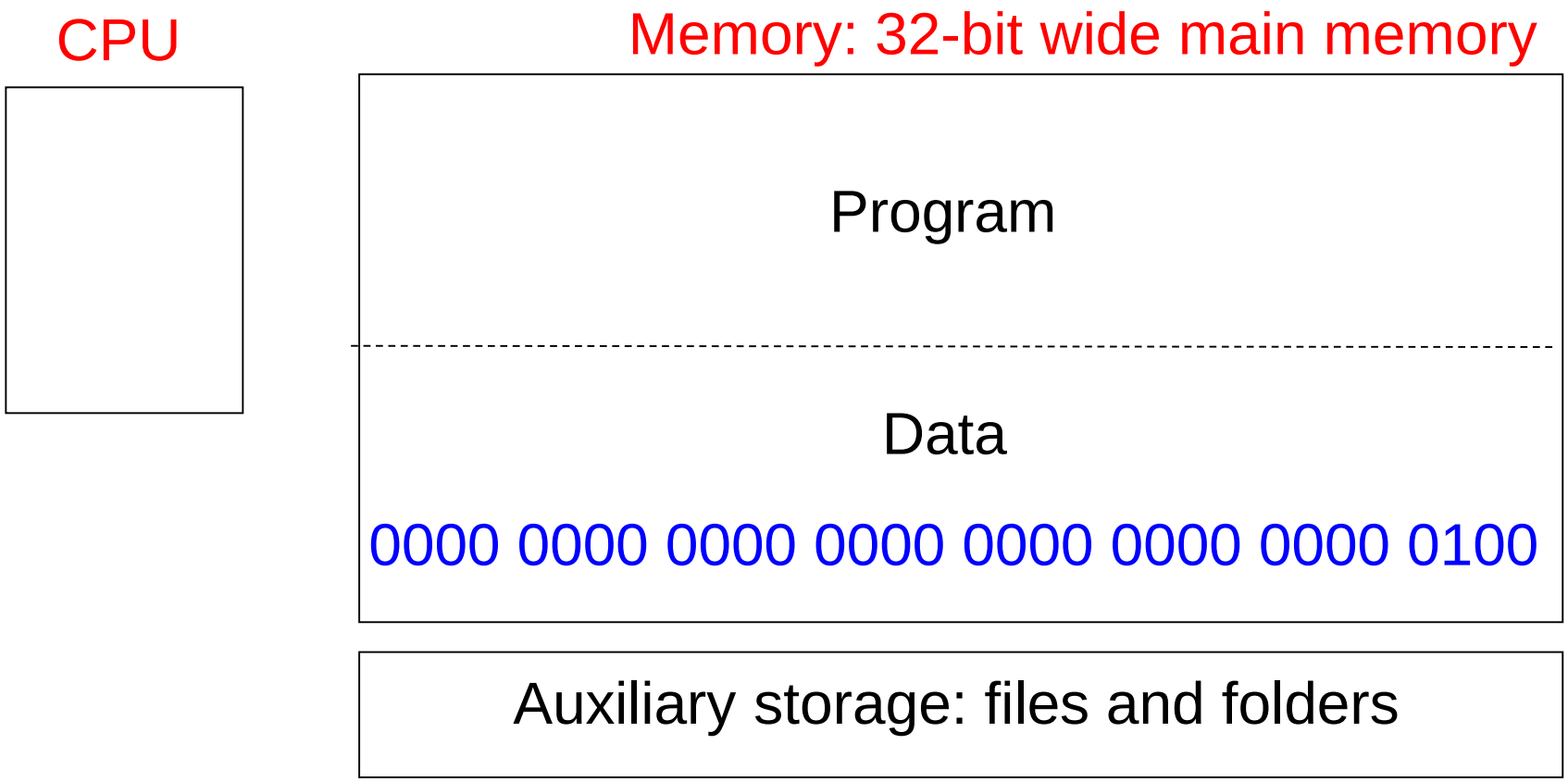
			CS	CS
yotta	Y	10^{24}	2^{80}	
zetta	Z	10^{21}	2^{70}	
exa	E	10^{18}	2^{60}	
peta	P	10^{15}	2^{50}	
tera	T	10^{12}	2^{40}	
giga	G	10^{9}	2^{30}	1,073,741,824
mega	M	10^{6}	2^{20}	1,048,576
kilo	K	10^{3}	2^{10}	1,024

Where are we?

- Storage
  - Passed 10^{18} (i.e., exa) bytes years ago
  - Millions of 1TB disks
    - Daily disk failures
- Computation
  - Passed 10^{15} (i.e., peta) FLOPS years ago
  - Supercomputers (competition)
    - Millions of processors

32-bit Computers

- Memory: many slots to store data



I/O: Monitor, keyboard, LAN-Internet, …

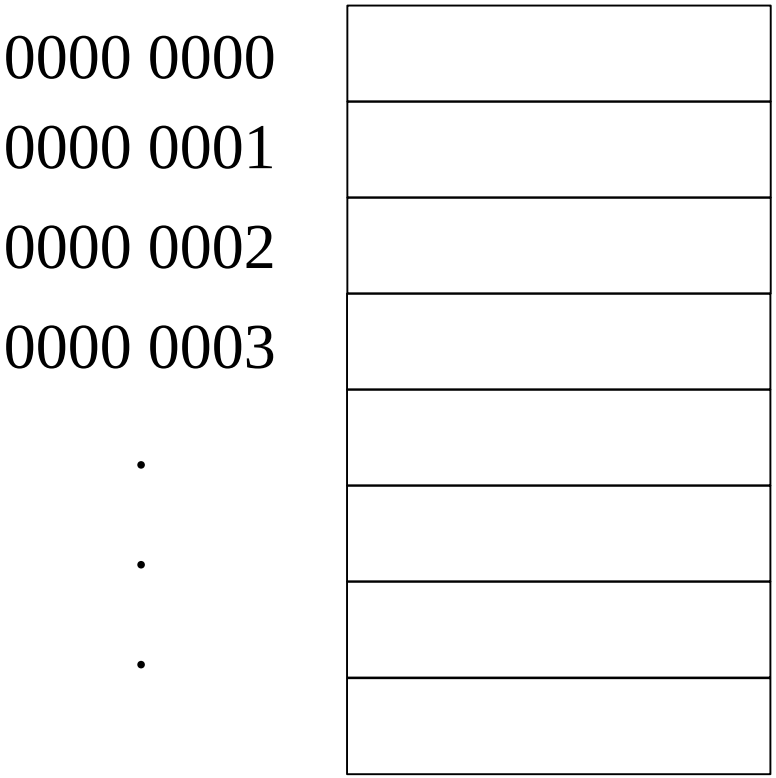
Hexadecimal Notation

- 0000 0000 0000 0000 0000 0000 0000 0100 (32-bit binary)
  - 00000004 in hex (8 hex digits)
  - Pure mnemonic (for easy human recognition)

Binary	Hex	Binary	Hex	Binary	Hex	Binary	Hex
0000	0	0100	4	1000	8	1100	C, c
0001	1	0101	5	1001	9	1101	D, d
0010	2	0110	6	1010	A, a	1110	E, e
0011	3	0111	7	1011	B, b	1111	F, f

Memory Model

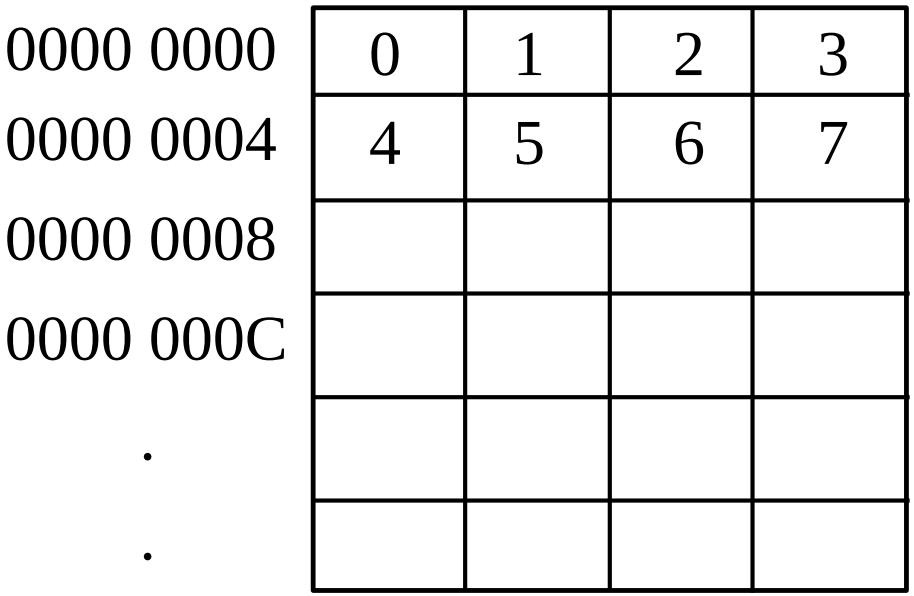
- How many bits to store in a single memory address?
  - Bit, byte or what? (And why?)



- Byte addressing

Memory Model, 32-Bit Machines

- Require 32-bit access, 4열 종대
  - Addresses for "char", "16-bit int", "32-bit int"?
  - Addresses for instructions?



- What happens in 16-bit processors?

Binary integers

- Negative integers
- Real numbers

(Chapter 3 미리보기)

Negative Numbers

- How to represent negative integers?

Unsigned binary	Two's complement
000 = +0	000 = +0
001 = +1	001 = +1
010 = +2	010 = +2
011 = +3	011 = +3
100 = +4	100 = -4
101 = +5	101 = -3
110 = +6	110 = -2
111 = +7	111 = -1

Real Numbers

- Fixed-point representation: a few bits below binary point

2^2	2^1	2^0		2^{-1}	2^{-2}	Weight
1	0	1	.	1	1	Number

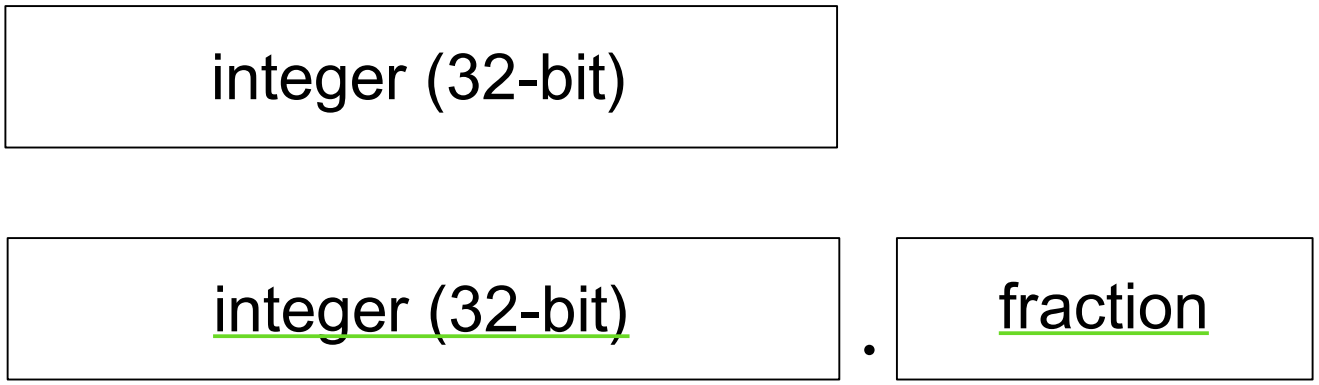
$$N = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} \quad \text{Value}$$

  - Equivalent decimal value N = 4 + 1 + 0.5 + 0.25 = 5.75

- HW support for more precision (fixed-point arithmetic)

Fixed-Point Arithmetic

- Hardware support for more precision



- Many/∞ bits below binary point
  - Truncation, lose accuracy (실수연산은 근사연산)

Floating-Point Numbers (Chap. 3)

- What is floating-point representation?
  - Very big and small numbers (i.e., scientific numbers)
    - 1.001×2^{-97}, 1.110×2^{68}
    - Need too many bits: store mantissa and exponent
  - Arithmetic becomes quite complex
    - Quite different from binary arithmetic (근사연산)
- May use 32-bit, 64-bit or others

1.001	-97

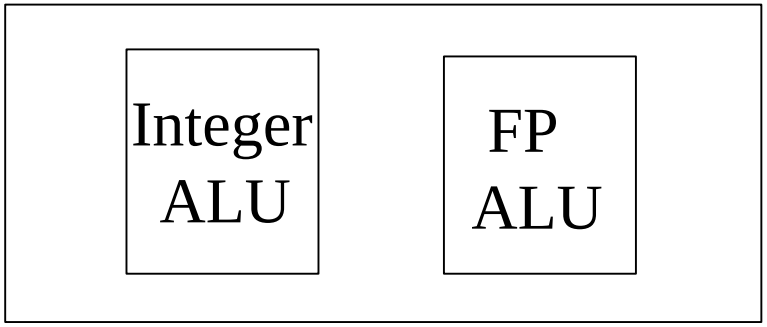
1.110	68

Floating-Point Numbers (Chap. 3)

- Two types of numbers: integers and floating-point
  - Two types of ALUs: integer ALU and FP ALU
  - Does your PC have both?
    - What about your smartphone?

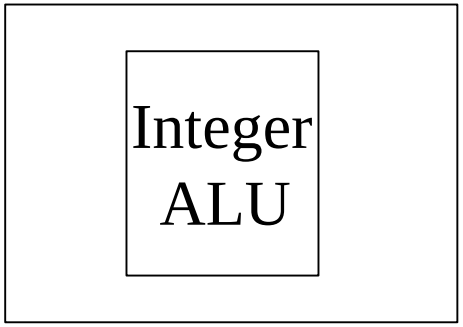
Integer and FP ALUs

- Processors for general-purpose computers



(과학계산 포함)

- Processors for embedded systems



Smaller,
cheaper,
low-power
(모바일 응용)

  - Fixed-point hardware support

Built-in Data Types in C

- Integers
  - short or long, signed or unsigned
  - Size is machine dependent (but int is at least 16 bits)
- Characters (char)
  - Single byte capable of holding one character
  - In a sense, small integers
- Floating point numbers (float, double)
  - Single or double precision (IEEE 754; 32/64 bits)

† Boolean, string (implicit in C)

Data: Beyond Numbers

- Different types of data
  - Numbers
  - Text
  - Audio, image, video
- Computation: more than arithmetic

Different Types of Data

- The term "multimedia"
  - Text, audio, image, video
    - All in binary patterns (integers)

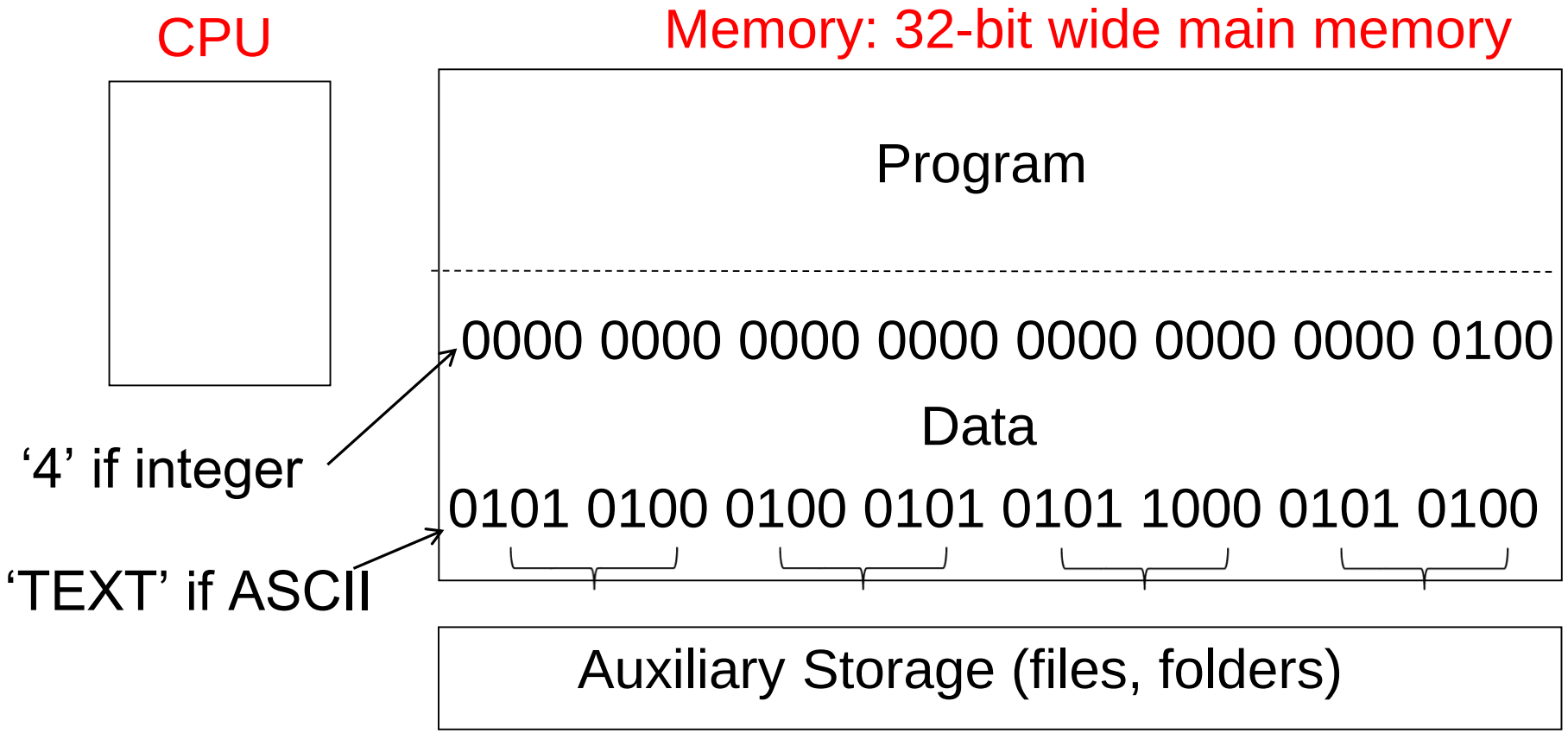
ASCII Code

USASCII code chart

Bits $b_7 b_6 b_5$ →	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
b_4 b_3 b_2 b_1 ↓ Row / Column →	0	1	2	3	4	5	6	7
0 0 0 0	0	NUL	DLE	SP	0	@	P	`	p
0 0 0 1	1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	2	STX	DC2	"	2	B	R	b	r
0 0 1 1	3	ETX	DC3	#	3	C	S	c	s
0 1 0 0	4	EOT	DC4	$	4	D	T	d	t
0 1 0 1	5	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	6	ACK	SYN	&	6	F	V	f	v
0 1 1 1	7	BEL	ETB	'	7	G	W	g	w
1 0 0 0	8	BS	CAN	(	8	H	X	h	x
1 0 0 1	9	HT	EM	)	9	I	Y	i	y
1 0 1 0	10	LF	SUB	*	:	J	Z	j	z
1 0 1 1	11	VT	ESC	+	;	K	[	k	{
1 1 0 0	12	FF	FS	,	<	L	\	l	|
1 1 0 1	13	CR	GS	-	=	M	]	m	}
1 1 1 0	14	SO	RS	.	>	N	^	n	~
1 1 1 1	15	SI	US	/	?	O	_	o	DEL

32-bit Computers

- Memory: many slots to store data



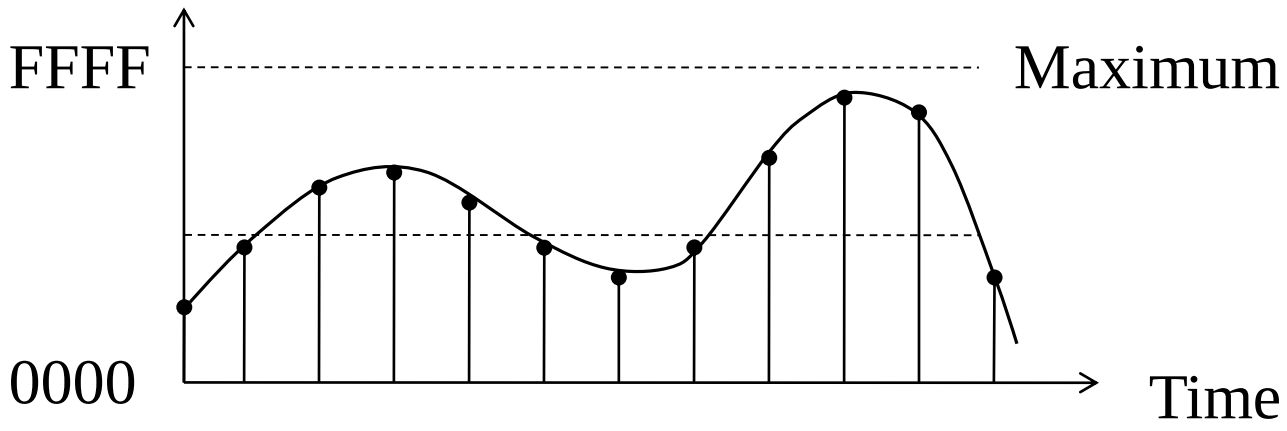
I/O: Monitor/keyboard, LAN-Internet, …

Representing Text (Integers)

- Each character is assigned a unique bit pattern
  - ASCII
    - 7-bit to represent 128 symbols in English text
  - ISO: a number of 8-bit extensions to ASCII
    - To accommodate different language groups
  - Unicode
    - Alphabets in world's languages plus symbols
      † "Internationalization"
    - Characters encoded in 1 byte to 4 bytes

Storing Audio (Integers)

- Speech or music represented by sequence of integers
- Dominant standard: MP3 (소리의 질: 샘플빈도, bit 수)
  - 44100 samples per second (sampling theorem)
  - 16 bits per sample: 0000 to FFFF in hexadecimal



- 음성 인식, computer music

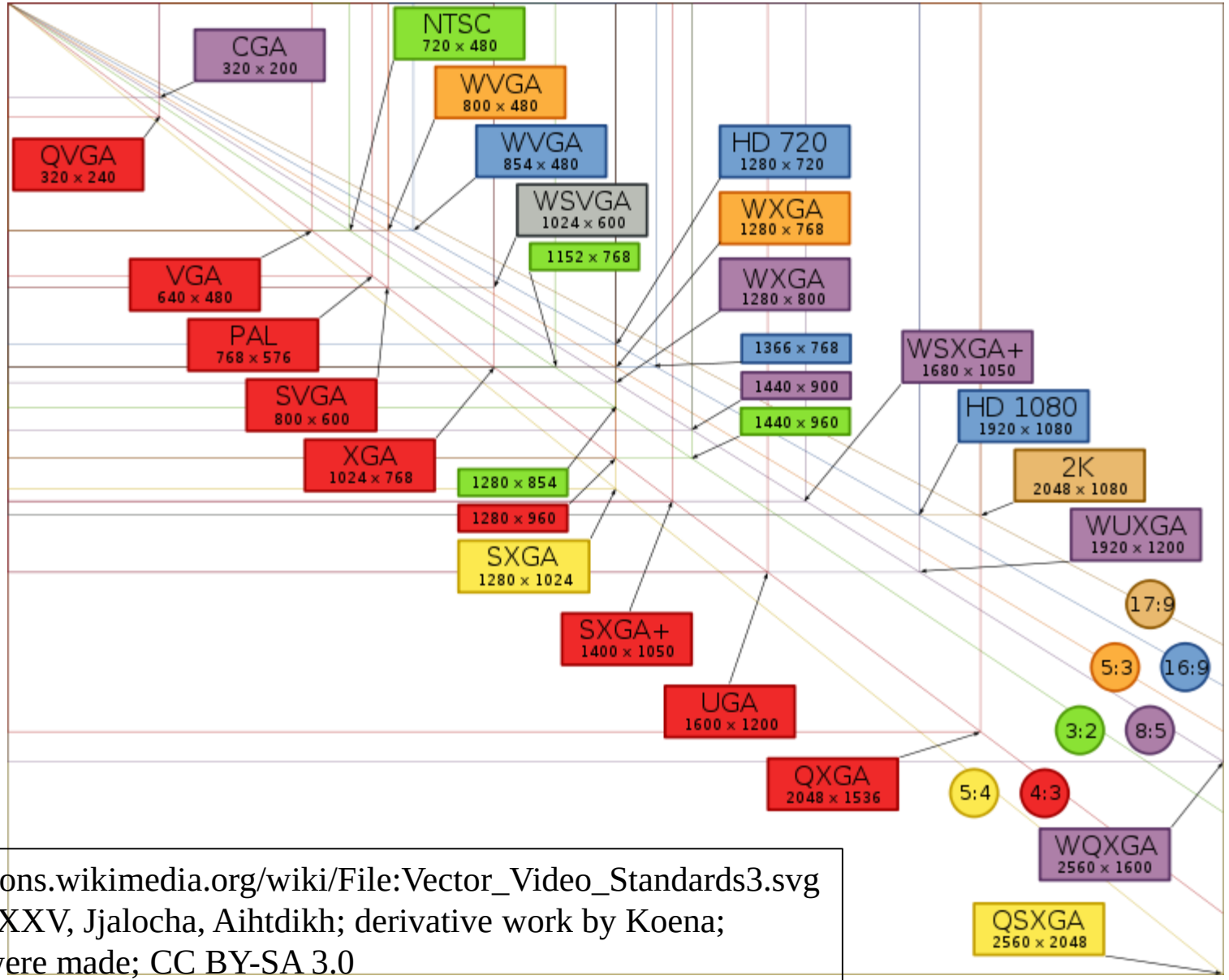
Storing Images (Integers)

- JPEG standard
  - True color: 24 bits per pixel (R,G,B 각각 8 bits)

Color	Red	Green	Blue	Color	Red	Green	Blue
Black	0	0	0	Yellow	255	255	0
Red	255	0	0	Cyan	0	255	255
Green	0	255	0	Magenta	255	0	255
Blue	0	0	255	White	255	255	255

- Full HD resolution: $(1920 \times 1080) \times 3$ bytes $\approx$ 6 MB
  - Compress to reduce size
- 물체 인식 (자율 주행), Photoshop

Display Resolution



https://commons.wikimedia.org/wiki/File:Vector_Video_Standards3.svg
uploaded by XXV, Jjalocha, Aihtdikh; derivative work by Koena;
no changes were made; CC BY-SA 3.0

Storing Video (Integers)

- Images (or frames) over time
  - e.g., 30 frames per second
- MPEG (Moving Picture Experts Group) standard
  - Lossy compression
    - MPEG-4
  - † Lossless compression (e.g., ALZip)

Summary

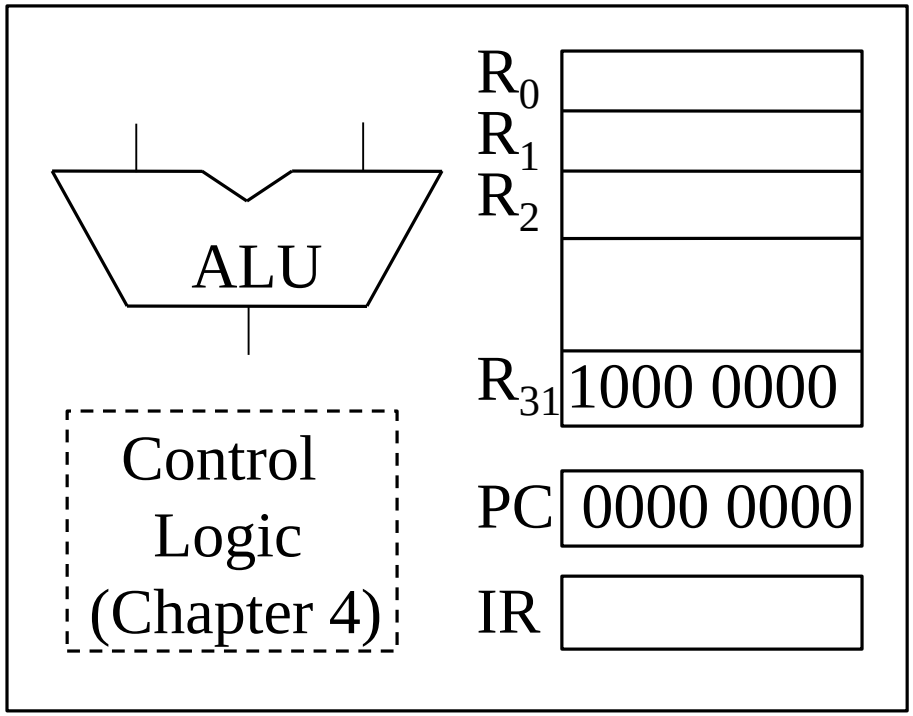
- Looked into "data" first
  - Numeric data for computation
    - Binary integers
    - Floating-point numbers
      - † Fixed-point arithmetic
  - Business computing: text data (integers)
  - Multimedia data: audio, image, video (integers)

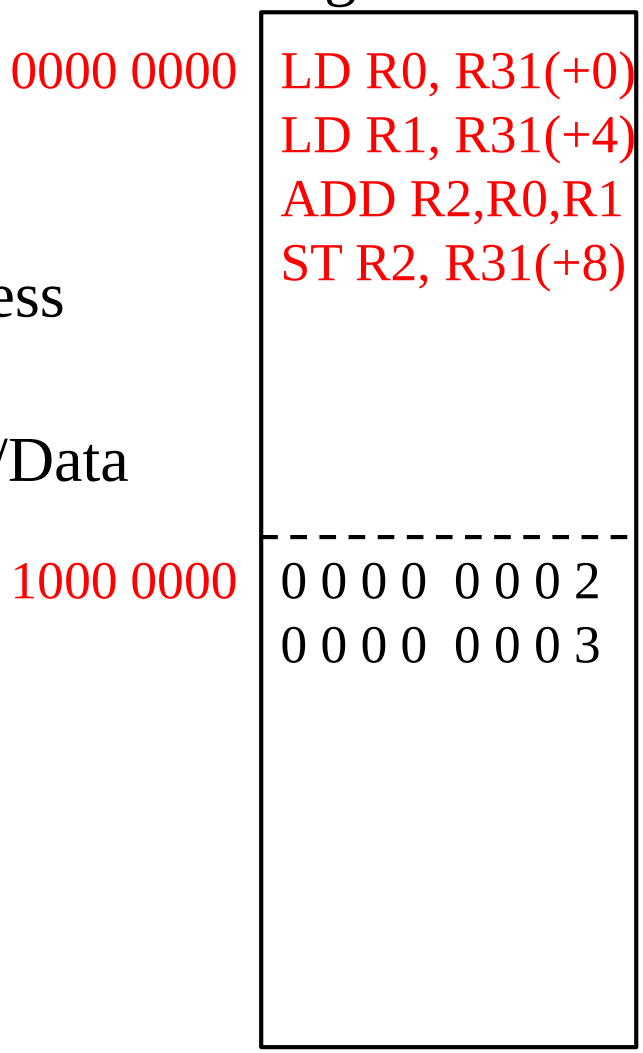
All in Binary Form: Address, Data and Programs

(Let's look into programs)

Machines Called Computers

- Data, address in binary form
  - Programs as well





† I/O devices are just like memory

Class Topics (클래스 홈페이지 참조)

- Part 1: Fundamental concepts and principles
  1) Invention of computers and digital logic design
  2) Abstractions to deal with complexity
  3) Data (versus code)
  4) Machines called computers
  5) Underlying technology and evolution since 1945
- Part 2: 빠른 컴퓨터를 위한 설계 (ISA design)
- Part 3: 빠른 컴퓨터를 위한 구현 (pipelining, cache)

To Think About (skip)

Why "Digital" Computers?

- Nature: continuous differential equation (i.e., analog)
- Quantization errors
  - Can reduce it with more bits
- Processing and communication errors